SIEMENS

SIMATIC

Process Control System PCS 7 Advanced Process Library (V9.0)

Function Manual

03/2017
A5E39147331-AA

Legal information

Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

⚠ DANGER

indicates that death or severe personal injury **will** result if proper precautions are not taken.

⚠ WARNING

indicates that death or severe personal injury **may** result if proper precautions are not taken.

⚠ CAUTION

indicates that minor personal injury can result if proper precautions are not taken.

NOTICE

indicates that property damage can result if proper precautions are not taken.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

⚠ WARNING

Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Division Process Industries and Drives
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E39147331-AA
℗ 05/2017 Subject to change

Table of contents

Security information

1

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines, and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions only form one element of such a concept.

Customer is responsible to prevent unauthorized access to its plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.

Additionally, Siemens' guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit:

http://www.siemens.com/industrialsecurity.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under

http://www.siemens.com/industrialsecurity.

Basics of APL

2

2.1 Functions of the blocks

2.1.1 General information

2.1.1.1 User-configured message classes

User-configured message classes

The message classes Alarm, Warning and Tolerance, the corresponding abbreviations in symbols and colors, and the terms used in this documentation are not valid for user-configured message classes.

These terms and colors depend on the project-specific setting. The user-configured messages classes are only supported for block symbols as of V8.

2.1.1.2 Forcing operating modes

Forcing operating modes

The "forcing of operating modes" function lets you set the function block into a different operating mode using interconnectable input parameters, regardless of the currently active control. This can, for example, be:

- Forces tracking for closed-loop controllers and control valves
- Enabling and disabling at motors
- Opening and closing of valves

It is only possible to force operating modes with "Large" blocks in the following operating modes:

- Manual mode
- Automatic mode
- Local mode (only if `Feature2.Bit8` = 1)

Forcing operating modes has the highest priority over all three operating modes.

Note

It is not possible to force operating modes in local mode.

Forcing operating modes at closed-loop controllers

In control engineering, this procedure is also known as forced tracking of values. Refer to the Tracking and limiting a manipulated variable (Page 192) section for more on this.

Forcing operating modes at motors and valves

The input parameter `xxxxForce = 1` (for example `OpenForce` and `CloseForce` at a valve) is used for forced controlling of the function block and thus an intervention in the function of the block, irrespective of currently active controls, interlock conditions and monitoring errors. If the input parameters are inconsistent (for example `OpenForce = 1` and `CloseForce = 1` at valves), an error number (Page 119) is output at the parameter `ErrorNum` and the control remains unchanged.

Note

If you have set the parameters for the advance warning time `WarnTiMan` and the idle time `IdleTime` to values higher than 0, the control will only take effect once the set times have elapsed.

Note

With block VlvAnl, the warning time is ignored in tracking `MV_TrkOn` = 1 and in forced tracking `MV_ForOn`.

The Enabling direct changeover between forward and reverse (Page 145) feature bit has no effect when forcing the operating modes of the MotRevL and MotSpdCL blocks. Direct switchover between forward and reverse is always possible.

Display in the faceplate and in the block icon

If an operating mode is forced, this is displayed in the block icon and in the standard view of the faceplate:

Block icon: In the block icon, the display for motors, valves and dosers involves the use of a red F and a crossed-out padlock.

There is no display for closed-loop controllers.

Faceplate: An information text on the forced operating mode is displayed in the standard view of the faceplate, for example, "Forced stop" for motors. This is also indicated by a crossed-out padlock:

Messaging

No messages are assigned to the forcing of operating modes. However, if you want to have corresponding messages, you can use the freely interconnectable input parameters to generate the messages. Refer also to the Generating instance-specific messages (Page 200) section for more on this.

2.1.1.3 Resetting the block in case of interlocks or errors

Resetting the block

The block must be reset when an interlock has been set via the `Protect` input ("Protection"), Trip ("Protection") or an error has occurred ("Runtime" or "Control", external error `FaultExt` or `CSF` with Feature Bit 18).

Note

"Small" blocks do not feature protection (`Protect`).

The `RdyToReset` output signals when a reset can be carried out via the `RstLi` input parameter or the automatic commands.

There are different ways to reset the block:

- Reset by interconnection (input `RstLi`).
- Reset by the operator using a button in the faceplate (input `RstOp`).
- Reset with a 0-1 edge transition in the corresponding automatic or local signal (except with motor protection). Refer to the following sections for more information.

Note

The reset via input RstLi or RstOp does not depend on the selected operating mode.

The operator must have the appropriate authorization to use the reset function in the faceplate (`OS_Perm`). After a reset, the output parameter `P_Rst` is set for a cycle.

Resetting monitoring errors and interlocks in manual and automatic mode

You can influence the reaction using the following `Feature` Bits:

- `Feature` Bit 9: Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
- `Feature` Bit 30: Set reset depending on the operating mode or the LiOp parameter (Page 162)
- `Feature` Bit 31: Activating reset of protection / error in manual mode (Page 164)

Note

The following applies for valves:

With `MonSafePos = 0`, no reset is required; the valve can be moved in spite of the response fault.

Resetting monitoring errors, external errors and interlocks in local mode

The monitoring error can occur in local mode if you have set 1, 3 or 5 for the input parameter `LocalSetting` (see Local mode (Page 79)). When `LocalSetting` is set to 2 or 4, a monitoring error can only occur when a rapid stop is triggered.

The following applies with `LocalSetting` 1 or 3:

The monitoring error, the external error and the interlocks cannot be reset when the control and feedback signals do not match.

- When the control and feedback signals match, the monitoring error, external error and the interlocks are reset by stopping (`StopLocal = 1`) the drive.
- With `Vlv2WayL` in the `MonSafePos = 1` setting, a monitoring error is reset by `Pos0Local = 1`.
- With `VlvL`, `VlvMotL` in the `MonSafePos = 1` setting, a monitoring error is reset with the local command, which moves the valve to the neutral position.
- With `Vlv2WayL`, `VlvMotL` and `VlvL` in the `MonSafePos = 0` setting, no reset of the monitoring error is required. The currently pending control is in effect.
- With `Vlv2WayL`, `VlvL` and `VlvMotL`, an external error is reset with the local command, which moves the valve to the neutral position
- With `DoseL`, you must acknowledge the protection (`Protect`) and flow alarms with a positive edge at the "`CancelLocal`" or "`PauseLocal`" output parameter.

The following applies with `LocalSetting` 2, 4 or 5:

No reset required.

Resetting motor protection (`Trip`) in local mode

In local mode, the "Motor protection" display is reset in the faceplate and not using the Reset button available there. The display disappears as soon as `Trip = 1`, the activation signals and feedback match and a command for stopping the drive has been issued.

Note

A motor protection signal (Trip parameter) with signal status 16#00 or 16#28 is used to activate motor protection. This is indicated by "Motor protection" in the standard view of the faceplates.

Resetting monitoring errors, external errors and interlocks using the "Forcing operating states" function

With "Forcing operating states", monitoring errors, external errors, interlocks or the motor protection function are reset under the following conditions and a reset pulse is output at the `P_Rst` output:

- The block is in an operating mode in which a reset is necessary and
- a monitoring error, an external error, a "Protection" interlock or the motor protection function is ready to be reset. This can be seen in the faceplate with the reset button or with the Request 0/1 indicator in the faceplate. When `Feature` Bit 19 = 1, the block is ready to reset as soon as the protection `(Protect = 0)` or motor protection `(Trip = 0)` interlock is set, whereby enabled motor protection prevents the motor from starting.

See also the following chapter: Forcing operating modes (Page 41).

Tabular overview for resetting for interlocks and errors

	Permit	**Interlock**	**Protect**
Meaning	Activation enable ("Permission")	Interlock without reset ("Interlock")	Interlock with reset ("Protection")
Description	The activation enable (input `Permit` = 1) makes it possible to leave the neutral position of the block in response to operator input or a command from the program (CFC/SFC). The activation enable has no effect if the block is not in the neutral position.	A pending interlock condition brings the block to the neutral position (input `Intlock` = 0).	A pending interlock condition brings the block to the neutral position (input `Protect` = 0).
Mode: **Automatic**	Takes effect if block is in the neutral position. After the interlock condition has gone, the currently pending control function becomes active again.	After the interlock condition has gone, the currently pending control function becomes active again.	**Feature Bit 9 and 30 = 0:** Reset via faceplate or `RstLi` = 1 **Feature Bit 9 = 1 and 30 = 0:** Reset via faceplate or `RstLi` = 1 or a 0-1 edge transition in the control **Feature Bit 9 = 0 and 30 = 1:** Reset via `RstLi` = 1 **Feature Bit 9 and 30 = 1:** Reset via `RstLi = 1` or a 0-1 edge transition in the control = 1 or 0-1 edge transition in the control

	Permit	**Interlock**	**Protect**
Mode: **Local**	Takes effect if block is in the neutral position. After the interlock condition has gone, the currently pending control function becomes active again.	After the interlock condition has gone, the currently pending control function becomes active again.	The following applies with `LocalSetting` = 1 or 3: **Generally:** When the control and feedback signals match, reset via `StopLocal` = 1. **Vlv2WayL, VlvMotL und VlvL:** Reset via local command, which moves the valve into the neutral position. **DoseL:** Reset via a positive edge at "`CancelLocal`" or "`PauseLocal`". The following applies with `LocalSetting` = 2,4 or 5: No reset required
Mode: **Manual**	Takes effect if block is in the neutral position. It is possible to leave the neutral position with an operation in the faceplate.	The faceplate can be operated again after the interlock condition has gone.	**Feature Bit 30 and 31 = 0:** Resetting not necessary **Feature Bit 30 = 1 and 31 = 0:** Resetting not necessary **Feature Bit 30 = 0 and 31 = 1:** Reset via faceplate or `RstLi` = 1 **Feature Bit 30 and 31 = 1:** Reset via faceplate

	Trip	**Error**	**Rapid stop**
Meaning	Motor protection	Monitoring errors and external errors	Rapid stop
Description	The motor protection function is used to switch off the motor when there is a heat overload (input `Trip` = 0).	• Monitoring the startup and stop characteristics for motors or the runtime of valves • Monitoring the operation of motors or the maintenance of the position of valves • External error FaultExt: Block goes to error state without a message being output. • External control system fault CSF with set Feature Bit 18: block reports an external control system fault and goes to error state.	A rapid stop stops the drive immediately.

	Trip	Error	Rapid stop
Mode: **Automatic**	**Feature Bit 9 and 30 = 0:** Reset via faceplate or RstLi = 1 **Feature Bit 9 = 1 and 30 = 0:** Reset via faceplate or RstLi = 1 or a 0-1 edge transition in the control **Feature Bit 9 = 0 and 30 = 1:** Reset via RstLi = 1 **Feature Bit 9 and 30 = 1:** Reset via RstLi = 1 or a 0-1 edge transition in the control = 1 or 0-1 edge transition in the control		**Feature Bit 9 = 0:** Reset via faceplate or RstLi = 1 **Feature Bit 9 = 1:** Reset via faceplate or RstLi = 1 or a 0-1 edge transition in the control
Mode: **Local**	The following applies with LocalSetting = 1 or 3: When the control and feedback signals of the drive match, reset via StopLocal = 1. The following applies with LocalSetting = 2, 4 or 5: No reset required.	The following applies with LocalSetting = 1 or 3: • When the control and feedback signals of the drive match, reset via StopLocal = 1. • With Vlv2WayL, VlvMotL and VlvL – **Monitoring error with MonSafePos = 1:** Reset via the local command, which moves the valve into the neutral position. – **Monitoring error with MonSafePos = 0:** No resetting required; the currently pending control function is active. – **External error:** Reset via the local command, which moves the valve into the neutral position. • With DoseL, resetting via a positive edge at "CancelLocal" or "PauseLocal". The following applies with LocalSetting = 2, 4 or 5: No reset required.	The rapid stop function is unlocked in the faceplate via the "Reset" button (RstOp = 1). In CFC, unlocking is carried out using the input parameter RstLi = 1
Mode: **Manual**	**Feature Bit 30 and 31 = 0:** Resetting not necessary **Feature Bit 30 = 1 and 31 = 0:** Resetting not necessary **Feature Bit 30 = 0 and 31 = 1** Reset via faceplate or RstLi = 1 **Feature Bit 30 and 31 = 1:** Reset via faceplate		The rapid stop function is unlocked in the faceplate via the "Reset" button (RstOp = 1). In CFC, unlocking is carried out using the input parameter RstLi = 1

2.1.1.4 Neutral position for motors, valves and controllers

Neutral position for motors, valves and controllers

The neutral position always represents the deenergized state.

Neutral position for motors

The neutral position for motors is always the stopped motor.

The neutral position is adopted when:

- The function Monitoring the feedbacks (Page 97) was triggered.
- One of the interlock conditions is active (see Interlocks (Page 99)).
- An external error via `FaultExt` or `CSF` was triggered (see Error handling (Page 119)).
- The Motor protection function (Page 99) was triggered.
- The function Rapid stop for motors (Page 106) was triggered.
- During start-up (see Set startup characteristics (Page 137)).
- The "Out of service" mode is active.
- One of the automatic commands has bad signal status (16#00 or 16#28) and the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1.

Neutral position for valves

There are different forms of the deenergized state for valves:

- Valve is closed in a de-energized state
- Valve is open in a de-energized state
- Valve is stopped in a de-energized state (e.g. motor valve)

The input parameter `SafePos` is used to set these properties of the valve:

- `SafePos = 0`: Valve is closed in a de-energized state
- `SafePos = 1`: Valve is open in a de-energized state
- `SafePos = 2`: Valve is stopped in a de-energized state (e.g. motor valve)

The neutral position is adopted when:

- The function Monitoring the feedbacks (Page 97) was triggered.
- One of the interlock conditions is active (see Interlocks (Page 99)).
- An external error via `FaultExt` or `CSF` was triggered (see Error handling (Page 119)).
- During start-up (see Set startup characteristics (Page 137)).

- The "Out of service" mode is active.
- One of the automatic commands has bad signal status (16#00 or 16#28) and the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1.

Neutral position for the VlvAnL and VlvPosL blocks (actuator)

The possible neutral positions are set by the `SafePos` parameter:

- `SafePos = 0`: Neutral position of the control valve is "Closed" (`MV.Value = MV_LoLim`)
- `SafePos = 1`: Neutral position of the control valve is "Open" (`MV.Value = MV_HiLim`)
- `SafePos = 2`: Neutral position of the control valve is "Stop" (`MV.Value` remains unchanged)

The control valve VlvAnL is brought to the neutral position when the `FbkAuxVCloseOut = 1` auxiliary valve is closed.

The neutral position of the control valve VlvAnL and VlvPosL is adopted when:

- The function Monitoring the feedbacks (Page 97) was triggered.
- One of the interlock conditions is active (see Interlocks (Page 99)).
- An external error via `FaultExt` or `CSF` was triggered (see Error handling (Page 119)).
- During start-up (see Set startup characteristics (Page 137)).
- The "Out of service" mode is active.
- One of the automatic commands has bad signal status (16#00 or 16#28) and the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1.

In case of startup or “Out of service” mode, the valve VlvAnL has additional `Feature` bits for adopting the neutral position:

- During start-up, if the `Feature` bit Set startup characteristics (Page 137) and the `Feature` bit Neutral position manipulated variable takes effect at startup (Page 165) are set.
- In the "Out of service" mode, if the `Feature` bit Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165) is set.

Neutral position for continuous controllers (does not apply to controller modules)

- Only the limits or a special analog value for the manual value are taken into consideration for the neutral position with continuous controllers.
  The input parameter `SafePos` is used to specify the neutral position:
  - `SafePos = 0` corresponds to the low limit (`ManLoLim` or `MV_LoLim` for PIDConS)
  - `SafePos = 1` corresponds to the high limit (`ManHiLim` or `MV_HiLim` for PIDConS)
- The input parameters `SafePos` and `SafePos2` are used to determine the neutral position:

SafePos	SafePos2	Neutral position	Remarks
=0	=0	ManLoLim for PIDConL, PIDConR or MV_LoLim for PIDConS	Low limit for manipulated variable in manual mode
=1	=0	ManHiLim for PIDConL, PIDConR or MV_HiLim for PIDConS	High limit for manipulated variable in manual mode
x	=1	MV_SafePos	Manipulated variable neutral position
x	=2	Last manipulated variable (stop)	Last manipulated variable (stop)

x: Not relevant

The neutral position is adopted:

- The interlock conditions is active (see Interlocks (Page 99) only at PIDConL and PIDConR).

In case of startup or "Out of service" mode, the controller has additional Feature bits for adopting the neutral position:

- During start-up, if the Feature bit Set startup characteristics (Page 137) and the Feature bit Neutral position manipulated variable takes effect at startup (Page 165) are set.
- In the "Out of service" mode, if the Feature bit Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165) is set.

Neutral position for step controllers (does not apply to controller modules)

You can use the input parameter SafePos to determine if the step controller should close, open or stop the valve when it enters the neutral position:

SafePos = 0: close valve

SafePos = 1: open valve

SafePos = 2: stop valve

When the neutral position (fully opened or fully closed) is reached and a limit stop signal (FbkOpened or FbkClosed) is set, the valve is stopped (Stop = 1).

The neutral position is adopted:

- during start-up if the Feature bit Set startup characteristics (Page 137) and the Feature bit Neutral position manipulated variable takes effect at startup (Page 165) are set.
- in the "Out of service" mode if the Feature bit Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165) is set.

Safety control for controllers of the FM 355 or FM 355-2 modules

The controller modules have their own mechanism for feedforwarding a safety value (see Temperature Controller FM 355-2 manual or Controller Module FM 355 manual)

2.1.1.5 Specifying warning times for control functions at motors and valves

Specifying warning times for control functions at motors and valves

This function is only supported by "Large" blocks.

You can generate warning signals when, for example, motors are started or valves are opened. Warning signals can be generated in the following modes:

- Manual mode (input parameter `WarnTiMan`)
- Automatic mode (input parameter `WarnTiAut`)

You specify the warning times in seconds using the input parameters `WarnTiMan` and `WarnTiAut`. If, for example, a motor is started then, this is displayed at the output parameter with `WarnAct = 1`. The motor then starts after the configured warning time has expired and `WarnAct` is reset (`WarnAct = 0`).

A corresponding warning is not output if the warning times (`WarnTiMan` or `WarnTiAut`) are specified with a smaller value than the `SampleTime` parameter.

Note

In this case, the warning time is only active if the block is controlled from the de-energized state.

Disabling warnings

Configure each parameter with 0 seconds to generate no warnings.

2.1.1.6 Output signal as a static signal or pulse signal

Output signal as a static signal or pulse signal

You can output the control signals for motors, valves and dosers as:

- Static signal or as a
- Pulse signal with configurable pulse length.

You can find the signals in the I/O table of the individual blocks.

Note

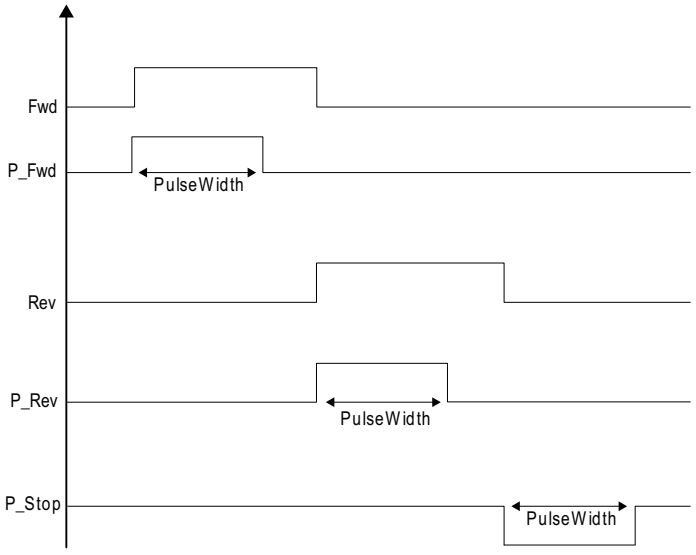
The pulse signal is available only for the "large" blocks.

Output signal as a static signal

The control settings are made available as a static signal in the blocks in the form of interconnectable output parameters. The MotRevL block, for example, provides these as static signals via the alternative output parameters `Fwd`, `Rev` and `Run`.

Output signal as a pulse signal

The control is made available as pulse signals at the blocks using interconnectable output parameters. You specify the pulse length of the output signals in seconds using the input parameter `PulseWidth`. The MotRevL block, for example, provides these as pulse signals via the output parameters `P_Fwd`, `P_Rev` and `P_Stop`.



Note

Almost all output parameters for pulse control, for example `P_Fwd`, `P_Rev`, `P_Ctrl`, have a positive effective direction, i.e. a 0→1→0 pulse triggers activation.

The only exception is the `P_Stop` output parameter with a negative effective direction, i.e. a 1→0→1 pulse triggers activation.

2.1.1.7 Recording the first signal for interlock blocks

Recording the first signal

You activate the function described below using the `Feature` bit " Activating recording of the first signal (Page 149)".

The number of the input that caused the last output signal change from 1 to 0 (good state to locked) is displayed for you in bit coding at the `FirstIn` output. The cause may be:

- A signal change at the input or a change in inversion
  Example: With an OR logic operation, the single 1 changes to 0. The output then changes from 1 to 0.
- A change to the I/O
  Example: Excluding the single 1 then results in an output of 0 with an OR logic operation.

- A change to the signal status
  If the signal status of the input, which forms the output value alone and has the value 1, changes from 16#80 to 16#00, the output value changes from 1 to 0.
- `FirstIn` is not changed if the following events occur, despite a change to the output:
  – Change in output value from or to `DefaultOut`

If several signals are at the same time responsible for the change, all responsible inputs are indicated in the faceplate and output in bit coding in the `FirstIn` output. If the input signals change without this causing the signal at the output to change, `FirstIn` does not change.

Inputs which are not interconnected or which are excluded are not taken into account.

You can reset `FirstIn` to 0 if you set the `RstLi` input from 0 to 1 (positive edge) or you can operate the `RstOp` input using the faceplate ("Reset" button).

If at least one bit in `FirstIn` is set, other signal changes are not taken into account.

2.1.1.8 Outputting a signal for start readiness

Outputting a signal for start readiness

The `RdyToStart = 1` output parameter is used to indicate start readiness in automatic mode.

Start readiness is output when the following conditions are met:

- No group error pending (group fault with consideration of feature bit 18 for `CSF` and `MonSafePos` for feedback errors)
- No interlock is active
- No forced operating mode or manipulated variable is active
- No rapid stop is active (only applies to motors or `VlvMotL`)
- The block is in automatic mode (not with controller blocks)
- The waiting time for the restart must have expired (only applies to motors or `VlvMotL`)

The start readiness is shown at the following block groups via the `RdyToStart` output parameter:

- Motors
- Valves
- Dosers
- Software controller
- Hardware controller

Start readiness for motors

The start readiness for motor blocks is formed as follows:

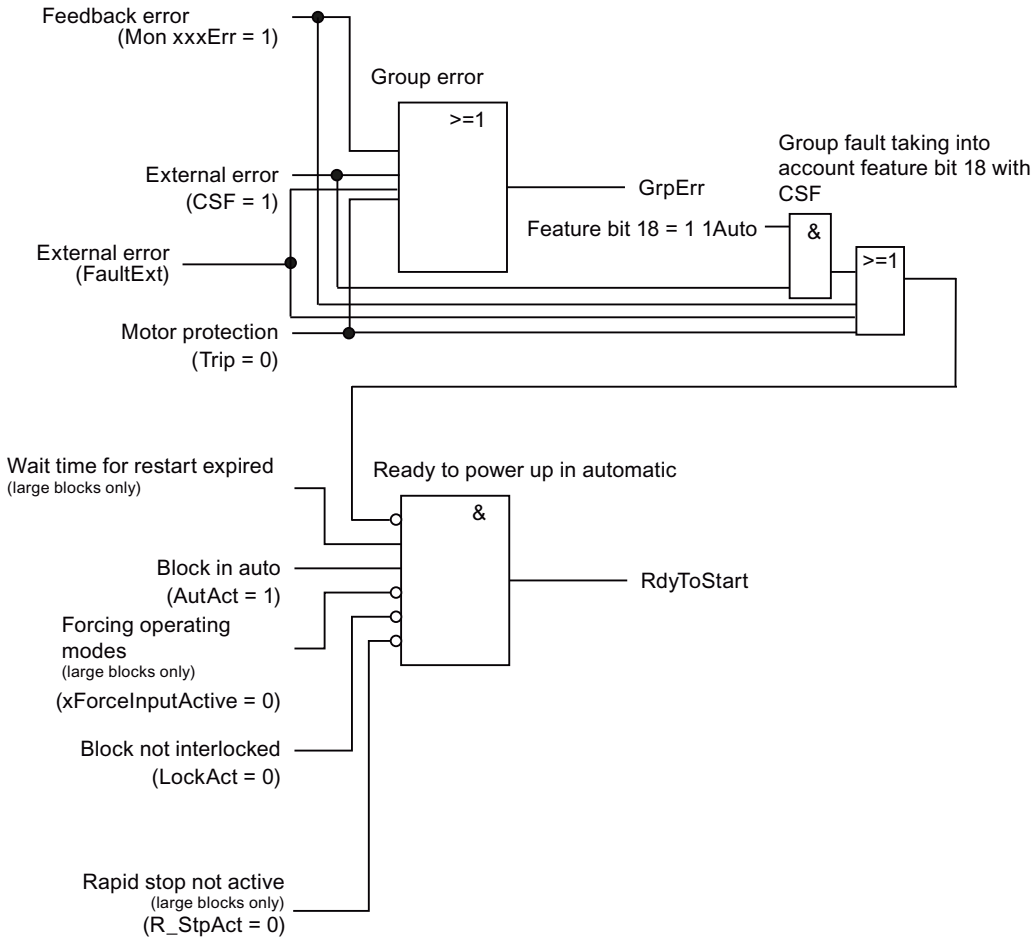


Figure 2-1 Output signal for start readiness for motors

Start readiness for valves

The start readiness for valve blocks is formed as follows:

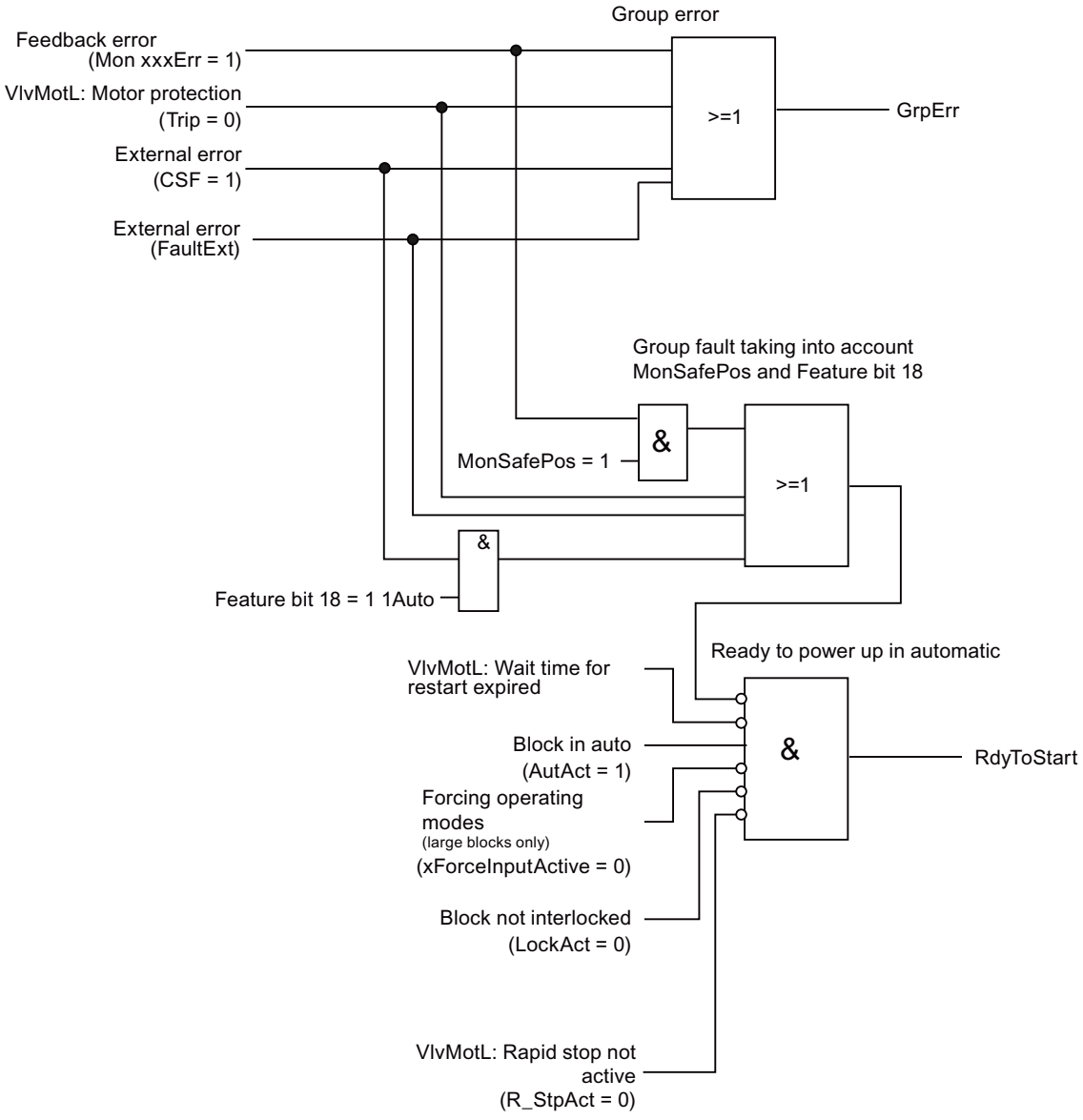


Figure 2-2 Output signal for start readiness for valves

Start readiness for dosers

The start readiness for dosers is formed as follows:

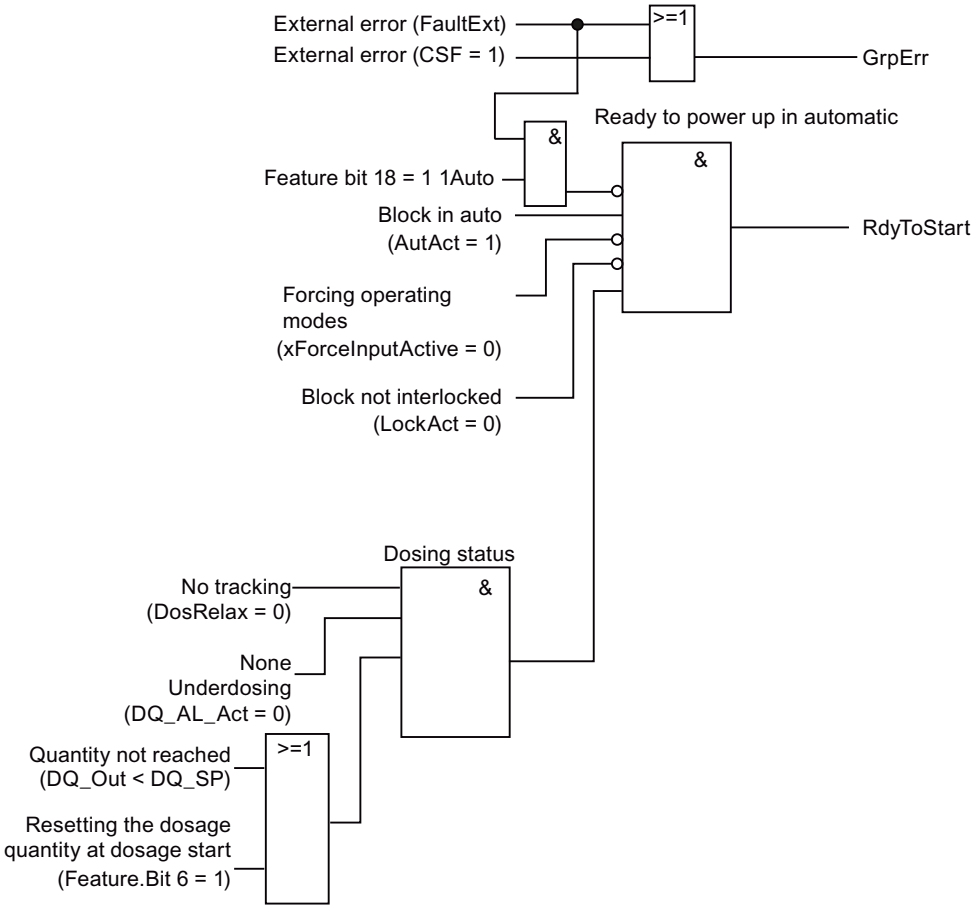


Figure 2-3 Output signal for start readiness for dosers

Start readiness for software controllers

The start readiness for software controllers is formed as follows:

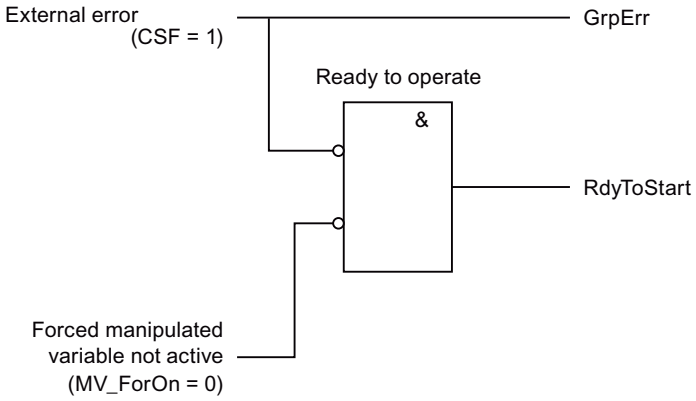


Figure 2-4 Output signal for start readiness for software controllers

Start readiness for hardware controllers

The start readiness for hardware controllers is formed as follows:

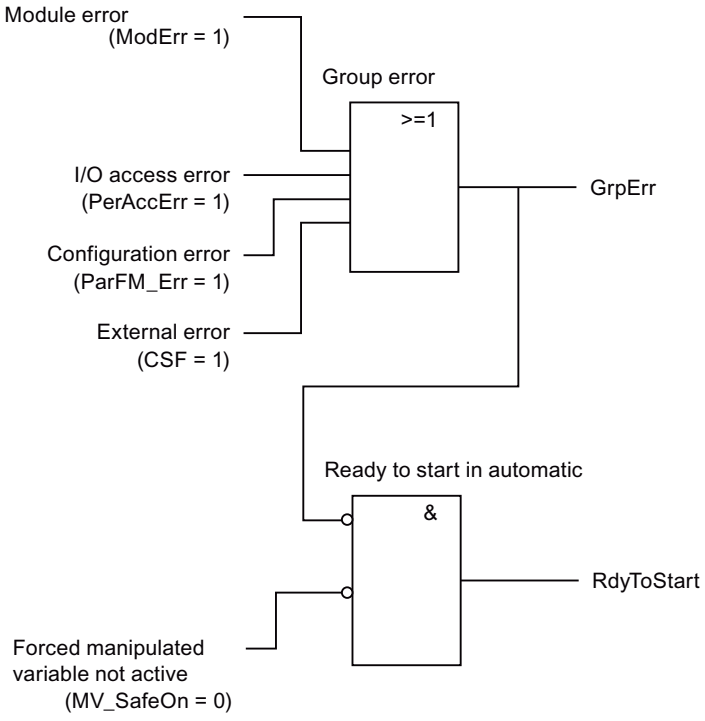


Figure 2-5 Output signal for start readiness for hardware controllers

2.1.1.9 Simulating signals

Simulating signals

Simulation means the manipulation of a signal regardless of the actual source of the signal or logic that generates this signal.

Simulation is carried out either at the field device (externally from the control system) itself or at a block (internally in the control system).

In either case, the associated status in the signal is set to the simulation value (see also Forming and outputting signal status for blocks (Page 108)).

During the simulation, every block is considered in isolation. There are two different forms of simulation here, namely:

- Block-external simulation and
- Block-internal simulation.

Block-external simulation

Block-external simulation is characterized by the fact that:

- The simulation function is not executed in the block itself and
- A signal whose status has the simulation state, for example, a simulation of the signal at another block or directly in the I/O device, is applied at an input parameter.

The block-external simulation has the following effects on the functionality of the block:

- The technological functions are not influenced
- All the process-relevant output signals do not receive the simulation status. In the case of technologic blocks, process-related output signals are parameters that actively affect the process, e.g. "`Start`" for block MotL.
- In the case of blocks with operator control or monitoring functions (for example faceplates), these signals are identified in the faceplate with the status for the simulation as follows:

- Blocks with one or more input parameters for signals with "Generate status from individual status" receive a group status in accordance with the priority table. This group status is displayed in the status bar of the block icon and of the faceplate with the simulation status as follows:

- The interlocking functions of the block are not influenced.

Note

For the output channel blocks, you need to specify the exact block response with external simulation using the `Feature` bit Outputting a de-energized value for block-external simulation (Page 147).

Block-internal simulation

Block-internal simulation is characterized by the "simulation" function being run in the block itself.

With operator control and monitoring blocks, all process values that cannot be controlled (e.g. `PV`, `AV`, `In`) can be simulated. This is used primarily as an aid for commissioning and servicing of the system. For example, the control settings of a motor can be simulated and the feedback values corrected without the monitoring functions being active.

For blocks that can be operated and monitored, simulation can take place via the faceplate as well as interconnectable inputs:

- `SimLiOp` = 0: The simulation is activated/deactivated via faceplate (parameter view) at the input `SimOn`.

- `SimLiOp` = 1: The simulation is activated/deactivated via the input `SimOnLi`. The interconnectable simulation values (e.g. `SimPVLi`, `SimAVLi`, `SimInLi`) will become effective in the process. In this case, the input `SimOn` is written back with `SimOnLi`.

The `Feature` bit Activating the run time of feedback signals (Page 152) can be used to delay tracking of the feedback signals for motors and valves (for example, `Fbkxxx`).

Simulation can also be carried out for blocks (such as channel blocks) that cannot be controlled and monitored by the operator.

The control is simulated in the CFC by setting parameters directly in the block with the input parameters `SimOn = 1` and `Simxxxx =` for the desired simulation value (e.g. `SimPV` , `SimAV` or `SimIn`).

Note

With channel blocks, ensure that the `Mode` parameter is set correctly during simulation. Otherwise this is displayed on the `Bad = 1` output parameter with a higher-level error.

If the block is not in simulation, the simulation value (`SimPV`, `SimAV` or `SimRbk`) process value (`PV`, `AV` or `Rbk`) is tracked.

Simulation is triggered during runtime in the faceplate's parameter view by clicking on the "Simulation" button.

This simulation is characterized by the fact that:

- The simulation can only be enabled / disabled with the operator authorization level for system authorization.
- The technological functions are not influenced.
- All the process-related output signals receive the "simulation" status. In the case of technologic blocks, process-related output signals are parameters that actively affect the process, e.g. "`Start`" for block MotL.
- In the case of blocks with operator control or monitoring functions (for example faceplates), these signals are identified in the faceplate with the status for the simulation as follows:

- The group status of the block is displayed in the status bar of the block icon and of the faceplate with the simulation status as follows:

- All the process values displayed in the faceplate that cannot be operated-controlled in normal operation (e.g. `PV`).
- When the block control can be manipulated, the readback and feedback values (for example `Rbk`, `FbkSpd1`) are adjusted according to the manipulation of the control.
- Associated values (for example `UserAna1`) cannot be simulated.
- The interlocking functions of the block are activated in accordance with input parameter `BypProt = 0` or deactivated (`BypProt = 1`). This is shown as follows in the faceplate and block icon:

Enabled	
Disabled	

Block-internal simulation for controllers

How block-internal simulation for controllers works (`SimOn = 1`):

- In manual mode, both the simulated process value `SimPV` and the simulated repeated manipulated variable `SimRbk` can be entered in the faceplate as a simulated value.
- When a switchover to automatic mode is performed, the simulated process value `SimPV` is set so that it is equal to the setpoint `SP` (= tracking). This means the control deviation is no longer present and the pending manipulated variable (from the bumpless manual-automatic switchover, for example) remains constant. `SimRbk` can still be controlled.

Note

If you switch a controller block to block-internal simulation during automatic mode and the controller is connected to the actual process on the actuator side, you will open the control loop as a result.

The actuating signals calculated on the basis of the simulated actual value are switched to the process, but the resulting motion in the process is no longer visible in terms of the controller actual value, as a copy of the setpoint is present at this point instead, where it takes the form of a simulated actual value. The process could move away from the setpoint without the controller doing anything to counteract this and without you seeing this happen in the controller faceplate.

Manipulated variable step changes occur during switchover to automatic mode if an error signal was already present before the switchover.

The following applies to program mode:

- Program mode with setpoint specification should be considered the same as automatic mode from a control engineering point of view. Block-internal simulation reacts in the same way as it does in automatic mode: The process value `PV` is set so that it is equal to the setpoint `SP`, which in this case is derived from the input parameter `AdvCoMV`.
- Program mode with manipulated variable specification should be considered the same as manual mode from a control engineering point of view. Block-internal simulation reacts in the same way as it does in manual mode: The simulated process value `SimPV` can be entered as a simulated value. In this case the manipulated variable `MV` is derived from the input parameter `AdvCoMV`.

2.1.1.10 Dead band

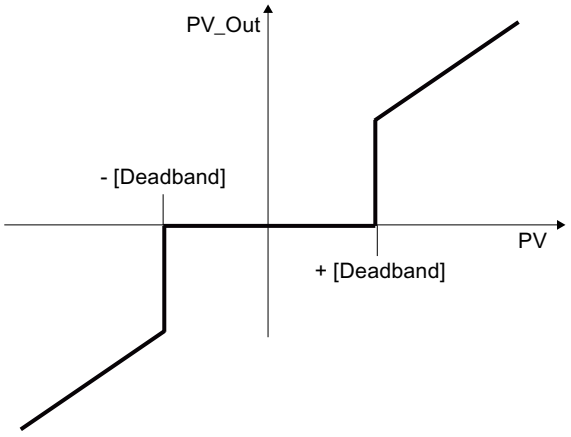
Dead band

To suppress values fluctuating around zero, you can set a dead band (`Deadband`):

`Deadband = 0`: Dead band is disabled

`Deadband ≠ 0`: Dead band is enabled

With a negative dead band (`Deadband < 0.0`), calculation is continued internally with this value

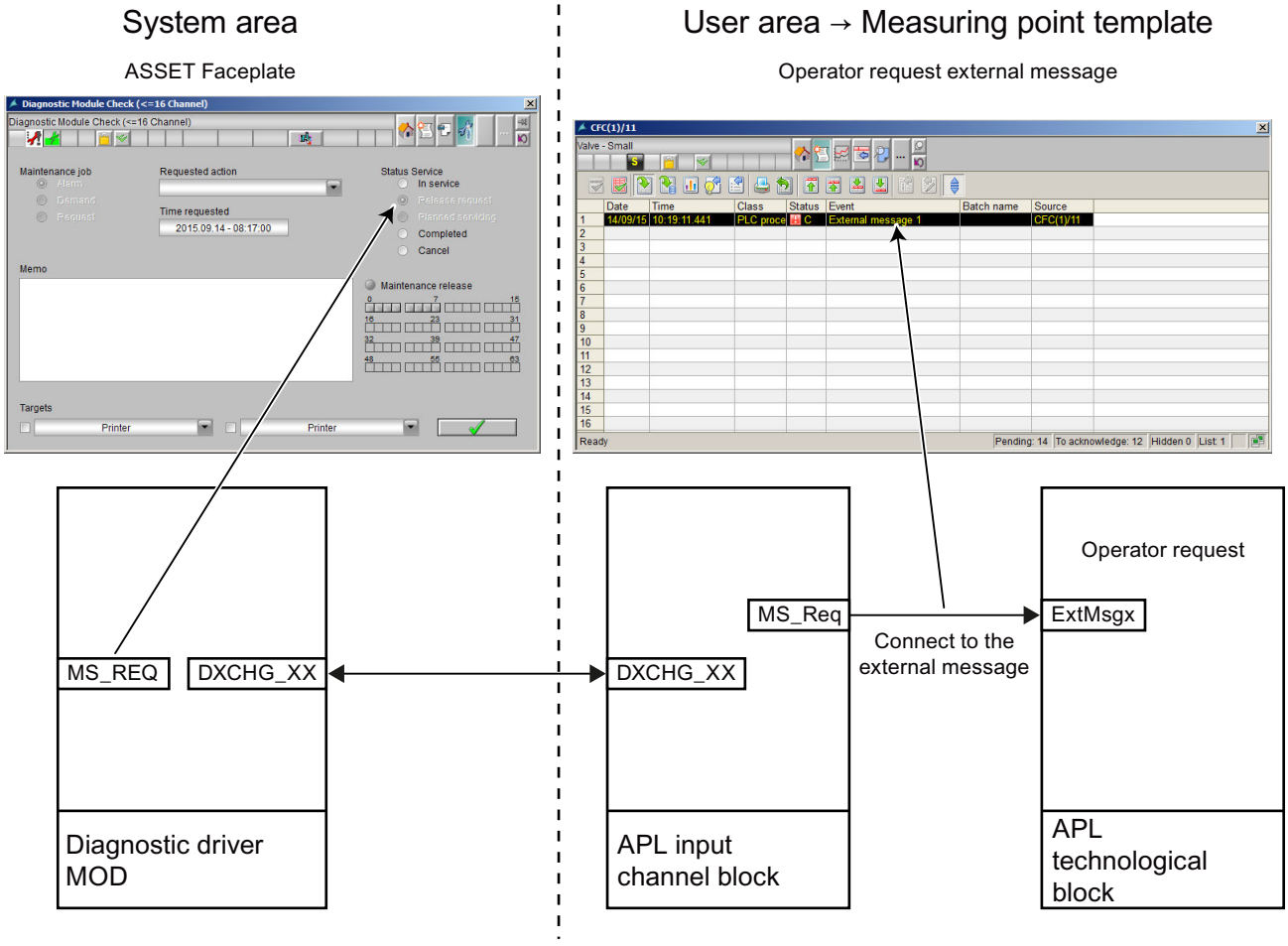


2.1.1.11 Request for maintenance release

Issuing a request for maintenance (MS → APL)

An operator can issue a request for maintenance release from the Maintenance view of the ASSET faceplate ("Status service" → "Release request"). After issuing a maintenance request, the operator can set the maintenance release. This will display the message which the operator has configured. Later, operator can set the Maintenance Release in the technological Block (Page 64). It indicates that the maintenance operator can start the maintenance.

Functional sequence in APL



- The output parameter `MS_Req` of the channel block will be set to 1 when an operator issues a maintenance request in Maintenance view of the ASSET faceplate.
- The output parameter `MS_Req` has to be connected to the input parameter `ExtMsgx` of the desired technological block for which the user wants to issue an operator request message.
- The operator has to configure the "Message class" and enter a suitable text in the "Event" field for the operator request message in the "PCS7 Message Configuration" window of the technological block.
- With the configured message and issued maintenance release, the maintenance operator can start the maintenance.
- The operator request message stays active until the user changes to "In service" or "Cancel" mode in the Maintenance view of ASSET faceplate.
- Once the maintenance is done, the operator can change the mode to "Completed". This will deactivate the "In service" mode (`OosAct` = 0).

2.1.1.12 Release for maintenance

Issuing a release for maintenance

The release for maintenance serves as information about a process tag at which maintenance, service or calibration should be carried out. You can use the signal for release for maintenance to transfer the information about the enabling of a process tag from the OS to a Maintenance Station.

Note

The block must be in either "Manual," "On" or "Out of service" mode to set the release for maintenance.

You set the release for maintenance (operator control permission "System control" required) in the parameter view using the input parameter `MS_RelOp = 1`. A release for maintenance is then made available via the interconnectable output parameter `MS_Release = 1` for further processing. In order to make this information of the Maintenance Station available, you have to interconnect the output parameter `MS_Release` of the technologic block with the input parameter `MS_Release`of the corresponding channel block.

The issuing of a release for maintenance does not have any influence on the function of the block. An operation message is generated.

Use of the state "In progress" on the Maintenance Station

The status "In progress" is implemented on the Maintenance Station for a process tag or a field device using the channel blocks and the interconnectable output parameter `OosAct = 1`. You can interconnect the output parameter `OosAct` of the channel block with the input parameter `OosLi` of a technologic block.

Use the `Feature` bit Reaction to the out of service mode (Page 176) to specify, in case the input parameter is `OosLi = 1`, if:

- there is a switchover to the "out of service" mode and the symbol for the "In progress" (see table) status is displayed. You can change to manual mode at any time.
- Only the "In progress" display (see table) in the block icon and in the faceplate of the assigned technologic block is made.

Display	Meaning
	In progress

Function sequence in the APL

- The OS operator issues the release for maintenance (`MS_RelOp = 1`) in the technologic block's parameter view.
- The technologic block then sets the `MS_Release = 1` output parameter.
- The channel block's `MS_Rel` input is now also 1.

- The channel block signals the release for maintenance to the diagnostics block via the `DXCHG` parameter.
- The release for maintenance is only signaled to the Maintenance Station once all 0 bits of the parameter `DXCHG_XX` are set on the diagnostics block.

- The channel block determines the "in progress" state of the Maintenance Station using the `MS` input parameter and makes this information available at the `OosAct` output parameter.
- On the technologic block, the "working" state is displayed at input parameter `OosLi` and forwarded for display to the faceplate.

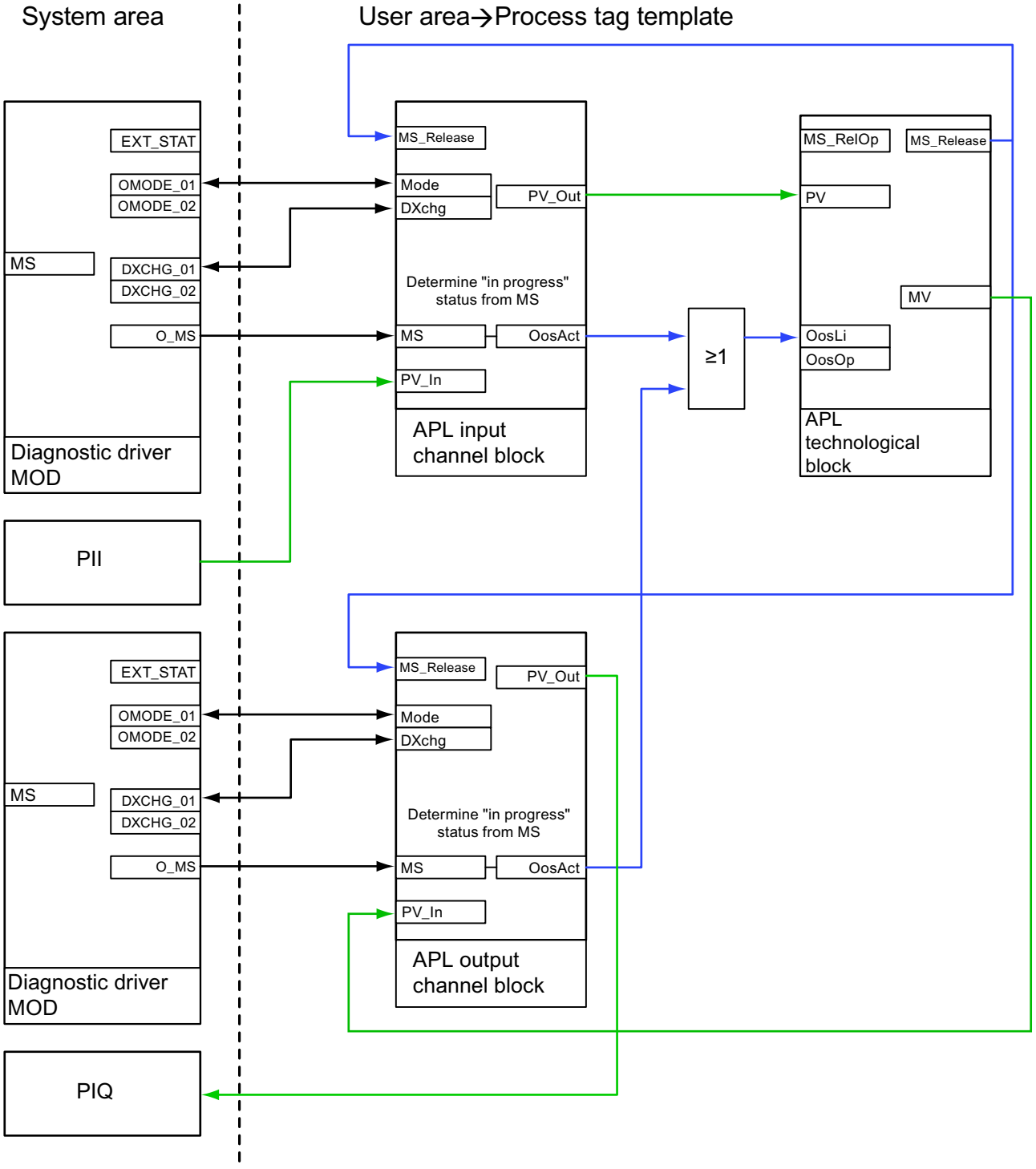


Figure 2-6 Release for maintenance

Key to diagram:

PII	Process image of inputs
PIQ	Process image of outputs
Black lines	Automatic system connections
Green lines	Process value connections made by the planner
Blue lines	Connections for release for maintenance made by the planner

Note

For additional information on the topic of maintenance please refer to PCS 7 OS process management.

2.1.1.13 SIMATIC BATCH functionality

SIMATIC BATCH functionality

Some blocks have an interface to SIMATIC BATCH. You use them when you connect `BatchEn`, `BatchID`, `BatchName`, `StepNo` and `Occupied` I/Os to the corresponding SIMATIC BATCH blocks. A new `BatchName` becomes effective if the `BatchID` is changed. Refer to the SIMATIC BATCH documentation.

Please refer to the descriptions of the individual blocks for information about whether a block supports the SIMATIC BATCH functionality.

2.1.1.14 Flutter suppression for channel blocks

Flutter suppression

The time-controlled "Flutter suppression" function is used to delay the outgoing of a message by a configurable period.

Flutter suppression is used for

- OB82 events - diagnostic messages
- OB83 events - fault

used.

The flutter time is entered at the channel block at the `FlutTmIn` parameter. The high byte of the `DataXchg` parameter of the channel blocks contains the flutter time.

Flutter suppression comes into effect when `FlutEN` = 1 and `FlutTmIn` > 0 is set at the channel block.

There is only one flutter message per module. The delay times and fault messages are channel-specific. The fault messages are extended by at least the delay time. Flutter occurs when the status of fault messages changes from "Outgoing" back to "Incoming" within the delay time.

The last fluttering channel and its set delay time deactivates the flutter message.

The following channel blocks have this function:

FbAnIn - Analog input channel block for field devices (Page 2015)

FbAnTot - Analog totalizer channel block for field devices (Page 2106)

FbAnOu - Analog output channel block for field devices (Page 2024)

FbDiIn - Digital input channel block for field devices (Page 2034)

FbDiOu - Digital output channel block for field devices (Page 2043)

Pcs7AnIn - Analog input channel block (Page 2084)

Pcs7AnOu - Analog output channel block (Page 2096)

Pcs7DiIn - Digital input channel block (Page 2113)

Pcs7DiIT - Digital input channel block with time stamp (Page 2121)

Pcs7DiOu - Digital output channel block (Page 2130)

Pcs7Cnt1 - Controlling and reading FM 350 modules (Page 2138)

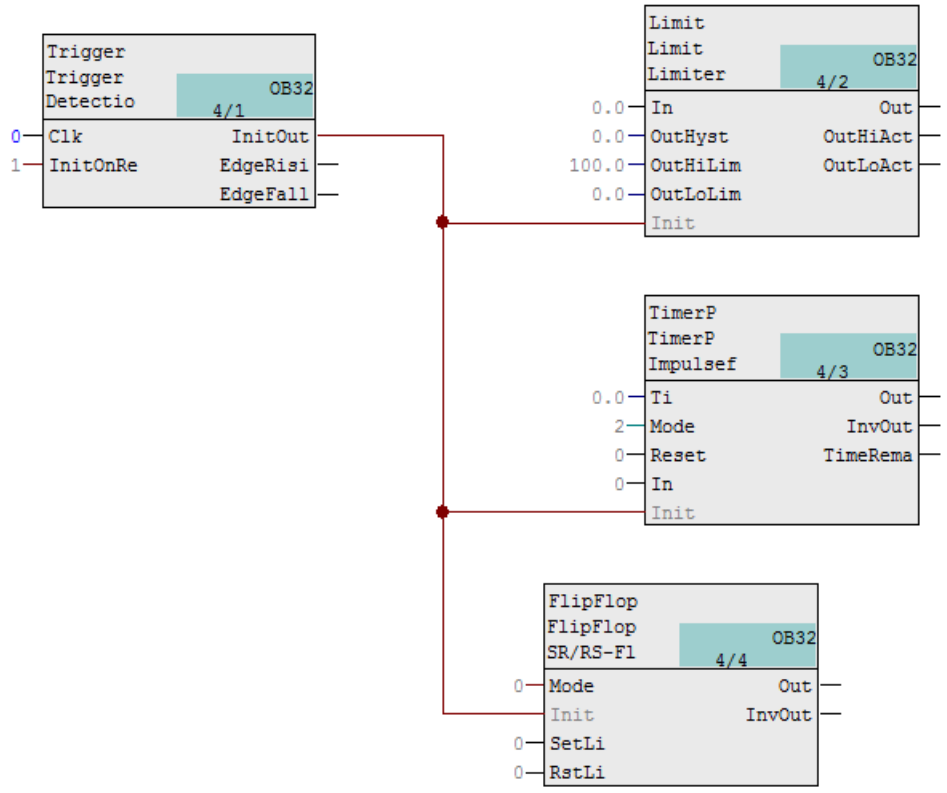
Pcs7Cnt2 - Control and read an 8-DI_NAMUR module of the ET 200iSP (Page 2148)

Pcs7Cnt3 - Control and read the 1 COUNT 24V/100kHz module for count mode (Page 2157)

2.1.1.15 Startup characteristics over Trigger block

Startup characteristics over Trigger block

With the function block Trigger, you can influence the startup characteristics of the blocks Limit, TimerP, FlipFlop, and Trigger. Connect the output parameter `InitOut` of the Trigger block with the input parameter `Init` of the block Limit, TimerP, or FlipFlop to initialize these blocks. In the second cycle of OB30 to OB38, the output parameter `InitOut` of the Trigger block will be reset after a startup.



2.1.2 Operating modes of the blocks

2.1.2.1 Overview of the modes

Overview of the individual modes

The available operating modes are assigned to the block families:

- Motors, valves and dosers
- Controllers
- Blocks without "Manual" and "Automatic" modes

You can find an overview below. Click on one of the operating modes to go directly to the relevant detailed description.

You can find a state graph for the operating modes (Page 83) at the end of this section.

Operating modes for motors, valves and dosers

The following operating modes are available:

1. Local mode (Page 79)
2. Automatic mode (Page 75)
3. Manual mode (Page 75)
4. Out of service (Page 71)

The mode with the lowest number in the list above has the highest priority. "Manual" and "Automatic" modes have the same priority. General information on the individual modes is available in the following sections. The sections also include block-specific information, for example, non-standard parameter assignment. Refer to the description and function of the relevant blocks.

Operating modes for controllers

The following operating modes are available:

1. Automatic mode (Page 72)
2. Manual mode (Page 72)
3. Program mode for controllers (Page 78)
4. Out of service (Page 71)

The mode with the lowest number has highest priority. "Manual" and "Automatic" modes have the same priority. General information on the individual modes is available in the following sections. The sections also include block-specific information, for example, non-standard parameter assignment. Refer to the description and function of the relevant blocks.

Operating modes for blocks without "Manual" and "Automatic" operation

The following operating modes are available:

1. On (Page 71)
2. Out of service (Page 71)

The mode with the lowest number has highest priority. General information on the individual modes is available in the following sections. The sections also include block-specific information, for example, non-standard parameter assignment. Refer to the description and function of the relevant blocks.

Note

Note that the operating modes are realized differently in the individual block families.

2.1.2.2 On

"On" operating mode

The "On" operating mode tells you that the block algorithm is being processed (output parameter `OnAct = 1`). This operating mode is only available for blocks that have faceplates but not the following operating modes:

- Manual mode or
- Automatic mode or
- Local mode

The "On" mode can only be activated via a control on the faceplate (input parameter `OnOp = 1`). The block must be in the "Out of service" operating mode for this to be possible.

2.1.2.3 Out of service

Using the "out of service" operating mode

The "Out of service" operating mode is available to all blocks that have an operating mode switchover and a direct connection to the process (with a connection to a process tag, for example).

The Out of service" operating mode is intended for purposes of maintenance and servicing (replacing the device, for example). All of the block's functions are disabled. No incoming or outgoing messages are generated. The only function still possible is an operating mode switchover.

All outputs for motors and valves are set to the neutral position in this operating mode.

The timer for the function "Restart disable after the shutoff of motor (Page 1392)" is reset in this mode.

Note

Notes on VlvMotL and VlvPosL

If the neutral position of VlvMotL is "Closed" or "Open", the corresponding control outputs "Closed" or "Open" are set. The control outputs are reset after reaching the neutral position or with an active torque signal. The "Seal valve" function (`Feature` bit 8) is a component.

Note

Note on VlvMotL

If the timer of the "Restart disable after the shutoff of motor (Page 1392)" function is running and the block is not in the neutral position, it is not possible to switch to "Out of service" mode.

For controllers, the neutral position manipulated variable (high or low manual limit of the manipulated variable) is only used if the `Feature` bit Neutral position manipulated variable takes effect at startup (Page 165) is active. Otherwise the manipulated variable remains at the latest value like all the other output parameters.

See also the section Neutral position for motors, valves and controllers (Page 48) for more on this.

The last value available is output permanently for all other blocks.

Requirement for the "out of service" mode

Prerequisite for switching to this operating mode is that the block is in "Manual mode" or "On" mode.

Activating the "Out of service" operating mode using the faceplate

The "Out of service" operating mode can only be switched on by using the faceplate when it is in the default block view (`OosOp = 1` parameter) and even then, only if `ModLiOp = 0`.

To switch the operating mode using the faceplate, refer to the descriptions relating to the standard view of the individual blocks.

Switching on the Out of service operating mode by using the interconnection

The "Out of service" operating mode is switched on by using the configurable parameter `OosLi = 1`. This is only possible if the block is in manual mode or "On" mode and the `Feature` bit Reaction to the out of service mode (Page 176) was set to 1.

Regardless of operating mode, the parameter view of the faceplate will always display the status of the parameter `OosLi =1` with the symbol for the status "In progress" (see table) next to the Maintenance enable button.

Display	Meaning
	In progress

See also the section Release for maintenance (Page 64) for more on this.

Exiting the "Out of service" operating mode

From this operating mode, a block can only be switched by an operator action at the faceplate into the following operating modes:

- "On"
- "Manual mode"

2.1.2.4 Manual and automatic mode for control blocks

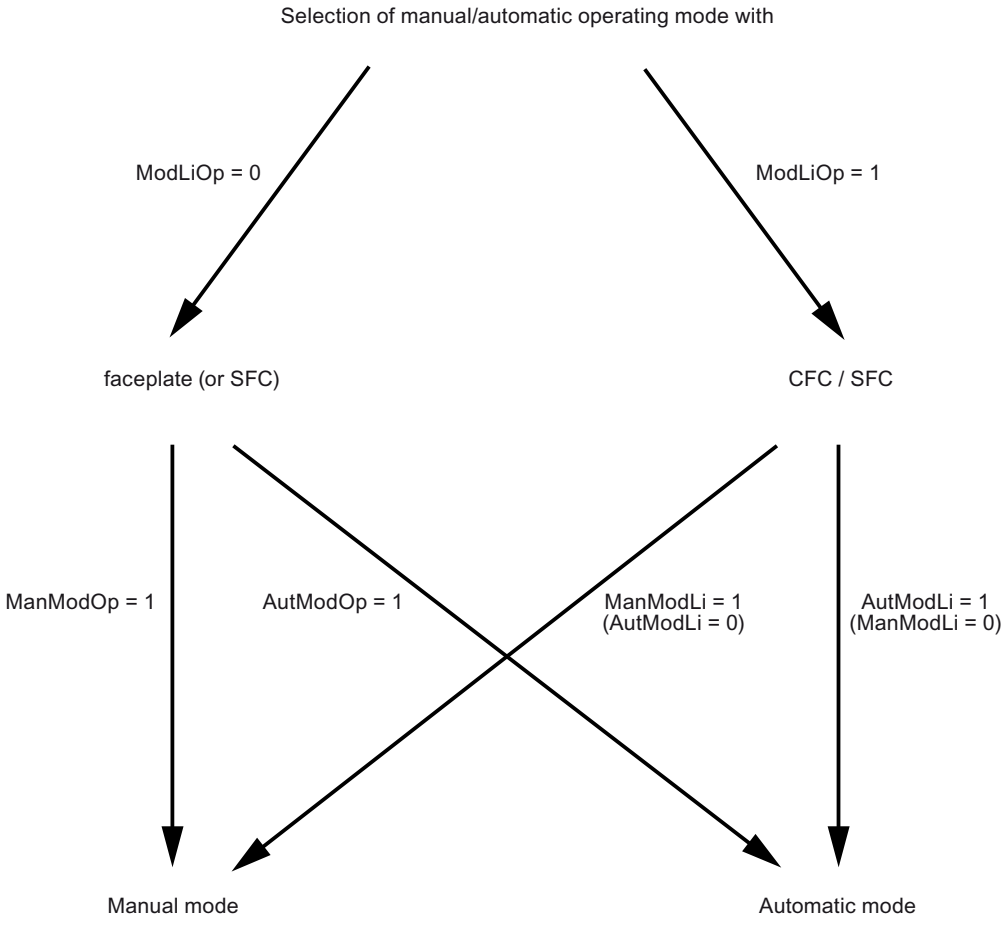
"Manual" and "Automatic" modes for controller blocks

In "manual mode", the control settings for the device are made manually by the operator. The operator decides how to change the block's manipulated variable (output signal). The manipulated variable can be analog or binary.

In "automatic mode", the control settings for the controller are made automatically as calculated by the block algorithm.

Changing between operating modes

The switchover between manual and automatic modes takes place as shown in the following schematic:



Switchover initiated in the faceplate (`ModLiOp` = 0)**:** The switchover between operating modes is carried out in the standard view of the faceplate. In the function block, the parameters `ManModOp` for "manual mode" and `AutModOp` for "automatic mode" are used.

If both signals (`ManModOp =1, AutModOp = 1`) are set, `ManModOp = 1` has priority.

Switchover per interconnection (CFC or SFC instance) (`ModLiOp = 1`)**:** The switchover between the operating modes is carried out with an interconnection on the function block. The parameters `ManModLi` for "manual mode" and `AutModLi` for "automatic mode" are used in pushbutton operation. In switching mode (requirement: `Feature Bit 4 = 1`, see Setting switch or button mode (Page 166)) connection `AutModLi` is used exclusively.

If both signals (`ManModLi =1, AutModLi = 1`) are set, `ManModLi = 1` has priority.

Note

You can access the variable parameters `AutModOp` and `ManModOp` from a normal SFC (in contrast to the instance of an SFC type). The SFC can thus change the operating mode without revoking the access rights of the operator (i.e. without setting `ModLiOp = 1`).

Switchover from automatic mode to manual mode

When changing over from "automatic mode" to "manual mode", the last valid control settings (**M**anipiulated **V**alue `MV`)for the controller set in "automatic mode" remain valid until you change the control settings manually.

Switchover from automatic mode to manual mode

The switchover from manual to automatic mode can take place with or without the internal setpoint tracking the process value. You specify this behavior on the `SP_TrkPV` I/O, which can also be operated from the faceplate in the parameter view (Option "SP = PV"). For the blocks PIDConL and PIDStepL you can also change the behavior for the switchover via the parameter `Feature` bit Disabling bumpless switchover to automatic mode for controllers (Page 172):

- **Switchover with internal setpoint tracking process variable**(`SP_TrkPV = 1`) means that in "Manual" mode the setpoint (`SP`) tracks the process value (`PV`) (bumpless switchover). After switching back to "Automatic" mode, the manipulated variable remains constant until the setpoint value (`SP`) is changed or the process value (`PV`) changes.
- **Switchover without internal setpoint tracking process variable** (`SP_TrkPV = 0`) means that the block immediately recalculates the value of the manipulated variable based on the setpoint and process value (`PV`) when the mode is changed. The `Feature` parameter is used to choose between the two variants:
  - **Switchover without P step** (standard setting, `Feature` bit = 0):
    During switchover, the I action of the controller is set in such a way that the switchover is carried out without a P step (virtually bumpless referring to the manipulated variable). A control deviation is only regulated via the I action.
  - **Switchover with P step** (`Feature` bit = 1):
    During switchover, the I action of the controller is set in such a way that the switchover is carried out with a P step (not bumpless referring to the manipulated variable). A control deviation is regulated via the P and the I action.

    Note

    Points to note about switchovers with a P step change:

    - The P action must be active for the setting "Switchover with P step" (`PropSel = 1`)
    - If the P action is in the feedback (`PropFacSP = 0`), the "Switchover with P step" setting has no effect.
    - If the switchover function with the internal setpoint tracking the process variable is active (`SP_TrkPV = 1`), the "Switchover with P step" setting has no effect.

Reaction of signals when operating mode is changed

Using the `Feature` bit Resetting the commands for changing the mode (Page 160), you can choose whether the block automatically resets the signal for changing the operating mode.

Switch on program mode

A few controller blocks allow you to operate in program mode. Refer to the relevant sections for the controller blocks to learn whether a control block allows program mode.

Also refer to the section Program mode for controllers (Page 78) for information on program mode.

2.1.2.5 Manual and automatic mode for motors, valves and dosers

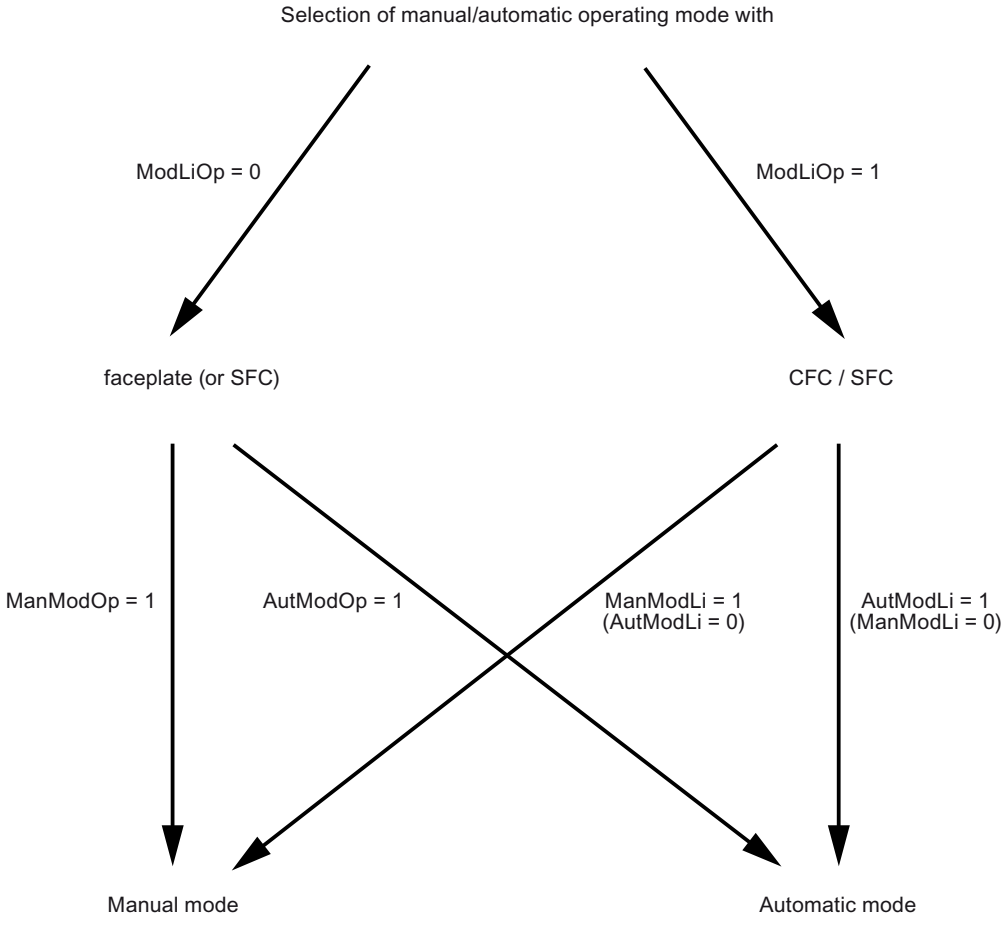
Manual and automatic mode for motors, valves and dosers

In "manual mode", the control settings for the device are made manually by the operator. The operator decides how to change the block's manipulated variable (output signal). The manipulated variable can be analog or binary in accordance with the function block.

In "automatic mode", the control settings for the device are made by the block algorithm via interconnected inputs or inputs controlled by SFC.

Changing between operating modes

The switchover between "manual and automatic mode" takes place as shown in the following schematic:



Note

The two selections (manual and automatic) cannot both be set to "1" in switching mode.

Switchover using faceplates (`ModLiOp = 0`)**:** The switchover between operating modes is carried out in the standard view of the faceplate. In the function block, the parameters `ManModOp` for "manual mode" and `AutModOp` for "automatic mode" are used.

Switchover per interconnection (CFC or SFC instance) (`ModLiOp = 1`)**:** The switchover between the operating modes is carried out with an interconnection on the function block. The parameters `ManModLi` for "manual mode" and `AutModLi` for "automatic mode" are used in

pushbutton operation. In switching mode (requirement: `Feature Bit 4 = 1`, see Setting switch or button mode (Page 166)) connection `AutModLi` is used exclusively.

Note

The `Feature` Bit 4 is available only for the "large" blocks.

Note

You can access the variable parameters `AutModOp` and `ManModOp` from a normal SFC (in contrast to the instance of an SFC type). The SFC can thus change the operating mode without revoking the access rights of the operator (i.e. without setting `ModLiOp = 1`).

Switchover from automatic mode to manual mode

When changing over from "automatic mode" to "manual mode", the last valid control settings for the block set in "automatic mode" remain valid until you change the control settings manually.

Note

For VlvAnL, the bumplessness of the switchover depends on the parameter `MV_TrkExt`.

Switchover from manual to automatic mode

You can set the following options for changing over from "manual mode" to "automatic mode" using the `Feature` bit Bumpless switchover to automatic mode (Page 172). Refer to the I/O descriptions for the relevant block.

- A switchover from manual to automatic mode is possible at any time (standard setting, `Feature` bit = 0). The control settings for the automatic mode become effective immediately.
- Switchover from manual to automatic mode is only possible if the control settings for the manual and automatic modes match (Bumpless switchover), (`Feature` bit = 1). An error message is output if they do not match. In this case, you will need to adapt the control settings in "manual mode" to the control settings in "automatic mode".

  Note

  The "Bumpless switchover to automatic mode" function is supported only by "large" blocks.

Reaction of signals when operating mode is changed

Using the `Feature` bit Resetting the commands for changing the mode (Page 160), you can choose whether the block automatically resets the signal for changing the operating mode.

Resetting the commands for the control settings

With the `Feature` bit Enabling resetting of commands for the control settings (Page 160), you select how the block handles commands for the control settings (for example motor on) via the interconnected input parameters.

2.1.2.6 Program mode for controllers

Program mode for controllers - interface for higher-level control functions

The interface for primary controller functions (external Advanced Control software package) provides primary controller functions, which run on an external PC as an OPC client, the option of using the control from the controller function block and specifying the setpoint or manipulated variable from a remote location. This procedure is called program mode.

You can use the feature bit Enabling program mode (Page 158) to specify whether or not the controller block is intended for program mode.

Program mode requires an enable signal (input parameter `AdvCoEn = 1`) from a central control block. If this enable signal goes from 1 to 0, for example, due to errors in the OPC communication, the controller block switches to manual mode if it is in program mode with manipulated variable specification or to automatic mode if it is in program mode with setpoint specification.

You activate program mode in the standard view of the controller faceplate. In addition to switching from manual to automatic mode, you are also given the option of using program mode as the operating mode. You exit program mode by operator input or by switching back into manual or automatic mode.

A 0-1 edge transition of the interconnectable input parameter `AdvCoMstrOn` activates program mode depending on the conditions described below. You can use this to put an entire group of downstream controller blocks into program mode at the same time from a central control block. Both the input parameter `AdvCoOn` and the interconnectable input parameter `AdvCoMstrOn` can be used at the same time, since the parameter `AdvCoMstrOn` only reacts to edges of the binary signal.

Program mode is deactivated with a 1 - 0 edge transition.

Note

The output parameters `ManAct` or `AutAct` display the mode to which the controller changes with a 1 - 0 edge transition.

The output parameter `AdvCoRdy = 1` indicates if the PID controller is ready to switch to program mode. At a central control block, you can use an AND operation for all `AdvCoRdy` signals of the downstream controllers to enable central switchover.

The output parameter `AdvCoAct = 1` indicates of the block is in program mode.

Selecting the type of program mode

There are two types of program mode:

- Program mode driven by setpoint (in automatic mode only)
- Program mode driven by manipulated variable (in manual mode only, not for step controllers without position feedback)

Program mode with setpoint: If you set the input parameter `AdvCoModSP = 1`, the analog value provided by the OPC client (`AdvCoMV`) is used as an external setpoint for the controller. The controller and faceplate otherwise react as they do with automatic mode and an external setpoint. Refer to section Setpoint specification - internal/external (Page 127) for more about this.

Requirements for program mode with setpoint:

- `AdvCoModSP = 1`,
- `AdvCoEn = 1`,
- Controller is preferably in manual mode, but it may also be in automatic mode
- The setpoint can be set to internal as well as external.

Program mode with manipulated variable: If you set the input parameter `AdvCoModSP = 0`, the analog value provided by the OPC client (`AdvCoMV`) is used as an external manipulated variable for the controller. The algorithm of the PID controller is bypassed. The controller and faceplate otherwise react as they do with tracking (`MV_TrkOn = 1`). Refer to section Tracking and limiting a manipulated variable (Page 192) for more about this.

Requirements for program mode with manipulated variable:

- `AdvCoModSP = 0`,
- `AdvCoEn = 1`,
- Controller is preferably in manual mode, but it can also be in automatic mode.
- The setpoint can be set to internal as well as external.

Note

Program mode with manipulated variable is not available for step controllers without position feedback (available in PIDStepL, FmCont and FmTemp). `ErrorNum = 50` is output on the controller block and the controller cannot switch into program mode `(AdvCoAct=0)`.

2.1.2.7 Local mode

Areas of application for local mode

This operating mode is used for motors, valves and dosing units. The control settings are made directly or via a control station that is located "locally". In addition, you can set different control strategies with the parameter `LocalSetting`.

With `LocalSetting = 0`, you prevent a change to "local mode".

Note

Differences between "Large" and "Small" blocks

The operating mode described here is valid for "Large" blocks. For "Small" blocks, `LocalSetting` can be configured only on a limited basis. For more information, refer to the respective description for the operating modes of the blocks.

Changing to local mode

Changing to local mode is only possible from the manual and automatic operating modes. The change to this mode is initiated by:

- An operation on the faceplate (input parameter `LocalOp = 1`, valid if `LocalSetting = 3` or `LocalSetting = 4` and `ModLiOp = 0`) or
- The interconnected input parameter (`LocalLi = 1`, valid if `LocalSetting = 1` or `LocalSetting = 2`).

Exiting local mode

You leave local mode using:

- An operation on the faceplate (`LocalSetting = 3` or `LocalSetting = 4` and `ModLiOp = 0`) or
- the interconnected input parameter (`LocalSetting = 1` or `LocalSetting = 2`).
  In order to exit local mode via the interconnected input parameter, you can configure various reactions using a `Feature` bit Exiting local mode (Page 176).

Operator input in "local mode" using a faceplate

You are not permitted to functionally operate the block in local mode. You can only use the faceplate to exit local mode if you have also activated "local mode" using the faceplate. The rules you specified for exiting "local mode" apply here.

Input in "local mode" via interconnected inputs

In "local mode", the way the block functions is influenced via interconnected input parameters according to the settings of the `LocalSetting` parameter. You have the following options:

- `LocalSetting = 1` and `LocalSetting = 3`
  - The control settings for the block are adjusted (tracking) via an interconnected input parameter. The interconnected input parameter includes the control signal for the local operator station on the system.
  - The runtime monitoring of the block is effective in accordance with your configuration.
  - The interlocking functions of the block are activated in accordance with input parameter `BypProt = 0` or deactivated (`BypProt = 1`).

  Note

  The block VlvAnL does not support `LocalSetting` = 1/3.

- `LocalSetting = 2` and `LocalSetting = 4`
  - The control settings for the block are made based on internal adjustment of the feedback value.
  - Runtime monitoring of the block is active only with rapid stop, external fault, motor protection, and if both feedback signals are set (discrepancy).
    An exception is VlvMotL:

  Note

  VlvMotL

  The motor and valve feedback signals are monitored if the motor feedback signlas exist and are connected with `FbkOpening` and `FbkClosing` (`Feature` bit 12 = 0). The motor feedback signals are only monitored in the end positions of the valve and in discrepancy. For example, if the valve is in end position `FbkOpen` =1 and the motor feedback `FbkOpening` =1 is pending, an error message is generated upon expiration of the monitoring time. If there are no motor feedback signals (`Feature` bit 12 = 1), monitoring of the valve and motor feedback signals does not take place.

  VlvAnL: The auxiliary valve is controlled via internal tracking of the feedback signals `FbkAuxVCloseOut` and `FbkAuxVOpenOut`. The control of the main valve via the feedback value `Rbk` is not affected by this.

The texts for labeling the command buttons in the faceplates of the motor and valve blocks can now be assigned for each specific instance.
The configuration of the texts is performed with the "Text 1" property of the respective control inputs of the motor and valve blocks in the CFC.

If no instance-specific text is configured, the previous default texts are used and displayed in the faceplate.

The following table shows the assignment of the command buttons to the corresponding block input:

The interlock functions of the block are deactivated.

Overview of behavior in local mode

LocalSetting =	**0**	**1**	**2**	**3**	**4**	**5 (VlvS only)**
Switch on operating mode	Cannot be set	CFC/SFC	CFC/SFC	Faceplate	Faceplate	CFC
Changing the operating mode: Local mode/to manual mode only (`Feature = 0`)	-	CFC/SFC	CFC/SFC	-	-	CFC/SFC
Changing the operating mode: Local mode/previous mode (`Feature = 1`)	-	CFC/SFC	CFC/SFC	-	-	CFC/SFC
Operating in the faceplate	-	Only rapid stop and resetting of rapid stop	Only rapid stop and resetting of rapid stop (only for "Large" blocks)	Only switching of operating mode, rapid stop, internal/ external setpoint switchover and resetting of rapid stop	Only switching of operating mode, rapid stop, and resetting of rapid stop	-
Executing local commands	-	Yes	No	Yes	No	No
Reaction of the block	-	Monitoring the feedbacks	Tracking of feedback, monitoring feedback during rapid stop, external errors, motor protection or discrepancy	Monitoring the feedbacks	Tracking of feedback, monitoring feedback during rapid stop, external errors, motor protection or discrepancy	Monitoring the feedbacks
Interlock activated	-	Yes: (`BypProt = 0`) No: (`BypProt = 1`)	only at output `LockAct` with `Feature` Bit 27 = 1 and `BypProt = 0`	Yes: (`BypProt = 0`) No: (`BypProt = 1`)	only at output `LockAct` with `Feature` Bit 27 = 1 and `BypProt = 0`	only at output `LockAct` with `Feature` Bit 27 = 1 and `BypProt = 0`

2.1.2.8 State graph of the operating modes

State graph of the operating modes

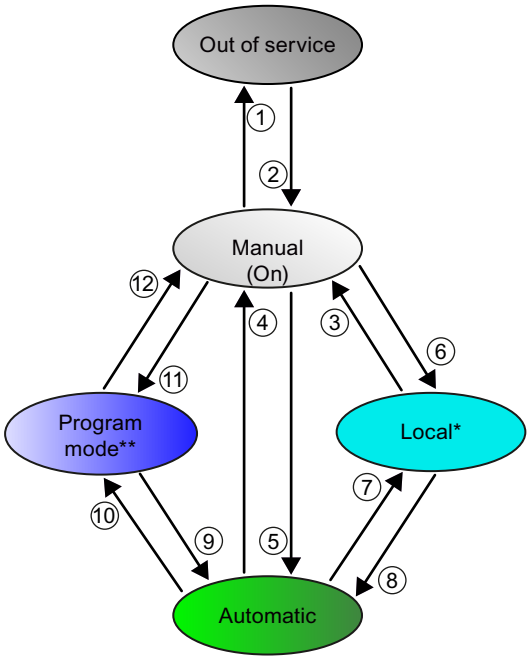


Figure 2-7 State graph of the operating modes

* This operating mode is used for motors, valves, and dosing units.

** This operating mode is used for controllers only.

Number in graphic (top)	Condition for status change
(1)	**Manual (on) → Out of service** • Via faceplate `(OosOp = 1)` if `ModLiOp = 0` or • Via edge transition 0 → 1 of `OosLi` if Feature bit Reaction to the out of service mode (Page 176)= 1
(2)	**Out of service → Manual (on)** • Via faceplate `(ManModOp = 1)`
(3)	**Local mode → Manual** • Via faceplate `(ManModOp = 1)` if `ModLiOp = 0` and `LocalSetting = 3` or `LocalSetting = 4` or • Via `LocalLi = 0` if `LocalSetting = 1`, `LocalSetting = 2` or `LocalSetting = 5`. See section Exiting local mode (Page 176) for more conditions.

Number in graphic (top)	Condition for status change
(4)	**Automatic → Manual** • Via faceplate (`ManModOp = 1`) if `ModLiOp = 0` or • Via `ManModLi = 1` if `ModLiOp = 1` and Feature bit Setting switch or button mode (Page 166)`= 0` or • Via `AutModLi = 0` if `ModLiOp = 1` and Feature bit Setting switch or button mode (Page 166)`= 1`
(5)	**Manual → Automatic** • Via faceplate (`AutModOp = 1`) if `ModLiOp = 0` or • Via `AutModLi = 1` if `ModLiOp = 1`
(6)	**Manual → Local mode** • Via faceplate `(LocalOp = 1)` if `ModLiOp = 0` and `LocalSetting = 3` or `LocalSetting = 4` or • Via `LocalLi = 1` if `LocalSetting = 1`, `LocalSetting = 2` or `LocalSetting = 5`
(7)	**Automatic → Local mode** • Via faceplate `(LocalOp = 1)` if `ModLiOp = 0` and `LocalSetting = 3` or `LocalSetting = 4` or • Via `LocalLi = 1` if `LocalSetting = 1`, `LocalSetting = 2` or `LocalSetting = 5`
(8)	**Local mode → Automatic** • Via faceplate `(AutModOp = 1)` if `ModLiOp = 0` and `LocalSetting = 3` or `LocalSetting = 4` or • Via `LocalLi = 0` if `LocalSetting = 1`, `LocalSetting = 2` or `LocalSetting = 5`. See section Exiting local mode (Page 176) for more conditions.
(9)	**Program mode → Automatic** • Via faceplate (`AutModOp = 1`) if `ModLiOp = 0` or • Via `AutModLi = 1` if `ModLiOp = 1` or • Via edge transition 1 → 0 of `AdvCoMstrOn` if automatic is set before program mode.
(10)	**Automatic → Program mode** Requirement for switchover in program mode: `AdvCoEn = 1` • Via faceplate (`AdvCoOn= 1`) if `ModLiOp = 0` or • Via `AdvCoMstrOn = 1`
(11)	**Manual → Program mode** Requirement for switchover from manual to program mode: `AdvCoEn = 1` and `AdvCoModSP = 0` • Via faceplate (`AdvCoOn= 1`) if `ModLiOp = 0` or • Via `AdvCoMstrOn = 1`
(12)	**Program mode → Manual** • Via faceplate (`ManModOp = 1`) if `ModLiOp = 0` or • Edge transition 1 → 0 of `AdvCoMstrOn` if manual is set before program mode.

2.1.3 Monitoring functions

2.1.3.1 Monitoring functions in the Advanced Process Library

Monitoring functions in the Advanced Process Library

This and the following chapters encompass the standard monitoring functions in the Advanced Process Library. The monitoring functions include:

- Limit value monitoring
- Feedback monitoring
- Motor protection

Some of the configured time values (e.g. `MonTiStatic`, `MonTiDynamic`) are limited at the low end to the sampling time by the block algorithm and written back to the block input. "Reset Program" (after a "Download Entire Program" for example) writes the parameter values changed in this way to the offline data storage system.

For further and detailed information, refer to the following chapters. For the block-specific monitoring functions, also refer to the description of the particular block.

2.1.3.2 Group display for limit monitoring, CSF and ExtMsgx

Group display for limit monitoring, CSF and ExtMsgx

The SumMsgAct output parameter assembles the following signals of a block and makes them available to you:

- Limit monitoring of the process value
- Limit monitoring of the count value
- Limit monitoring of the feedback
- Limit monitoring of setpoint difference, manipulated variable difference and error signal
- External control system fault (CSF)
- Freely selectable messages ExtMsg1..4

Note

The signal status of the individual signals is not taken into consideration for forming the group error.

2.1.3.3 Limit monitoring

Limit monitoring of the process value

You can monitor the process value to the following high and low alarm, warning and tolerance limits:

- `PV_AH_Lim`: Limit for high alarm
- `PV_AL_Lim`: Limit for low alarm
- `PV_WH_Lim`: Limit for high warning
- `PV_WL_Lim`: Limit for low warning
- `PV_TH_Lim`: Limit for the high tolerance
- `PV_TL_Lim`: Limit for the low tolerance

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. Please take into consideration the validity of terms for user-configured message classes (Page 41).

Note

Special note for "Small" blocks

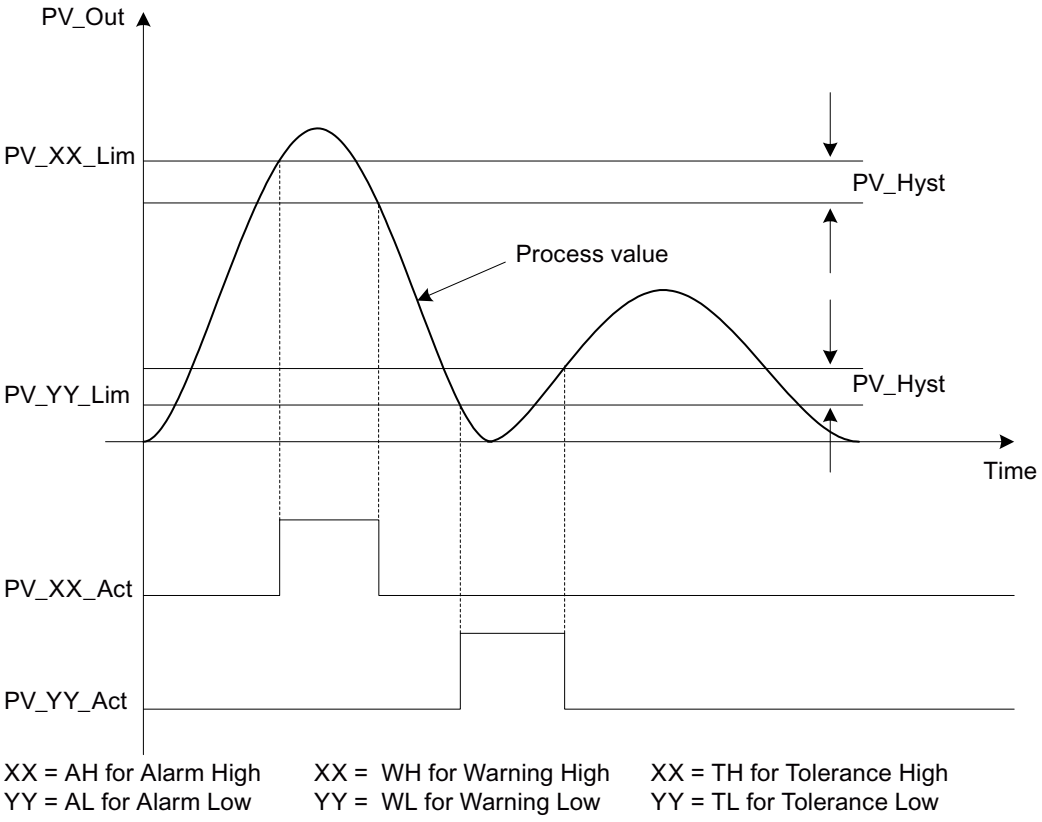
"Small" blocks provide only the monitoring for alarms and warnings.

Result of the limit monitoring

The result of limit monitoring is made available at the interconnectable output parameters:

- `PV_AH_Act = 1:`: Limit for high alarm reached or exceeded
- `PV_AL_Act = 1:`: Limit for low alarm reached or undershot
- `PV_WH_Act = 1:`: Limit for high warning reached or exceeded
- `PV_WL_Act = 1:`: Limit for low warning reached or undershot
- `PV_TH_Act = 1:`: Limit for high tolerance reached or exceeded
- `PV_TL_Act = 1:`: Limit for low tolerance reached or undershot

made available (see figure). The `SumMsgAct = 1` output parameter is also set when at least one limit value is reached or violated.



You can use `Feature` Bit 29 Signaling limit violation (Page 169) to determine whether the output parameter triggers limit monitoring with the value "0" or "1".

You can use `Feature` Bit 28 Disabling operating points (Page 144) to disable limit monitoring when message suppression is enabled (`MsgLock = 1`).

Activating limit monitoring

Monitoring is always enabled using the input parameters:

- `PV_AH_En = 1`: Monitoring of the high alarm limits
- `PV_AL_En = 1`: Monitoring of the low alarm limits
- `PV_WH_En = 1`: Monitoring of the high warning limits
- `PV_WL_En = 1`: Monitoring of the low warning limits
- `PV_TH_En = 1`: Monitoring of the high tolerance limits
- `PV_TL_En = 1`: Monitoring of the low tolerance limits

Predefinition: When the block is installed, monitoring of the tolerance limits is disabled, meaning that the parameters are configured with 0. To activate monitoring, assign 1 to these parameters.

All other monitoring functions are enabled.

Message suppression

	Symbol for message suppression

The corresponding message is suppressed using the parameters:

- `PV_AH_MsgEn = 0`: Alarm (high) messages are suppressed
- `PV_AL_MsgEn = 0`: Alarm (low) messages are suppressed
- `PV_WH_MsgEn = 0`: Warning (high) messages are suppressed
- `PV_WL_MsgEn = 0`: Warning (low) messages are suppressed
- `PV_TH_MsgEn = 0`: Tolerance (high) messages are suppressed
- `PV_TL_MsgEn = 0`: Tolerance (low) messages are suppressed

The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

Hysteresis

You can specify a hysteresis (`PV_Hyst`) for the limits, for example, to suppress signal flutter. Refer to the Limit monitoring with hysteresis (Page 97) chapter for more on this.

Alarm delays

You can set alarm delays for incoming and outgoing alarms, warnings and tolerances. Refer to the Area of application of the alarm delays (Page 195) chapter for more on this.

Operating in the faceplate

You can also change the limits and the hysteresis using the faceplate. Refer to the Limit operation and display in the faceplate (Page 309) chapter for more on this.

See also

Two time values per limit pair (Page 197)

Two time values for each individual limit (Page 198)

User-configured message classes (Page 41)

Limit monitoring of the count value

You can monitor the count value to the following high and low alarm, warning and tolerance limits:

- `OutAH_Lim`: Limit for high alarm
- `OutAL_Lim`: Limit for low alarm
- `OutWH_Lim`: Limit for high warning
- `OutWL_Lim`: Limit for low warning
- `OutTH_Lim`: Limit for the high tolerance
- `OutTL_Lim`: Limit for the low tolerance

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. Please take into consideration the validity of terms for user-configured message classes (Page 41).

Note

Monitoring the limits

The limits monitored depend on the direction of counting:

- In Mode 1 (summing up or integrating), the high limits are monitored:
  - OutAH_Lim
  - OutWH_Lim
  - OutTH_Lim
- In Mode 2 (summing down or integrating), the low limits are monitored:
  - OutAL_Lim
  - OutWL_Lim
  - OutTL_Lim

Example for limit monitoring with a counter

Monitoring of the high limit is performed only if the counter is running in "up" direction.

If you count "up" but with negative values, for example, from 100 down, you need to adjust the high limit accordingly (for example, the high limit could be -15).

This behavior applies to the blocks CountScL, CountOh and TotalL.

Result of the limit monitoring

The result of limit monitoring is made available at the interconnectable output parameters:

- `OutAH_Act = 1`: Limit for high alarm reached or exceeded
- `OutAL_Act = 1`: Limit for low alarm reached or undershot
- `OutWH_Act = 1`: Limit for high warning reached or exceeded
- `OutWL_Act = 1`: Limit for low warning reached or undershot

- `OutTH_Act = 1`:: Limit for high tolerance reached or exceeded
- `OutTL_Act = 1`:: Limit for low tolerance reached or undershot

made available (see figure). The `SumMsgAct = 1` output parameter is also set when at least one limit value is reached or violated.

You can use `Feature` Bit 29 Signaling limit violation (Page 169) to determine whether the output parameter triggers limit monitoring with the value "0" or "1".

You can use `Feature` Bit 28 Disabling operating points (Page 144) to disable limit monitoring when message suppression is enabled (`MsgLock = 1`).

Activating limit monitoring

Monitoring is always enabled using the input parameters:

- `OutAH_En = 1`: Monitoring of the high alarm limits
- `OutAL_En = 1`: Monitoring of the low alarm limits
- `OutWH_En = 1`: Monitoring of the high warning limits
- `OutWL_En = 1`: Monitoring of the low warning limits
- `OutTH_En = 1`: Monitoring of the high tolerance limits
- `OutTL_En = 1`: Monitoring of the low tolerance limits

Predefinition: When the block is installed, monitoring of the tolerance limits is disabled, meaning that the parameters are configured with 0. To activate monitoring, assign 1 to these parameters.

All other monitoring functions are enabled.

Message suppression

The corresponding message is suppressed using the parameters:

- `OutAH_MsgEn = 0`: Alarm (high) messages are suppressed
- `OutAL_MsgEn = 0`: Alarm (low) messages are suppressed
- `OutWH_MsgEn = 0`: Warning (high) messages are suppressed
- `OutWL_MsgEn = 0`: Warning (low) messages are suppressed
- `OutTH_MsgEn = 0`: Tolerance (high) messages are suppressed
- `OutTL_MsgEn = 0`: Tolerance (low) messages are suppressed

The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

Operating in the faceplate

You can also change the limits using the faceplate. Refer to the Limit operation and display in the faceplate (Page 309) chapter for more on this.

See also

User-configured message classes (Page 41)

Limit monitoring of an additional analog value

Limit monitoring of an additional analog value

Limit monitoring is performed for an additional analog value on the basis of the AV block, see Description of AV (Page 427) chapter.

You can monitor an additional analog value to the following high and low limits for alarms warnings and tolerances at the technologic block:

- `AV_AH_Lim`: Limit for high alarm
- `AV_AL_Lim`: Limit for low alarm
- `AV_WH_Lim`: Limit for high warning
- `AV_WL_Lim`: Limit for low warning
- `AV_TH_Lim`: Limit for the high tolerance
- `AV_TL_Lim`: Limit for the low tolerance

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. Please take into consideration the validity of terms for user-configured message classes (Page 41).

Note

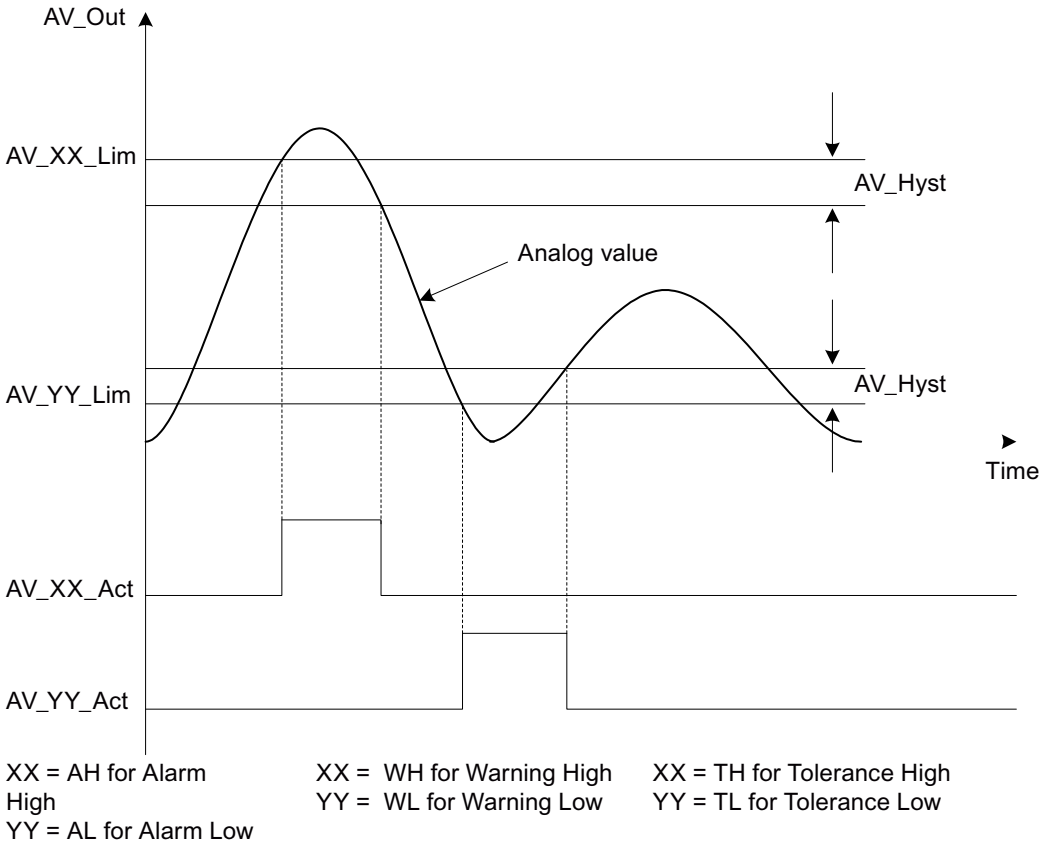
The AV block and the technologic block must be built into the same cyclic interrupt OB.

Result of the limit monitoring

The result of limit monitoring is made available at the interconnectable output parameters of the AV block:

- `AV_AH_Act = 1:` Limit for high alarm reached or exceeded
- `AV_AL_Act = 1:` Limit for low alarm reached or undershot
- `AV_WH_Act = 1:` Limit for high warning reached or exceeded
- `AV_WL_Act = 1:` Limit for low warning reached or undershot
- `AV_TH_Act = 1:` Limit for high tolerance reached or exceeded
- `AV_TL_Act = 1:` Limit for low tolerance reached or undershot

made available (see figure).



You can use `Feature` Bit 29 Signaling limit violation (Page 169) to determine whether the output parameter triggers limit monitoring with the value "0" or "1".

You can use `Feature` Bit 28 Disabling operating points (Page 144) to disable limit monitoring when message suppression is enabled (`MsgLock = 1`).

Activating limit monitoring

Monitoring is always enabled using the input parameters of the AV block:

- `AV_AH_En = 1`: Monitoring of the high alarm limits
- `AV_AL_En = 1`: Monitoring of the low alarm limits
- `AV_WH_En = 1`: Monitoring of the high warning limits
- `AV_WL_En = 1`: Monitoring of the low warning limits
- `AV_TH_En = 1`: Monitoring of the high tolerance limits
- `AV_TL_En = 1`: Monitoring of the low tolerance limits

Predefinition: When the block is installed, monitoring of the tolerance limits is disabled, meaning that the parameters are configured with 0. To activate monitoring, assign 1 to these parameters.

All other monitoring functions are enabled.

Message suppression

The corresponding message is suppressed at the block AV using the parameters:

- `AV_AH_MsgEn = 0`: Alarm (high) messages are suppressed
- `AV_AL_MsgEn = 0`: Alarm (low) messages are suppressed
- `AV_WH_MsgEn = 0`: Warning (high) messages are suppressed
- `AV_WL_MsgEn = 0`: Warning (low) messages are suppressed
- `AV_TH_MsgEn = 0`: Tolerance (high) messages are suppressed
- `AV_TL_MsgEn = 0`: Tolerance (low) messages are suppressed

The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

Hysteresis

You can specify a hysteresis (`AV_Hyst`) at the technologic block for the limits, for example, to suppress signal flutter. Refer to the Limit monitoring with hysteresis (Page 97) chapter for more on this.

Alarm delays

You can set alarm delays for incoming and outgoing alarms, warnings and tolerances. Refer to the Area of application of the alarm delays (Page 195) chapter for more on this.

Operating in the faceplate

You can also change the limits and the hysteresis using the faceplate. Refer to the Limit operation and display in the faceplate (Page 309) chapter for more on this.

See also

User-configured message classes (Page 41)

Limit monitoring of the feedback

Limit monitoring of position feedback

The position feedback of the manipulated variable can be monitored for the following high and low warning limits:

- `RbkWH_Lim`: Limit for high warning
- `RbkWL_Lim`: Limit for low warning

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. Please take into consideration the validity of terms for user-configured message classes (Page 41).

Result of limit monitoring of the position feedback

The result of limit monitoring of the position feedback is made available at the interconnectable output parameters:

- `RbkWH_Act = 1`: High limit reached or exceeded
- `RbkWL_Act = 1`: Low limit reached or undershot

made available. The `SumMsgAct = 1` output parameter is also set when at least one limit value is reached or violated.

When the limits are reached or exceeded, messages that can be suppressed are output.

You can use `Feature` Bit 29 Signaling limit violation (Page 169) to determine whether the output parameter triggers limit monitoring with the value "0" or "1".

You can use `Feature` Bit 28 Disabling operating points (Page 144) to disable limit monitoring when message suppression is enabled (`MsgLock = 1`).

Activating limit monitoring

Monitoring is always enabled using the input parameters:

- `RbkWH_En = 0`: Monitoring of the high warning limit is disabled
- `RbkWL_En = 0`: Monitoring of the low warning limit is disabled

Predefinition: When the block is installed, monitoring is enabled (default is 1).

Message suppression

The corresponding message is suppressed using the parameters:

- `RbkWH_MsgEn = 0`: Messages from the high limit monitoring are suppressed
- `RbkWL_MsgEn = 0`: Messages from the low limit monitoring are suppressed

The output of messages is not suppressed when the block is installed (for example, `RbkWH_MsgEn = 1`). Messages can only be output if limit monitoring of the position feedback has been enabled.

Hysteresis

You can specify a hysteresis (`RbkHyst`) for the limits, for example, to suppress signal flutter. Refer also to chapter Limit monitoring with hysteresis (Page 97) for more on this.

Alarm delays (only for the PIDConR and MotSpdCL blocks)

You can set alarm delays for incoming and outgoing warnings. Refer to the Area of application of the alarm delays (Page 195) chapter for more on this.

Operating in the faceplate

You can also change the limits and the hysteresis using the faceplate. Refer to the Limit operation and display in the faceplate (Page 309) chapter for more on this.

See also

User-configured message classes (Page 41)

Limit monitoring of setpoint, manipulated variable and control deviation

Limit monitoring of setpoint, manipulated variable and error signal

The setpoint, manipulated variable and error signal can be monitored for the following high and low alarm limits:

- `ER_AH_Lim`: Limit for high alarm
- `ER_AL_Lim`: Limit for low alarm

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. Please take into consideration the validity of terms for user-configured message classes (Page 41).

Result of the limit monitoring

The result of limit monitoring is made available at the interconnectable output parameters:

- `ER_AH_Act =1`: High limit violated (reached or exceeded)
- `ER_AL_Act =1`: Low limit (reached or undershot)

made available. The `SumMsgAct = 1` output parameter is also set when at least one limit value is reached or violated.

When the limits are reached or exceeded, messages that can be suppressed are output.

You can use `Feature` Bit 29 Signaling limit violation (Page 169) to determine whether the output parameter triggers limit monitoring with the value "0" or "1".

You can use `Feature` Bit 28 Disabling operating points (Page 144) to disable limit monitoring when message suppression is enabled (`MsgLock = 1`).

Activating limit monitoring

Alarm monitoring is enabled using the input parameters:

- `ER_AH_En = 1`: Monitoring of the high alarm limit
- `ER_AL_En = 1`: Monitoring of the low alarm limit

Predefinition: When the block is installed, monitoring is disabled.

Message suppression

The corresponding message is suppressed using the parameters:

- `ER_AH_MsgEn = 0`: Messages from the high limit monitoring are suppressed
- `ER_AL_MsgEn = 0`: Messages from the low limit monitoring are suppressed

The output of messages is not suppressed when the block is installed (for example, `ER_AH_MsgEn = 1`). Messages can only be output if limit monitoring has been enabled.

Note

With the MotSpdCL block, messages are only output if you have enabled `Feature` Bit 5 (Alarm setpoint difference (Page 170)).

Hysteresis

You can specify a hysteresis (`ER_Hyst`) for these limits, for example, in order to suppress signal flutter . Refer also to chapter Limit monitoring with hysteresis (Page 97) for more on this.

Alarm delays

You can set alarm delays for incoming and outgoing alarms. Refer to the Area of application of the alarm delays (Page 195) chapter for more on this.

Operating in the faceplate

You can also influence the limits and the hysteresis by means of the faceplate. Refer to the Limit operation and display in the faceplate (Page 309) chapter for more on this.

See also

User-configured message classes (Page 41)

Limit monitoring with hysteresis

Limit monitoring with hysteresis

You can additionally define a hysteresis for all limit monitoring functions (parameter `xxx_Hyst`, xxx can, for example, be `PV` for the process value). You use the hysteresis, for example, to suppress signal flutter.

Enter the hysteresis as a physical variable at the block and faceplate (if you have the appropriate operator control permissions (WinCC)).

For the WinCC operator control permissions, refer to the help on WinCC.

2.1.3.4 Feedbacks

Monitoring the feedbacks

Feedback monitoring

You can use the following monitoring functions:

- Monitoring the start-up and stop characteristics for motors or the runtime of valves
- Monitoring the operation of motors or the maintenance of the position of valves
- Disabling feedback

This monitoring function is enabled via the `Monitor = 1` input.

Static and dynamic errors are reset by disabling the monitoring (`Monitor = 0`). If you reactivate monitoring during the plant runtime, only dynamic monitoring (`MonTiDynamic`) will be performed.

Monitoring the start-up and stop characteristics for motors or the runtime of valves

Monitoring of the startup characteristics is implemented using the parameter `MonTiDynamic`. The monitoring time specifies the period within which the feedback value, for example, `FbkStart` with motors, must be available in response to a control signal. If this is not the case, the text "Control error" is displayed in the standard view of the faceplate. An error message is generated at the same time. The block then goes to its neutral position. In the case of motors, this is always the stop state. With other blocks, this is an neutral position you have specified (`SafePos` parameter). The block signals this at the corresponding output parameter of the error message with 1, for example, with `MonDynErr = 1` for motors.

With the motors MotL, MotRevL, MotSpdL, MotSpdCL and the motor valve VlvMotL, the `Feature` bit 13 can be set to use a separate monitoring time `MonTiDyStop` for the stop behavior. The monitoring time for the stop behavior is displayed at the `MonDynStopErr` output.

Parameters are set in seconds.

Note

In manual mode, you can control all valves (including the motor valve) despite the `MonSafePos = 1` setting and with "End position error" even without reaching the neutral position.

Monitoring the operation of motors or the maintenance of the position of valves

The following applies for "Large" blocks: Monitoring of the operation or the maintenance of the position of valves is implemented using the parameter `MonTiStatic`. The monitoring time specifies the period in which the feedback value can change its value briefly without an error message being output. An example would be a running motor with the feedback via the input parameter `FbkStart`. This parameter should be static in accordance with the control function. However, its value can change within the monitoring time. If the change in the `FbkStart` parameter takes longer than the monitoring time, the text "End position error" is displayed in the standard view of the faceplate. An error message is generated at the same time. The block then goes to its neutral position. In the case of motors, this is always the stop state. With other blocks, this is an neutral position you have specified (`SafePos` parameter). The block signals this at the corresponding output parameter of the error message with 1, for example, with `MonStaErr = 1` for motors.

Parameters are set in seconds.

Note

Please note that `MonTiDynamic ≥ MonTiStatic` and `MonTiDynamic ≥ SampleTime` have to be configured. If something is set outside these limits, the block always returns the respective limit at the input.

If `SampleTime` changes, `MonTiDynamic` may be tracked to the new value for `SampleTime`. `MonTiStatic` is tracked if `MonTiDynamic < MonTiStatic` changes. With `MonTiStatic = 0`, each feedback change without change of the control immediately results in a runtime error.

The following applies for "Small" blocks: These blocks operate like "Large" blocks; however, the monitoring time is set to 0 within the block and cannot be changed. Any change is displayed immediately at the output parameter `MonStaErr` with 1.

Disabling feedback for valves

You can also disable feedback completely. Please refer to section Disabling feedback for valves (Page 99) for further information.

Note

This function is only supported by "Large" blocks.

Resetting the block in case of interlocks or errors

In the event of an interlock or error, the block has to be reset. Refer to the Resetting the block in case of interlocks or errors (Page 43) section for more on this.

Disabling feedback for valves

Disabling monitoring of feedback for valves

This function is only supported by "Large" blocks.

If you operate a block without feedback, use the parameter `NoFbkOpen = 1 bzw. NoFbkClose = 1`. This means, for example, that you do not have any feedbacks for the opened state of the valve. Monitoring is thus disabled for this feedback. The feedback at the block is adjusted according to the control signal.

2.1.3.5 Motor protection function

Motor protection function

The motor protection function is used to turn off the motor if there is thermal overload (`Trip = 0`, interconnectable input parameter).

If the motor is turned off by the motor protection function, a message (process control message) is generated. This is indicated in the faceplate by the "Motor protection" text. You can influence the reset using a various `Feature` bits. Refer to the Resetting the block in case of interlocks or errors (Page 43) section for more on this.

You can find more information in Section Influence of the signal status on the interlock (Page 103).

2.1.4 Interlocking functions

2.1.4.1 Interlocks

Interlocks at blocks

A maximum of three types of interlock can be used depending on the block. Three separate inputs named `Intlock`, `Protect` and `Permit` are available for these functions. The block "MotRevL" has separate inputs for the activation enable **("Enable")** and the interlock without reset **("Interlock")** and interlock with reset **("Protection")** depending on Feature 2 bit 16. For more information, see section Separate interlock for each direction or position (Page 169).

MotRevL (Feature 2 bit 16 = 1):

- `Permit`: Enable forward
- `PermRev`: Enable reverse

- `Intlock`: Interlock forward
- `IntlRev`: Interlock reverse
- `Protect`: Protection forward
- `ProtRev`: Protection reverse

Note

Interlocks for "small" blocks and controller blocks

Ensure that "small" blocks and controller blocks PIDConL and PIDConR have only the `Intlock` parameter. The other two interlocks are not included in these block variants.

The following interlock types exist:

All blocks with the interlock function and MotRevL with Feature 2 bit 16 = 0

- **Activation enable ("Permission"):** The activation enable (input `Permit = 1`) makes it possible to leave the neutral position of the block in response to operator input or a command from the program (CFC/SFC). The activation enable has no effect if the block is not in the neutral position. See also the section Neutral position for motors, valves and controllers (Page 48) for information on the neutral position.
- **Interlock without reset ("Interlock"):** An active interlock condition brings the block to the neutral position (input `Intlock = 0`). After the interlock condition has gone, the currently active control function becomes active again in automatic or local mode. In manual mode the faceplate can be operated again after the interlock condition has gone.
- **Interlock with reset ("Protection"):** An active interlock condition brings the block to the neutral position (input `Protect = 0`). After the interlock conditions are cleared, the operator or an activation sequence must perform a reset to once again enable activation of the control according to the input parameters.
  You can influence the reset using a various `Feature` bits. Refer to the Resetting the block in case of interlocks or errors (Page 43) section for more on this.

Only MotRevL with Feature 2 bit 16 = 1

- **Activation enable forward ("Enable forward"):** The activation enable (input `Permit` = 1) makes it possible to leave the neutral position of the block in the forward direction in response to operator input or a command from the program (CFC/SFC). The activation enable has no effect if the block is not in the neutral position. See also the section Neutral position for motors, valves and controllers (Page 48) for information on the neutral position.
- **Activation enable reverse ("Enable reverse"):** The activation enable (input `PermRev` = 1) makes it possible to leave the neutral position of the block in the reverse direction in response to operator input or a command from the program (CFC/SFC). The activation enable has no effect if the block is not in the neutral position. See also the section Neutral position for motors, valves and controllers (Page 48) for information on the neutral position.
- **Interlock forward without reset ("Interlock forward"):** A pending forward interlock condition only puts the block in the neutral position when the motor is running in this direction (input `Intlock = 0`). Once the forward interlock condition is cleared, the currently active forward control function becomes active again in automatic or local mode. In manual mode, the faceplate can be operated forward again.

- **Interlock reverse without reset ("Interlock reverse"):** A pending reverse interlock condition only puts the block in the neutral position when the motor is running in this direction (input `IntlRev = 0`). Once the reverse interlock condition is cleared, the currently active reverse control function becomes active again in automatic or local mode. In manual mode, the faceplate can be operated reverse again.
- **Interlock forward with reset ("Protection forward"):** A pending forward interlock condition only puts the block in the neutral position when the motor is running in this direction (input `Protect = 0`). Once the interlock conditions are cleared, the operator or an activation sequence must perform a reset to once again enable activation of the forward control according to the input parameters. You can influence the reset using a various `Feature` bits. Refer to the Resetting the block in case of interlocks or errors (Page 43) section for more on this.
- **Interlock reverse with reset ("Protection reverse"):** A pending reverse interlock condition only puts the block in the neutral position when the motor is running in this direction (input `ProtRev = 0`). Once the interlock conditions are cleared, the operator or an activation sequence must perform a reset to once again enable activation of the reverse control according to the input parameters. You can influence the reset using a various `Feature` bits. Refer to the Resetting the block in case of interlocks or errors (Page 43) section for more on this.

Display of the interlock in the faceplate and in the block icon

The interlock state is visualized in the faceplate and in the block icon by a status display (padlock) as follows:

- Open padlock: No interlock pending
- Closed padlock: One or more interlocks are pending
- No padlock: Individual interlocks are not active
  - `Perm_En = 0` or `Permit.ST = 16#FF`: of the input parameter `Permit` has no effect, the button in the faceplate is invisible
  - `PermRevEn = 0` or `PermRev.ST = 16#FF` or `Feature2 Bit16 =0`: The `PermRev` input parameter has no effect, the button in the faceplate is invisible (only for "MotRevL")
  - `Prot_En = 0` or `Protect.ST = 16#FF`: of the input parameter `Protect` has no effect, the button in the faceplate is invisible
  - `ProtRevEn = 0` or `ProtRev.ST = 16#FF` or `Feature2 Bit16 =0`: The input parameter has no effect, the button in the faceplate is invisible (only for "MotRevL")
  - `Intl_En = 0` or `Intlock.ST = 16#FF`: of the input parameter `Intlock` has no effect, the button in the faceplate is invisible
  - `IntlRevEn = 0` or `IntlRev.ST = 16#FF` or `Feature2 Bit16 =0`: The input parameter has no effect, the button in the faceplate is invisible (only for "MotRevL")

The block icon indicates the prioritized group status according to the active operating state. See also the section Forming the group status for interlock information (Page 104) for more on this.

The faceplate visualizes the state of each interlock type separately.

The padlock is not shown in the block icon if all parameters for enabling the button are set to 0 or all parameters have the signal status `16#FF`.

The padlock is not shown in the block icon if all parameters for enabling the button are set to 0 (`Perm_En = 0, Prot_En = 0, Intl_En = 0`) or all parameters have the signal status `16#FF` (`Permit.ST = 16#FF, Protect.ST = 16#FF, Intlock.ST = 16#FF`).

Note

Motors and values are not put into the neutral position if one of the interlock inputs is active (for example `Intlock = 0`) and the corresponding signal status is `16#FF` (`Intlock.ST = 16#FF`).

Influence of the signal status on the interlock

See also the section Influence of the signal status on the interlock (Page 103) for more on this.

Outputting "Interlock active" using the `LockAct` parameter

If an interlock is set at the parameter:

- `Intlock`, `IntlRev` (only for MotRevL)
- `Permit`, `PermRev` (only for MotRevL)
- `Protect`, `ProtRev` (only for MotRevL)
- `Trip` (only for motors and motor valves),

the `LockAct` parameter is set automatically to active (=1). The parameter `LockAct` is set to 0 if the interlock is no longer present and those interlocks which require acknowledgement have been acknowledged.

You can bypass the interlock using `BypProt = 1` in local mode and during simulation. This also makes `LockAct = 0`.

Note

The `LockAct` parameter is not set despite a pending interlock, if a value in the block is forced. See also the section Forcing operating modes (Page 41) for more on this.

Messaging

No messages are assigned to the interlock types. However, if you want to have a message when an interlock condition is violated, you can use the freely interconnectable input parameters to generate the messages. See also the section Generating instance-specific messages (Page 200) for more on this.

2.1.4.2 Disabling interlocks

Disabling individual interlocks

You can disable the block interlocks that are implemented using the input parameters `Intlock`, `Protect` and `Permit`.

If you want to disable the block interlock, you have to set the following parameters:

- `Perm_En = 0` or `Permit.ST = 16#FF`: The `Permit` input parameter has no effect
- `PermRevEn = 0` or `PermRev.ST = 16#FF` or `Feature2 Bit16 =0`: The `PermRev` input parameter has no effect (only for "MotRevL")
- `Prot_En = 0` or `Protect.ST = 16#FF`: The `Protect` input parameter has no effect
- `ProtRevEn = 0` or `ProtRev.ST = 16#FF` or `Feature2 Bit16 =0`: The `ProtRev` input parameter has no effect (only for "MotRevL")
- `Intl_En = 0` or `Intlock.ST = 16#FF`: The `Intlock` input parameter has no effect
- `IntlRevEn = 0` or `IntlRev.ST = 16#FF` or `Feature2 Bit16 =0`: The `IntlRev` input parameter has no effect (only for "MotRevL")

Note

"Small" blocks have only the `Intlock` parameter

With "Small" blocks, you can only assign parameters to "Interlock without reset" (input parameter `Intlock` or for the deactivation of the interlock for the input parameter `Intl_En`).

Disabling of all the interlocks (only for local operation and for simulation)

You can use the input parameter `BypProt = 1` to disable all the interlocks, irrespective of the parameter assignment of the individual interlock, in local mode as well as for the "simulation" function.

2.1.4.3 Influence of the signal status on the interlock

Influence of the signal status on the interlock

There are three ways in which the signal status affects the interlocks:

- Simulation signal status
- Signal status "Bad, device related" (value 16#00) or "Bad, process related" (value 16#28)
- Signal status ≠ "Simulation" and "Bad, device related"

"Simulation" signal status

An interlock signal is displayed differently depending on the "Separate evaluation for excluded and simulated interlock signals (Page 151)" function and the status of the interlock (see "Forming the group status for interlock information (Page 104)").

Signal status "Bad, device related" (value 16#00) or "Bad, process related" (value 16#28)

An interlock signal with this status is always processed as an active interlock signal in the block and displayed with the following icons in the faceplate:

for 16#00 or

for 16#28 and

A motor protection signal (`Trip` parameter) with signal status 16#00 or 16#28 is used to activate motor protection. This is indicated by "Motor protection" in the standard view of the faceplates.

A torque-monitoring signal (`TorqOpen`, `TorqClose` parameters) with signal status 16#00 or 16#28 is used to activate torque monitoring for motor valve VlvMotL.

Signal status ≠ "Simulation" as well as "Bad, device related" and "Bad, process related"

Only signal states "Simulation", "Bad, device related" and "Bad, process related" have an effect on the processing in the block; all others are only displayed with their relevant icon in the faceplate.

2.1.4.4 Forming the group status for interlock information

Forming the group status for interlock information

A group status for interlock information is required for:

- Interlock state for:
  - Interlocked
  - Not interlocked
  - Disabled

Group status for the interlock state

All the effective interlock states are combined and displayed in the block icon. The interlock states are displayed with the following prioritization:

1. Function interlocked, shown in the block icon with a closed padlock

2. Function not interlocked, shown in the block icon as an open padlock

3. Function disabled, shown in the block icon as a crossed-out, closed padlock

Overview: Display for the interlock status in faceplate

If the Separate evaluation for excluded and simulated interlock signals (Page 151) function is not enabled (Feature bit =0), the bypass signals in the structures Permission, Protection, and Interlock are not evaluated.

Interlocks are shown in the faceplate as follows:

Parameter: BypProt	Parameter: Perm_En[2] or Prot_En[2] or Intl_En [2]	Value: Permission[2] or Protection[2] or Intlock [2]	Status: Permission[2] or Protection[2] or Intlock[2]	Button	Icon	Icon
x	x	X	16#FF	-	-	-
x	0	x	x	-	-	-
0	1	1	16#00	Visible		
0	1	1	16#28	Visible		
0	1	0	16#60	Visible		
0	1	1	16#60	Visible		
0	1	0	≠ 16#FF	Visible		Status based on priority
0	1	1	≠ (16#FF, 16#60, 16#00, 16#28)	Visible		Status based on priority
1 (for local mode and simulation only)	1	x	≠ 16#FF	Visible		Status based on priority

Comments on table:

- X: The value is irrelevant for the display of the icon.
- 2): MotRevL with Feature 2 Bit 16 = 1: "Perm_En", "Intl_En", "Prot_En", "Enable", "Interlock" and "Protection" also include the direction

Note

If values are forced in the block, this is indicated in the block icon by a crossed-out, closed padlock.

If the Separate evaluation for excluded and simulated interlock signals (Page 151) function is enabled (Feature bit =1), the bypass signals in the structures Permission, Protection, and Interlock are evaluated. The bypass signals can be read from the upstream interlock block at the output Bypass.

Excluded interlock signals are shown in the faceplate as follows:

Parameter: Perm_En2) or Prot_En2) or Intl_En2)	**Status: Permission2) or Protection2) or Intlock2)**	**Bypass: Permission2) or Protection2) or Intlock2)**	**Button**	**Icon**	
x	16#FF	x	-	-	
0	x	x	-	-	
1	≠ 16#FF	0	Visible		1)
1	≠ 16#FF	1	Visible		1)

Comments on table:

- X: The value is irrelevant for the display of the icon.
- 1): Icon corresponding to the status of the interlock signal (for 16#80 no icon)
- 2): MotRevL with Feature 2 Bit 16 = 1: "Perm_En", "Intl_En", "Prot_En", "Enable", "Interlock" and "Protection" also include the direction

2.1.4.5 Rapid stop for motors

Rapid stop for motors

Rapid stop has the highest priority in all operating modes (manual and automatic mode as well as local mode) and operating states (such as the forcing of states). It is activated via the faceplate. This depends on the setting at the Feature bit Enabling rapid stop via faceplate (Page 167).

Note

"Small" blocks do not feature rapid stop.

The "rapid stop" function is supported only by "Large" blocks.

You issue the command for rapid stop state using the RapidStp = 1 input parameter.

When you click on the "Rapid Stop" button in the faceplate, the drive stops immediately, shown as follows in the faceplate:

The `R_StpAct = 1` output parameter is set to implement the rapid stop function for local mode. You need to interconnect this parameter with the corresponding channel block and in the I/O to realize the rapid stop function in the hardware.

Rapid stop is unlocked for all operating modes using the "Reset" button in the faceplate (`RstOp = 1`); in CFC it is unlocked using the `RstLi = 1` input parameter. In automatic mode, the unlocking can also be performed via a 0-1 edge transition in the control if the `Feature` is `Bit9 = 1`.

Rapid stop can be selected even with the motor in stop state. In this case, the motor start is prevented.

2.1.4.6 Bypassing signals

Bypassing signals

With this function, you can flag a block as bypassed. With a connection from the output parameter `BypassAct` to an input parameter `BypLixx` of an interlock block, you can bypass the corresponding input of the interlock block.

Additionally, you can define over the `Feature` bit Substitution value is active if the block is in bypass (Page 184) if the block shall use the process value or a substitution value in case of an active bypass.

Block	Process value/signal	Process value/signal
MonAnL	`PV`	`BypPV`
PIDConL, PIDConR, and PIDStepL	`PV`	`BypPV`
MonDiL	`In`	`BypIn`
TotalL	`Out`	`BypOut`

With the parameter `BypLiOp`, you can decide whether the switching on/off will be done by the input parameters which are connected in CFC or by the faceplate from an operator.

- `BypLiOp` = 0: In the "Parameter view", the operator can switch on/off the bypass function over the parameter `BypPVOp` or `RstBypOp`. If `BypLock` = 1, you cannot make any change to the parameters `BypPVOp` or `ResBypOp`. The button for switching the bypass functionality on/off in the "Parameter view" is deactivated.
- `BypLiOp` = 1: With a connection of the parameter `BypPVLi` or `RstBypLi`, the bypass function will be switched on/off.

If the bypass function is active, the output parameter `BypassAct` will be set from the block.

2.1.5 Form signal status

2.1.5.1 Forming and outputting signal status for blocks

General information on forming and outputting the signal status

The process values of the function blocks are generated and transferred along with a signal status as a structured variable. This contains a statement about the signal quality. The function blocks determine the appropriate signal status for their process outputs depending on the signal status of the process inputs, which are involved in calculating the process outputs. If multiple process inputs are involved in calculating a process output, the signal status is formed according to prioritization defined by function block groups. The highest priority is the signal status with the value 0.

The blocks are grouped into the following function block groups:

- Technologic blocks (Page 108)
- Digital logic blocks (Page 110)
- Analog logic blocks (Page 111)
- Redundancy blocks (Page 112)
- Blocks with configurable status prioritization (Page 114)
- Interlock blocks (Page 115)
- Mathematical blocks (Page 117)
- PCS 7 channel blocks (Page 118)
- Channel blocks for field devices (Page 118)

All blocks of a group use the same priority specifications and form the signal status of the process outputs based on them.

Note

The status / quality of control inputs for logic functions and parameters have no influence on the status / quality of process values and logic functions of the blocks.

The status / quality of process values inherit the results of mathematic and logic functions, which are directly related to the process value.

The status/quality of process values immediately inherit the results of monitoring and limiting functions directly related to the process value.

2.1.5.2 Forming and outputting the signal status for technologic blocks

Forming the signal status for technology blocks

For more general information on forming the status signal, refer to the section: Forming and outputting signal status for blocks (Page 108).

In technology blocks, a group status is formed from the input parameters (see description of the relevant blocks) according to the priority table below (highest priority is 0). This group status is displayed in the status bar of the faceplate and of the block icon.

The group status is set to 16#68 (Uncertain, device related) with an undefined signal status at a control input, which is involved in the formation of the group status.

Signal status icon	Priority	Value	Meaning
	0	16#60	Manipulated value (for example, substitute value, simulation, last valid value)
	1	16#00	Bad, device related
	2	16#28	Bad, process related
	3	16#68	Uncertain, device related
	4	16#78	Uncertain, process related
	5	16#A4	Maintenance request
	6	16#80	Good

Interconnectable output parameters for limits (for example, `PV_AH_Act`) that can be influenced directly by an interconnectable input parameter (for example, `PV`) inherit the status from the associated output parameter (for example, `PV_Out`).

If an output parameter for limits is influenced directly by several interconnectable input parameters (limit monitoring), it receives the status of the input parameter with the highest priority (see overview above). Thus, for example, the control deviation is formed from the setpoint (`SP`) and the process value (`PV`). The output parameter for limits `ER_AH_Act`, for example, which signals an active violation of the high limit for the error signal, has a signal status based on the group status formed from the process value and setpoint (`ER`).

Evaluation of the signal status in case of interlocks in technology blocks

The signal states of the interconnectable input parameters of the interlocking and protective signals are treated exactly like process values with the following exceptions:

- The signal status is displayed in the faceplate at the buttons for calling the series-connected interlocking blocks.
- If the input signal has "Simulation" status and the input signal is therefore inactive (for example, `Protect` = 1), the input signal in this case is interpreted as a bypassed signal and is displayed by the icon for bypassing:

- If the input signal has "Simulation" status and the input signal is therefore inactive (for example, `Protect` = 0), the input signal in this case is interpreted as a simulated signal and is displayed by the icon for simulation:

- If the input signal has the "Bad, device related" or "Bad, process related" signal status, this is evaluated as an active input signal, regardless of its value, i.e. safety interlock signal (`Protect` = 1) that is inactive due to its value, triggers a safety interlock when its status is "Bad, device related" or "Bad, process related".

Display of the signal status in the faceplate and block icon for technology blocks

The signal status is displayed for each individual input parameter in the faceplate next to the process values or the interlock buttons. The group status is displayed in the block icon and in the group display of the faceplate.

Note

The interlocks and additional values are not included in the formation of the group status.

2.1.5.3 Forming and outputting the signal status of digital logic blocks

Forming the signal status of digital logic blocks

For more general information on forming the status signal, refer to the section: Forming and outputting signal status for blocks (Page 108).

The status for the result of the output is formed within digital logic blocks from all input parameters, according to the following priority table (highest priority is 0)

Signal status icon	Priority	Value	Meaning
	0	16#00	Bad, device related
	1	16#28	Bad, process related
	2	16#60	Manipulated value (for example, substitute value, simulation, last valid value)
	3	16#68	Uncertain, device related
	4	16#78	Uncertain, process related

Signal status icon	Priority	Value	Meaning
	5	16#A4	Maintenance request
	6	16#80	Good

The group status is set to 16#68 (uncertain, device-related), with an undefined signal status at a control input that is involved in forming the group status.

A signal status 16#FF at a control input is not used for the calculation of the group status. If all relevant control inputs are 16#FF, the group status is 16#80.

If only one process input is decisive for calculating the output value, the status of the process input is transferred to the status of the output.

Special notes for the Andxx blocks

- If the output value is 1, it has the signal status with the highest priority of all input signals.
- If the output value is 0, it has the signal status with the lowest priority of all input signals, which have a value of 0.

Special notes for the Orxx blocks

- If the output value is 1, it has the signal status with the lowest priority of all input signals, which have a value of 1.
- If the output value is 0, it has the signal status with the highest priority of all input signals.

Special notes for the Xor04 block

The worst signal status of all input parameters is always selected and output with the `Out` output parameter.

2.1.5.4 Forming and outputting the signal status of analog logic blocks

Forming the signal status of analog logic blocks

For more general information on forming the status signal, refer to the section: Forming and outputting signal status for blocks (Page 108).

The signal status of the Out output value within the block is taken directly from the `Inx` input value.

Special notes for the CompAn02 block

This block evaluates the signal status of the two input parameters `In1` and `In2` as shown in the following table.

Signal status icon	Priority	Value	Meaning
	0	16#60	Manipulated value (for example, substitute value, simulation, last valid value)
	1	16#00	Bad, device related
	2	16#28	Bad, process related
	3	16#68	Uncertain, device related
	4	16#78	Uncertain, process related
	5	16#A4	Maintenance request
	6	16#80	Good

2.1.5.5 Forming and outputting the signal status of redundancy blocks

Forming the signal status of redundancy blocks RedAn02 and RedDi02

For more general information on forming the status signal, refer to the section: Forming and outputting signal status for blocks (Page 108).

The signal status is evaluated according to the following priority:

Signal status icon	Priority	Value	Meaning
	0	16#60	Manipulated value (for example, substitute value, simulation, last valid value)
	1	16#80	Good
	2	16#A4	Maintenance request
	3	16#78	Uncertain, process related
	4	16#68	Uncertain, device related

Signal status icon	Priority	Value	Meaning
	5	16#28	Bad, process related
	6	16#00	Bad, device related

The output value is generated based on the signal status of the process value with the above table of priorities. In addition, the output parameters `SimAct`, `Uncertain`, and `LossRed` are still set according to the signal status.

In1.ST	In2.ST	Out.ST	Out.Value	SimAct.Value	Uncertain.Value	LossRed.Value
16#80	16#80	16#80	In1	0	0	0
16#80	16#60	16#60	In2	1	1	0
16#80	16#A4	16#80	In1	0	1	0
16#80	16#78	16#80	In1	0	1	0
16#80	16#68	16#80	In1	0	1	0
16#80	16#28	16#68	In1	0	1	1
16#80	16#00	16#68	In1	0	1	1
16#60	16#80	16#60	In1	1	1	0
16#60	16#60	16#60	In1	1	1	0
16#60	16#A4	16#60	In1	1	1	0
16#60	16#78	16#60	In1	1	1	0
16#60	16#68	16#60	In1	1	1	0
16#60	16#28	16#60	In1	1	1	1
16#60	16#00	16#60	In1	1	1	1
16#A4	16#80	16#80	In2	0	1	0
16#A4	16#60	16#60	In2	1	1	0
16#A4	16#A4	16#A4	In1	0	1	0
16#A4	16#78	16#A4	In1	0	1	0
16#A4	16#68	16#A4	In1	0	1	0
16#A4	16#28	16#68	In1	0	1	1
16#A4	16#00	16#68	In1	0	1	1
16#78	16#80	16#80	In2	0	1	0
16#78	16#60	16#60	In2	1	1	0
16#78	16#A4	16#A4	In2	0	1	0
16#78	16#78	16#78	In1	0	1	0
16#78	16#68	16#78	In1	0	1	0
16#78	16#28	16#68	In1	0	1	1
16#78	16#00	16#68	In1	0	1	1
16#68	16#80	16#80	In2	0	1	0
16#68	16#60	16#60	In2	1	1	0
16#68	16#A4	16#A4	In2	0	1	0
16#68	16#78	16#78	In2	0	1	0

In1.ST	In2.ST	Out.ST	Out.Value	SimAct.Value	Uncertain.Value	LossRed.Value
16#68	16#68	16#68	In1	0	1	0
16#68	16#28	16#68	In1	0	1	1
16#68	16#00	16#68	In1	0	1	1
16#28	16#80	16#68	In2	0	1	1
16#28	16#60	16#60	In2	1	1	1
16#28	16#A4	16#68	In2	0	1	1
16#28	16#78	16#68	In2	0	1	1
16#28	16#68	16#68	In2	0	1	1
16#28	16#28	16#28	In1	0	1	1
16#28	16#00	16#28	In1	0	1	1
16#00	16#80	16#68	In2	0	1	1
16#00	16#60	16#60	In2	1	1	1
16#00	16#A4	16#68	In2	0	1	1
16#00	16#78	16#68	In2	0	1	1
16#00	16#68	16#68	In2	0	1	1
16#00	16#28	16#28	In2	0	1	1
16#00	16#00	16#00	In1	0	1	1

2.1.5.6 Forming and outputting the signal status for blocks with configurable status prioritization

Forming the signal status for blocks with configurable status priority

For more general information on forming the status signal, refer to the section: Forming and outputting signal status for blocks (Page 108).

The SelPrio parameter is used for this block to define the priority setting for linking the individual states. You have the option between the following specifications:

Priority	SelPrio = 0	SelPrio = 1	SelPrio = 2	SelPrio = 3	SelPrio = 4	SelPrio = 5	SelPrio = 6	SelPrio = 7
0	16#60	16#80	16#A4	16#80	16#A4	16#60	16#80	16#60
1	16#00	16#00	16#00	16#A4	16#80	16#80	16#A4	16#80
2	16#28	16#28	16#28	16#60	16#60	16#A4	16#78	16#A4
3	16#68	16#68	16#68	16#00	16#00	16#00	16#68	16#78

Priority	SelPrio = 0	SelPrio = 1	SelPrio = 2	SelPrio = 3	SelPrio = 4	SelPrio = 5	SelPrio = 6	SelPrio = 7
4	 16#78	 16#78	 16#78	 16#28	 16#28	 16#28	 16#28	 16#68
5	 16#A4	 16#60	 16#60	 16#68	 16#68	 16#68	 16#00	 16#28
6	 16#80	 16#A4	 16#80	 16#78	 16#78	 16#78	 16#60	 16#00

Special notes for the MuxAn03 block

The status priority for this block, which is used by the block to process the status of the `PV1` ... `PV3` process value inputs, can be set with the `SelPrio` parameter.

Note

The parameter `SelPrio` can take a value from 0 to 7 . `SelPrio = 6` is set by default.

If you enter a value greater than 7, the setting for 7 is used. If you enter a value lower than 0, the setting for 0 is used.

For information on evaluating the status of the process value inputs, refer to the section MuxAn03 functions (Page 1903).

2.1.5.7 Forming and outputting the signal status for interlock blocks

Forming the signal status for interlock blocks

For more general information on forming the status signal, refer to the section: Forming and outputting signal status for blocks (Page 108).

The block determines the signal status of the output signal, based on the signal status of the input values from the configured logical operation according to the following table (highest priority is 0):

Signal status icon	Priority	Value	Meaning
	0	16#00	Bad, device related
	1	16#28	Bad, process related
	2	16#60	Manipulated value (for example, substitute value, simulation, last valid value)

Signal status icon	Priority	Value	Meaning
	3	16#68	Uncertain, device related
	4	16#78	Uncertain, process related
	5	16#A4	Maintenance request
No icon	6	16#80	Good
No icon	-	16#FF	Input is not connected

General rules

- If a given input causes a signal change at the output, the signal status of the input with the highest priority (worst signal status) is set at the output, since each input can influence the output:
  - Logical AND operation (logic = 1):
    If the output value is 1, it has the signal status with the highest priority of all interconnected input signals.
  - Logical OR operation (logic = 0):
    If the output value is 0, it has the signal status with the highest priority of all interconnected input signals.
- If several inputs have the same priority and the output therefore cannot be changed, the signal status of the input with the lowest priority (highest signal status) is set at the output, since the output is uniquely determined by the signal with the best signal status:
  - Logical AND operation (logic = 1):
    If the output value is 0, it has the signal status with the lowest priority of all interconnected input signals, which have a value of 0.

    Note

    With `feature bit` 23 = 1 (evaluation signal status), input signals with a bad signal status (`16#28` or `16#00`) are processed with the value 0, regardless of their actual value
  - Logical OR operation (logic = 0):
    If the output value is 1, it has the signal status with the lowest priority of all interconnected input signals, which have a value of 1.
- If no inputs are interconnected, the signal status of the output is set to simulation (`16#60`).

Display of the signal status in the faceplate and block icon for interlocking blocks

The signal status is displayed for each individual parameter (except for the analog values) in the faceplate next to the process values.

If you bypass a signal, it is displayed in the faceplate of the interlocking block next to the button for excluding, as well as in the block icon as follows.

Note

If an excluded interlock signal can become switchover-relevant, the status of the output Out.ST is set to simulation 16#60, as long as it is not forced by other interlock inputs to Bad, device-related 16#00 or Bad, process-related 16#28.

An excluded interlock input has no influence on the status when the Separate evaluation for excluded and simulated interlock signals (Page 151) function is enabled.

The currently valid status for the output signal is also displayed in the faceplate.

2.1.5.8 Forming and outputting the signal status for mathematical blocks

Forming the signal status for mathematical blocks

For more general information on forming the status signal, refer to the section: Forming and outputting signal status for blocks (Page 108).

The status for the result of the output is formed within mathematical blocks from all process inputs involved in the calculation, according to the following table (highest priority is 0)

Signal status icon	Priority	Value	Meaning
	0	16#00	Bad, device related
	1	16#28	Bad, process related
	2	16#60	Manipulated value (for example, substitute value, simulation, last valid value)
	3	16#68	Uncertain, device related
	4	16#78	Uncertain, process related
	5	16#A4	Maintenance request
	6	16#80	Good

The group status is set to 16#68 (uncertain, device-related), with an undefined signal status at a control input that is involved in forming the group status.

A 16#FF signal status at a control input is not used for the calculation of the group status. If all relevant control inputs are 16#FF, the group status in the search for the worst group status is 16#80 and 16#00 in the search for the best group status.

If only one process input is decisive for calculating the output value, the status of the process input is transferred to the status of the output.

Note

Special notes for the Integral mathematical block

Due to its application area (time integral, I-component for the configuration of a controller), the Integral block generates the signal status like the technological blocks.

Special notes for the mathematical blocks Addxx, Mulxx, Div02 and Sub02

If the result of the mathematic operation is a floating-point number that cannot be displayed, the result of the status is set to 16#28. Floating-point numbers that cannot be displayed are labeled in the CFC with #+Inf (+ infinite), #-Inf (- infinite) or with #NaN (not a number).

2.1.5.9 Forming and outputting the signal status for PCS 7 channel blocks

Forming the signal status for PCS7 channel blocks

For more general information on forming the status signal, refer to the section: Forming and outputting signal status for blocks (Page 108).

The signal status for PCS7 channel blocks can assume the following values:

Signal status icon	Value	Meaning
	16#80	Good
	16#78	Uncertain, process related: Limitation of input parameter `PV_In` is active (analog output channel blocks only)
	16#60	Manipulated value (for example, substitute value, simulation, last valid value)
	16#00	Bad, device related (value not valid)

2.1.5.10 Forming and outputting the signal status for channel blocks for field devices

Forming the signal status for channel blocks (field devices)

For more general information on forming the status signal, refer to the section: Forming and outputting signal status for blocks (Page 108).

The signal status of channel blocks for field devices can assume the following values:

Signal status icon	Value	Meaning
	16#80	Good
	16#78	Uncertain, process related
	16#68	Uncertain, device related
	16#60	Manipulated value (for example, substitute value, simulation, last valid value)
	16#28	Bad, process related
	16#00	Bad, device related (value not valid)
	16#A4	Maintenance required, maintenance demanded

2.1.6 Error handling

2.1.6.1 Error handling

Error handling

The channel and technologic blocks feature error handling routines. A distinction must be made between the following areas:

- Error numbers
- External process control error (CSF)
- Process-specific errors
- Invalid signal states
- Mode switchover error
- Errors in channel blocks

Error numbers

Most blocks have an output parameter `ErrorNum` that can be used to output internal error states of the block as error numbers.

With some blocks, input parameters are checked for permissible values. They are therefore only used to prevent the output value from remaining invalid when the input value is once again in the valid range. If an invalid value is detected, and the corresponding output value is held at the last displayed value instead of an invalid value being displayed. If blocks do not have

this check, an invalid value can appear at the output. However, a valid value is displayed again at the output as soon as the input values of the block have changed correspondingly.

Any value set over an interconnection or as a result of a parameter assignment that is outside the range of values (e.g. "Not a Number") is not processed by the block algorithm. The last valid value is processed instead.

In addition to the errors stated above, a limit violation is also signaled for example. Each error number is assigned to a specific error.

If there is more than one error, all error numbers have the same priority. The routine always displays the error number of the error most recently detected in a block cycle.

External process control error (CSF)

An external process control error always lies outside the process - it exists in the form of device or other hardware faults. If, for example, a run-time error occurs at a valve, there is an error or fault in the pneumatic system.

A process control error is output if an external error is set at the input `CSF`. You can enable this output function, for example, by interconnecting output `Bad` of the channel block with input `CSF` of the technologic block.

The error message "$$BlockComment$$ External error occurred" is output at `CSF = 1`.

This state is visualized in the group display by an "S" character in the faceplate overview and in the block icon.

With motor, valve and dosing blocks, there is also the possibility that the block switches to error processing. `Feature` bit 18 Activating error state for external process control error CSF (Page 150) must be set for this. The device goes into error processing and moves the drive to the neutral position or switches the dosing operation to off for a doser. If the dosing operation is already finished, it remains in the completed state. Resetting the error is described in Resetting the block in case of interlocks or errors (Page 43).

External error (FaultExt)

The `FaultExt` input can be used to pass an external error without the generation of a message. The device goes into error processing and moves the drive to the neutral position or switches the dosing operation to off for a doser. If the dosing operation is already finished, it remains in the completed state. Separate messages can be generated by interconnecting the external error to `ExtMsgx` messages that can be freely selected. Resetting the error is described in Resetting the block in case of interlocks or errors (Page 43).

Process-specific errors

Process-specific errors can have the following causes:

- Runtime monitoring: If the feedback signals do not match the control settings after a selected time has expired, a process-related error is output.
- Feedback monitoring: Refer to the Monitoring the feedbacks (Page 97) section for more on this.

If the block algorithm detects a monitoring error while monitoring is enabled, the corresponding output parameter is set to 1 in the block. The "$$BlockComment$$ Feedback error xxx" error message is also output, where xxx, for example, stands for the valve.

This state is visualized in the group display by an "S" character in the faceplate overview and in the block icon.

The block must be reset after the monitoring error was cleared and if automatic mode is set.

Invalid input signals

This error is output if inconsistencies are detected between associated I/Os. The close and open commands cannot be output simultaneously to the valve, for example.

If the block algorithm detects an invalid combination of input signals, an error number `(ErrorNum)` is output that depends on the block type.

In the case of motors, valves and batches, the faceplates' standard view will additionally output the text "invalid signal".

Mode switchover error

This error is reported if you change the mode of the block from:

- Manual to automatic mode or
- Local mode to automatic mode

and the previous and target state are **inconsistent** (bumpless switchover). You can only change the block mode if the subsequent state corresponds with the previous state.

Bumpless switchover can be activated / deactivated using the `Feature` connection on the Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172) or Disabling bumpless switchover to automatic mode for controllers (Page 172) bit.

Bumpless switchover from local to automatic mode is undertaken using the `LocalSetting` parameter, as described in section Local mode (Page 79).

In the standard view of the faceplate, the text "Changeover error" is displayed in the event of an unwanted switchover with bumps.

The block retains local mode if the operator changes the mode from local to automatic and the error mentioned above occurs. The block changes to manual mode if the mode is changed from local to automatic over interconnected inputs and the error mentioned above occurs.

Errors in channel blocks

The following errors may be displayed by the channel blocks:

- Channel error
- Device or module fault
- Higher-level error
- Invalid measuring range

2.1.6.2 Outputting group errors

Outputting group errors

The `GrpErr` output parameter assembles the faults of a block and makes them available to you. A group error is compiled from the following error information:

- Feedback errors (static or dynamic feedback monitoring)
- External error (CSF and `FaultExt`)
- Motor protection (only for motors)
- Module errors (only for hardware controllers)
- I/O access errors (only for hardware controllers)
- Parameter assignment errors (only for hardware controllers)

For information on how the signal is formed for the group error at the `GrpErr` output parameter, refer to the corresponding block descriptions in the "Functions" chapter.

Note

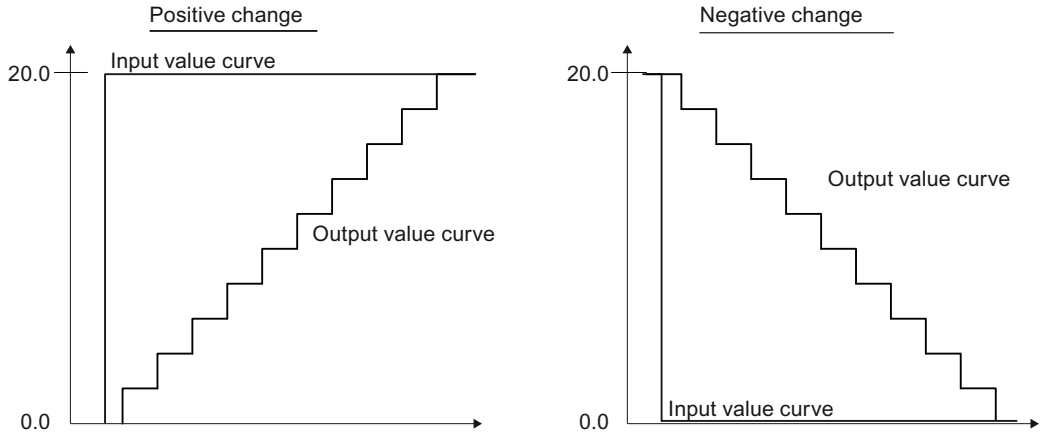
The signal status of the individual signals is not taken into consideration for forming the group error. The outputs therefore always have the status, `16#80`.

2.1.7 Ramp function

2.1.7.1 Using setpoint ramp

Using setpoint ramp

Starting at the current internal setpoint, the setpoint can be set to a target setpoint value over a ramp-shaped function. In the faceplate, you can start the function in ramp view (`SP_RmpOn = 1`).



Use the `SP_RmpModTime` input parameter or the ramp view of the faceplate to specify whether the setpoint ramp is defined by time or by gradients:

- If you select time `(SP_RmpModTime = 1)`: The ramp of the setpoint is calculated automatically by the block so that after the ramp has started `(SP_RmpOn = 1)`, the setpoint will reach the target setpoint (`SP_RmpTarget`) after the selected time `(SP_RmpTime)`.
- If you select ramp `(SP_RmpModTime = 0)`: The inclination of the ramp matches the selected rates of change `SP_UpRaLim` (positive) or `SP_DnRaLim` (negative).

Once the setpoint has reached the target setpoint, the function is terminated automatically `(SP_RmpOn = 0)`. The ramp trip can be prematurely aborted in the faceplate by setting `SP_RmpOn = 0`.

Requirements for using a setpoint ramp

Block	Manipulated variable	Gradient limit	Operating mode
Controller	Internal	Off	Automatic
OpAnL	Internal	Off	
MotSpdCL	Internal	Off	

If the requirements are not fulfilled during the ramp trip, the ramp trip is automatically canceled.

Note

Special note for the MotSpdCL block

In the case of an interlock, a monitoring error, motor protection or rapid stop, the motor is switched off and the internal ramp setpoint is reset to the starting setpoint of the ramp trip.

2.1.7.2 Gradient limit of the setpoint

Gradient limit of the setpoint

The gradient limit is activated via the `SP_RateOn = 1` input parameter.

The values are set at the `SP_UpRaLim` and `SP_DnRaLim` parameters depending on the `TimeFactor`.

- `TimeFactor = 0`: Unit of the gradient limiting is Unit/Second
- `TimeFactor = 1`: Unit of the gradient limiting is Unit/Minute
- `TimeFactor = 2`: Unit of the gradient limiting is Unit/Hour
- `SP_UpRaLim` sets the gradient high limit
- `SP_DnRaLim` sets the gradient low limit

Note

Parameters`SP_UpRaLim` and `SP_DnRaLim` are always evaluated according to their magnitude.

With the `Feature` Bit Gradient limitation with time duration (Page 181), you can also use the parameters `SP_RmpModTime` and `SP_RmpTime` in the gradiant limit function. If this `Feature` Bit = 1 and `SP_RmpModTime` = 1 is parameterized, the ramp rate is calculated with a change of setpoint such that the new value of the setpoint will reach after the time of `SP_RmpTime`.

Displaying active limitation

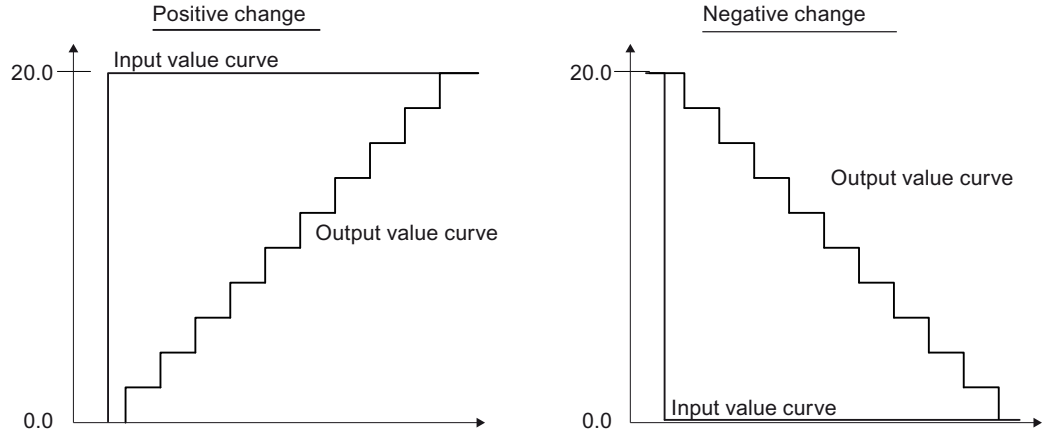
A gradient limit is indicated at the following output parameters:

- `SP_UpRaAct = 1`: Gradient has a high limit
- `SP_DnRaAct = 1`: Gradient has a low limit

2.1.7.3 Using a manipulated variable ramp

Using a manipulated variable ramp

Starting at the current internal manipulated variable, the manipulated variable can be brought to a target value in the form of a ramp. In the faceplate, you can start the function in ramp view (`MV_RmpOn = 1`).



Use the `MV_RmpModTime` input parameter or the ramp view of the faceplate to specify whether the manipulated variable ramp is defined by time or by gradients:

- If you select time `(MV_RmpModTime = 1)`: The gradients of the manipulated variable are calculated automatically by the block so that after the ramp has started `(MV_RmpOn = 1)`, the manipulated variable will reach the target value (`MV_RmpTarget`) after the configured time `(MV_RmpTime)`.
- Specification with gradients `(MV_RmpModTime = 0)`: The ramp slope matches the configured rates of change `MV_UpRaLim` (positive) or `MV_DnRaLim` (negative).

Once the manipulated variable has reached the target value, the function is terminated automatically `(MV_RmpOn = 0)`. The ramp trend can be prematurely aborted in the faceplate by setting `MV_RmpOn = 0`.

Requirements for using a manipulated value ramp:

Block	Manipulated variable	Gradient limit	Operating mode
VlvAnL without auxiliary valve	Internal	Off	Manual
VlvAnL with auxiliary valve	Internal	Off	

If the requirements are not fulfilled during the ramp trip, the ramp trip is automatically canceled.

2.1.7.4 Gradient limiting of the manipulated variable

Gradient limiting of the manipulated variable

The gradient limit is activated via the `MV_RateOn = 1` input parameter.

The values are set at the `MV_UpRaLim` and `MV_DnRaLim` parameters depending on the `TimeFactor`.

- `TimeFactor = 0`: Unit of the gradient limiting is Unit/Second
- `TimeFactor = 1`: Unit of the gradient limiting is Unit/Minute
- `TimeFactor = 2`: Unit of the gradient limiting is Unit/Hour
- `MV_UpRaLim` sets the gradient high limit
- `MV_DnRaLim` sets the gradient low limit

Note

Parameters `MV_UpRaLim` and `MV_DnRaLim` are always evaluated according to their magnitude.

With the `Feature` Bit Gradient limitation with time duration (Page 181), you can also use the parameters `MV_RmpModTime` and `MV_RmpTime` in the gradiant limit function. If this `Feature` Bit = 1 and `MV_RmpModTime` = 1 is parameterized, the ramp rate is calculated with a change of setpoint such that the new value of the setpoint will reach after the time of `MV_RmpTime`.

Displaying active limitation

A gradient limit is indicated at the following output parameters:

- `MV_UpRaAct = 1`: Gradient has an high limit
- `MV_DnRaAct = 1`: Gradient has a low limit

2.1.8 Internal/external setting

2.1.8.1 Applying the dynamically activated dead band during the PV settling time

`Feature` bit

Number of the `Feature` bit: 30.

Dynamic deactivation of the dead band by the `PV` settling time

You can use this `Feature` bit to improve the `PV` settling time in the range of the dead band.

The deadband is deactivated until the `PV` settles more in the center than at the edges of the dead band. The probability that the controlled variable is within the dead zone time is increased,

and further control actions are no longer necessary. This leads to reduced wear and less energy consumption.

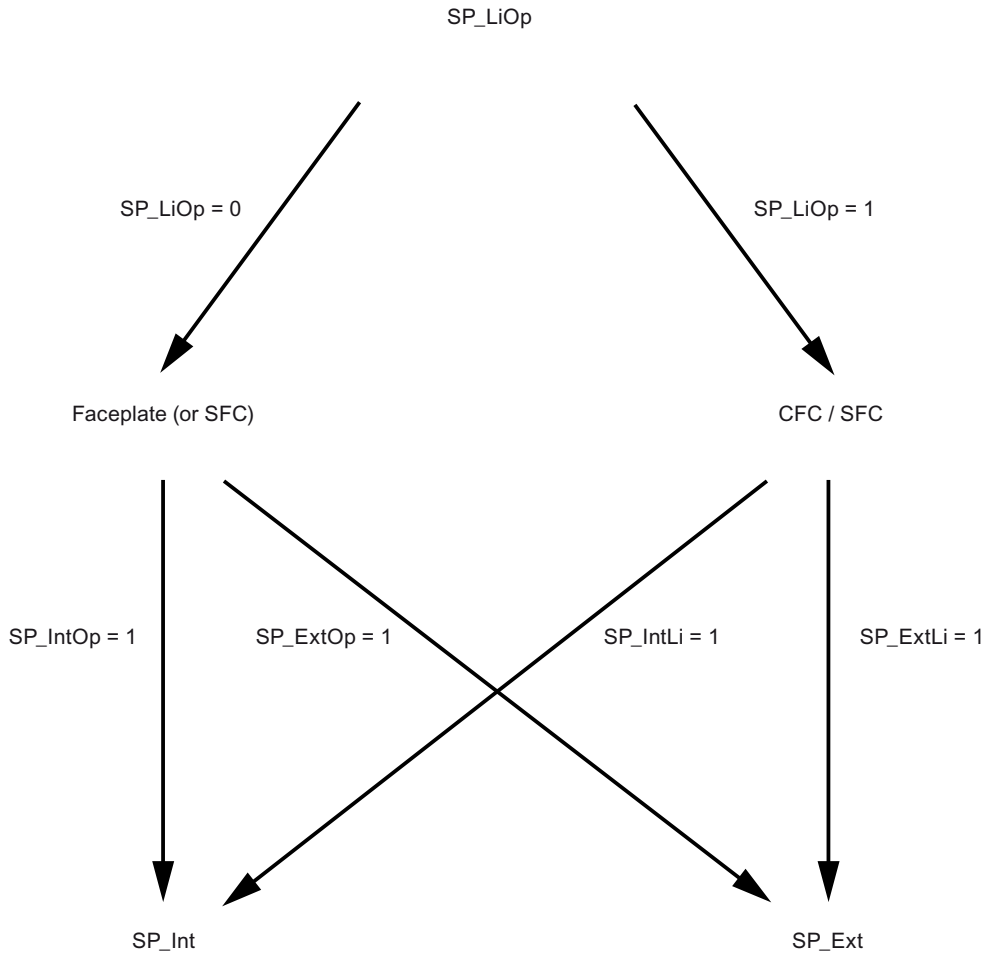
Bit = 0: Dead band width is constant

Bit = 1: The dead band is deactivated for the `PV` settling time. When the `PV` settling time is over, the dead zone gets the value of the `DeadBand` parameter.

2.1.8.2 Setpoint specification - internal/external

Setpoint specification internal & external

Some blocks have a function that allows setpoints to be specified. This specification is carried out either by means of a CFC/SFC program or by means of the faceplate (operator). With doser blocks and frequency converters, the operator can specify the internal setpoint value (`SP_Int`) or a higher-level open-loop control will specify an external setpoint value (`SP_Ext`). In principle, the blocks operate according to the same scheme:



First you define whether the setpoint specification is to be carried out by means of a CFC/SFC program or by means of the faceplate. In the next step you specify whether the internal or the external setpoint is to be used.

Setpoint specification by means of faceplate or interconnection

With the SP_LiOp parameter, you define whether the setpoint will be set by a CFC/SFC program or using the faceplate.

- Parameterize Sp_LiOp with 0 so that the setpoint specification is carried out by means of the faceplate.
- Parameterize SP_LiOp with 1 so that the setpoint specification is carried out by means of a CFC / SFC program.

Setpoint specification internal & external

You have to set the corresponding parameters depending on how the setpoint specification is to be carried out.

If the setpoint is set in the faceplate (SP_LiOp = 0), you have to set the parameter:

- SP_IntOp = 1 in order to achieve an internal setpoint specification by means of the faceplate.
- SP_ExtOp = 1 to have an external setpoint set in the faceplate.

If both signals are set, SP_IntOp = 1 has priority.

If the setpoint is set by a CFC / SFC program (SP_LiOp =1), you have to set the parameter:

1. SP_IntLi = 1 to have an internal setpoint set by a CFC / SFC program.
2. SP_ExtLi = 1 in order to achieve an external setpoint specification by means of a CFC / SFC program.

Note

For PIDConL, PIDStepL, FmCont, FmTemp: If both signals are set, SP_IntLi = 1 has priority.

Bumpless switchover from external to internal setpoint

The parameter SP_TrkExt = 1 is used so that the internal setpoint tracks the external setpoint to achieve a Bumpless switchover from the external to the internal setpoint. This allows unwanted jumps at the output parameter to be avoided.

2.1.8.3 Manipulated variable specification - internal/external

Manipulated variable specification internal and external

The VlvAnl block provides a function for specifying manipulated variables. This specification is carried out either using a CFC/SFC program or using a faceplate (operator).

If a auxiliary valve is used for operation, it is possible to switch between internal and external in the both manual and automatic mode.

If no auxiliary valve is used for operation, the external manipulated variable is used for automatic mode and the internal manipulated variable is used for manual mode. It is not possible to switch between internal and external.

Note

If no auxiliary valve is used and either the "Open" or "Close" command is active, a new manipulated variable (internal or external) only takes effect after the change. In automatic mode, the manipulated variable is ignored as long as the "Open" or "Close" command is pending.

Manipulated variable specification using a faceplate or interconnection

You can use the `MV_LiOp` parameter to determine if the manipulated variable should be set by a CFC/SFC program or via the faceplate.

- Set `MV_LiOp` to 0 for manipulated variable specification to be performed with the faceplate.
- Set `MV_LiOp` to 1 for manipulated variable specification to be performed by a CFC/SFC program.

Manipulated variable specification internal and external

You need to set the corresponding parameters depending on the selected method for manipulated variable specification.

If the manipulated variable is to specified via the faceplate `MV_LiOp = 0`, you have to set the parameter:

- `MV_IntOp = 1` in order to achieve internal manipulated variable specification via the faceplate.
- `MV_ExtOp = 1` in order to achieve external manipulated variable specification via the faceplate.

If both signals are set, the most recently active state is retained.

If the manipulated variable is set by a CFC/SFC program (`MV_LiOp = 1`), you have to set the parameter:

- `MV_IntLi = 1` to have an internal manipulated variable set by a CFC/SFC program.
- `MV_ExtLi = 1` in order to achieve an external manipulated variable specification by a CFC/SFC program.

If both signals are set, `MV_IntLi = 1` has priority.

Note

It is only possible to switch between internal and external when operating with a auxiliary valve.

Bumpless switchover of the manipulated variable from external to internal

The parameter `SP_TrkExt = 1` is used so that the internal setpoint tracks the external setpoint to achieve a Bumpless switchover from the external to the internal setpoint. This allows unwanted jumps at the output parameter to be avoided.

Forming the manipulated variable externally

With external manipulated variable specification, the manipulated variable is restricted to the `MV_HiLim` and `MV_LoLim` limits and sent to the `MV_ExtOut` output.

2.1.9 Configurable response using the Feature I/O

2.1.9.1 Configurable functions using the Feature I/O

Configurable functions using the `Feature` I/O

Some blocks have an input called `Feature`. This input can be used to influence the way in which the block works.

The `Feature` bits are assigned in the following order:

Bit number	Meaning	Block
0	Set startup characteristics (Page 137)	AV, Average, CountOh, CountScL, DeadTime, Derivative, DoseL, FmCont, FmTemp, Integral, Lag, MeanTime, ModPreCon, MonAnL, MonAnS, MonDi08, MonDiL, MonDiS, MotL, MotS, MotRevL, MotSpdCL, MotSpdL, OpAnL, Pcs7AnOu, Pcs7DiOu, PIDConL, PIDConR, PIDStepL, RateLim, Ratio, ShrdResS, TotalL, Vlv2WayL, VlvAnL, VlvMotL, VlvL, VlvS
1	Reaction to the out of service mode (Page 176)	ConPerMon, CountOh, CountScL, DoseL, Event, EventNck, EventTs, FmCont, ModPreCont, MonAnL, MonAnS, MonDi08, MonDiL, MonDiS, MotL, MotS, MotRevL, MotRevSpdL, OpAnL, OpDi01, OpDi03, OpTrig, PIDConL, PIDConR, PIDStepL, Ratio, SelA16In, TotalL, Vlv2Way, VlvAnL, VlvL, VlvS, VlvMotL

Bit number	Meaning	Block
2	Resetting the commands for changing the mode (Page 160)	DoseL, FmCont, FmTemp, ModPreCon, MotL, MotS, MotRevL, MotSpdCL, MotSpdL, PIDConL, PIDConR, PIDStepL, VIv2WayL, VIvAnL, VIvL, VIvS, VIvMotL, SelA16In
	Separate evaluation for excluded and simulated interlock signals (Page 151)	Intlk02, Intlk04, Intlk08, Intlk16, OpDi01, OpDi03
3	Enabling resetting of commands for the control settings (Page 160)	DoseL, MotL, MotS, MotRevl, MptSpdCL, MotSpdL, VIv2WayL, VIvAnL, VIvL, VIvMotL
	Control priority in the event of an invalid input command (Page 174)	VIvS
4	Setting switch or button mode (Page 166)	DoseL, FMCont, FMTemp, ModPreCon, MotL, MotRevL, MotSpdCL, MotSpdL, PIDConL, PIDStepL, VIv2WayL, VIvAnL, VIvL, VIvMotL, SelA16In
	Setting switch or button mode for local commands (Page 180)	MotL, MotRevL, MotSpdCL, MotSpdL, VIvL, VIv2WayL, VIvMotL, VIvPosL
5	Specifying the dosing type (Page 145)	DoseL
	Control via auxiliary valve (Page 171)	VIvAnL
	Alarm setpoint difference (Page 170)	MotSpdCL
	Specifying switching mode (Page 167)	MotSpdL
	Use the last value following a complete download as the current value during startup of the block (Page 153)	TotalL
	Display only input values that are interconnected in the faceplate (Page 156)	SelA16In
	Activate OS_Perm bits (Page 156)	Intlk02, Intlk04, Intlk08, Intlk16
	Limit output Out (Page 143)	Polygon
	Setting the scaling for the process values (Page 147)	FbDrive
	Activate LowCutOff (Page 154)	Pcs7AnIn
	Evaluation of the signal status of the interlock signals (Page 141)	OpDi01, OpDi03
		DoseL, MotL,MotRevL,MotS, MotSpdCL,MotSpdL,VIv2WayL, VIvAnL,VIvL,VIvS, VIvMotL
	Retain last output value in case of bad input signal status (Page 183)	MeanTime

Bit number	Meaning	Block
6	Resetting the dosing quantity when dosing starts (Page 161)	DoseL
	Block as summing unit or integrator (Page 142)	TotalL
	Disabling opening and closing (Page 157)	VlvAnL
	Substitute value switch in the event of an error (Page 148)	Polygon
	Failure handling (Page 136)	FbDrive
	Vibrate after torque monitoring (Page 177)	VlvPosL
	Ramp rate calculation (Page 178)	PIDConL, PIDConR, PIDStepL, MotSpdCL, OpAnL, FMTemp, FMCont
	External/internal selection specification (Page 179)	SelA16In
	Operator can change the setpoint via faceplate also in the "Local" mode (Page 182)	MotSpdCL
7	Enabling direct changeover between forward and reverse (Page 145)	MotRevL, MotSpdCL
	Summing characteristics continuous or triggered (Page 173)	TotalL
	Activating calculation of the flow rate for dosing by scale (Page 143)	DoseL
	Ramp rate calculation (Page 178)	VlvAnL
	Analog input 1 is reserved for the operator (Page 182)	SelA16In
	Define the setpoint after stop and start of the motor (Page 183)	MotSpdCL
8	Unit for the rate of change (Page 146)	RateLim
	Sealing the valve (Page 175)	DoseL
	Reporting with BATCH parameters (Page 155)	Event, EventTS
	Sealing the valve (Page 175)	VlvMotL
	Switch to substitute value (Page 148)	FlowCorr
	Separate delay times for each alarm (Page 169)	MonAnL, PIDConL, PIDConR
	Forcing operating modes in the "Local" mode (Page 183)	MotL, MotRevL, MotSpdCL, MotSpdL, VlvL, Vlv2WayL, VlvMotL, VlvPosL, DoseL
	Inverter enable (Page 186)	MotSpdCL
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)	DoseL, MotL, MotS, MotRevL, MotSpdCL, MotSpdL, Vlv2WayL, VlvAnL, VlvL, VlvS, VlvMotL
	Substitution value is active if the block is in bypass (Page 184)	MonAnL, MonDiL, PIDConL, PIDConR, PIDStepL, TotalL
	Enable external message (Page 184)	EventTs, Event16Ts

Bit number	Meaning	Block
10	Exiting local mode (Page 176)	DoseL, MotL, MotS, MotRevL, MotSpdCL, MotSpdL, Vlv2WayL, VlvAnL, VlvL, VlvS, VlvMotL
	Condition monitoring information at MOD_Blocks (Page 144)	FbAnIn, FbAnOu, FbDiIn, FbDiOu, FbAnTot, Pcs7AnIn, Pcs7AnOu, Pcs7DiIn, Pcs7DiOu, Pcs7Cnt1, Pcs7Cnt2, Pcs7Cnt3, FmCont, FmTemp
	Condition monitoring information at the channel blocks (Page 144)	FbAnIn, FbAnOu, FbAnTot, FbDiIn, FbDiOu, FbDrive, FbEnMe, FbSwtMMS, Pcs7AnIn, Pcs7AnOu, Pcs7Cnt1, Pcs7Cnt2, Pcs7Cnt3, Pcs7DiIn, Pcs7DiIT, Pcs7DiOu, FmCont, FmTemp
	Considering bad quality of automatic commands or external values (Page 185)	MotL, MotS, MotRevL, MotSpdCL, MotSpdL , Vlv2WayL, VlvL, VlvS, VlvMotL, VlvPosL, VlvAnL, DoseL, OpAnL, OpAnS, OpDi01, OpDi03, OpTrig
11	Stopping dosing at a flow alarm (Page 136)	DoseL
	Activating the run time of feedback signals (Page 152)	MotL, MotS, MotRevL, MotSpdL, MotSpdCL, VlvL, Vlv2WayL, VlvMotL, VlvAnL
	Gradient limitation with time duration (Page 181)	PIDConL, PIDConR, PIDStepL
12	Automatic post dosing for underdosing in automatic mode (Page 142)	DoseL
	Control zone with specified I component (Page 159)	PIDConL
	Motor feedback is not available (Page 156)	VlvMotL
	Position feedback signals are active (Page 168)	Vlv2WayL
	Gradient limitation with time duration (Page 181)	MotSpdCL, VlvAnL, OpAnL
13	Creep rate is always detected in the dosing quantity (Page 167)	DoseL
	Control zone with frozen I component (Page 159)	PIDConL
	Separate monitoring time for stopping the motor (Page 168)	MotL, MotRevL, MotSpdCL, MotSpdL, VlvMotL
14	Enabling rapid stop via faceplate (Page 167)	MotL, MotRevL, MotSpdCL, MotSpdL, VlvMotL
	External control deviation (Page 150)	PIDConL
15	Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165)	ModPreCon, PIDConL, PIDConR, PIDStepL, VlvAnL
	Frequency converter with separate device feed (Page 150)	MotSpdL
	Frequency converter with separate device feed (Page 150)	DoseL
	Motor stop in end position depends only on the corresponding feedback signal (Page 184)	VlvMotL

Bit number	Meaning	Block
16	Neutral position manipulated variable takes effect at startup (Page 165)	ModPreCon, PIDConL, PIDConR, PIDStepL, VlvAnL
	Process value with separate scale range (Page 161)	OpAnL
	Setpoint specification with own scale and unit of the parameter (Page 170)	MotSpdCL
	Separate interlock for each direction or position (Page 169)	MotRevL
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)	DoseL, MotL, MotRevL, MotSpdCL, MotSpdL, Vlv2WayL, VlvAnL, VlvL, VlvMotL
	With accelerated return of the integral action from the manipulated variable limit (Page 179)	PIDConR
18	Disabling bumpless switchover to automatic mode for controllers (Page 172)	PIDConL, PIDConR, PIDStepL
	Activating error state for external process control error CSF (Page 150)	DoseL, MotL, MotS, MotRevL, MotSpdCL, MotSpdL, Vlv2WayL, VlvAnL, VlvL, VlvS, VlvMotL
19	Enabling program mode (Page 158)	PIDConR
	Reset even with locked state (Page 164)	MotL, MotS, MotRevL, MotSpdCL, MotSpdL, VlvMotL
20	Enabling bumpless change to the proportional gain, derivative time and amplification of the differentiator (Page 171)	PIDConR
	Disable calculation of impulse control in LocalSetting 2 and 4 (Page 185)	MotL, MotRevL, MotSpdCL, MotSpdL, VlvL, VlvMotL, VlvPosL, Vlv2WayL
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)	MotL, MotRevL, MotSpdCL, MotSpdL, VlvMotL, VlvL, Vlv2WayV, VlvAnl, DoseL
	Switching operator controls for external setpoint to visible (Page 143)	PIDConR
	First-in detection response to deactivation (Page 175)	Intlk02, Intlk04, Intlk08, Intlk16
22	Update acknowledgment and error status of the message call (Page 159)	AssetM, AV, ConPerMon, CountOh, CoutnScL, DoseL, Event, EventTs, FmCont, FmTemp, MonAnL, MonDi08, MonDiL, MotL, MotRevL, MotSpdCL, MotSpdL, OpAnL, PIDConL, PIDConR, PIDStepL, TotalL, Vlv2WayL, VlvAnL, VlvL, VlvMotL
23	Evaluation of signal status (Page 141)	Intlk02, Intlk04, Intlk08, Intlk16
	Specifying the influence of the signal status on the dosing process (Page 146)	DoseL
	SP following PV in open loop has no priority over SP_Ext and SP limits (Page 178)	PIDConL, PIDConR, PIDConS, PIDStepL, FmTemp, FmCont

Bit number	Meaning	Block
24	Enabling local operator authorization (Page 157)	ConPerMon, CountOh, CountScL, DoseL, FmCont, FmTemp, GainSched, Intlk02, Intlk04, Intlk08, Intlk16, ModPreCon, MonAnL, MonAnS, MonDi08, MonDiL, MonDiS, MotL, MotS, MotRevL, MotSpdCL, MotSpdL, OpAnL, OpDi01, OpDi03, OpTrig, PIDConL, PIDConR, PIDStepL, Ratio, SelA16In, TotalL, Vlv2WayL, VlvAnL, VlvL, VlvS, VlvMotL
	With acknowledge overdosage (Page 178)	DoseL
25	Suppression of all messages (Page 173)	ConPerMon, DoseL, FmCont, FmTemp, MonAnL, MonAnS, MonDiL, MonDiS, MotL, MotS, MotRevL, MotSpdCL, MotSpdL, PIDConL, PIDConR, PIDStepL, Vlv2WayL, VlvAnL, VlvL, VlvS, VlvMotL
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)	ConPerMon, FmCont, FmTemp, PIDConL, PIDConR, PIDStepL, AV, MotL, MotRevL, MotSpdL, VlvMotL, MotSpdCL, VlvAnL, DoseL, CountOh, CountScL, TotalL, MonAnL, MonAnS, CntOhSc, PIDConS
27	Selecting values associated with messages (Page 155)	Event, EventNck, EventTs
	Interlock display with LocalSetting 2 or 4 (Page 177)	MotL, MotS, MotRevL, MotSpdCL, MotSpdL, Vlv2WayL, VlvAnL, VlvL, VlvS, VlvMotL
	Enable configuration of the dribbling quantity (Page 158)	DoseL
28	Output invalid raw value (Page 173)	Pcs7AnIn, Pcs7DiIn, Pcs7DiIT
	Disabling operating points (Page 144)	AssetM, AV, ConPerMon, CountOh, CountScL, DoseL, FmCont, MonAnL, PIDConL, PIDStepL, VlvAnL, FmTemp, PIDConR, MotL, MotRevL, MotSpdCL, MotSpdL, TotalL, VlvMotL
	Reading messages (Page 146)	FbDrive
29	Output substitute value if raw value is invalid (Page 148)	FbAnIn, FbDiIn, Pcs7AnIn, Pcs7DiIn, Pcs7DiIT
	Signaling limit violation (Page 169)	AV, ConPerMon, CountOh, CountScL, DoseL, FmCont, FmTemp, MonAnL, MotL, MotRevL, MotSpdCL, MotSpdL, PIDConL, PIDConR, PIDStepL, TotalL, VlvAnL, VlvMotL
	Transmission of status information of devices (Page 174)	FbDrive, FbSwtMMS

Bit number	Meaning	Block
30	Outputting last valid value if raw value is invalid (Page 153)	FbAnIn, FbDiIn, Pcs7AnIn, Pcs7DiIn, Pcs7DiIT
	Outputting a de-energized value for block-external simulation (Page 147)	FbAnOu, FbDiOu, Pcs7AnOu, Pcs7DiOu, FbDrive, FbSwtMMS
	Set reset depending on the operating mode or the LiOp parameter (Page 162)	DoseL, MotL, MotS, MotRevL, MotSpdL, MotspdCL, VlvL, VlvS, VlvMotL, VlvAnL, Vlv2WayL, CountScL, CountOh, CntOhSc; TotalL
	Applying the dynamically activated dead band during the PV settling time (Page 140)	PIDConL, PIDConR
31	Activating recording of the first signal (Page 149)	Intlk02, Intlk04, Intlk08, Intlk16
	Activating reset of protection / error in manual mode (Page 164)	DoseL, MotL, MotS, MotRevL, MotSpdL, MotspdCL, VlvL, VlvS, VlvMotL, VlvAnL, Vlv2WayL

See also

Flow setpoints in percent (Page 145)

Use an internal or external setpoint for the absolute fine dosing quantity (Page 152)

Polygon functions (Page 1868)

2.1.9.2 Stopping dosing at a flow alarm

Feature bit

Number of the `Feature` bit: 11

Stopping dosing at a flow alarm

You can use this feature bit to enable stopping dosing at a flow alarm.

The default setting is 0.

Bit = 0: Disabled, dosing is not stopped when a flow alarm occurs

Bit = 1: Enabled, dosing is stopped when a flow alarm occurs

2.1.9.3 Failure handling

Feature bit

Number of the Feature Bit: 6

Failure handling

Use this Feature bit to configure "Failure handling" in the case of device failure.

The default setting is 0.

Bit = 0: Commands `Local` and `Ackn` only active

Bit = 1: All commands active

See also

Configurable functions using the Feature I/O (Page 130)

2.1.9.4 Set startup characteristics

Feature bit

Number of the `Feature` bit: 0

Set startup characteristics

With this `Feature` bit, you set the startup characteristics of the function blocks, for example, for:

- Motors, valves and controllers
- Channel blocks
- Monitoring blocks, e.g. `MonAnL` and `MonDiL`.
- Mathematical and analog logic blocks
- The OpAnL, OpAnS, OpDi01, and OpDi03 block
- The TimeTrg block
- Counter blocks
- The Average block
- The TimeTrig block
- The Trigger block

The default setting is 0.

Note

This `Feature` bit has no function in the "Out of service" operating mode. The process tag remains in the "Out of service" operating mode after a warm restart of the CPU.

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. Please take into consideration the validity of terms for user-configured message classes (Page 41).

Note

With a Run-Stop-Run transition of the CPU and internally pending messages, non-stuck-through messages with time stamps and auxiliary values beginning with `RunUpCycle` occur for blocks with the startup characteristic `Feature` bit `= 0` after expiration of the `RunUpCycle` counter in the following cases:

- Alarm, warning or tolerance messages from the operating points (motor, valve, dosing, controller and analog monitoring blocks)
- Feedback errors (motor and valve blocks)
- Output signals of digital process tags (MonDiL, MonDi08)
- Flutter limits violated (MonDiL)

The restart routines of the blocks reset the following outputs in OB100:

- Operating point outputs `xx_AH_Act`, `xx_AL_Act`, `xx_WH_Act`, `xx_WL_Act`, `xx_TH_Act`, `xx_TL_Act` or `GradHUpAct`, `GradHDnAct`, `GradLAct`
- Feedback error outputs `MonDynErr` and `MonStaErr`
- Output binary signals `Out`, `Out1..8` for MonDiL or MonDi08
- Flutter suppression `FlutAct` for MonDiL

This causes an outgoing message when initializing Alarm8_P in OB100 and an incoming message after expiration of the `RunUpCycle` counter on the cyclic interrupt level.

Note

During a complete download with AS stop, the blocks (with `Feature.Bit0 = 1`) cannot resume operation in their previous mode and control when restarted.

Set startup characteristics for motors, valves and controllers

Bit = 0: Starting the block in manual mode and in neutral position. With controllers, the setpoint is set to (`SP_Int`) internally. See also the section Neutral position for motors, valves and controllers (Page 48) for more on this.

The controller blocks PIDConL, PIDConS, PIDConR, PIDStepL, ModPreCon, MPC10x10, and the valve block VlvAnL have the additional `Feature` bit 16, Neutral position manipulated variable takes effect at startup (Page 165), which specifies whether the neutral position is approached

Bit = 1: Starting the block with the last stored values, in other words in the last operating mode set (manual, automatic or local mode) and at the last valid position.

Note

Special note following complete download to the CPU

Following a complete download to the CPU, the motor protection signal `Trip` is evaluated during the initial run as good (=1).

When a motor protection signal is pending, this causes a non-struck-through message with time stamp and auxiliary values beginning with `RunUpCycle` after the complete download and after expiration of the `RunUpCycle` counter.

Startup characteristics for the ShrdResS block

Bit = 0: The output command interface is reset to 0.

Bit = 1: The block leaves the output command interface unchanged.

Defining the startup characteristics for channel blocks

Bit = 0: The channel block uses either the process value `PV_In` or the value of `SimPV_In` as the startup value, depending on the setting of the input parameter `SimOn` (`PV_In = PV_Out` or `SimPV_In = PV_Out`).

Bit = 1: The channel block uses the value `StartVal` as the startup value (`StartVal = PV_Out`).

Defining the startup characteristics for monitoring blocks

Bit = 0: The most recently stored values are reset on startup.

Bit = 1: The most recently used value at the output parameter `Out` is output on startup.

Defining the startup characteristics for mathematical, analog logic blocks

Bit = 0: The `Out` output parameter is reset to 0 on startup.

Bit = 1: The most recently saved value is output at the `Out` output parameter on startup.

Defining startup characteristics for the OpAnL/OpAnS blocks

Bit = 0: The internal setpoint is used for startup.

Bit = 1: The most recently saved value is output at the `Out` output parameter on startup.

Defining startup characteristics for the OpDi01/OpDi03 blocks

Bit = 0: The output parameters `Out` or `Out1`...`Out3` will be reset on startup.

Bit = 1: The most recently used value at the output parameter `Out` or `Out1`...`Out3` is outputted on startup.

Defining the startup characteristics for counter blocks

Bit = 0: On startup, the counter is stopped and reset to the value specified in the input parameter.

Bit = 1: On startup, counting continues with the most recently stored value.

Input parameters for the startup characteristics of the counter blocks:

- Block CountOh: Input parameter `PresetTime`
- Block CountScL: Input parameter `PresetVal`
- Block TotalL: Input parameter `PresetVal`

Messages can be suppressed for a short time after startup. You can set the number of cycles using the input parameter `RunUpCyc`.

Note

Advanced configuration of the startup characteristics for the counter blocks

Note that you can further affect the startup characteristics via the `Feature Bit 5` as a function of this `Feature Bits 0`. Refer to the section: Use the last value following a complete download as the current value during startup of the block (Page 153).

Defining startup characteristics for the Average block

Bit = 0: At startup, averaging begins with the value that is currently at the input parameter `(Out = In, NumCycles = 1)`.

Bit = 1: When starting-up, the last `Out` and `NumCycles` values saved are used as the last value for averaging `(Out ≠ In)`.

Defining startup characteristics for the TimeTrig block

Bit 0 =0: Periodic trigger and single trigger can be shut off. The trigger pulse `Trigger` is reset.

Bit 0 =1: The activation of the periodic trigger and single trigger as well as the single trigger point are retained.

The InPerTrigOn, InSglTrigOn and InSglTrigDT inputs are applied for this. If you want to use the reset for a complete download, you must read back the marked parameters in addition to the operated and monitored parameters before a complete download.

See also

User-configured message classes (Page 41)

2.1.9.5 Applying the dynamically activated dead band during the PV settling time

`Feature` bit

Number of the `Feature` bit: 30.

Dynamic deactivation of the dead band by the PV settling time

You can use this Feature bit to improve the PV settling time in the range of the dead band.

The deadband is deactivated until the PV settles more in the center than at the edges of the dead band. The probability that the controlled variable is within the dead zone time is increased, and further control actions are no longer necessary. This leads to reduced wear and less energy consumption.

Bit = 0: Dead band width is constant

Bit = 1: The dead band is deactivated for the PV settling time. When the PV settling time is over, the dead zone gets the value of the DeadBand parameter.

2.1.9.6 Evaluation of signal status

Feature bit

Number of the Feature bit: 23

Evaluation of signal status

You can use the Feature bit to specify if the signal status of the inputs is to be checked for the values 16#00 or 16#28. The signal status of the inputs itself remains unchanged here.

The default setting is 0.

Bit = 0: No evaluation of the signal status for 16#00 or 16#28.

Bit = 1: the signal status is determined, an input with ST = 16#00 or 16#28 is forwarded with value = 0. The "Negate signal" function at the input of the block has no influence on the reaction in this case.

2.1.9.7 Evaluation of the signal status of the interlock signals

Feature / Feature2 bit

Number of the Feature / Feature2 bit: 5

Family

Family: Operate number of the Feature bit: 5

Family: Control, drives, dosage number of the Feature2 bit: 5

Evaluation of the signal status of the interlock signals

You can use the Feature / Feature2 bit to specify if the signal status of the interlock inputs is to be checked for the values 16#00 or 16#28. The signal status of the inputs itself remains unchanged here.

The default setting is 0.

Bit = 0: the signal status is determined, an input with `ST = 16#00` or `16#28` is forwarded with value = 0.

Bit = 1: No evaluation of the signal status for `16#00` or `16#28`.

2.1.9.8 Automatic post dosing for underdosing in automatic mode

Feature bit

Number of the `Feature` bit: 12

Automatic post dosing for underdosing in automatic mode

Use this `Feature` bit to enable automatic post dosing for underdosing in automatic mode.

The default setting is 0.

Bit = 0: Disabled, no automatic post dosing is started for underdosing in automatic mode.

Bit = 1: Enabled, automatic post dosing is started for underdosing in automatic mode.

2.1.9.9 Block as summing unit or integrator

Feature bit

Number of the `Feature` bit: 6

Specifying the summing unit or integrator function mode

You can define the summing response of the block via this `Feature` bit.

The default setting is 0.

Bit = 0: The block operates as a summing unit.

Bit = 1: The block operates as an integrator.

Note

Special note for the summing unit

Note that the characteristics of the block as a summing unit `(Feature Bit 6 = 0)` can be further affected via the `Feature Bit 7` .

Refer to the section: Summing characteristics continuous or triggered (Page 173).

2.1.9.10 Switching operator controls for external setpoint to visible

Feature bit

Number of the `Feature` bit: 21

Switching operator controls for external setpoint to visible

Use this `Feature` bit to switch all operator controls for the external setpoint to visible in the faceplates for the OS operator. This is always required when the external setpoint should actually be used, for example, for slave controllers in cascade and ration controlling.

The default setting is 0, which keeps the faceplate as clear as possible for simple applications.

Bit = 0: The operator controls for the external setpoint are **not visible** in the faceplate.

Bit = 1: The operator controls for the external setpoint are **visible** in the faceplate.

2.1.9.11 Limit output Out

Feature bit

Number of the `Feature` Bit: 5

Limit output `Out`

Use this `Feature` bit to define the limit for the output `Out`.

The default setting is 0.

Bit = 0: Beyond the end interpolation points, the `Out` output is extrapolated based on the first two or last two interpolation points.

Bit = 1: The `Out` output is limited to the first and last interpolation point.

See also

Configurable functions using the Feature I/O (Page 130)

2.1.9.12 Activating calculation of the flow rate for dosing by scale

Feature bit

Number of the `Feature` bit: 7

Calculation of the flow rate for dosing by scale

Use this `Feature` bit to activate calculation of the flow rate for dosing by scale.

The default setting is 0.

Bit = 0: Deactivated

Bit = 1: Calculation activated.

The flow is determined by the change to the dosing quantity per second.

2.1.9.13 Condition monitoring information at MOD_Blocks

Feature bit

Number of the `Feature` bit: 10

Bit = 0: Not forwarded

Bit = 1: forwarded

2.1.9.14 Condition monitoring information at the channel blocks

Feature bit

Number of the `Feature` bit: 10

Transferring condition monitoring information to the Maintenance Station

With this `Feature` bit, the data which is entered at the parameters `MS_Ext` and `TextRef` will be transferred to the corresponding driver block and the same data will be displayed on the asset faceplate of the Maintenance Station. For more information, refer to the *Maintenance Handbook*.

The default setting is 0.

Bit = 0: Data will not be transferred.

Bit = 1: Data will be transferred.

2.1.9.15 Disabling operating points

Feature bit

Number of the `Feature` bit: 28

Disabling operating points

You can use this `Feature` bit to determine if the operating point function of a limit for disabling the message (`MsgLock = 1`) should also be disabled.

The default setting is 0.

Bit = 0: Operating point is not suppressed

Bit = 1: Operating point is suppressed

2.1.9.16 Enabling direct changeover between forward and reverse

Feature bit

Number of the `Feature` bit: 7

Direct changeover between forward and reverse

With this `Feature` bit, you can enable direct reversal of the direction of motors.

The default setting is 0.

Bit = 0: Direct reversal of the direction is disabled.

You can only change the direction of the motor by first stopping and starting the motor again in the required direction. The motor can only be started again after the time set in the `IdleTime` parameter has elapsed.

Bit = 1: Direct changeover is enabled.

You can reverse the motor direction directly. The motor block reverses the direction automatically. The motor is stopped and is started in the other direction when the time set in the `IdleTime` parameter has elapsed.

2.1.9.17 Specifying the dosing type

Feature bit

Number of the `Feature` bit: 5

Specifying the dosing type

You can specify the dosing type to be used for the block using this `Feature` bit.

The default setting is 0.

Bit = 0: Flow

Bit = 1: Scales

2.1.9.18 Flow setpoints in percent

`Feature` bit

Number of the `Feature` bit: 15

Flow setpoints in percent

You can use this `Feature` bit to specify if the doser block should display the flow setpoints in the faceplate as percentages.

The default setting is 0.

Bit = 0: Deactivated. The display of the flow setpoints is made in the unit specified with the `PV_Unit` parameter.

Bit = 1: Activated: The display of the flow setpoints is made in the unit %.

2.1.9.19 Specifying the influence of the signal status on the dosing process

Feature bit

Number of the `Feature` bit: 23

Specifying the influence of the signal status on the dosing process

You can use this `Feature` bit to specify how the doser should react dependent on the signal status.

The default setting is 0.

Bit = 0: Dosing process does not stop with bad signal status of the process value `PV`.

Bit = 1: Dosing process stops with bad signal status of the process values `PV`. The doser is also set to the "Off" state (see state diagram: DoseL functions (Page 997)).

See also the following section:

- Forming and outputting signal status for blocks (Page 108)

2.1.9.20 Unit for the rate of change

Feature bit

Number of the `Feature` bit: 8

Specifying the unit for the rate of change

You can use this `Feature` bit to specify the unit for the rate of change:

The default setting is 0.

Bit = 0: unit for the rate of change in the unit of measurement to or from the field device

Bit = 1: unit for the rate of change as a percentage to or from the field device

2.1.9.21 Reading messages

Feature bit

Number of the `Feature` bit: 28

Use of the input data

You can use this `Feature` bit to specify the format for reading messages.

Default setting is 0

Bit = 0: Messages are read as PV freely configurable

Bit = 1: Messages are read as MsgNamur

Note

This feature bit is only used when "Message frame type 20" and the "PZD 6" parameter are active.

2.1.9.22 Setting the scaling for the process values

Feature Bit

Number of the Feature Bit: 5

Calculation for the outputs CurrentLi, Power1Li, Power2Li, FreeLi

Feature Bit 5 defines whether the PZDIN**X**Scale is used for the calculation for the other process values.

Bit = 0: **xxxx**.Value := (PZD**X**In*(100/16384.0));

Bit = 1: **xxxx.**Value := (PZD**X**In*((PZDIn**X**Scale.HIGH- PZDIn**X**Scale.LOW)/16384.0))+ PZDInXScale.LOW;

2.1.9.23 Outputting a de-energized value for block-external simulation

Feature bit

Number of the Feature bit: 30

Outputting a de-energized value for block-external simulation

Use this `Feature` bit for channel blocks to specify which value is to be output during the block-external simulation (Simulating signals (Page 58)).

The default setting is 1.

Bit = 0: If process value input has the status `16#60`, the output value will be set to the input value.

Bit = 1: If process value input has the status `16#60`, the output value will be set to “0”.

2.1.9.24 Switch to substitute value

Feature bit

Number of the `Feature` bit: 8

Switch to substitute value

You can use this `Feature` bit to specify whether a substitute value should be applied or the last valid value should be retained when certain conditions occur.

The default setting is 0.

Bit = 0: Retain last valid value

Bit = 1: Apply the substitute value

See also

Output invalid raw value (Page 173)

Outputting last valid value if raw value is invalid (Page 153)

2.1.9.25 Substitute value switch in the event of an error

Feature bit

Number of the `Feature` bit: 6

Substitute value switch in the event of an error

You can specify the substitute value activation at errors using this `Feature` bit.

The default setting is 0.

Bit = 0: No substitute value switch

Bit = 1: The substitute value `SubV_In` is output at `Out` for `ErrorNum` ≠

See also

Configurable functions using the Feature I/O (Page 130)

2.1.9.26 Output substitute value if raw value is invalid

Feature bit

Number of the `Feature` bit: 29

Output substitute value if raw value is invalid

Use this `Feature` bit to activate output of the substitute value (input parameter `SubsPV_In`) for channel blocks if there is an invalid raw value or the device is in the initialization phase (input parameter `PV_ST` 16#4C..4F).

The default setting is 0.

Bit = 0: If there is an invalid raw value or the device is in the initialization phase, the substitute value is not output.

Bit = 1: If there is an invalid raw value or the device is in the initialization phase, the substitute value is output. The signal status of the output value is set to "Manipulated value (for example, substitute value, simulation, last valid value).

If there is an invalid raw value or the device is in the initialization phase, the output parameter `Bad = 1` is set automatically.

Prioritizing the `Feature` bits for channel blocks:

You need to assign parameters for three `Feature` bits for the response to an invalid raw value for the channel blocks.

If more than one of these `Feature` bits are set (=1), the following priority applies:

- Output invalid raw value (Page 173) (`Feature` bit 28 = highest priority)
- Output substitute value if raw value is invalid (`Feature` bit 29)
- Outputting last valid value if raw value is invalid (Page 153) (`Feature` bit 30 = lowest priority)

The raw value is output of none of the `Feature` bits 28, 29 or 30 is set.

2.1.9.27 Activating recording of the first signal

Feature Bit

Number of the `Feature` bit: 31

Activating recording of the first signal

Use this `Feature` bit to activate recording of the first signal with interlock blocks. Please also refer to the section Recording the first signal for interlock blocks (Page 52).

Default setting is 0

Bit = 0: Recording of the first signal is deactivated.

Bit = 1: Recording of the first signal is activated.

2.1.9.28 External control deviation

Feature bit

Number of the `Feature` bit: 14

External control deviation

You can use this `Feature` bit to specify whether the external control deviation is to be activated.

The default setting is 0.

Bit = 0: The external control deviation is deactivated, the internal control deviation is active.

Bit = 1: The external control deviation is activated.

2.1.9.29 Activating error state for external process control error CSF

Feature bit

Number of the `Feature` bit: 18

Activating error state for external process control error CSF

You can use this `Feature` bit to specify whether the block should switch to the error state at an external process control error CSF =1.

The default setting is 0.

Bit = 0: : Block does not go to error state with process control error CSF = 1.

Bit = 1: Activated: Block goes to error state with process control error CSF = 1.

2.1.9.30 Frequency converter with separate device feed

Feature bit

Number of the `Feature` bit: 15

Frequency converter with separate device feed

You can use this `Feature` bit to specify if a frequency converter with a separate device feed or the inverter enable should be used.

The default setting is 0.

Bit = 0: Motor without device feed or without inverter enable

Bit = 1: Frequency converter with separate device feed or with inverter enable

2.1.9.31 Separate evaluation for excluded and simulated interlock signals

Feature bit

Number of the `Feature2` bit: 2

Separate evaluation for excluded and simulated interlock signals

You can use this `Feature` bit to set the reaction of the block to an interlock signal of an interlock block that is excluded or simulated.

The default setting is 0

Bit = 0: Switch-relevant excluded and simulated interlock signals are processed with the status 16#60 and are displayed as simulated or excluded, depending on the interlock status.

Bit = 1: Excluded and simulated interlock signals are evaluated separately. All excluded interlock signals in a sequence of interlock blocks are displayed as excluded. Switch-relevant simulated interlock signals are processed with the status 16#60 and displayed as simulated.

Note

If the `Feature` Bit is enabled, the signal cannot be inverted via CFC during interconnections at the inputs, as, otherwise, the bypass display would not update.

The interlocking inputs of the following blocks are affected:

- **Intlk02**, **Intlk04**, **Intlk08**, **Intlk16**: `In01`, `In02`, ...
- **DoseL**, **MotL**, **MotRevL**, **MotSpdL**, **MotSpdCL**, **VlvL**, **Vlv2WayL**, **VlvAnL**, **VlvMotL**, **VlvPosL**: `Permit`, `Interlock`, `Protect`
- **MotRevL**: `Permit`, `Interlock`, `Protect` , `PermRev`, `IntlRev`, `ProtRev`
- **MotS**, **VlvS**, **OpDi01**, **OpDi03**, **PIDConL**, **PIDConR**: `Interlock`

The inversion of the interlocking signals occurs via the `InvIn01`, `InvIn02`, ... inputs on the interlocking blocks.

See also

OpDi01 functions (Page 373)

OpDi03 functions (Page 386)

Intlk02 functions (Page 1548)

Intlk04 functions (Page 1560)

Intlk16 functions (Page 1588)

Intlk08 functions (Page 1573)

MotL functions (Page 1064)

MotS functions (Page 1096)

MotRevL functions (Page 1122)

MotSpdL functions (Page 1215)

Vlv2WayL functions (Page 1295)

VlvL functions (Page 1335)

VlvMotL functions (Page 1392)

VlvAnL functions (Page 1437)

MotSpdCL functions (Page 1162)

VlvS functions (Page 1366)

DoseL functions (Page 997)

Influence of the signal status on the interlock (Page 103)

Forming the group status for interlock information (Page 104)

Forming and outputting the signal status for interlock blocks (Page 115)

2.1.9.32 Use an internal or external setpoint for the absolute fine dosing quantity

Feature Bit

Number of the `Feature` bit: 8

Use an internal and external setpoint for the absolute fine dosing quantity

Use this `Feature` bit to determine whether the doser processes the internal and external setpoint for the fine dosing quantity in an absolute manner and is displayed or operated in an absolute manner in the faceplate.

The default setting is 0.

Bit = 0: Deactivated. The internal and external I/Os as well as the display and operation of the fine dosing quantity setpoint are processed in the % unit.

Bit = 1: Activated: The internal and external I/Os as well as the display and operation of the fine dosing quantity setpoint are processed in the unit that was set using the parameter `DQ_Unit`.

2.1.9.33 Activating the run time of feedback signals

Feature Bit

Number of the `Feature` bit: 11

Activating the run time of feedback signals

Use this `Feature` bit to activate the run time of feedback signals.

The default setting is 0.

Bit = 0: Deactivated: Tracking of feedback for simulation immediately after the trigger signal.

Bit = 1: Activated: Tracking of feedback for simulation after the trigger signal and expiration of the monitoring time (`MonTiDynamic`). The feedback signals are generated after expiration of the monitoring time.

2.1.9.34 Outputting last valid value if raw value is invalid

Feature bit

Number of the `Feature` bit: 30

Outputting last valid value if raw value is invalid

Use this `Feature` bit to activate output of the last valid value for channel blocks if there is an invalid raw value or the device is in the initialization phase (input parameter PV_ST = 16#4C.. 4F).

The default setting is 0.

Bit = 0: If there is an invalid raw value or the device is in the initialization phase, the last valid value is not output.

Bit = 1: If there is an invalid raw value or the device is in the initialization phase, the last valid value is output. The signal status of the output value is set to local "Manipulated value (for example, substitute value, simulation, last valid value).

If there is an invalid raw value or the device is in the initialization phase, the output parameter `Bad = 1` is set.

Prioritizing the `Feature` bits for channel blocks:

You need to assign parameters for three `Feature` bits for the response to an invalid raw value for the channel blocks.

If more than one of these `Feature` bits are set (=1), the following priority applies:

- Output invalid raw value (Page 173) (`Feature` bit 28 = highest priority)
- Output substitute value if raw value is invalid (Page 148)(`Feature` bit 29)
- Output the last valid value if raw value is invalid (`Feature` bit 30, lowest priority)

The raw value is output of none of the `Feature` bits 28, 29 or 30 is set.

2.1.9.35 Use the last value following a complete download as the current value during startup of the block

Feature bit

Number of the `Feature` bit: 5

Use the last value during startup of the block after a complete download of the CPU

You can use this `Feature` bit to define the startup characteristics of the block as a function of the `Feature` bit `0`.

This feature bit is used on the following parameters:

- Block TotalL: `OldOut`, `OldCntOut`
- Block CountScL: `OldOut`
- Block CountOh: `OldDays`, `OldHours`, `OldMinutes`, `OldSeconds`

For the settings on `Feature` bit 0, refer to the section: Set startup characteristics (Page 137).

Note

If you want to use this function, you must read back the marked parameters in addition to the operated and monitored parameters before a complete download.

The default setting is 0.

Bit = 0: Define the startup characteristics as a function of `Feature` bit 0:

- `Feature` bit 0 = `0`: The block is set to the default value (input parameter `Preset`) during startup.
- `Feature` bit 0 = 1: On startup, counting continues with the most recently stored value.

Bit = 1: During startup of the block after a complete download, the last value `(Oldxxx)` is used as the current value regardless of `Feature` bit 0 (Set startup characteristics (Page 137)).

2.1.9.36 Activate LowCutOff

Feature bit

Number of the `Feature` bit: 5

Activate LowCutOff

You use this `Feature` bit to activate the use of LowCutOff bits.

The default setting is 0.

Bit = 0: Low signal cut-off - OFF (default)

Bit = 1: Low signal cut-off - ON

2.1.9.37 Selecting values associated with messages

Feature Bit

Number of the `Feature` bit: 27

Selecting values associated with messages

Use this `Feature` bit to select which values associated with messages are to be output.

The default setting is 0.

Bit = 0: The signal status of the binary input is output as the value associated with messages.

Bit = 1: The associated analog value is output as the value associated with messages.

2.1.9.38 Reporting with BATCH parameters

Feature bit

Number of the `Feature` bit: 8

Reporting with BATCH parameters

You can use this `Feature` bit to specify whether the block transfers the BATCH parameters

- `BatchID`: Batch ID
- `BatchName`: Batch name
- `StepNo`: Batch step number

Are transferred as associated values to the OS during messaging.

The default setting is 0.

Bit = 0: The block does not transfer any BATCH parameters to the OS.

Bit = 1: The block transfers BATCH parameters to the OS.

Note

Information about the setting "Bit = 1:

The Event block can no longer transfer the `In8.ST` or `AV8.Value` as an associated value.

The EventTS block can no longer transfer the signal status of the signal `In7` and `In8` or `InTS7` and `InTS8` as an associated value.

You will find more information in the following chapters:

- Event messaging (Page 1611)
- EventTs messaging (Page 1637)

2.1.9.39 Motor feedback is not available

Feature bit

Number of the `Feature` bit: 12

Motor feedback is not available

You can use this `Feature` bit to specify whether or not motor feedback should be available.

The default setting is 0.

Bit = 0: Motor feedback is available.

Bit = 1: Motor feedback is not available. The motor feedback is derived from the control signals internally in the block; this disables the monitoring of the motor feedback.

2.1.9.40 Display only input values that are interconnected in the faceplate

Feature bit

Number of the `Feature` bit: 5

Display only input values that are interconnected in the faceplate

You can use this feature bit to specify whether only interconnected input values `In01` to `In16` (status not equal to 16#FF) should be displayed in the faceplate.

The default setting is 0.

Bit = 0: Display all input values.

Bit = 1: Display only interconnected input values (status not equal to 16#FF) and in case inputs are not connected, the faceplate size should shrink to the last connected input.

2.1.9.41 Activate OS_Perm bits

Feature bit

Number of the `Feature` bit: 5

Display only input values that are interconnected in the faceplate

You use this `Feature` bit to activate the use of additional `OS_Perm` bits in the faceplate.

The default setting is 0.

Bit = 0: `OS_Perm` bits XXX inactive

Bit = 1: `OS_Perm` bits XXX active

With XXX:

Block	OS_Perm bit X
Intlk02	16-17
Intlk04	16-19
Intlk08	16-23
Intlk16	16-31

OS_Perm bit 16	OS_Perm bit 0	OS_Perm bit 1	Operation exclusion "Set" In01	Operation exclusion "Reset" In01
0	0	0	No	No
0	0	1	No	Yes
0	1	0	Yes	No
0	1	1	Yes	Yes
1	X	X	Yes	Yes

2.1.9.42 Disabling opening and closing

Feature bit

Number of the `Feature` bit: 6

Disabling the Open and Close commands

Only applicable when no auxiliary valve is used (bit 5 = 0)

Bit = 0: Open and Close commands affect control valve

Bit = 1: Disable the Open and Close commands

2.1.9.43 Enabling local operator authorization

Feature bit

Number of the `Feature` bit: 24

Enabling local operator authorization

You can use this `Feature` bit to enable local permission for a technologic block.

The default setting is 0.

Bit = 0: Disabled

Bit = 1: Enabled

For information on this operator permission, refer to the section Operator control permissions (Page 248).

2.1.9.44 Enable configuration of the dribbling quantity

Feature bit

Number of the `Feature` bit: 27

Enable configuration of the dribbling quantity

With this `Feature` bit, you can enable configuration of the dribbling quantity during dosing.

The default setting is 0.

Bit = 0: Disabled, the dribbling quantity can only be configured in the "End" state or with the automatic determination of the dribbling quantity (`DribbCor` = 1).

Bit = 1: Enabled, the dribbling quantity can always be configured, but not in the "Dribbling" state. The automatic determination of the dribbling quantity must be switched off (`DribbCor` = 0).

2.1.9.45 Enabling program mode

Feature bit

Number of the `Feature` bit: 19

Enabling program mode

You can use this `Feature` bit to specify whether or not the controller block should be used for program mode.

Default setting is 0.

Bit = 0: The block is not intended for program mode.

Bit = 1: The block can be used for program mode. The operator control elements required for this are then visible in the faceplate.

Note

This `Feature` bit only applies to the PIDConR block.

What is the program mode?

Program mode provides primary controller functions (external Advanced Control software package), which run on an external PC as an OPC client, the option of using the control from the controller function block and specifying the setpoint or manipulated variable from a remote location.

2.1.9.46 Update acknowledgment and error status of the message call

Feature bit

Number of the `Feature` bit: 22

Update acknowledgment and error status of the message call

You can use the `Feature` bit to determine if the acknowledgment and error status of the message call at the block output should be updated.

The default setting is 0.

- **Bit = 0**: The `MsgErr`, `MsgStat` and `MsgAckn` block outputs are set to the default settings and not updated. The block will run faster with this setting.
- **Bit = 1**: The `MsgErr`, `MsgStat`and `MsgAckn` block outputs re updated based on the feedback of the lower level message blocks. The lower level message blocks are called every other cycle as long as an acknowledgment is expected or error information is pending.

2.1.9.47 Control zone with frozen I component

Feature bit

Number of the `Feature` bit: 13

Freezing the I component for control with control zone

You can use this `Feature` bit to specify how the controller should react to control zone operation. The I component must be enabled for this TI <> 0.

The default setting is 0.

Bit = 0: The I component is set in such a way that the manipulated variable reacts bumplessly when the error signal reoccurs in the control zone.

Bit = 1: If the error signal is outside the control zone, the I component is frozen internally. When it enters the control zone, the controller starts with the value of the I component that was in effect when it left the control zone.

The reaction depends on additional `Feature` bits and controller settings. See also the following section:

- Using control zones (Page 190)

2.1.9.48 Control zone with specified I component

Feature bit

Number of the `Feature` bit: 12

Specifying I component for control with control zone

You can use this `Feature` bit to specify how the controller should react to control zone operation. The I component must be enabled for this TI <> 0.

The default setting is 0.

Bit = 0: The I component is set in such a way that the manipulated variable reacts bumplessly when the error signal reoccurs in the control zone.

Bit = 1: If the error signal is outside the control zone, the I component is set internally as follows:

- `I_Part := MV_Offset`.

The reaction depends on additional `Feature` bits and controller settings. See also the following section:

Using control zones (Page 190)

2.1.9.49 Resetting the commands for changing the mode

Feature Bit

Number of the `Feature` bit: 2

Resetting the commands for changing the mode

Using this `Feature` bit, you define how the block handles the incoming control commands:

- `SP_IntLi` and `SP_ExtLi` for controllers
- `AutModLi` and `ManModLi` for controllers, drives, and dosage block
- `SaPo_Ext` for KalFilt block
- `OffLi`, `UpLi`, and `DnLi` for TotalL block

The default setting is 0.

Bit = 0: The control commands are not reset by the block. If there are two pending control commands for changing mode, the mode is not changed. In this case, the note text "Invalid command" is displayed in the faceplate.

Bit = 1: The control commands are reset by the block. This, for example, ensures that if a control command is sent from the SFC, the command is reset automatically after a step is exited.

2.1.9.50 Enabling resetting of commands for the control settings

Feature bit

Number of the `Feature` bit: 3

Enabling resetting of commands for the control settings

With this `Feature` bit, you select how the block handles commands for the control settings (for example motor on) via the interconnected input parameters.

The default setting is 0.

Bit = 0: The control commands are not reset by the block. If there are two commands relating to the control settings at the same time, the status of the control settings is retained. In this case, the "Invalid signal" message is displayed in the standard view of the faceplate.

Bit = 1: The control commands are reset by the block. This, for example, ensures that if a control command is sent from the SFC, the command is reset automatically after a step is exited.

2.1.9.51 Resetting the dosing quantity when dosing starts

Feature bit

Number of the `Feature` bit: 6

Resetting the dosing quantity when dosing starts

Use this feature bit to enable resetting the dosing quantity when dosing starts.

The default setting is 0.

Bit = 0: Disabled, the dosing quantity is not reset when dosing starts

Bit = 1: Enabled, the dosing quantity is reset when dosing starts.

2.1.9.52 Process value with separate scale range

Feature bit

Number of the `Feature` bit: 16

Process value with separate scale range

You can use this `Feature` bit to assign a separate scale range to the process value. The default setting is 0.

Bit = 0: Apply scale range from `SP`.

Bit = 1: `PV` has a separate scale range.

See also

Neutral position for motors, valves and controllers (Page 48)

Neutral position manipulated variable takes effect at startup (Page 165)

Use the last value following a complete download as the current value during startup of the block (Page 153)

2.1.9.53 Resetting via input signals in the event of interlocking (Protection) or errors

Feature Bit

Number of the `Feature` bit: 9

Resetting the block in the event of interlocking (only Protection: Input parameter `Protect`) or errors via input signals

With this `Feature` bit, you define how automatic control is to be re-enabled after an active interlock.

The default setting is 0.

Bit = 0: After an interlock (only Protection: Input parameter `Protect`) or errors, the system can only be restarted using a reset command. Reset is initiated either by operator input in the faceplate or via the interconnectable input parameter (`RstLi = 1`) in the block. Thereafter, the currently pending command takes effect in automatic mode.

Bit = 1: It is also possible to reset with a 0-1 edge change in the control signal in automatic mode.

2.1.9.54 Set reset depending on the operating mode or the LiOp parameter

`Feature` Bit

Number of the `Feature` Bit: 30

Set reset depending on the operating mode (motor, valve and dosing blocks)

When the "Protection" interlock, feedback error ("Status error", "Control error") or "Motor protection" signal is present again, use this `Feature` bit to specify if a reset can be made depending on the mode only by the operator in manual mode or only by the automatic I/Os in automatic mode.

Resetting to manual mode is enabled with `Feature` Bit 31 (Activating reset of protection / error in manual mode (Page 164)). Also refer to the Resetting the block in case of interlocks or errors (Page 43) chapter.

The default setting is 0.

Bit = 0: Reset does not depend on the operating mode

Bit = 1: In manual mode, manual reset by the operator is only possible if `Feature` Bit 31 is set, otherwise no reset is required in manual mode.

In automatic mode, reset can only be made with automatic I/Os, regardless of `Feature` Bit 31. This is performed either with a 0-1 edge transition at the `RstLi` input or, when `Feature` Bit 9 is set, with a 0-1 edge transition at the automatic inputs, for example `OpenAut`, `CloseAut`.

Note

Rapid stop is unlocked for all operating modes using the "Reset" button in the faceplate (`RstOp = 1`); in CFC it is unlocked using the `RstLi = 1` input parameter.

Note

The local operating mode does not depend on this `Feature` Bit and has a separate reset mechanism.

Resetting the dosing mode depending on the operating mode (dosing blocks)

You can also use this feature bit to set whether or not the reset of the dosing quantity is dependent on the operating mode.

The default setting is 0.

Bit = 0: Reset does not depend on the operating mode

Bit = 1: In manual mode, only a manual reset by the operator is possible at the input `RstDQ_Op`. In automatic mode, a reset is only possible by a 0-1 edge transition at the input `RstDQ_Li`.

Note

The operating mode Local is independent of this feature bit. The dosing quantity can be reset by the operator at the input `RstDQ_Op` or via a 0-1 edge transition at the input `RstDQ_Li`.

Set reset depending on the LiOp parameter (counter blocks)

You can use this `Feature` Bit to determine whether the setting or resetting to a default value should be made dependent on the `LiOp` parameter.

The default setting is 0.

Bit = 0: Setting or resetting to a default value does not depend on the `LiOp` parameter.

Bit = 1: Setting or resetting to a default value depends on the `LiOp` parameter.

`LiOp` **=0:** Setting or resetting to a default value can only be made via the faceplate or at the parameter for the faceplate.

`LiOp` **=1**: Setting or resetting to a default value can only be made at the parameter for interconnections.

2.1.9.55 Activating reset of protection / error in manual mode

`Feature` bit

Number of the `Feature` bit: 31

Activating reset of protection / error in manual mode

Use this `Feature` bit to specify whether a reset is necessary once the "Protection" interlock signal, feedback errors ("Runtime error", "Control deviation"), or "Motor protection" are present again. See also the following section: Resetting the block in case of interlocks or errors (Page 43).

The default setting is 0.

Bit = 0: No reset required in manual mode.

Bit = 1: Reset required in manual mode. The reset is performed using the "Reset" button (`RstOp = 1`) or, in CFC, using the input parameter `RstLi`.

Note

Rapid stop is unlocked for all operating modes using the "Reset" button in the faceplate (`RstOp = 1`); in CFC it is unlocked using the `RstLi = 1` input parameter.

Note

The local operating mode has a separate reset mechanism.

2.1.9.56 Reset even with locked state

Feature Bit

Number of the `Feature` bit: 19

Reset even with locked state

With this `Feature` bit, you specify if it is possible to perform a reset with an active "Protection" or "Motor protection" type interlock. This can be used, for example, to reset hardware interlocks.

The default setting is 0.

Bit = 0: No reset is possible with a "Protection" type interlock or with active motor protection.

Bit = 1: Reset is possible with a "Protection" type interlock or with active motor protection.

2.1.9.57 Neutral position manipulated variable takes effect at startup

Feature Bit

Number of the `Feature` bit: 16

Neutral position manipulated variable takes effect at startup

You can use this `Feature` bit to specify if the block should go to the neutral position at startup.

Default setting is 0

Bit = 0: The block does not go to the neutral position at startup

With control valve VlvAnL, the main valve is closed and the auxiliary valve (if configured) opened.

Bit = 1: The block goes to the neutral position at startup

With control valve VlvAnL, the main valve and auxiliary valve (if configured) go to the neutral position.

You can find more information in Section Neutral position for motors, valves and controllers (Page 48).

Note:

`Feature` bit 16 is only effective when `Feature` bit 0 = 0.

2.1.9.58 Neutral position manipulated variable takes effect with "out of service" operating mode

Feature Bit

Number of the `Feature` bit: 15

Neutral position manipulated variable takes effect with "out of service" operating mode

You can use this `Feature` bit to specify if the block should go to the neutral position when it transitions to the "Out of service" operating mode.

Default position is "0".

Bit = 0: The block does not go to the neutral position at the transition to the "out of service" operating mode.

Bit = 1: The block goes to the neutral position at the transition to the "out of service" operating mode.

Refer to the Neutral position for motors, valves and controllers (Page 48) section for more information.

2.1.9.59 Setting switch or button mode

Feature bit

Number of the `Feature` bit: 4

Setting switch or button mode (input signal as pulse signal or as static signal)

You can use this `Feature` bit to determine whether a separate interconnectable 1-active control input has to be used for every automatic command of the block or two automatic commands are assigned to a control input.

The `Feature` bit affects the following control inputs:

- starting and stopping a motor
- Opening and closing a valve
- switching modes (parameters `AutModLi` and `ManModLi`)
- Setpoint specification internal and external (parameters`SP_ExtLi` and `SP_IntLi`)

is given in the form of a pulse (pushbutton operation) or a static signal (switching mode).

You can find the commands for controlling the block in the relevant section on block operating modes. They are always the parameters that are used for the automatic operation of a block.

Bit = 0: Button mode: Each automatic command is assigned to a control input. This has a latching reaction and is 1-active.

Example with a motorMotRevL: In this case, use the interconnectable input parameters.

- `FwdAut` = `1` for the command "Start forward"
- `RevAut` = `1` for the command "Start backwards"
- `StopAut` = `1` for the stop command and
- `AutModLi` = `1` for setting "Automatic" operating mode
- `ManModLi` = `1` for setting "Manual" operating mode

Bit = 1: Switching mode: two static automatic commands are assigned to a control input.

Example with a motorMotRevL: In this case, use the interconnectable input parameters.

- `FwdAut` = `1` for the command "Start forward"
- `RevAut` = `1` for the command "Start backwards" and
- `FwdAut` = `0` and `RevAut` = `0` for the stop command
- `AutModLi` = `1` for setting "Automatic" operating mode
- `AutModLi` = `0` for setting "Manual" operating mode

The `StopAut` and `ManModLi` control inputs are irrelevant in this case.

2.1.9.60 Specifying switching mode

Feature Bit

Number of the `Feature` bit: 5

Specifying switching mode

Use this `Feature` bit to specify switching mode for the motor block.

The default setting is 0.

Bit = 0: Switching mode "On over speed 1" with the `SwOverTi > 0` parameter

Bit = 1: Switching mode "Off over speed 1" with the `SwOverTi > 0` parameter

Note

This `Feature` bit is only used with the `SwOverTi > 0` parameter. Refer also to the MotSpdL functions (Page 1215) section for more on this.

2.1.9.61 Creep rate is always detected in the dosing quantity

Feature Bit

Number of the `Feature` bit: 13

Enable: Creep rate is always detected in the dosing quantity

Use the `Feature` bit to specify the response for detecting the creep rate in the dosing quantity.

The default setting is 0.

Bit = 0: Disabled, the creep rate is only detected in the dosing quantity over the limit `CR_AH_Lim` (high alarm for creep rate). With `CR_AH_En = 0`, the creep rate has no effect on the dosing quantity calculation.

Bit = 1: Enabled, the creep rate is always detected in the dosing quantity.

Note

Creep rate is the flow in the states "End", "Off", and "Pause".

2.1.9.62 Enabling rapid stop via faceplate

Feature bit

Number of the `Feature` bit: 14

Enabling rapid stop via faceplate

You can use the `Feature` bit "Enable rapid stop via faceplate" to specify if the OS operator can use rapid stop for the block via the standard view of the faceplate.

The default setting is 0.

Bit = 0: The "Rapid stop" button is not visible in the faceplate.

Bit = 1: The OS operator can use the button for rapid stop.

2.1.9.63 Position feedback signals are active

Feature bit

Number of the Feature bit: 12

Position feedback signals are active

You can use this `Feature` bit to activate the position feedback signals (`FbkP1` and `FbkP2`) and deactivate the separate valve feedback signals (`FbkV0`, `FbkV1` and `FbkV2`). In the trend view and preview, the position feedback signals and control signals `Pos0Out, Pos1Out, Pos2Out` are used for display.

Default: 0

Bit = 0: The valve feedback signals `FbkV0`, `FbkV1`, `FbkV2` are active. The position feedback signals `FbkP1`, `FbkP2` are deactivated.

Bit = 1: The position feedback signals `FbkP1`, `FbkP2` are activate. The valve feedback signals `FbkV0`, `FbkV1`, `FbkV2` are deactivated.

Note

`SafeV0`, `SafeV1, SafeV2` and `DefPos1`, `DefPos2` are only used for configuring the icon displays in the OS.

2.1.9.64 Separate monitoring time for stopping the motor

Feature bit

Number of the `Feature` bit: 13

Separate monitoring time for stopping the motor

You can use this `Feature` bit to activate a separate monitoring time for stopping the motor.

The default setting is 0.

bit = 0: A monitoring time for starting and stopping the motor

Bit = 1: Separate monitoring time for stopping the motor

2.1.9.65 Separate interlock for each direction or position

Feature bit

Number of the `Feature`2 bit: 16

Separate interlock for each direction or position

You can use this `Feature` bit to enable the use of the "Separate interlock for each direction or position" function.

Default setting is 0

Bit = 0: One interlock for each direction or position.

Bit = 1: Separate interlock for each direction or position.

2.1.9.66 Separate delay times for each alarm

Feature bit

Number of the `Feature` bit: 8

Separate delay times for each alarm

You can use this `Feature` bit to activate the alarm delay type Two time values for each individual limit (Page 198).

The default setting is 0.

Bit = 0: The alarm delay type Two time values for each individual limit (Page 198) is deactivated.

Bit = 1: The alarm delay type Two time values for each individual limit (Page 198) is activated.

See also

User-configured message classes (Page 41)

Two time values per limit pair (Page 197)

2.1.9.67 Signaling limit violation

Feature bit

Number of the `Feature` bit: 29

Signaling limit violation

With this `Feature` bit, you specify how limit violation should be sent to the respective limit outputs.

The default setting is 0.

Bit = 0: Output value of the limit output = 1 (1 active)

Bit = 1: Output value of the limit output = 0 (0 active)

Note

To learn about the parameters with which you can influence this behavior, refer to the description of the connections for the respective blocks.

2.1.9.68 Alarm setpoint difference

Feature bit

Number of the `Feature` bit: 5

Setpoint difference should be alarmed

Use this `Feature` bit to activate alarming when a setpoint difference occurs.

The default setting is 0.

Bit = 0: Disabled: The message for the `ExtMsg2` and `ExtMsg3` parameters are output.

Bit = 1: Enabled: The messages for the `ER_H_Lim` or `ER_L_Lim` parameter are output instead of the message parameters `ExtMsg2` and `ExtMsg3`.

2.1.9.69 Setpoint specification with separate display area and custom unit

Feature bit

Number of the `Feature` bit: 16

Setpoint specification with separate display area and custom unit

You can use this `Feature` bit to specify the display area and the unit of the setpoint specification.

The default setting is 0.

Bit = 0: Setpoint specification and readback value have the same display area (`RbkOpScale`, `RbkUnit`)

Bit = 1: The setpoint specification has a separate display area and a custom unit (`SP_OpScale`, `SP_Unit`)

2.1.9.70 Control via auxiliary valve

Feature Bit

Number of the `Feature` bit: 5

Control via a auxiliary valve

You can use this `Feature` bit to specify if control should be performed by a auxiliary valve.

The default setting is 0.

Bit = 0: Control without auxiliary valve / no auxiliary valve present

Bit = 1: Control via auxiliary valve

2.1.9.71 Enabling bumpless change to the proportional gain, derivative time and amplification of the differentiator

Feature Bit

Number of the `Feature` bit: 20

Enabling bumpless change to the proportional gain, derivative time and amplification of the differentiator

Use this `Feature` bit to enable bumpless change to proportional gain `Gain`, derivative action time `TD` and gain of the differentiator in automatic mode.

The default setting is 0.

Bit = 0: The bumpless switchover is deactivated.

Bit = 1: The bumpless switchover is activated.

2.1.9.72 Enable bumpless switchover to "Automatic" mode for operator only

Feature bit

Number of the `Feature` bit: 21

Enable bumpless switchover to automatic mode for operator only

You can use this `Feature` bit to specify whether bumpless switchover to automatic mode for valves, motors and dosing unit should be enabled only if switching via the faceplate is in effect, or whether switching using interconnectable inputs `AutModLi` and `ManModLi` (`ModLiOp = 1`) is also possible.

The default setting is 0

Bit = 0: The function "Bumpless switchover to automatic mode for valves, motors and dosers" works when switching via the faceplate and switching using interconnectable inputs `AutModLi` and `ManModLi` (`ModLiOp = 1`).

Bit = 1: The "Bumpless switchover in automatic mode for valves, motors and dosers" function only works for switching via the faceplate. Bumping switchover can be activated via the inputs `AutModLi` and `ManModLi` (`ModLiOp = 1`).

2.1.9.73 Disabling bumpless switchover to automatic mode for controllers

Feature Bit

Number of the `Feature` bit: 18

Changeover with or without P step change when the internal setpoint is not in tracking mode

Use this `Feature` bit to specify if a changeover should occur with or without a P step change when the internal setpoint (`SP_TrkPv = 0`) does not track the process value.

The default setting is 0.

Bit = 0: Changeover without P step change (bumpless)

Bit = 1: Changeover with P step change (not bumpless)

For more detailed information, refer to the description of Manual and automatic mode for control blocks (Page 72).

2.1.9.74 Enabling bumpless switchover to automatic mode for valves, motors, and dosers

Feature bit

Number of the `Feature` bit: 17

Bumpless switchover

You can use this `Feature` bit to enable the bumpless switchover from local/manual mode to automatic mode.

Default setting is 0

Bit = 0: Bumpless switchover is disabled. You can switch from local/manual mode to automatic mode at any time.

Bit = 1: Bumpless switchover from local/manual mode to automatic mode is enabled. A switchover from local/manual mode to automatic mode is only possible if the control settings of the local/manual mode and automatic modes match. If switchover occurs at a different point in time, this is indicated in the faceplate with the text "Switchover error".

Refer to the Manual and automatic mode for motors, valves and dosers (Page 75) section for more on this.

A second feature bit is used to specify if bumpless switchover to automatic mode is only possible via the faceplate or if switching is also possible via the interconnectable parameters `AutModLi` and `ManModLi` (`ModLiOp = 1`).

Refer to the section Enable bumpless switchover to "Automatic" mode for operator only (Page 171).

2.1.9.75 Summing characteristics continuous or triggered

Feature bit

Number of the `Feature` bit: 7

Define summing characteristics

You can use this `Feature` bit to define the summing characteristics of the block as a function of the `Feature` bit6 = 0. If you have set the `Feature` bit 6 = 1, this `Feature` bit has no function.

The default setting is 0.

Bit = 0: Triggered summing characteristics.

Bit = 1: Continuous summing characteristics.

2.1.9.76 Suppression of all messages

Feature Bit

Number of the `Feature` bit: 25

Suppression of all messages

You can use this `Feature` bit to determine whether all messages of the block are to be suppressed.

Bit = 0: Process messages are suppressed.

Bit = 1: All messages are suppressed.

2.1.9.77 Output invalid raw value

Feature bit

Number of the `Feature` bit: 28

Output invalid raw value

Use this `Feature` bit to activate output of the invalid raw value for channel blocks.

Default setting is 1

Bit = 0: The invalid raw value is not output. Either the substitute value (`Feature` bit Output substitute value if raw value is invalid (Page 148)) or the last valid value (`Feature` bit Outputting last valid value if raw value is invalid (Page 153)) is output.

Bit = 1: The invalid raw value is output. The signal status of the output value is set to "Bad, device related" or "Bad, process related".

If there is no valid raw value the output parameter `Bad` = `1` is set automatically.

Prioritizing the `Feature` bits for channel blocks:

You need to assign parameters for three `Feature` bits for the response to an invalid raw value for the channel blocks.

If more than one of these `Feature` bits are set (=1), the following priority applies:

- Output invalid raw value (`Feature` bit 28 = highest priority)
- Output substitute value if raw value is invalid (Page 148)(`Feature` bit 29)
- Outputting last valid value if raw value is invalid (Page 153) (`Feature` bit 30 = lowest priority)

The raw value is output of none of the `Feature` bits 28, 29 or 30 is set.

2.1.9.78 Transmission of status information of devices

Feature bit

Number of the `Feature` bit: 29

Transmission of status information of devices

You can use this `Feature` bit to specify if status information of the device should be transferred to the upstream diagnostics block.

The default setting is 1.

Bit = 0: No transfer of status information.

Bit = 1: Transfer status information of the device to the upstream diagnostics block.

2.1.9.79 Control priority in the event of an invalid input command

Feature2 Bit

Number of the `Feature`2 bit: 3

Control priority in the event of an invalid input command

You can use this function to define the control priority in the event of an invalid input command.

Default setting: 0

Bit = 0: In the event of an invalid input command, the control output is retained.

Bit = 1: In the event of an invalid input command, the control output switches to the "neutral position".

2.1.9.80 Sealing the valve

Feature Bit

Number of the `Feature` bit: 8

Sealing the valve

You can use this `Feature` to enable the valve sealing function. The torque shutoff is then also evaluated when the valve is closed.

The default setting is 0.

Bit = 0: "Seal valve" function is disabled.

Bit = 1: "Seal valve" function is enabled.

2.1.9.81 First-in detection response to deactivation

Feature Bit

Number of the `Feature` Bit: 21

First-in detection response to deactivation

You can use this `Feature` Bit to define the first-in detection response of the interlock blocks depending on the `FirstInEn = 0` input parameter.

The default setting is 0.

- **Bit = 0**: When first-in detection is deactivated via `FirstInEn = 0`, the output parameter `FirstIn` is reset.
- **Bit = 1**: When first-in detection is deactivated via `FirstInEn =0`, the output parameter `FirstIn` is maintained. When first-in detection is activated for the first time by an edge change of `FirstInEn 0->1`, the output parameter `FirstIn` is reset.

2.1.9.82 Reaction of the switching points in the "Out of service" operating mode

Feature Bit

Number of the `Feature` bit: 26

Reaction of the switching points in the "Out of service" operating mode

You can use this `Feature` bit to specify the reaction of the switching points to the "Out of service" operating mode.

The default setting is 0.

Bit = 0: Last state of the switching points before switching to the "Out of service" operating mode is retained.

Bit = 1: The state of the switching points is reset to "Good".

2.1.9.83 Reaction to the out of service mode

Feature bit

Number of the `Feature` bit: 1

Reaction to the out of service mode

You can use this `Feature` bit to define the reaction of the technologic block based on the interconnectable input parameter `OosLi = 1`.

The default setting is 0.

- **Bit = 0**: The symbol for the "In progress" status (see below) appears in the block icon and in the faceplate of the assigned technologic block. A 0-1 edge transition at the input parameter `OosLi` has no further influence on the reaction of the technological block; the previous status is retained. No switch to the "Out of service" mode is performed.
- **Bit = 1**: The mode switches to "Out of service" assuming that the block is "On" or "Manual" mode. If this is not the case, the mode does not change. The symbol for the "In progress" (see below) status also appears in the block icon and in the faceplate of the assigned technologic block regardless of the mode change. No message is output to indicate whether or not the mode change took place.

The status display for "In progress" appears as follows:

A 1-0 edge transition at the input parameter `OosLi` has no influence on the reaction of the technologic block, the previous status is retained.

See also the section Release for maintenance (Page 64) for more on this.

2.1.9.84 Exiting local mode

Feature bit

Number of the `Feature` bit: 10

Reaction to exiting local mode

Use this `Feature` bit to define how "Local mode" is be exited with `LocalSetting` = `1` or `LocalSetting` = `2` and if the mode is not specified by `AutModLi` or `ManModLi`.

Default setting is 0

Bit = 0: Exiting local mode in manual mode (bumpless because the control signals are continuously adjusted).

Bit = 1: When local mode is exited, the mode changes back to the last mode that was active prior to local mode (not bumpless).

For more detailed information, refer to the description of Local mode (Page 79).

2.1.9.85 Interlock display with LocalSetting 2 or 4

Feature bit

Number of the `Feature` bit: 27

Interlock display with LocalSetting 2 or 4

Use this `Feature` bit to specify the display of interlocks with `LocalSetting 2` or `4` at the faceplate and at the faceplate output `LockAct` .

The default setting is 0

Bit = 0: `LocalSetting 2`and`4` crossed out locks are displayed in the standard view. `LockAct` is not set with interlock.

Bit = 1: `LocalSetting 2`and`4` locks are displayed in the standard view according to the interlock. `LockAct` is set according to interlock. This setting is used for hardware interlock.

Note

A decreasing motor protection (`Trip.Value`=0) is displayed at the output parameter `LockAct` , regardless of the `Feature` bit setting.

This feature bit is also applicable for the LocalSetting 5 in VlvS where LocalSetting 4 is not available

2.1.9.86 Vibrate after torque monitoring

Feature2 bit

Number of the `Feature2` bit: 6

Vibrate after torque monitoring

You can use this `Feature2` bit to determine whether vibration should be started after a torque monitoring error `TorqOpen` or `TorqClose`.

The default setting is 0.

Bit = 0: No vibration.

Bit = 1: Vibration is started after a torque monitoring error.

2.1.9.87 With acknowledge overdosage

Feature2 bit

Number of the `Feature2` bit: 24

With acknowledge overdosage

You can use this `Feature2` bit to change the acknowledgment behavior of overdosage and underdosage.

The default setting is 0.

Bit = 0: After an overdosage is identified, the block changes directly to the “End” state.

Bit = 1: After an overdosage is identified, the block changes and continues to be in the “Off” state. An acknowledgment is necessary to change the dosing state to “End”. In the standard view, status display shows the state “Ack Dos End”.

2.1.9.88 Ramp rate calculation

Feature bit

Number of the `Feature` bit: 6 (7 for VlvAnL)

Define ramp rate

You can use this `Feature` bit to define the ramp rate calculation based on the actual start point and the actual ramp target value.

The default setting is 0.

Bit = 0: New ramp rate will not be calculated after a change in the ramp target value.

Bit = 1: Ramp rate is calculated with the actual start point and the actual ramp target value.

2.1.9.89 SP following PV in open loop has no priority over SP_Ext and SP limits

Feature bit

Number of the `Feature` bit: 23

SP following PV in open loop has no priority over SP_Ext and SP limits

You can use this `Feature` bit to specify whether the setpoint follows the process value in "manual mode" with tracking (`SP_TrkPV` = 1). You can specify this function in case of setpoint is external or limited.

The default setting is 0.

Bit = 0: Setpoint follows process value (`SP` = `PV`) irrespective of whether setpoint is external or limited.

Bit = 1: If the setpoint is external, setpoint follows the external setpoint (`SP` = `SP_Ext`) and will be limited to the external limits (`SP_ExHiLim` and `SP_ExLoLim`). If the setpoint is internal, setpoint follows the process value (`SP` = `PV`) and will be limited to the internal limits (`SP_InHiLim` and `SP_InLoLim`).

2.1.9.90 With accelerated return of the integral action from the manipulated variable limit

Feature bit

Number of the `Feature` bit: 17

With accelerated return of the integral action from the manipulated variable limit

When switching to the automatic mode (manually, by tracking, forced tracking, or program mode with manipulated variable specification) or when the controller is initialized by the `InitPid` input, the integral action (reset) is moved to preserve bumpless operation in case of an impending control deviation. The greater the control deviation and the controller gain, the greater is this shift. The result may be that the integral action is set far outside the manipulated variable limits. If the control deviation decreases after switching to the automatic mode, it may happen that in controllers with large `TI` values, the manipulated variable stays within the limit for a longer duration.

You can use this `Feature` bit to accelerate the return of the integral action from the limit. In case of a pending control deviation, the integral action is moved only in the direction of the control range to the extent that the manipulated variable can remove itself quickly and bumplessly from the manipulated variable limits (`MV_HiLim`/`MV_LoLim`).

The default setting is 0.

Bit = 0: Without accelerated return of the integral action.

Bit = 1: With accelerated return of the integral action when the integral action and the manipulated variable are outside or at the manipulated variable limits.

Note

In regular control mode, the integral action cannot move outside the manipulated variable limits because the manipulated variable is limited first and then fed back to the reset memory (see also anti-windup in PIDConR functions (Page 800)).

2.1.9.91 External/internal selection specification

Feature bit

Number of the `Feature` bit: 6

External/internal selection specification

You can use this `Feature` bit to switch from the faceplate between internal and external selection of the inputs `In01` to `In10` (`SelInt` or `SelExt`).

The default setting is 0.

- **Bit = 0**:

`LiOp` = 0	Selection of `In01` to `In16` is internal; `SelInt` is active and can be used via faceplate.
`LiOp` = 1	Selection of `In01` to `In16` is external; `SelExt` is active and can be used via interconnection.

- **Bit = 1**:

`LiOp` = 0	With the input parameter `SelIntOp` or `SelExtOp`, you can change the selection between internal (`SelInt`) and external (`SelExt`) via faceplate.
`LiOp` = 1	With the input parameter `SelIntLi` or `SelExtLi`, you can change the selection between internal (`SelInt`) and external (`SelExt`) via interconnection.

2.1.9.92 Setting switch or button mode for local commands

Feature2 bit

Number of the `Feature2` bit: 4

Setting switch or button mode for local commands (input signal as pulse signal or as static signal)

You can use this `Feature2` bit to determine whether a separate interconnectable 1-active control input has to be used for every local command of the block or two local commands are assigned to a control input.

The `Feature2` bit affects the following control inputs for local mode in the form of a pulse (pushbutton operation) or a static signal (switching mode):

- Starting and stopping a motor
- Opening and closing a valve

You can find the commands for controlling the block in the relevant section on block operating modes. They are always the parameters that are used for the local operation of a block.

Bit = 0: Button mode: Each local command is assigned to a control input. This has a latching reaction and is 1-active.

1. Example with a motor MotRevL: In this case, use the interconnectable input parameters.

- `FwdLocal` = 1 for the command "Start forward"
- `RevLocal` = 1 for the command "Start backwards"
- `StopLocal` = 1 for the stop command

Bit = 1: Switching mode: two static local commands are assigned to a control input.

2. Example with a motor MotRevL: In this case, use the interconnectable input parameters.

- `FwdLocal` = 1 for the command "Start forward"
- `RevLocal` = 1 for the command "Start backwards"
- `FwdLocal` = 0 and `RevLocal` = 0 for the stop command

The `StopLocal` control input is irrelevant in this case.

3. Example with motor MotSpdL: In this case, use the interconnectable input parameters.

- `Spd1Local` = 1 for the command Motor to run with Speed1
- `Spd2Local` = 1 for the command Motor to run with Speed2
- `Spd1Local` = 0 and `Spd2Local` = 0 for the stop Command

4. Example with motor MotSpdCL: In this case, use the interconnectable input parameters.

- `FwdLocal` = 1 for the command "Start forward"
- `RevLocal` = 1 for the command "Start backwards"
- `FwdLocal` = 0 and `RevLocal` = 0 for the stop command

5. Example with VlvPosL: In this case, use the interconnectable input parameters.

- `OpenLocal` = 1, Command for Valve to "Open”
- `CloseLocal` = 1, Command for Valve to "Close”
- `OpenLocal` = 0 and `CloseLocal` = 0 for the stop command

6. Example with Vlv2WayL: In this case, use the interconnectable input parameters.

- `Pos1Local` = 1, Command for Valve1 to "Open"
- `Pos2Local` = 1, Command for Valve2 to "Open"
- `Pos1Local` = 0 and `Pos2Local` = 0, Command for "Position 0" (neutral position),

7. Example with VlvL

- `OpenLocal` = 1, Command for Valve to "Open"
- `OpenLocal` = 0, Command for Valve to "Close"

With `Feature2.Bit4` = 1, two static local commands are assigned to a control input.

2.1.9.93 Gradient limitation with time duration

Feature bit

Number of the `Feature` bit:

- 11 for PIDConL, PIDConR, and PIDStepL
- 12 for MotSpdCL, VlvAnL, and OpAnL

Gradient limitation with time duration

You can use this `Feature` bit to ramp up or ramp down the setpoint `SP` respectively `MV` as a function of time duration. With this `Feature` bit, the gradient limitation function can also work with the time duration.

The default setting is 0.

Bit = 0: Gradient limitation works only with the gradient values: `SP`/`MV_UpRaLim`, `SP`/`MV_DnRaLim`

Bit = 1: Gradient limitation also works with the time duration with the following parameters: `SP`/`MV_RmpModTime`, `SP`/`MV_RmpTime`.

2.1.9.94 Analog input 1 is reserved for the operator

Feature bit

Number of the `Feature` bit: 7

Analog input 1 is reserved for the operator

You can use this `Feature` bit to reserve the analog input 1 (`In01`) for the operator.

The default setting is 0.

Bit = 0: Operator cannot use the analog input 1 (`In01`).

Bit = 1: Operator can change the value of analog input 1 (`In01`) if the input is not connected and `OS_Perm.Bit22` is set to 1.

2.1.9.95 Operator can change the setpoint via faceplate also in the "Local" mode

Feature2 bit

Number of the `Feature2` bit: 6

Operator can change the setpoint via faceplate also in the "Local" mode

You can use this `Feature2` bit to enable the input of a setpoint for the operator in the "Local" mode.

The default setting is 0.

Bit = 0: Operator cannot change the setpoint in the "Local" mode.

Bit = 1: Operator can change the setpoint via faceplate in the "Local" mode.

2.1.9.96 Define the setpoint after stop and start of the motor

Feature2 bit

Number of the `Feature2` bit: 7

Define the setpoint after stop and start of the motor

You can use this `Feature2` bit to define the setpoint after stop and start of the motor.

The default setting is 0.

Bit = 0: The setpoint is retained after the motor is stopped.

Bit = 1: After the motor is stopped, the setpoint will be set to `SP_Off`. If the motor is started and the setpoint is in internal mode, the setpoint will be set to the low limitation `SP_LoLim`. Up to this value, the internal setpoint can be changed. In case the setpoint is external after the start of the motor, the setpoint will be changed directly to the external setpoint. The low limitation will be limited downwards to `SP_Off` and will be written back if the limitation is reached.

2.1.9.97 Retain last output value in case of bad input signal status

Feature bit

Number of the `Feature` bit: 5

Retain last output value in case of bad input signal status

You can use this `Feature` bit to specify that the block holds its output value if the input signal status is bad (16#00 or 16#28).

The default setting is 0.

Bit = 0: Default state of the block.

Bit = 1: Retain last output value in case of bad input signal status.

2.1.9.98 Forcing operating modes in the "Local" mode

Feature2 bit

Number of the `Feature2` bit: 8

Forcing operating modes in the "Local" mode

You can use this `Feature2` bit to specify whether forcing operating modes is possible in the "Local" mode.

The default setting is 0.

Bit = 0: In the "Local" mode, forcing operating modes is not possible.

Bit = 1: In the "Local" mode, forcing operating modes is possible.

2.1.9.99 Motor stop in end position depends only on the corresponding feedback signal

Feature bit

Number of the `Feature` bit: 15

Motor stop in end position depends only on the corresponding feedback signal

With this `Feature` bit, you can specify whether the motor stop in end position depends only on the corresponding feedback signal or on both feedback signals.

The default setting is 0.

Bit = 0: Motor stop in end position depends on two feedback signals (`FbkOpen` and `FbkClose`). For example, motor stop in the end position "Open" depends on both feedback signals "Opened" and "Closed". In this case, `FbkOpen` = 1 and `FbkClose` = 0 are parameterized such that the drive output control of the motor "Open" is reset.

Bit = 1: Motor stop in end position depends only on the corresponding feedback signal. For example, motor stop in the end position "Open" depends only on the feedback signal "Opened". In this case, only `FbkOpen` = 1 is parameterized such that the drive output control of the motor "Open" is reset.

2.1.9.100 Substitution value is active if the block is in bypass

Feature bit

Number of the `Feature` bit: 9

Substitution value is active if the block is in bypass

You can use this `Feature` bit to specify whether the block uses the process value or a substitution value in case of an active bypass.

The default setting is 0.

Bit = 0: Block will use the process value (`PV`/`In`/`Out`).

Bit = 1: Block will use the substitution value (`BypPV`/`BypIn`/`BypOut`).

2.1.9.101 Enable external message

Feature bit

Number of the `Feature` bit: 9

Enable external message

You can use this `Feature` bit to specify if you want to use the quality code of `Inx`/`InTSx` or the external values `ExtValxx` as the associated value.

The default setting is 0.

Bit = 0: Use quality code of Inx or InTSx as the associated value.

Bit = 1: Use external values `ExtValxx` as the associated value.

2.1.9.102 Considering bad quality of automatic commands or external values

Feature/Feature2 bit

Number of the `Feature`/`Feature2` bit: 10

Considering bad quality of automatic commands or external values

You can use this `Feature/Feature2` bit to consider the bad signal status (16#00 or 16#28) in the parameter for the automatic commands or external values to move the block into the neutral position.

The default setting is 0.

Bit = 0: A bad signal status will not be considered.

Bit = 1: A bad signal status (16#00 or 16#28) in the automatic commands or external values will be considered from the block. The block goes to a defined neutral position.

2.1.9.103 Disable calculation of impulse control in LocalSetting 2 and 4

Feature bit

Number of the `Feature` bit: 20

Disable calculation of impulse control in `LocalSetting` 2 and 4

You can use this `Feature` bit to disable the tracking of feedback signals in the "Local" mode setting 2 and 4.

The default setting is 0.

Bit = 0: Calculation of impulse controls in `LocalSetting` 2 and 4.

Bit = 1: Disable calculation of Impulse controls in `LocalSetting` 2 and 4.

2.1.9.104 Inverter enable

Feature bit

Number of the `Feature` bit: 8

Inverter enable

You can use this `Feature` bit to specify whether, after the activation of `Feature` bit 15 (`Feature.Bit15` = 1), a frequency converter with a separate device infeed is to be used or the inverter enable.

The default setting is 0.

Bit = 0: Frequency converter with separate device infeed

Bit = 1: Inverter enable

2.1.10 Functions for controllers

2.1.10.1 Delay alarm for control deviation at setpoint step changes

Alarm delay for blocks with the function "Delay alarm for control deviation at setpoint step changes"

This type of alarm delay is used when temporary violations of set alarm thresholds of the control deviation are to be suppressed at setpoint step changes. The alarm delay is parameterized at the following inputs:

Parameter for the delay time	Explanation
ER_AH_DFac	Delay factor at positive setpoint step changes for incoming alarms at the control deviation monitoring ER_AH_Lim
ER_AL_DFac	Delay factor at negative setpoint step changes for incoming alarms at the control deviation monitoring ER_AL_Lim

The effective delay time is calculated from the delay factor and the setpoint difference:

- Positive setpoint step change: ER_A_DCOut = Maximum
  The maximum is formed from the parameters ER_A_DC as well as
  ER_AH_DFac• Setpoint difference
- Negative setpoint step change: Maximum
  The maximum is formed from the parameters ER_A_DC as well as
  -1• ER_AL_DFac• Setpoint difference

The effective delay time is specified at the output parameter:

Parameter	Explanation
ER_A_DCOut	Effective delay time at setpoint step changes for incoming alarms during control deviation monitoring

Before a setpoint change, the effective delay time amounts to
ER_A_DCOut = ER_A_DC.

For a setpoint step change, the effective delay time increases depending on the factors ER_AH_DFac as well as ER_AL_DFac.

When the control loop has settled again, meaning that
(ER_AL_Lim + ER_Hyst) <= ER <= (ER_AH_Lim – ER_Hyst)
and that the delay time for outgoing alarms (ER_A_DG) has expired, the output is reset again to ER_A_DC: ER_A_DCOut = ER_A_DC
.

Activating the alarm delay

The alarm delay factors have a default value of 0, meaning that the function is deactivated. Specify a delay factor > 0 in order to use the function.

Pending alarms

Pending alarms are output at the output parameters ER_AL_Act and ER_AH_Act.

Calculation of the output parameter `ER_A_DCOut`

`ER_A_DC` is assigned by default to the output before a setpoint change.

`ER_A_DCOut` = `ER_A_DC`
In the case of a setpoint change in the positive direction during automatic mode, the output is calculated as follows:

`ER_A_DCOut` = Maximum (`ER_A_DC`, `ER_AH_DFac` * Setpoint difference)

In the case of a setpoint change in the negative direction during automatic mode, the output is calculated as follows:

`ER_A_DCOut` = Maximum (`ER_A_DC`, -1*`ER_AL_DFac` * Setpoint difference)

When the control circuit has stabilized again, meaning (`ER_AL_Lim` + `ER_Hyst`) <= `ER` <= (`ER_AH_Lim` – `ER_Hyst`), and the delay time for outgoing alarms `ER_A_DG` has expired, the output is reset again to `ER_A_DC`: `ER_A_DCOut` = `ER_A_DC`.

Activating and deactivating the function

The function is deactivated (default) when `ER_AH_DFac` = 0.0 and `ER_AL_DFac` = 0.0.

2.1.10.2 Inverting control direction

Inverting control direction

For some processes (for example cooling processes), negative control gain is necessary. This is achieved by the inverting the control direction by means of the input parameter `NegGain = 1`. The gain is always entered positively at the input parameter `Gain` . If there is an inversion, this is indicated at the output parameter `GainEff` by a negative number.

2.1.10.3 Error signal generation and dead band

Error signal generation and dead band

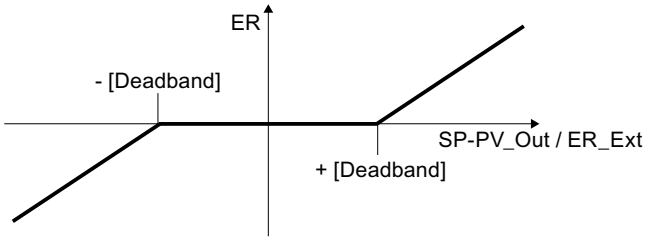
The signal error is formed from the effective setpoint `SP` and the process value `PV` (`ER = SP - PV_Out`) (PIDConR `ER =PV_Out - SP`) and is available at the `ER` output.

In the case of activated error signal generation (only at block PIDConL, `Feature` bit 14), `ER` is formed by `ER_Ext`.

To suppress disturbances in the steady state, you can assign a dead band `(Deadband)`:

- `Deadband = 0`: Dead band is disabled
- `Deadband ≠ 0`: Dead band is enabled

With a negative dead band (`Deadband < 0.0`), calculation is continued internally with this value



A dead zone in the controller can help reduce the movements of the actuator and thus reduce the energy consumption and wear of the actuator. This applies especially to mechanical actuators, such as valves and pumps. The dead zone may also required in order to avoid small, stable, continuous oscillations (working movement), which are caused by the quantization of the control actions in the control loop.

Note

Calming the control signals via dead band is only possible when no structure splitting is active, i.e. when `PropFacSP` = 1 and `DiffToFbk` = 0. Controller actions in the feedback branch directly process the measured actual value instead of the control deviation and are therefore not affected by the dead band.

- Remaining control deviations smaller than the dead band are ignored by the controller, i.e. it does nothing even it can. Therefore, steady states of the actual value can occur, the average time of which significantly differs from the setpoint.
- When such steady states occur at the edges of the dead zone, it must be expected that even the smallest problems will cause control intervention (i.e. actuator energy consumption).
- Large dead zones have a negative effect on the control response of the control loop for setpoint changes. This is because the controller "stops work", for example, in the rising phase of a positive step response when entering the dead band. This can lead to creeping settling time, or cause the controller to be activated again when the actual value exceeds the high limit of the dead band in the form of an overshoot.
- After a fault event that leads to exiting the dead band, the controller brings the actual value back only to the edge of the dead band, which has the above-mentioned disadvantages.

By setting `Feature` bit 30, you can activate dynamic adjustment of the dead band. This allows you to benefit from the advantages of a dead band without having to tolerate the above-mentioned disadvantages.

When large control deviations occur (i.e. the dead band is exited), it is temporarily deactivate until the controller returns the actual value to the proximity of the setpoint. The controller also demonstrates control response for setpoint changes without a dead band in this way.

There is then a wait until it can be assumed that the process has settled on operating point. Only then is the dead band activated again. When it is reactivated, the manipulated variable is set to the mean of a sliding time window. This ensures that the actual value to be controlled settles more in the middle of the dead band than at the edges. This avoids unnecessary shifting of the mean value of the controlled variable from the setpoint, and thus poor stationary control performance. The probability that the controlled variable is within the dead zone time is increased, and further control actions are no longer necessary. This leads to reduced wear and less energy consumption.

Configuration and commissioning

- Use a process tag type that contains the ConPerMon function block for monitoring the control performance in addition to the controller.
- Check whether the additional interconnections between the monitoring block and controller are available:
  - `PIDConL.MV_Mean:= ConPerMon.MV_Mean`
  - `PIDConL.SettliDeadBand:= ConPerMon.SettliDeadBand`
- Put the controller into operation as usual with the PID Tuner, and then initialize the ConPerMon block.
- Perform a setpoint step with activated control performance monitoring and without dead band (`PIDConL.DeadBand:=0`). Various characteristics of the step response can be determined by the ConPerMon block in this way. Check if the determined settling time at the `ConPerMon.SettlingTime` output parameter appears plausible.
- Specify the width of the dead band at the `PIDConL.DeadBand` input parameter.
- Check the actuating signal signal in the steady state of the control loop. Has the desired calming of the signal form been achieved?

- Activate `Feature` bit 30.
- Check the control response of the control loop. If the control loop has not sufficiently settled as expected near the center of the dead band after a setpoint change, it may help to increase the value of the `PIDConL.SettliDeadBand` input parameter.

Notes on selecting the width of the dead band

The width of the dead band primarily depends on the desired precision of the control, in other words, after the maximum, permanent, control deviations permissible by process engineering and after the change of the manipulated variable based on a minimum possible change of the manipulated variable (caused, for example, by a switching actuator). With regard to the minimization of control interventions, the following adjustment rules help to select the dead band wide enough that unavoidable variances of the control variables due to measurement noise or quantization noise do not lead to permanent control movements:

- Assuming a normal distribution of the statistical control variables around the setpoint, two to three times the standard deviation of the actual value in the steady state is used as the width of the dead band. This information can be obtained from the control loop monitoring block: `PIDConL.DeadBand:= (2...3)*ConPerMon.RefStdDev`.
- If working movements in the control loop are systematically caused by the manipulated variable quantization, for example, due to a pulse width modulation with a defined minimum pulse duration and pulse period, a step controller with defined minimum step size or an electro-pneumatic positioner, the width of the dead band depends on the manipulated variable quantization multiplied by the process gain.
  Example: A control signal resolution of 5% for temperature regulation with the process gain 1.5°C/% means that only temperatures in a grid of 5%, 1.5°C/% = 7.5°C can be reached exactly. The dead band must then be selected wide enough sp that at least one grid point falls within the dead band in which the process can remain in a steady-state, i.e. DeadBand > 3.8°C in this case.

2.1.10.4 Using control zones

Using control zones

The control zone function is mainly used for temperature processes in which the time lag is small compared to the recovery time (no more than a tenth of the recovery time). This is typically the case when the temperature controller accesses the actuator directly (e.g. heating, steam valve), and typically not the case with master controllers of cascade interconnections (e.g. control of the internal temperature of a reactor via the shell temperature).

The control zone can result in large overshoots in processes with an excessively large time lag.

During commissioning with the PCS 7 PID-Tuner, you can check whether the process order is smaller than 2 and the process type "VZ2 model" is displayed. An initial approach for dimensioning the control zone: `ConZone= MV_HiLim/Gain`.

If `ConZone > 0`, the controller works with a control zone; if `ConZone <= 0`, the "control zone" function is deactivated. With a negative control zone, an error ID is output at the

ErrorNum parameter. When the control zone is enabled, the controller operates according to the following algorithm:

- If the process value PV exceeds the setpoint SP by more than ConZone, the value MV_LoLim is output as the manipulated variable (controlled mode, in order to return to the control zone as fast as possible).
- If the process value PV falls below the setpoint SP by more than ConZone, MV_HiLim is output (controlled mode).
- (Only possible with PIDConL, not with FmTemp) In controlled mode of the control zone and when the I component is enabled (TI<>0), the I component can be set in a variety of ways when the control zone is entered, depending on three Feature bits:

Feature bit 12	**Feature** bit 13	**Feature** bit 18	Response
0	0	0	I_Part := MV.Value -P_Part -FFwd ; The controller operates bumplessly when entering automatic control mode. (This corresponds to the reaction up to V8.0)
0	0	1	I_Part := MV.Value -FFwd ; A P jump of the manipulated variable occurs when entering automatic control mode (use only in exceptional cases)
1	x	x	I_Part := MV_Offset.Value; The controller starts with the default value MV_Offset as I component when entering automatic control mode
0	1	x	I component is frozen. When it enters the control zone, the controller starts with the value of the I component that was in effect when it left the control zone. This can be useful for continuous processes, where it can be assumed that the controller is at the same operating point as it was when it left the control zone, nearly in a steady state.

x = any

The I component is always set to MV_Offset when the I component is disabled (TI=0). I.e.: The control zone operates with the specified I_Part := MV_Offset with P or PD controlling.

Freezing of the I component with IntHoldPos or IntHoldNeg has no effect during controlled mode due to the control zone.

- If the process value PV stays within the control zone (ConZone), the manipulated variable assumes the value of the PID algorithm (automatic closed-loop mode).

Note

The change from controlled closed-loop mode to automatic closed-loop mode is based on a hysteresis of 20% of the control zone. Make sure that the control zone has an adequate width before you manually activate the control zone. An insufficient width of the control zone leads to oscillation of the manipulated variable and of the process value.

Advantages of the control zone:

Once the control zone is entered, the manipulated variable is quickly reduced by the applied D component. The control zone is therefore only useful if the derivative component is activated. Without the derivative component, only the reducing proportional component would reduce the manipulated variable to any significant degree. The control zone speeds up settling without overshoot or undershoot if the value of the output minimum or maximum manipulated variable is a long way from the stationary manipulated variable required for the new operating point.

2.1.10.5 Setpoint limiting for external setpoints

Setpoint limiting for external setpoints

With this function you can limit the external setpoint to a range by means of the parameters `SP_ExHiLim` (high limit) and `SP_ExLoLim` (low limit). If the setpoint lies outside the range defined by you, it is limited to the valid range.

If the external setpoint lies on or above the limit `SP_ExHiLim`, this is displayed at the output `SP_ExHiAct = 1`.

If the external setpoint lies on or below the limit of `SP_ExLoLim`, this is displayed at the output `SP_ExLoAct = 1`.

The external setpoint limits can be tracked internally via the interconnection of the output parameters `SP_InHiOut` or `SP_InLoOut` following `SP_ExHiLim` or `SP_ExLoLim`. You can control the two limit pairs from the faceplate.

2.1.10.6 Tracking setpoint in manual mode

Tracking setpoint in manual mode

To allow a bumpless transfer to automatic mode, the setpoint tracks the process value. When tracking,(`SP_TrkPV = 1`) the internal setpoint `SP_Int` is tracked to the process value `PV`.

Additional information on setpoint tracking is available in Manual and automatic mode for control blocks (Page 72).

2.1.10.7 Tracking and limiting a manipulated variable

Manipulated variable tracking

You adjust the manipulated value (tracking) in order to implement a Bumpless switchover of controllers. A typical use case is cascade control: If the assigned secondary controller is no longer in automatic mode with external setpoint, the primary controller must track it.

To adjust the manipulated variable (tracking), you have to set the parameter `MV_TrkOn = 1`. Now the manipulated variable is taken from the interconnected tracking value `MV_Trk` and passed to the output `MV`. The MV output is limited to the `MV_HiLim` and `MV_LoLim` parameters.

Manual mode takes priority over tracking to allow a plant operator to set the controller to manual mode using the faceplate even when tracking the manipulated variable and therefore continue to normal operation.

The text "Tracking" is also displayed in the standard view of the faceplate.

Activating forced tracking mode for the manipulated variable

Forced tracking is used to set the controller output of a higher-level controller to a value that can be specified.

You can use forced tracking, for example, to implement a centralized emergency stop in the plant. This can be put into effect regardless of the operating mode the controller is currently in.

To force adjustment of the manipulated variable (tracking), you have to set the parameter `MV_ForOn = 1`. Now the manipulated variable is taken from the interconnected tracking value `MV_Forced` and passed to the output `MV`.

With forced tracking, it is not possible to limit the manipulated variable, nor can the plant operator change to manual mode in the faceplate. The text "Forced tracking" is also displayed in the standard view of the faceplate.

Note

This function is not available for the FmCont, FmTemp, and ModPreCon blocks.

Limiting the value of a manipulated variable in automatic mode

In automatic mode, the manipulated variable is set to its automatic manipulated variable limits as specified by the input parameters `MV_HiLim` and `MV_LoLim` and output at the output parameter `MV`. Reaching the limit is then displayed at the output parameter `MV_HiAct = 1` for the high limit and `MV_LoAct = 1` for the low limit.

Interconnecting the output parameter `ManHiOut` or `ManLoOut` to `MV_HiLim` or `MV_LoLim` allows the automatic manipulated variable limits to tracked to manual manipulated variable limits. You can keep both limit pairs in synch and control the manual manipulated variable limits in the faceplate.

Limiting the value of a manipulated variable in manual mode

In manual mode, the manipulated variable is set to its manual manipulated variable limits as specified by the input parameters `ManHiLim` and `ManLoLim` and output at the output parameter `MV`.

See also

Forcing operating modes (Page 41)

2.1.10.8 Feedforwarding and limiting disturbance variables

Feedforward control and limitation

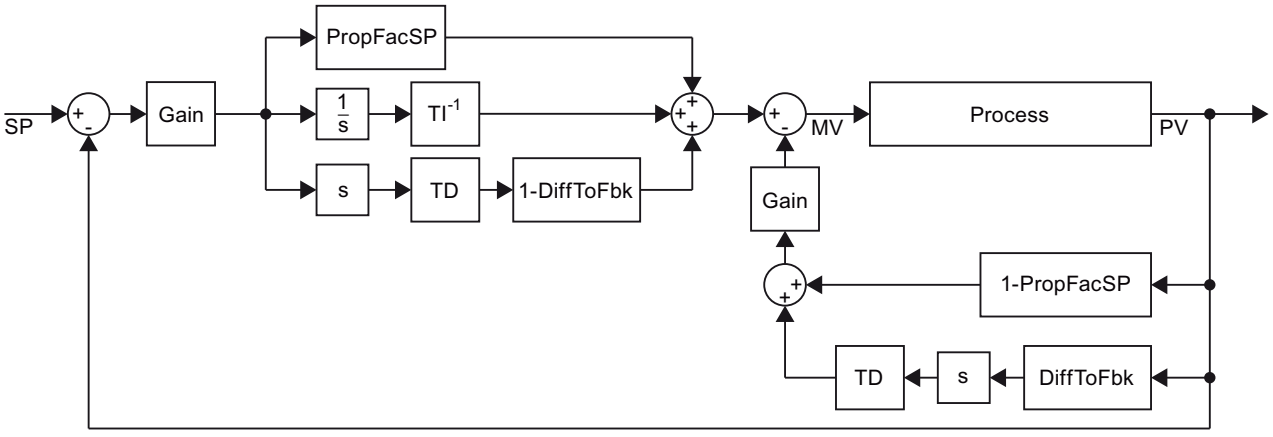
The feedforward control is used in order to compensate measurable disturbance variables, such as temperature or pressure, that can have an effect on the process. In automatic mode, the disturbance variable is added to the result of the PID algorithm.

The disturbance variable is connected to the `FFwd` Parameter. It is limited to the `FFwdHiLim` and `FFwdLoLim` limits. If the disturbance variable is outside or at the limits, this is indicated by the `FFwdHiAct = 1` or `FFwdLoAct = 1` output parameters.

2.1.10.9 Structure segmentation at controllers

Structure segmentation at controllers

In order to avoid jumps at the manipulated value (controller output) during setpoint changes, proportional and derivative actions can be switched into the feedback path. I.e.: The proportional action (proportionally) and the derivative action are then only influenced by the process value.



Switching proportional action into the feedback path

A proportion of the P action can be placed in the feedback using the `PropFacSP` parameter. Setpoint jumps then only affect the proportional action proportionally.

`PropFacSP = 0`: Proportional action is completely in the feedback path. Setpoint jumps do not affect the proportional action.

`PropFacSP = 1`: Proportional action is not in the feedback path. Proportional action is affected by setpoint and process variable (default setting).

Switching derivative action into the feedback path

The parameter `DiffToFbk` can be used to switch the D action into the feedback path. Setpoint jumps then no longer affect the derivative action directly.

`DiffToFbk = 0`: Derivative action is not in the feedback path. Derivative action is affected by setpoint and process variable (default setting).

`DiffToFbk = 1`: Derivative action is in the feedback path. Setpoint jumps do not affect the derivative action.

Implementation of the structure segmentation

The implementation of the structure segmentation differs from the figure shown above such that changing `PropFacSP` does not change `MV`. Control behavior corresponds to the figure.

2.1.11 Messaging

2.1.11.1 Area of application of the alarm delays

Area of application

A sensible area of application for setting alarm delays can, for example, be a motor. When it is started, an elevated starting current can occur and this could be reported depending on the configured limit. Since this usually settles down to a value below the set limit, the alarm would not make sense. In this case, the alarm delay that is intended to bridge the duration of the active alarm is used.

Note

Alarms that are really wanted are, naturally, also delayed when an alarm delay is used. Therefore select the delay period prudently!

Alarm delays in the Advanced Process Library

There are three block types with a different application of the alarm delay for:

- One time value for all limits (Page 195)
- One time value per limit pair (Page 196)
- Two time values per limit pair (Page 197)
- Two time values for each individual limit (Page 198)

Note

An alarm delay is displayed with the following symbol in the limit view and in the faceplate overview:

2.1.11.2 One time value for all limits

Blocks with one time value for the alarm delay

This form of alarm delay is used for blocks without a unit designation in the name, for example, ConPerMon.

The alarm delay is used when brief violations of the set alarm thresholds are to be suppressed. The duration of the time delay is parameterized at the input `AlmDelay`. Parameter assignment is always carried out in seconds.

Activating the alarm delay

The alarm delay is disabled by default (`AlmDelay = 0`). To use the function, set a delay time [s] with the `AlmDelay` parameter.

2.1.11.3 One time value per limit pair

Blocks with one time value for the alarm delay per limit pair

The alarm delay is used when brief violations of the set alarm thresholds are to be suppressed.

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. Please take into consideration the validity of terms for user-configured message classes (Page 41).

The alarm delay is configured at the following inputs:

Parameter for the delay time	Explanation
`XXX_A_DC`	Delay time for incoming events of the class • Alarm **XXX_A**larm_**D**elay**C**oming XXX: Value to be monitored
`XXX_W_DC`	Delay time for incoming events of the class • Warning **XXX_W**arning_**D**elay**C**oming XXX: Value to be monitored

Activating the alarm delay

By default, the alarm delay is disabled for each limit, meaning that each individual parameter has 0 [s] pre-assigned.

To use the function, set a delay time [s] for each parameter.

Pending alarms

Pending alarms, warnings, or tolerances are output at the corresponding output parameters:

- `XXX_AH_Act = 1`: Alarm limit (high) reached or violated
- `XXX_AL_Act = 1`: Alarm limit (low) reached or violated
- `XXX_WH_Act = 1`: Warning limit (high) reached or violated
- `XXX_WL_Act = 1`: Warning limit (low) reached or violated

If a message is active at one of these outputs, this is indicated by a 1.

See also

User-configured message classes (Page 41)

2.1.11.4 Two time values per limit pair

Blocks with two time values for the alarm delay per limit pair

The alarm delay is used when brief violations of the set alarm thresholds are to be suppressed.

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. Please take into consideration the validity of terms for user-configured message classes (Page 41).

The alarm delay is configured at the following inputs:

Parameter for the delay time	Explanation
XXX_A_DC	Delay time for incoming events of the class • Alarm XXX_Alarm_DelayComing XXX: Value to be monitored
XXX_A_DG	Delay time for outgoing events of the class • Alarm XXX_Alarm_DelayGoing XXX: Value to be monitored
XXX_W_DC	Delay time for incoming events of the class • Warning XXX_Warning_DelayComing XXX: Value to be monitored
XXX_W_DG	Delay time for outgoing events of the class • Warning XXX_Warning_DelayGoing XXX: Value to be monitored
XXX_T_DC	Delay time for incoming events of the class • Tolerance XXX_Tolerance_DelayComing XXX: Value to be monitored
XXX_T_DG	Delay time for outgoing events of the class • Tolerance XXX_Tolerance_DelayGoing XXX: Value to be monitored

Activating the alarm delay

By default, the alarm delay is disabled for each individual pair, meaning that each individual parameter has 0 [s] pre-assigned.

To use the function, set a delay time [s] for each parameter.

Pending alarms

Pending alarms, warnings, or tolerances are output at the corresponding output parameters:

- `XXX_AH_Act = 1`: Alarm limit (high) reached or violated
- `XXX_AL_Act = 1`: Alarm limit (low) reached or violated
- `XXX_WH_Act = 1`: Warning limit (high) reached or violated
- `XXX_WL_Act = 1`: Warning limit (low) reached or violated
- `XXX_TH_Act = 1`: Tolerance limit (high) reached or violated
- `XXX_TL_Act = 1`: Tolerance limit (low) reached or violated

If a message is active at one of these outputs, this is indicated by a 1.

See also

User-configured message classes (Page 41)

2.1.11.5 Two time values for each individual limit

Alarm delay for blocks with two time values for each individual limit

This form of alarm delay is used for the blocks PIDConL, PIDConR, and MonAnL.

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. Please take into consideration the validity of terms for user-configured message classes (Page 41).

The alarm delay is used when brief violations of the set alarm thresholds are to be suppressed. The alarm delay is configured at the following inputs:

Parameter for the delay time	Explanation
XX_AH_DC	Delay time for incoming events of the class • Alarm (for high limit) **XX_A**larm**H**igh_**D**elay**C**oming XX: Value to be monitored
XX_AH_DG	Delay time for outgoing events of the class • Alarm (for high limit) **XX_A**larm**H**igh_**D**elay**G**oing XX: Value to be monitored
XX_AL_DC[1)]	Delay time for incoming events of the class • Alarm (for low limit) **XX_A**larm**L**ow_**D**elay**C**oming XX: Value to be monitored
XX_AL_DG[1)]	Delay time for outgoing events of the class • Alarm (for low limit) **XX_A**larm**L**ow_**D**elay**G**oing XX: Value to be monitored
XX_WH_DC	Delay time for incoming events of the class • Warning (for high limit) **XX_W**arning**H**igh_**D**elay**C**oming XX: Value to be monitored
XX_WH_DG	Delay time for outgoing events of the class • Warning (for high limit) **XX_W**arning**H**igh_**D**elay**G**oing XX: Value to be monitored
XX_WL_DC[1)]	Delay time for incoming events of the class • Warning (for low limit) **XX_W**arning**L**ow_**D**elay**C**oming XX: Value to be monitored
XX_WL_DG[1)]	Delay time for outgoing events of the class • Warning (for low limit) **XX_W**arning**L**ow_**D**elay**G**oing XX: Value to be monitored
XX_TH_DC	Delay time for incoming events of the class • Tolerance (for high limit) **XX_T**olerance**H**igh_**D**elay**C**oming XX: Value to be monitored
XX_TH_DG	Delay time for outgoing events of the class • Tolerance (for high limit) **XX_T**olerance**H**igh_**D**elay**G**oing XX: Value to be monitored

Parameter for the delay time	Explanation
XX_TL_DC[1)]	Delay time for incoming events of the class • Tolerance (for low limit) **XX_T**olerance**L**ow_**D**elay**C**oming XX: Value to be monitored
XX_TL_DG[1)]	Delay time for outgoing events of the class • Tolerance (for low limit) **XX_T**olerance**L**ow_**D**elay**G**oing XX: Value to be monitored

[1)] In the "MonAnL" block, the inputs for a low delay time do not have the separate "L" letters (e.g. XX_AL_DC: the input name is XX_A_DC).

Activating the alarm delay

By default, the alarm delay is disabled for each individual limit, meaning that each individual parameter has 0 [s] pre-assigned.

To use the function, set a delay time [s] for each parameter.

Pending alarms

Pending alarms, warnings, or tolerances are output at the corresponding output parameters:

- `XX_AL_Act = 1`: Alarm limit (low) reached or violated
- `XX_AH_Act = 1`: Alarm limit (high) reached or violated
- `XX_WL_Act = 1`: Warning limit (low) reached or violated
- `XX_WH_Act = 1`: Warning limit (high) reached or violated
- `XX_TL_Act = 1`: Tolerance limit (low) reached or violated
- `XX_TH_Act = 1`: Tolerance limit (high) reached or violated

If a message is active at one of these outputs, this is indicated by a 1.

See also

User-configured message classes (Page 41)

Separate delay times for each alarm (Page 169)

2.1.11.6 Generating instance-specific messages

Generating instance-specific messages

You can generate instance-specific messages for a binary signal of every block.

The number of interconnectable input parameters that can be used freely varies with relation to the blocks. The X in the parameter name designates the position.

You can specify the following messages for these instance-specific messages:

- Message class
- Priority of the message
- Message text
- Message auxiliary value
- Acknowledge behavior

Additional information is available in the descriptions of the message functionality of the individual blocks and in the PCS 7 Configuration Manual Operator Station under "How to configure the user-specific messages".

See also

Time stamp (Page 201)

2.1.11.7 Suppressing messages using the MsgLock parameter

The `MsgLock = 1` parameter is used to selectively suppress the following messages depending on `Feature` bit 25:

- All messages at the block

or

- All messages at the block, except for process control messages (for example, `CSF`, motor protection, feedback error) and external messages.

Messages already queued received the "outgoing" status with `MsgLock = 1` .

You can find additional information on `Feature` bit 25 in the section Suppression of all messages (Page 173)

Note

With the block OpAnL, this function is independent of `Feature` bit 25.

2.1.11.8 Time stamp

Time stamp

The time stamp is the assignment of time information to the status change of a binary process signal. The status change of the signal is signaled together with the time information.

Use the EventTS block to report time stamped signals.

For more information on time stamping and how to configure it, please refer to the "PCS 7 - High-Precision Time Stamping Function Manual".

Areas of application

Areas of application of the time stamp are for example:

- Accurately-timed detection of problems in process-related equipment. The time stamp enables you to explicitly identify signals that indicate the cause of the failure of a process unit.
- Analysis of system-wide interrelationships
- Detection and reporting of the sequence of time-critical signal changes

Forming the time information

The time information is generated by one of the following methods and is specified at the block by means of the input parameter `TimeStampOn`:

- `TimeStampOn = 0`: Use time stamp of the CPU (default)
- `TimeStampOn = 1`: Use time stamp of the I/O

Time stamping in the EventTS block

Connect the binary output parameter of another block (e.g. Pcs7DiIn) with a message input Inx (x = 1 ... 8) of the EventTS block.

When the EventTS block recognizes a change in the signal state at this message input, it uses the current time of the CPU as the time stamp. Only the signal changes that are slower than the cycle time of the block can be detected.

High-precision time stamp in the process I/O

You have configured the hardware of your system for high-precision time stamping as explained in the "PCS 7 - High-Precision Time Stamping Function Manual". The signal changes are recognized in the I/O devices and the time stamp assigned to them. This data is available at the output parameter `TS_Out` of the Pcs7DiIT block.

The high-precision time stamp is independent of the cycle time of the blocks. The actual time resolution for two different status changes depends on your plant configuration and the hardware you are using.

Interconnect the output parameter `TS_Out` of Pcs7DiIT with a message input `InTSx(x = 1 ... 8)` of the EventTS block.

Error handling

The system block ImDrvTs recognizes when the time stamp function in the I/O devices is defective and forwards this information to the Pcs7DiIT block. This then forms the time stamp using the current CPU time and sets the signal status of `TS_Out` output parameter to "Bad, due to device". The EventTS block then uses the current time of the CPU as the time stamp. You can find additional information in the "PCS 7 - High-Precision Time Stamping Function Manual".

2.1.12 Settings for operator control and monitoring

2.1.12.1 Display and operator input area for process values and setpoints

Display and operator input area for process values and setpoints

You specify the upper and lower area limits in the faceplate using interconnectable input parameters for the following:

- Display areas (bar display)
- Operator input area (for example setpoint and manipulated value)
- Input area for the limit values
  - Up to 7 numbers (including a decimal separator and a minus sign) are possible in the faceplates

The interconnectable input parameter is a structured variable that contains two analog values. Refer to the descriptions of the individual blocks for the relevant input parameters.

Using a structured variable for scaling

There are three possibilities with which you can influence the contents of the structured variables, namely with:

- The corresponding channel function block, for example, the FbAnIn block
- A conversion block, for example, the StruScIn block
- Direct parameter assignment at the block input

Additional information on data types is available in the documentation of CFC and STEP 7.

2.1.12.2 Opening additional faceplates

Opening additional faceplates

You can open standard views of other faceplates from various faceplate views. Here, you have the following options:

- Two buttons that you can assign freely and that are used to call faceplates of other blocks.
- Two predefined buttons for calling faceplates with a fixed assignment to the controller blocks.
- Buttons predefined for interlock functions

Note

"Small" blocks

With "Small" blocks, you can call up only one faceplate from the standard view.

Freely assignable buttons

From the standard view and from the preview, you can use a button to open the standard view of a block that can be selected freely. In order to use this function, in the CFC you need to interconnect the `SelFp1` input parameter for the button in the standard view or `SelFp2` for the button in the preview to any given output parameter of the block whose faceplate is to be opened. This makes the buttons in the faceplates visible.

Note

You can only configure the button in the standard view (`SelFp1`) with interlock blocks. There are no buttons with the GainSched block.

Button label

You can change the button labels in the "OS additional text" attribute to "SelFp1"/"SelFp2".

Note

There is also an alternative solution for labeling the buttons:

- Open the process picture in WinCC GraphicsDesigner.
- Open the object properties of the block icon.
- Under Configurations, assign the desired text to the attribute UserButtonText1 or UserButtonText2.

The main difference between this solution and the first solution is that the labels are only specified when the faceplate above the block icon is opened. This solution is therefore outdated.

Predefined buttons for controller blocks

You can open the standard view for the following blocks from a controller standard view or parameter view (for example PIDConL):

- ConPerMon (can be called from the standard view)
  - To do this, you need to interconnect the output parameter `CPI` of the ConPerMon block to the input parameter `CPI_In` of the controller block.
- GainSched (can be called from the parameter view)
  - To do this, you need to interconnect the output parameter `Link2Gain` of the GainSched block to the input parameter `Gain` of the controller block.

The labels of the buttons cannot be changed here.

Predefined buttons for interlocks

You can open the following interlock blocks from the standard view of the technology blocks:

- Activation enable
- Interlock without reset (interlock)
- Interlock with reset (protection)

The buttons intended for this are visible when the relevant input parameter (`Permit`, `Intlock` or `Protect`) is interconnected to an interlock block.

You can open the standard view for the following faceplates from the standard view of the interlock blocks:

- The blocks interconnected to the input values.
- The block interconnected to the output value.

The buttons intended for this are visible when the input parameters (for example `In01`) or the `Out` output parameter of the interlock block is interconnected to a block that has a faceplate.

Note

Interconnection of the `Out` output parameter to multiple blocks is not permitted. The reason for this is that a direct relationship must be established between the button in the faceplate and the faceplate to be opened by it.

2.1.12.3 Labeling of buttons and text

Labeling of buttons and text

You can change the text of buttons (such as start/stop) for each specific instance. For binary OpDiXX faceplates, you can also change the static text for the "x command" for each specific instance.

To do this, you need to specify how the buttons are labeled yourself using the attributes "Text0" and "Text1" in the object properties of the block. These texts are displayed in the standard view and preview of the faceplate.

Block type	Command button	Block I/O for parameter assignment "Text 1"
FmCont	Open Close Stop	OpenOp CloseOp StopOp
FmTemp	Open Close Stop	OpenOp CloseOp StopOp
MonDiL	Process value	Out
MonDiS	Process value	Out

Block type	Command button	Block I/O for parameter assignment "Text 1"
MotL	Start	StartMan
	Stop	StopMan
	Rapid stop	RapidStp
MotRevL	Start - ->	FwdMan
	Start < - -	RevMan
	Stop	StopMan
	Rapid stop	RapidStp
MotS	Start	StartMan
	Stop	StopMan
MotSpdCL	Start |->	FwdMan
	Start < -|	RevMan
	Stop	StopMan
	Rapid stop	RapidStp
MotSpdL	Start >	Spd1Man
	Start >>	Spd2Man
	Stop	StopMan
	Rapid stop	RapidStp
PIDConL	Automatic	AutModOp
	Manual	ManModOp
	Program	AdvCoOn
	Out of service	OosOp
	External	SP_ExtOp
	Internal	SP_IntOp
PIDConR	Automatic	AutModOp
	Manual	ManModOp
	Program	AdvCoOn
	Out of service	OosOp
	External	SP_ExtOp
	Internal	SP_IntOp
PIDStepL	Open	OpenOp
	Close	CloseOp
	Stop	StopOp
VlvAnl	Open	OpenOp
	Close	CloseOp
VlvL	Open	OpenMan
	Close	CloseMan
Vlv2WayL	Pos0	Pos0Man
	Pos1	Pos1Man
	Pos2	Pos2Man
VlvMotL	Open	OpenMan
	Close	CloseMan
	Stop	StopMan
	Rapid stop	RapidStp

Block type	Command button	Block I/O for parameter assignment "Text 1"
VlvPosL	Open	OpenMan
	Close	CloseMan
	Stop	StopMan
	Rapid stop	RapidStp
VlvS	Open	OpenMan
	Close	CloseMan

The default text is shown if no text is configured.

If the text is longer than can be displayed with the default font size, the font size is automatically reduced until the text is fully shown. The smallest font size is 7 point.

Refer to the function description of the faceplate to learn which parameters are affected by this function.

2.1.12.4 Displaying auxiliary values

Displaying auxiliary values

Up to two auxiliary values can be displayed in the standard view of some faceplates. This feature can be used, for example, with motors to indicate the motor current and winding temperature.

To do so interconnect the value that you want to have displayed with the input parameters `UserAna1` or `UserAna2`.

In the object properties (I/Os > Identifier) of the block in CFC, you can specify the text to be displayed for these parameters in the standard view of the faceplate.

2.1.12.5 Selecting a unit of measure

Coded unit of measure

The parameter `XXX_Unit` is used to specify the unit of measure for the corresponding input parameter (XXX stands for a specific parameter, for example, `PV_Unit`). Entry is carried out in the form of a code. Exactly one unit of measure is assigned to each code and is displayed on the faceplate.

You can interconnect the `XXX_Unit` input parameter of a technologic block with the `XXXUnit` output parameter of an analog input channel block. At the analog input channel block, enter the unit of measure at the `XXXUnit` input parameter (`XXX` stands for a specific parameter, for example `PV_InUnit, PVOutUnit`).

If the parameter value of `XXX_Unit` is out of range (that is, the value is not defined), the faceplate displays "!undef.!" in the place of unit.

Note

Special notes for channel blocks PCS7AnIn, PCS7AnOu, FbAnIn and FbAnOu

You can use the S7_enum attribute to display the unit in plain text in the CFC editor for these blocks.

Note

In block icons, the update/refresh time of the units is 1 h. If the unit is changed from CFC, the new unit will be visible in the block icon after 1 hour or if the process picture is reloaded. In faceplates, the update/refresh time of the units is 5 seconds, so the new unit will be visible after 5 s.

Customer-specific units

It is possible to use units which differ from the IEC 611582 standard. You can define your own units in the range 1 to 199 in an XML file.

The name of the XML should be APLCustomerUnits.xml and should be placed in the project path in the folder "GraCS" on both server and client.

Below is an example to describe the content of the XML file APLCustomerUnits.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<UserDefinedUnits>                          <!-- Root node start -->
    <!-- Define the first unit for different languages -->
    <Unit id="1">                           <!-- first node with unit ID 1 start -->
      <Name lcid="1031">German1</Name>      <!-- unit value (e.g. German1) in German -->
      <Name lcid="1033">English1</Name>     <!-- unit value (e.g. English1) in English -->
      <Name lcid="1034">Spanish1</Name>     <!-- unit value (e.g. Spanish1) in Spanish -->
      <Name lcid="1036">French1</Name>      <!-- unit value (e.g. French1) in French -->
      <Name lcid="1040">Italian1</Name>     <!-- unit value (e.g. Italian1) in Italian -->
      <Name lcid="1041">Japanese1</Name>   <!-- unit value (e.g. Japanese1) in Japanese -->
      <Name lcid="2052">Chinese1</Name>     <!-- unit value (e.g. Chinese1) in Chinese -->
    </Unit>                                 <!-- first node with unit ID 1 end -->
    <!-- Define the second unit for different languages -->
    <Unit id="2">
      <Name lcid="1031">German2</Name>      <!-- lcid="1031" is for German -->
      <Name lcid="1033">English2</Name>     <!-- lcid="1033" is for English -->
      <Name lcid="1034">Spanish2</Name>     <!-- lcid="1034" is for Spanish -->
      <Name lcid="1036">French2</Name>      <!-- lcid="1036" is for French -->
      <Name lcid="1040">Italian2</Name>     <!-- lcid="1040" is for Italian -->
      <Name lcid="1041">Japanese2</Name>    <!-- lcid="1041" is for Japanese -->
      <Name lcid="2052">Chinese2</Name>     <!-- lcid="2052" is for Chinese -->
    </Unit>
```

```
<!-- similarly define other units up to unit id = 199-->
</UserDefinedUnits>                          <!-- Root node end -->
```

The "Unit id" value should be given to `XXX_Unit` parameter in the function blocks.

Note

The XML is read-only once when WinCC runtime is activated. If any changes are done in the XML later, you must deactivate, close, and activate the WinCC runtime again to reflect the changes.

Using the unit of measure with controllers for the ConPerMon block

For controller blocks, the current unit of measure is output via output parameter `XX_UnitOut` . If you use the ConPerMon block, you must switch this output parameter with the corresponding input parameter `XXX_Unit` on the ConPerMon block.

Using the S7_unit attribute

If you set the `xxx_Unit` parameter to 0, the entry is displayed by the `S7_Unit` attribute in the faceplate and in the block symbol.

Overview of the units of measure

The units of measure are listed in the following tables:

List of the most commonly used units of measure in accordance with IEC 61158

Note

Due to space constraints on the system, not all units referred to can be displayed in totality. The display of the measurement units is left-aligned. If you want to abbreviate the unit of measurement, use the function **Customer-specific units**.

Value	Display	Description
1000	K	Kelvin
1001	°C	Degrees Celsius
1002	°F	Degrees Fahrenheit
1005	°	Degree
1006	'	Minute
1007	''	Second
1010	m	Meter
1013	mm	Millimeter
1018	ft	Foot
1023	m^2	Square meter
1038	L	Liter

Value	Display	Description
1041	hl	Hectoliter
1054	s	Second
1058	min	Minute
1059	h	Hour
1060	d	Day
1061	m/s	Meters per second
1077	Hz	Hertz
1081	kHz	Kilohertz
1082	1/s	Per second
1083	1/min	Per minute
1088	kg	Kilogram
1092	t	Metric ton
1100	g/cm^3	Grams per cubic centimeter
1105	g/L	Grams per liter
1120	N	Newton
1123	mN	Millinewton
1130	Pa	Pascal
1133	kPa	Kilopascal
1137	bar	Bar
1138	mbar	Millibar
1149	mmH_2O	Millimeters of water[1]
1175	W·h	Watt hour
1179	kW·h	Kilowatt hour
1181	$kcal_{th}$	Kilocalories[1]
1190	kW	Kilowatt
1209	A	Ampere
1211	mA	Milliampere
1221	A·h	Ampere hour
1240	V	Volt
1349	m^3/h	Cubic meters per hour
1353	L/h	Liters per hour
1384	mol	Mol
1422	pH	pH value

List of all units of measure in accordance with IEC 61158

Value	Display	Description
1000	K	Kelvin
1001	°C	Degrees Celsius
1002	°F	Degrees Fahrenheit
1003	°R	Degree Rankine
1004	rad	Radian

Value	Display	Description
1005	°	Degree
1006	'	Minute
1007	"	Second
1008	gon	Gon
1009	r	Revolution
1010	m	Meter
1011	km	Kilometer
1012	cm	Centimeter
1013	mm	Millimeter
1014	µm	Micrometer
1015	nm	Nanometer
1016	pm	Picometer
1017	Å	Angstrom [2]
1018	ft	Foot
1019	in	Inch
1020	yd	Yard
1021	mile	Mile
1022	nautical mile	Nautical mile
1023	m^2	Square meter
1024	km^2	Square kilometer
1025	cm^2	Square centimeter
1026	dm^2	Square decimeter
1027	mm^2	Square millimeter
1028	a	Are
1029	ha	Hectare
1030	in^2	Square inch
1031	ft^2	Square foot
1032	yd^2	Square yard
1033	$mile^2$	Square mile
1034	m^3	Cubic meter
1035	dm^3	Cubic decimeter
1036	cm^3	Cubic centimeter
1037	mm^3	Cubic millimeter
1038	L	Liter
1039	cl	Centiliter
1040	ml	Milliliter
1041	hl	Hectoliter
1042	in^3	Cubic inch
1043	ft^3	Cubic foot
1044	yd^3	Cubic yard
1045	$mile^3$	Cubic mile
1046	pint	Pint
1047	quart	Quart

Value	Display	Description
1048	gal	US gallon
1049	ImpGal	Imperial gallon
1050	bushel	Bushel
1051	bbl	Barrel = 42 gallons
1052	bbl(liq)	Liquid barrel = 31.5 gallons
1053	ft^3 std.	Standard cubic foot
1054	s	Second
1055	ks	Kilosecond
1056	ms	Millisecond
1057	µs	Microsecond
1058	min	Minute
1059	h	Hour
1060	d	Day
1061	m/s	Meters per second
1062	mm/s	Millimeters per second
1063	m/h	Meters per hour
1064	km/h	Kilometers per hour
1065	knot	Knot
1066	in/s	Inches per second
1067	ft/s	Feet per second
1068	yd/s	Yards per second
1069	in/min	Inches per minute
1070	ft/min	Feet per minute
1071	yd/min	Yards per minute
1072	in/h	Inches per hour
1073	ft/h	Feet per hour
1074	yd/h	Yards per hour
1075	mi/h	Miles per hour
1076	m/s^2	Meter/second squared
1077	Hz	Hertz
1078	THz	Terahertz
1079	GHz	Gigahertz
1080	MHz	Megahertz
1081	kHz	Kilohertz
1082	1/s	Per second
1083	1/min	Per minute
1084	r/s	Revolutions per second
1085	rpm	Revolutions per minute
1086	rad/s	Radians per second
1087	$1/s^2$	Per second squared
1088	kg	Kilogram
1089	g	Gram
1090	mg	Milligram

Value	Display	Description
1091	Mg	Megagram
1092	t	Metric ton
1093	oz	Ounce
1094	lb	Pound
1095	STon	US ton (short ton)
1096	LTon	British ton (long ton)
1097	kg/m^3	Kilograms per cubic meter
1098	Mg/dm^3	Megagrams per cubic meter
1099	kg/dm^3	Kilograms per cubic decimeter
1100	g/cm^3	Grams per cubic centimeter
1101	g/m^3	Grams per cubic meter
1102	t/m^3	Metric tons per cubic meter
1103	kg/L	Kilogram per liter
1104	g/ml	Grams per milliliter
1105	g/L	Grams per liter
1106	lb/in^3	Pounds per cubic inch
1107	lb/ft^3	Pounds per cubic foot
1108	lb/gal	Pounds per US gallon
1109	$STon/yd^3$	US tons per cubic yard
1110	°Twad	Degree Twaddell
1111	°Baum (hv)	Degree Baumé (heavy)
1112	°Baum (lt)	Degree Baumé (light)
1113	°API	Degrees API
1114	SGU	Specific gravity units
1115	kg/m	Kilograms per meter
1116	mg/m	Milligrams per meter
1117	tex	Tex
1118	$kg \cdot m^2$	Kilograms per square meter
1119	kg·m/s	Kilograms per meter per second
1120	N	Newton
1121	MN	Meganewton
1122	kN	Kilonewton
1123	mN	Millinewton
1124	µN	Micronewton
1125	$kg \cdot m^2/s$	Kilograms per square meter per second
1126	N·m	Newton meter
1127	MN·m	Meganewton meter
1128	kN·m	Kilonewton meter
1129	mN·m	Millinewton meter
1130	Pa	Pascal
1131	GPa	Gigapascal
1132	MPa	Megapascal

Value	Display	Description
1133	kPa	Kilopascal
1134	mPa	Millipascal
1135	µPa	Micropascal
1136	hPa	Hectopascal
1137	bar	Bar
1138	mbar	Millibar
1139	torr	Torr
1140	atm	Atmosphere
1141	psi	Pounds per square inch
1142	psia	Pounds per square inch (absolute)
1143	psig	Pounds per square inch (gauge)
1144	g/cm^2	Grams per square centimeter
1145	kg/cm^2	Kilograms per square centimeter
1146	inH_2O	Inches of water
1147	inH_2O (4°C)	Inches of water at 4 degrees Celsius
1148	inH_2O (68°F)	Inches of water at 68 degrees Fahrenheit
1149	mmH_2O	Millimeters of water[1]
1150	mmH_2O (4°C)	Millimeters of water at 4 degrees Celsius[1]
1151	mmH_2O (68°F)	Millimeters of water at 68 degrees Fahrenheit[1]
1152	ftH_2O	Feet of water[1]
1153	ftH_2O (4°C)	Feet of water at 4 degrees Celsius[1]
1154	ftH_2O (68°F)	Feet of water at 68 degrees Fahrenheit[1]
1155	inHg	Inches of mercury
1156	inHg (0°C)	Inches of mercury at 0 degrees Celsius
1157	mmHg	Millimeters of mercury
1158	mmHg (0°C)	Millimeters of mercury at 0 degrees Celsius
1159	Pa·s	Pascal second
1160	m^2/s	Square meters per second
1161	P	Poise
1162	cP	Centipoise
1163	St	Stokes
1164	cSt	Centistokes
1165	N/m	Newtons per meter
1166	mN/m	Millinewtons per meter
1167	J	Joule
1168	EJ	Exajoule

Value	Display	Description
1169	PJ	Petajoule
1170	TJ	Terajoule
1171	GJ	Gigajoule
1172	MJ	Megajoule
1173	kJ	Kilojoule
1174	mJ	Millijoule
1175	W·h	Watt hour
1176	TW·h	Terawatt hour
1177	GW·h	Gigawatt hour
1178	MW·h	Megawatt hour
1179	kW·h	Kilowatt hour
1180	cal_{th}	Calorie (thermo chemical)[1]
1181	$kcal_{th}$	Kilocalorie (thermo chemical)[1]
1182	$Mcal_{th}$	Megacalorie (thermo chemical)[1]
1183	Btu_{th}	British thermal unit[1]
1184	datherm	Decatherm
1185	ft·lbf	Foot pound
1186	W	Watt
1187	TW	Terawatt
1188	GW	Gigawatt
1189	MW	Megawatt
1190	kW	Kilowatt
1191	mW	Milliwatt
1192	µW	Microwatt
1193	nW	Nanowatt
1194	pW	Picowatt
1195	$Mcal_{th}/h$	Megacalorie per hour[1]
1196	MJ/h	Megajoule per hour
1197	Btu_{th}/h	British thermal units per hour[1]
1198	hp	Horsepower
1199	W/(m·K)	Watts per meter kelvin
1200	$W/(m^2·K)$	Watts per (square meter kelvin)
1201	$m^2·K/W$	Square meters kelvin per Watt
1202	J/K	Joules per kelvin
1203	kJ/K	Kilojoules per kelvin
1204	J/(kg·K)	Joules per (kilogram kelvin)
1205	kJ/(kg·K)	Kilojoules per (kilogram kelvin)
1206	J/kg	Joules per kilogram
1207	MJ/kg	Megajoules per kilogram
1208	kJ/kg	Kilojoules per kilogramm
1209	A	Ampere
1210	kA	Kiloampere
1211	mA	Milliampere

Value	Display	Description
1212	µA	Microampere
1213	nA	Nanoampere
1214	pA	Picoampere
1215	C	Coulomb
1216	MC	Megacoulomb
1217	kC	Kilocoulomb
1218	µC	Microcoulomb
1219	nC	Nanocoulomb
1220	pC	Picocoulomb
1221	A·h	Ampere hour
1222	C/m^3	Coulombs per cubic meter
1223	C/mm^3	Coulombs per cubic millimeter
1224	C/cm^3	Coulombs per cubic centimeter
1225	kC/m^3	Kilocoulombs per cubic meter
1226	mC/m^3	Millicoulombs per cubic meter
1227	$µC/m^3$	Microcoulombs per cubic meter
1228	C/m^2	Coulombs per square meter
1229	C/mm^2	Coulombs per square millimeter
1230	C/cm^2	Coulombs per square centimeter
1231	kC/m^2	Kilocoulombs per square meter
1232	mC/m^2	Millicoulombs per square meter
1233	$µC/m^2$	Microcoulombs per square meter
1234	V/m	Volts per meter
1235	MV/m	Megavolts per meter
1236	kV/m	Kilovolts per meter
1237	V/cm	Volts per centimeter
1238	mV/m	Millivolts per meter
1239	µV/m	Microvolts per meter
1240	V	Volt
1241	MV	Megavolt
1242	kV	Kilovolt
1243	mV	Millivolt
1244	µV	Microvolt
1245	F	Farad
1246	mF	Millifarad
1247	µF	Microfarad
1248	nF	Nanofarad
1249	pF	Picofarad
1250	F/m	Farad per meter
1251	µF/m	Microfarad per meter
1252	nF/m	Nanofarad per meter
1253	pF/m	Picofarad per meter
1254	C·m	Coulomb meter

Value	Display	Description
1255	A/m^2	Amperes per square meter
1256	MA/m^2	Megaamperes per square meter
1257	A/cm^2	Amperes per square centimeter
1258	kA/m^2	Kiloamperes per square meter
1259	A/m	Amperes per meter
1260	kA/m	Kiloamperes per meter
1261	A/cm	Amperes per centimeter
1262	T	Tesla
1263	mT	Millitesla
1264	µT	Microtesla
1265	nT	Nanotesla
1266	Wb	Weber
1267	mWb	Milliweber
1268	Wb/m	Webers per meter
1269	kWb/m	Kilowebers per meter
1270	H	Henry
1271	mH	Millihenry
1272	µH	Microhenry
1273	nH	Nanohenry
1274	pH	Picohenry
1275	H/m	Henries per meter
1276	µH/m	Microhenries per meter
1277	nH/m	Nanohenries per meter
1278	$A \cdot m^2$	Ampere square meters
1279	$N \cdot m^2/A$	Newton meter squared per ampere
1280	Wb·m	Weber meter
1281	Ω	Ohm[1]
1282	GΩ	Gigaohm[1]
1283	MΩ	Megaohm[1]
1284	kΩ	Kiloohm[1]
1285	mΩ	Milliohm[1]
1286	µΩ	Microohm[1]
1287	S	Siemens
1288	kS	Kilosiemens
1289	mS	Millisiemens
1290	µS	Microsiemens
1291	Ω·m	Ohms times meters[1]
1292	GΩ·m	Gigaohms times meters[1]
1293	MΩ·m	Megaohms times meters[1]
1294	kΩ·m	Kiloohms times meters[1]
1295	Ω·cm	Ohms times centimeters[1]
1296	mΩ·m	Milliohms times meters[1]
1297	µΩ·m	Microohms times meters[1]

Value	Display	Description
1298	nΩ·m	Nanoohms times meters[1]
1299	S/m	Siemens per meter
1300	MS/m	Megasiemens per meter
1301	kS/m	Kilosiemens per meter
1302	mS/cm	Millisiemens per centimeter
1303	µS/mm	Microsiemens per millimeter
1304	1/H	Per henry
1305	sr	Steradian
1306	W/sr	Watts per steradian
1307	W/(sr·m^2)	Watts per (steradian square meter)
1308	W/(m^2)	Watts per square meter
1309	lm	Lumen
1310	lm·s	Lumen second
1311	lm·h	Lumen hour
1312	lm/m^2	Lumens per square meter
1313	lm/W	Lumens per watt
1314	lx	Lux
1315	lx·s	Lux second
1316	cd	Candela
1317	cd/m^2	Candela per square meter
1318	g/s	Grams per second
1319	g/min	Grams per minute
1320	g/h	Grams per hour
1321	g/d	Grams per day
1322	kg/s	Kilograms per second
1323	kg/min	Kilograms per minute
1324	kg/h	Kilograms per hour
1325	kg/d	Kilograms per day
1326	t/s	Metric tons per second
1327	t/min	Metric tons per minute
1328	t/h	Metric tons per hour
1329	t/d	Metric tons per day
1330	lb/s	Pounds per second
1331	lb/min	Pounds per minute
1332	lb/h	Pounds per hour
1333	lb/d	Pounds per day
1334	STon/s	US tons per second
1335	STon/min	US tons per minute
1336	STon/h	US tons per hour
1337	STon/d	US tons per day
1338	LTon/s	British tons per second
1339	LTon/min	British tons per minute
1340	LTon/h	British tons per hour

Value	Display	Description
1341	LTon/d	British tons per day
1342	%	Percent
1343	% sol/wt	Percentage solids per weight unit
1344	% sol/vol	Percentage solids per volume unit
1345	% stm qual	Percentage steam quality
1346	°Plato	Degree plato
1347	m^3/s	Cubic meters per second
1348	m^3/min	Cubic meters per minute
1349	m^3/h	Cubic meters per hour
1350	m^3/d	Cubic meters per day
1351	L/s	Liters per second
1352	L/min	Liters per minute
1353	L/h	Liters per hour
1354	L/d	Liters per day
1355	ML/d	Megaliters per day
1356	ft^3/s	Cubic feet per second
1357	ft^3/m	Cubic feet per minute
1358	ft^3/h	Cubic feet per hour
1359	ft^3/d	Cubic feet per day
1360	ft^3/min std	Standard cubic feet per minute
1361	ft^3/h std	Standard cubic feet per hour
1362	gal/s	US gallons per second
1363	gal/min	US gallons per minute
1364	gal/h	US gallons per hour
1365	gal/d	US gallons per day
1366	Mgal/d	Mega US gallons per day
1367	ImpGal/s	Imperial gallons per second
1368	ImpGal/min	Imperial gallons per minute
1369	ImpGal/h	Imperial gallons per hour
1370	ImpGal/d	Imperial gallons per day
1371	bbl/s	Barrels per second
1372	bbl/min	Barrels per minute
1373	bbl/h	Barrels per hour
1374	bbl/d	Barrels per day
1375	W/m^2	Watts per square meter
1376	mW/m^2	Milliwatts per square meter
1377	$\mu W/m^2$	Microwatts per square meter
1378	pW/m^2	Picowatts per square meter
1379	$Pa \cdot s/m^3$	Pascal seconds per cubic meter
1380	N·s/m	Newton seconds per meter
1381	Pa·s/m	Pascal seconds per meter
1382	B	Bel
1383	dB	Decibel

Value	Display	Description
1384	mol	Mol
1385	kmol	Kilomole
1386	mmol	Millimole
1387	µmol	Micromole
1388	kg/mol	Kilograms per mole
1389	g/mol	Grams per mole
1390	m^3/mol	Cubic meters per mole
1391	dm^3/mol	Cubic decimeters per mole
1392	cm^3/mol	Cubic centimeters per mole
1393	L/mol	Liters per mole
1394	J/mol	Joules per mole
1395	kJ/mol	Kilojoules per mole
1396	J/(mol·K)	Joules per mole kelvin
1397	mol/m^3	Moles per cubic meter
1398	mol/dm^3	Moles per cubic decimeter
1399	mol/L	Moles per liter
1400	mol/kg	Moles per kilogram
1401	mmol/kg	Millimoles per kilogram
1402	Bq	Becquerel
1403	MBq	Megabecquerel
1404	kBq	Kilobecquerel
1405	Bq/kg	Becquerels per kilogram
1406	kBq/kg	Kilobecquerels per kilogram
1407	MBq/kg	Megabecquerels per kilogram
1408	Gy	Gray
1409	mGy	Milligray
1410	rd	Rad
1411	Sv	Sievert
1412	mSv	Millisievert
1413	rem	Rem
1414	C/kg	Coulombs per kilogram
1415	mC/kg	Millicoulombs per kilogram
1416	R	Röntgen
1417	1/Jm^3	Density of magnetic energy
1418	e/Vm^3	
1419	m^3/C	Cubic meters per coulomb
1420	V/K	Volts per kelvin
1421	mV/K	Millivolts per kelvin
1422	pH	pH value
1423	ppm	Parts per million
1424	ppb	Parts per billion
1425	ppth	Parts per trillion
1426	°Brix	Degrees Brix

Value	Display	Description
1427	°Ball	Degrees Balling
1428	proof/vol	Proof per volume
1429	proof/mass	Proof per mass
1430	lb/ImpGal	Pounds per Imperial gallon
1431	$kcal_{th}/s$	Kilocalories per second[1]
1432	$kcal_{th}/min$	Kilocalories per minute[1]
1433	$kcal_{th}/h$	Kilocalories per hour[1]
1434	$kcal_{th}/d$	Kilocalories per day[1]
1435	$Mcal_{th}/s$	Megacalories per second[1]
1436	$Mcal_{th}/min$	Megacalories per minute[1]
1437	$Mcal_{th}/d$	Megacalories per day[1]
1438	kJ/s	Kilojoules per second
1439	kJ/min	Kilojoules per minute
1440	kJ/h	Kilojoule per hour
1441	kJ/d	Kilojoules per day
1442	MJ/s	Megajoules per second
1443	MJ/min	Megajoules per minute
1444	MJ/d	Megajoules per day
1445	Btu_{th}/s	British thermal units per second[1]
1446	Btu_{th}/min	British thermal units per minute[1]
1447	Btu_{th}/d	British thermal units per day[1]
1448	µgal/s	Micro US gallons per second
1449	mgal/s	Milli US gallons per second
1450	kgal/s	Kilo US gallons per second
1451	Mgal/s	Mega US gallons per second
1452	µgal/min	Micro US gallons per minute
1453	mgal/min	Milli US gallons per minute
1454	kgal/min	Kilo US gallons per minute
1455	Mgal/min	Mega US gallons per minute
1456	µgal/h	Micro US gallons per hour
1457	mgal/h	Milli US gallons per hour
1458	kgal/h	Kilo US gallons per hour
1459	Mgal/h	Mega US gallons per hour
1460	µgal/d	Micro US gallons per day
1461	mgal/d	Milli US gallons per day
1462	kgal/d	Kilo US gallons per day
1463	µImpGal/s	Micro Imperial gallons per second
1464	mImpGal/s	Milli Imperial gallons per second
1465	kImpGal/s	Kilo Imperial gallons per second
1466	MImpGal/s	Mega Imperial gallons per second
1467	µImpGal/min	Micro Imperial gallons per minute
1468	mImpGal/min	Milli Imperial gallons per minute
1469	kImpGal/min	Kilo Imperial gallons per minute

Value	Display	Description
1470	MImpGal/min	Mega Imperial gallons per minute
1471	µImpGal/h	Micro Imperial gallons per hour
1472	mImpGal/h	Milli Imperial gallons per hour
1473	kImpGal/h	Kilo Imperial gallons per hour
1474	MImpGal/h	Mega Imperial gallons per hour
1475	µImpgal/d	Micro Imperial gallons per day
1476	mImpgal/d	Milli Imperial gallons per day
1477	kImpgal/d	Kilo Imperial gallons per day
1478	MImpgal/d	Mega Imperial gallons per day
1479	µbbl/s	Microbarrels per second
1480	mbbl/s	Millibarrels per second
1481	kbbl/s	Kilobarrels per second
1482	Mbbl/s	Megabarrels per second
1483	µbbl/min	Microbarrels per minute
1484	mbbl/min	Millibarrels per minute
1485	kbbl/min	Kilobarrels per minute
1486	Mbbl/min	Megabarrels per minute
1487	µbbl/h	Microbarrels per hour
1488	mbbl/h	Millibarrels per hour
1489	kbbl/h	Kilobarrels per hour
1490	Mbbl/h	Megabarrels per hour
1491	µbbl/d	Microbarrels per day
1492	mbbl/d	Millibarrels per day
1493	kbbl/d	Kilobarrels per day
1494	Mbbl/d	Megabarrels per day
1495	$\mu m^3/s$	Cubic micrometers per second
1496	mm^3/s	Cubic millimeters per second
1497	km^3/s	Cubic kilometers per second
1498	Mm^3/s	Cubic megameters per second
1499	$\mu m^3/min$	Cubic micrometers per minute
1500	mm^3/min	Cubic millimeters per minute
1501	km^3/min	Cubic kilometers per minute
1502	mm^3/min	Cubic megameters per minute
1503	$\mu m^3/h$	Cubic micrometers per minute
1504	mm^3/h	Cubic millimeters per minute
1505	km^3/h	Cubic kilometers per minute
1506	Mm^3/h	Cubic megameters per minute
1507	$\mu m^3/d$	Cubic micrometers per day
1508	mm^3/d	Cubic millimeters per day
1509	km^3/d	Cubic kilometers per day
1510	Mm^3/d	Cubic megameters per day
1511	cm^3/s	Cubic centimeters per second
1512	cm^3/min	Cubic centimeters per minute

Value	Display	Description
1513	cm^3/h	Cubic centimeters per hour
1514	cm^3/d	Cubic centimeters per day
1515	$kcal_{th}/kg$	Kilocalories per kilogram[1]
1516	Btu_{th}/lb	British thermal units per pound[1]
1517	kL	Kiloliter
1518	kL/min	Kiloliters per minute
1519	kL/h	Kiloliters per hour
1520	kL/d	Kiloliters per day
1551	S/cm	Siemens per centimeter
1552	µS/cm	Microsiemens per centimeter
1553	mS/m	Millisiemens per meter
1554	µS/m	Microsiemens per meter
1555	MΩ · cm	Megaohm centimeter[1]
1556	kΩ · cm	Kiloohm centimeter[1]
1557	Weight%	Weight percent
1558	mg/L	Milligram per liter
1559	µg/L	Microgram per liter
1560	%Sat	-
1561	vpm	-
1562	%vol	Volume percent
1563	ml/min	Milliliters per minute
1564	mg/dm^3	Milligrams per cubic centimeter
1565	mg/L	Milligram per liter
1566	mg/m^3	Milligrams per cubic meter
1567	ct	Carat (jewels) = $200.0 \cdot 10^{-6}$ kg
1568	lb (tr)	Pound (troy or apothecary) = 0.3732417216 kg
1569	oz (tr)	Ounce (troy or apothecary) = 1/12 lb (tr)
1570	fl oz (U.S.)	Ounce (U.S. fluid) = (1/128) gal
1571	cm^3	Cubic centimeter = 10^{-6} m^3
1572	af	acre foot = 43560 ft^3
1573	m^3 normal	Cubic meter
1574	L normal	Liter
1575	m^3 std.	Standard cubic meter
1576	L std.	Standard liter
1577	ml/s	Milliliters per second
1578	ml/h	Milliliters per hour
1579	ml/d	Milliliters per day
1580	af/s	Acre foot per second
1581	af/min	Acre foot per minute
1582	af/h	Acre foot per hour
1583	af/d	Acre foot per day

Value	Display	Description
1584	fl oz (U.S.)/s	Ounces per second
1585	fl oz (U.S.) /min	Ounces per minute
1586	fl oz (U.S.)/h	Ounces per hour
1587	fl oz (U.S.)/d	Ounces per day
1588	m^3/s normal	Standard cubic meters per second
1589	m^3/min normal	Standard cubic meters per minute
1590	m^3/h normal	Standard cubic meters per hour
1591	m^3/d normal	Standard cubic meters per day
1592	L/s normal	Standard liters per second
1593	L/min normal	Standard liters per minute
1594	L/h normal	Standard liters per hour
1595	L/d normal	Standard liters per second
1596	m^3/s std.	Standard cubic meters per second
1597	m^3/min std.	Standard cubic meters per minute
1598	m^3/h std.	Standard cubic meters per hour
1599	m^3/d std.	Standard cubic meters per day
1600	L/s std.	Standard liters per second
1601	L/min std.	Standard liters per minute
1602	L/h std.	Standard liters per hour
1603	L/d std.	Standard liters per day
1604	ft^3/s std.	Standard cubic feet per second
1605	ft^3/d std.	Standard cubic feet per day
1606	oz/s	Ounces per second
1607	oz/min	Ounces per minute
1608	oz/h	Ounces per hour
1609	oz/d	Ounces per day
1610	Paa	Pascal (absolute)
1611	Pag	Pascal (gauge)
1612	GPaa	Gigapasacal (absolute)
1613	GPag	Gigapascal (gauge)
1614	MPaa	Megapascal (absolute)
1615	MPag	Megapascal (gauge)
1616	kPaa	Kilopascal (absolute)
1617	kPag	Kilopascal (gauge)
1618	mPaa	Millipascal (absolute)
1619	mPag	Millipascal (gauge)
1620	µPaa	Micropascal (absolute)
1621	µPag	Micropascal (gauge)
1622	hPaa	Hectopascal (absolute)
1623	hPag	Hectopascal (gauge)
1624	gf/cm^2a	
1625	gf/cm^2g	
1626	kgf/cm^2a	

Value	Display	Description
1627	kgf/cm^{2}g	
1628	SD4°C	Standard density at 4°C
1629	SD15°C	Standard density at 15°C
1630	SD20°C	Standard density at 20°C
1631	PS	Metric horsepower
1632	ppt	Parts per trillion = 10^{12}
1633	hl/s	Hectoliters per second
1634	hl/min	Hectoliters per minute
1635	hl/h	Hectoliters per hour
1636	hl/d	Hectoliters per day
1637	bbl (liq)/s	Barrels (US liquid) per second
1638	bbl (liq)/min	Barrels (US liquid) per minute
1639	bbl (liq)/h	Barrels (US liquid) per hour
1640	bbl (liq)/d	Barrels (US liquid) per day
1641	bbl (fed)	Barrel (U.S. federal) = 31 gallons
1642	bbl (fed)/s	Barrels (US federal) per second
1643	bbl (fed)/min	Barrels (US federal) per minute
1644	bbl (fed)/h	Barrels (US federal) per hour
1645	bbl (fed)/d	Barrels (US federal) per day
1998	Unknown unit	To be used when the unit of measure is not known during configuration
1999	Special	Special units

[1] A notation different to the PA profile must be used in order to represent this unit in conformity with the system.

[2] "m2" is used instead of "m3" in the Chinese and Japanese documentation.

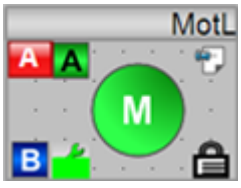
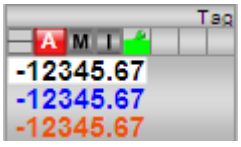
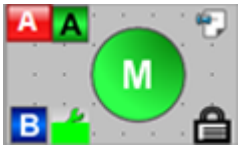
2.2 Functions of the block icons

2.2.1 Block icon structure

Block icon structure

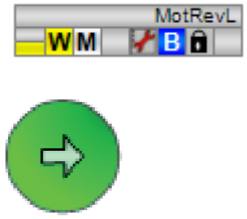
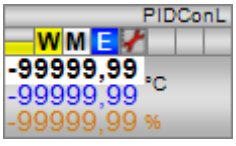
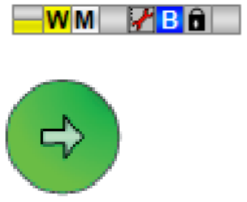
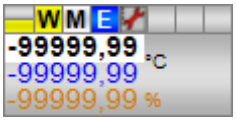
The new block icons are located in the template "@PCS7TypicalsAPL8.pdl" and "@TemplateAPL8.PDL".

There are two types of block icons (V8.0), those with a display of the instance-specific name and those without:

	Block icon of MotL with instance-specific name
	Block icon of PIDConL with instance-specific name
	Block icon of MotL without instance-specific name The toolbar only shows the information that is actually available.
	Block icon of PIDConL without instance-specific name The toolbar only shows the information that is actually available.

The old block icons are located in the template "PCS7TypicalsAPL7.pdl" and "@TemplateAPL7.PDL".

There are two types of block icons (V7.1), those with a display of the instance-specific name and those without:

	Block icon of MotRevL with instance-specific name
	Block icon of PIDConL with instance-specific name
	Block icon of MotRevL without instance-specific name The toolbar only shows the information that is actually available.
	Block icon of PIDConL without instance-specific name The toolbar only shows the information that is actually available.

You can select one of these block icons. See section Configuring the block icons (Page 233) for more information.

A block icon has several display areas:

- CPU stop
- Instance-specific name
- Icon for the block
- Analog value display
- Status bar for the block status

Displaying CPU stop

With a CPU stop, boxes are unavailable and a yellow warning triangle is displayed in the group display for blocks with messaging.

Instance-specific name

The name of the associated block is shown in the instance-specific name, for example for the PIDConL block:

You can change this name in the object properties of the instance block.

There are block icons with or without display of the instance-specific name. Refer to the individual block descriptions to learn about them.

You can reach the visible display for blocks without display of the instance-specific name in two different ways:

- **Displaying individual instance-specific names**: Click on the block icon while holding down the Shift key: The name remains visible as long as the process picture is displayed.
- **Displaying all instance-specific names at once**: All instance-specific names can be made visible in a process picture at once by clicking a button. To do this, copy this button into the process picture of the chart from the @PCS7TypicalsAPLV7.PDL/ @PCS7TypicalsAPLV8.PDL or insert this button in the "Key area" of WinCC. If you insert it into the key area, read the manual section "PCS 7 OS Process Control " > "Layout of the User Interface".
  The instance-specific names are hidden once more by clicking the button again.

Icon for the block

Block icons for technologic blocks have their own operable symbols (for example, for the MotRevL block) and represent a status display of the block.

This symbol can be positioned at various locations, 0°, 90°, 180° and 270°. Refer to the Configuring the block icons (Page 233) section for more on this. You can change the operating mode of the block by right-clicking on the status display. Refer to the Operation via the block icon (Page 234) section for more on this.

Note

Block icons for motors and valves are available in various mounting positions and forms of representation. However, a general state icon is always displayed in the faceplates that does not consider the mounting position and form of representation in the block icons.

Analog value display

For block icons with analog value displays (for example, for the PIDConL block), there are basic settings that differ depending of the associated unit of measure:

Refer to the Configuring the block icons (Page 233) section for more on this.

These analog values can be controlled according to the Operator control permissions (Page 248). See also the section Operation via the block icon (Page 234) for more on this.

As of V8.0 block icons, when the actual values are displayed with limit monitoring in the analog value displays, the background and font colors for an alarm, warning or tolerance correspond to the message colors.

Note

As of V8.0 block icons, when the actual values are displayed with limit monitoring in the analog value displays, the background and font colors for an alarm, warning or tolerance, control system fault or control system fault correspond to the message colors.

Status bar for the block status

The status bar of the block icon provides an overview of the overall status of the block (see figure below).

The arrangement of icons in the following tables is prioritized from high to low.

The following elements can be displayed:

Alarms, warnings, tolerances, and messages

Refer to the Monitoring functions in the Advanced Process Library (Page 85) section for more on this.

Icon	Meaning
	No messages are output.
	A fault has occurred.
	An alarm is triggered.
	An alarm for the high alarm limit is triggered.
	An alarm for the low alarm limit is triggered.
	An warning is triggered.
	A warning for the high warning limit is triggered.
	A warning for the low warning limit is triggered.
	A tolerance violation has occurred.
	A tolerance violation for the high tolerance limit has been triggered.

Icon	Meaning
	A tolerance violation for the low tolerance limit has been triggered.
	The block has an operator prompt.
	There is a process status determination.

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. Please take into consideration the validity of terms for user-configured message classes (Page 41).

Note

For user-configured message classes, the symbols for high/low alarm, warning and tolerance limits are displayed only for limit violations which are triggered within the APL blocks with the colors of the user-configured message classes.

Operating modes

Refer to the Overview of the modes (Page 69) section for more on this.

Icon	Meaning
	The block is in automatic mode.
	The block is in the "On" operating mode.
	The block is in manual mode.
	A program is running.
	The block is in local mode.
	The block is in the "Out of service" operating mode. In this operating mode, no other symbols are displayed and no values are shown in the analog value display except for: - Display for an active message in the memo view - Display for the maintenance enable

Note

No symbol is displayed for the "On" operating mode (no green "O" displayed) if the block only has the operating modes "On" and "Out of service".

Internal or external setpoint

Refer to the section Setpoint specification - internal/external (Page 127).

Icon	Meaning
	Setting the setpoint internally
	External setpoint specification
	SP/MV ramp active

Signal status

Refer to the Forming and outputting signal status for blocks (Page 108) section for more on this.

Icon	Meaning
	The block is in simulation.
	Signal status is "Bad, device related".
	Signal status is "Bad, process related".
	Signal status is "Uncertain, device related".
	Signal status is "Uncertain, process related".
	A maintenance request is pending.
	Block is released for maintenance

Tracking and forcing of values and bypasses

See sections Forcing operating modes (Page 41), Interlocks (Page 99) and Manual and automatic mode for control blocks (Page 72).

Icon	Meaning
	At least one value has been forced
	Value is tracked
	There is a bypass of an interlock or a bypass condition of an upstream Intlock FB or tag bypass is active.

Interlocks

Refer to the Interlocks (Page 99) section for more on this.

Icon	Meaning
	Block is not interlocked.
	Block is interlocked.
	Bypass protection.
	Rapid stop is commanded.

Memo display

Icon	Meaning
	A message is available in the memo view.

Customer-specific icons

In the block icons, the icons can be replaced with customer-specific icons. A subdirectory must therefore be created in the OS project directory "GraCS" and the customer-specific icons must be inserted there with the same name as before. The name of the subdirectory must be adapted in the block icon in the "**Directory for pictures**" property ("Configurations" group). The icons in the subdirectory are not changed by the OS project editor or a new version of APL faceplates.

Note

- The "Directory for pictures" property is an instance-specific property. This means that the adaptations of block icons already configured in the process pictures are not changed by an update of the block icons in the process picture.
- We recommend that you adapt this property in the process picture and not in the "@PCS7TypicalsAPLVx.PDL" or "@TemplateAPLVx.PDL" pictures. The "@" pictures are system pictures and are replaced by the new version of APL faceplates. Customer changes will be lost.
- In the WinCC Graphics Designer, the PCS 7 "Export Objects" wizard can be used to generate a list of adaptations of instance-specific properties in the block icons. The "Directory for pictures" property can be changed in the exported list and collectively transferred back into the block icons with the PCS 7 "Import Objects" wizard.
- It is possible to use several subdirectories. This can be useful to create different variants of block icons in the process pictures.

See also

Motor protection function (Page 99)

Forming the group status for interlock information (Page 104)

User-configured message classes (Page 41)

2.2.2 Configuring the block icons

Configuring the block icons

There are two ways to configure your block icons:

- Automatically
- Manually

The new block icons are located in the template "@PCS7TypicalsAPL8.pdl" and "@TemplateAPL8.PDL".

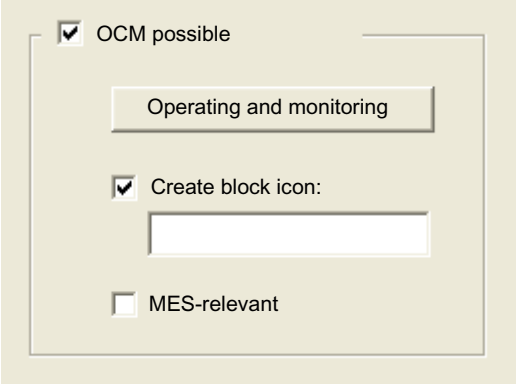
The old block icons are located in the template "PCS7TypicalsAPL7.pdl" and "@TemplateAPL7.PDL".

If you still want to continue using the V7.1 block icons in the project, you need to delete the V8.0 template or remove the "@" and deselect the update for the V8 template in the OS project editor on the basic data tab.

The block icons V7.1 are to remain in status V7.1 and be set to care and maintenance. New blocks are not to be integrated any longer.

Automatic configuration of block icons

There is a variety of block icons, which you can select for a block. You select a block icon by entering the number of the block icon in the field "OCM possible" > "Create block icon" in the object properties of the block instance:



You can find the names (numerical entry) of the respective block icons in the description for the blocks (Operator Control and Monitoring section).

If you do not enter a number for a block icon, the number 1 is always used for the block icon. The template for automatically generated block icons is @PCS7TypicalsAPLV7.PDL/ @PCS7TypicalsAPLV8.PDL.

Additional information

- Manual *Process Control System PCS 7; Engineering System*

Manual configuration of block icons

You can configure block icons manually by copying them from the @TemplateAPLV7.PDL/ @TemplateAPLV8.PDL template and inserting them into plant pictures.

The connection to the process tag is established with the "Connect faceplate with process tag" Wizard, see WinCC Information System, section "Making Process Pictures Dynamic" and "Standard Dynamics".

You can find information on exporting/importing and updating these objects in the WinCC Information System, section "Graphic Object Update Wizard". Use the "TemplateControlAPL.cfg" configuration file for these wizards.

Note

The procedure for changing the tooltip text of the block icons is described in the *APL Style Guide*. Otherwise, this property may not be changed.

2.2.3 Operation via the block icon

Operation via the block icon

The block icon can be used to operate elements directly if a relevant operator control permission (`OS_Perm`) is available. This operator control permission can be configured in the engineering system (ES).

The operation is performed by right-clicking on the element involved. The operable elements in the block icon include:

- Switching the operating mode
- Internal and external setpoint specification
- Changing the process value, setpoint and manipulated variable
- Changing the operating state

The operation is then performed in the same way as in the faceplate. Refer to the section Switching operating states and operating modes (Page 251) as well as Changing values (Page 253).

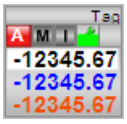
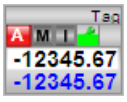
2.2.4 Block icons for PID and FM controller

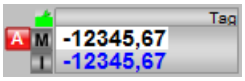
Block icons for PID and FM controller

A variety of block icons are available with the following functions:

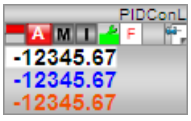
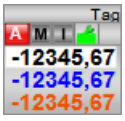
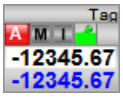
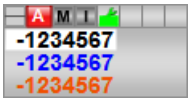
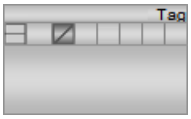
- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits (tolerance limits not with PIDConS) as well as control system faults
- Operating modes
- Internal and external setpoint specification
- Signal status, release for maintenance
- Memo display
- Process value (black, with and without decimal places, with and without color change at limit violations)
- Setpoint (blue, with decimal places)
- Feedback value (red, with decimal points), not available for types 3 and 4
  With FmCont and FmTemp as a step controller without feedback, there is no feedback value
- only `PIDStepL`: Position feedback value (green, with decimal points), not available for types 3 and 4
- Interlock without reset (only for PIDConR and PIDConL)

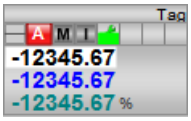
The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in the CFC	Special features
	1	
	2	
	3	
	4	

Icons	Selection of the block icon in the CFC	Special features
	5	Block icon in the full display
	6	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Icons	Selection of the block icon in CFC	Special features
Special case for `PIDStepL`:		
	-	In the case of the `PIDStepL` block only, the green value shows the position feedback (only visible if `WithRbk = 1` has been assigned parameters). With operation by means of the block icon, however, the manipulated variable is operated.

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

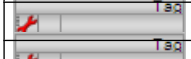
2.2.5 Block icon for interlock blocks

Properties of the block icon for interlock blocks Intlk02, Intlk04, Intlk08, Intlk16

A variety of block icons are available with the following functions:

- Signal status
- Memo display
- Output signal

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Block icon for
	1	Intlk02
	1	Intlk04
	1	Intlk08
	1	Intlk16
	2	Intlk02
	2	Intlk04
	2	Intlk08
	2	Intlk16

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Block icon for
	1	Intlk02
	1	Intlk04
	1	Intlk08
	1	Intlk16
	2	Intlk02
	2	Intlk04
	2	Intlk08
	2	Intlk16
	3	Intlk02
	3	Intlk04
	3	Intlk08
	3	Intlk16

Block icons (mini) without display of instance-specific names

These icons only show the output signal. They take the form of a small rectangle.

Display of the output signal

The display can show the following states for the output signal (priority from high to low):

- Gray: No inputs interconnected at the interlock block, the block is not used
- Blue: The output signal is 1, at least one input signal is bypassed. There is no gray state.
- Yellow: The signal status is 16#60; the output signal is simulated. There are no gray and blue states.
- Red: The output signal is 0; there is an interlock. There are no gray, blue, and yellow states.
- Green: The output signal is 1; the block is in the good state. There are no gray, blue, yellow, and red states.

Additional information on the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

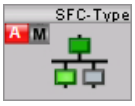
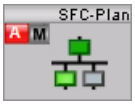
2.2.6 Block icons for SFC

Block icons for SFC

A variety of SFC block icons are available with the following functions:

- Alarm indicator
- Current active mode indicator ("Manual" or "Automatic" mode)
- Operator request icon
- Indicator for SFC operating status

The block icons from template @PCS7TypicalsAPLV8.PDL:

Block icons	Type 2	Type 3
SFC-Type		
SFC-Plan		

The following icons are dispayed in the SFC block icon:

1. Mode indicators:

Icon	Bit number	Tag name	Description
	7	.SFC_STATE	The block is in "Automatic" mode.
	7	.SFC_STATE	The block is in "Manual" mode.

2. Status indicators:

Icon	Bit number				Tag name	Description
	0	1	2	3		
	0	0	0	0	.SFC_STATE	IDLE
	0	0	0	1	.SFC_STATE	RUN
	0	0	1	0	.SFC_STATE	COMPLETED
	0	0	1	1	.SFC_STATE	HELD
	0	1	0	0	.SFC_STATE	ABORTED
	0	1	0	1	.SFC_STATE	STARTING
	0	1	1	0	.SFC_STATE	COMPLETING
	0	1	1	1	.SFC_STATE	ERROR_COMPLETING
	1	0	0	0	.SFC_STATE	HOLDING
	1	0	0	1	.SFC_STATE	RESUMING
	1	0	1	0	.SFC_STATE	ERROR
	1	0	1	1	.SFC_STATE	HELD_ERROR
	1	1	0	0	.SFC_STATE	RESU_ERROR
	1	1	0	1	.SFC_STATE	ABORTING
	1	1	1	0	.SFC_STATE	STOPPING
	1	1	1	1	.SFC_STATE	STOPPED

See also

Block icon structure (Page 226)

Configuring the block icons (Page 233)

Operation via the block icon (Page 234)

2.2.7 Adding block icons to static picture components

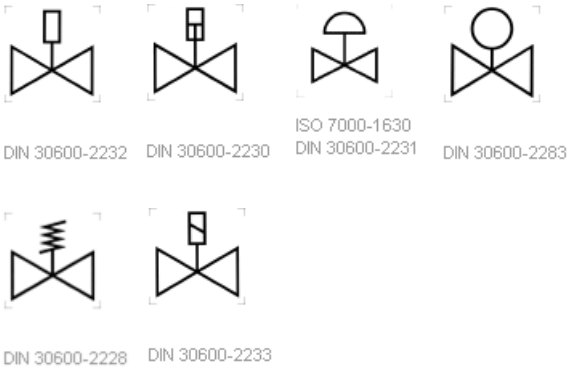
Static picture component for the block icons

The block icons for drives and valves contain no static picture components. However, you can add static picture components to the block icons by copying them from the @TemplateAPLV8.PDL/@TemplateAPLV7.PDL template and placing them over the block icons.

The following static picture components are available:

- Static picture components for motor blocks:

- Static picture components for valve blocks:



Note

The static picture components for the valves are visible here above the valve icon.

Note that all the block icons of the APL are located on layer 0 of the process picture. Layer 1 is intended for the static picture component. This ensures that the static picture component is always over the block icon. If you have changed this layer of the block icon, the static picture component should always be placed in a higher layer.

2.3 Functions of the faceplates

2.3.1 Structure of the faceplate

General functions of the faceplates

This section provides general information that applies to all faceplates.

Recommended screen resolution

The faceplates are shown in full on the screen with a resolution of 1280 x 1024. The full-screen mode (press the F11 key) must also be activated on the Web client.

Scaling the faceplates

You can scale the faceplates from 50% to 200%. The value to scale is given in the internal variable "@APLFaceplateScaleFactor" which is of type unsigned 32-bit value. The default value of the scale is 100. You can scale the faceplates from 101% to 200% by providing a value in the range of 101 to 200 in the internal variable. Similarly, you can scale down the faceplates from 50% to 99% by providing a value in the range of 50 to 99. The reference for calculation is default size of the faceplates which is 100%. This means if the value given is 150, the faceplates will be scaled to 150% of the default size. Similarly, if the value given is 90, the faceplate will be scaled down to 90% of the default size.

Note

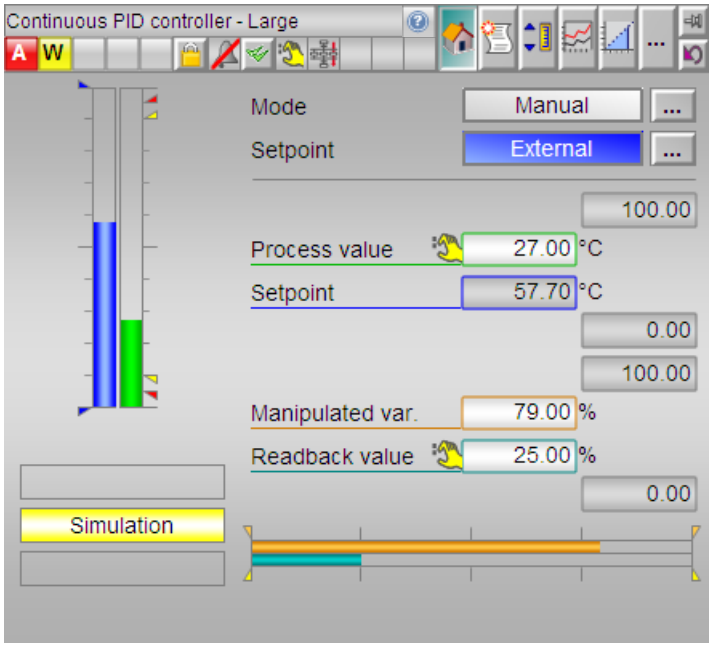
You must reopen the faceplate after changing the scale value. Already opened faceplate will not be scaled if the scale value is changed.

Displaying CPU stop

With a CPU stop, boxes are unavailable or hidden and a yellow warning triangle is displayed in the group display for blocks with messaging. No operations are possible.

Opening the faceplate

Click on the block icon in WinCC to open the faceplate with the standard view; in this example this is the standard view of PIDConL:



Note

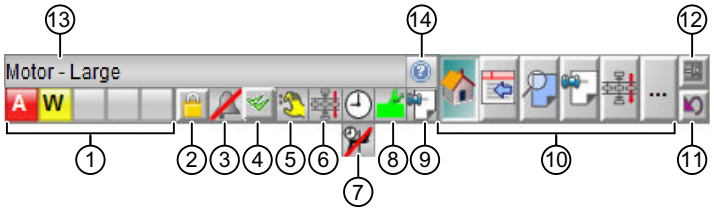
Some of the illustrations of the faceplate views and block icons in the help for the PCS 7 Advanced Process Library are examples or offline representations. The representations on runtime may vary.

The views differ depending on the block functions. All blocks that have faceplates provide a status bar where you can see the most important information relating to the block status. There are additional functions available that are described in the next sections.

Note

Display elements of block inputs with `16#FF` as their status default, are only shown when their block inputs are interconnected (status ≠ FF). Exception: The values for `Gain`, `TI` and `TD` are always displayed for controller blocks.

Displays and operator controls



The faceplate provides the following display and operator controls:

(1) Group display

(2) Lock alarms

(3) Suppress messages / alarm delays with the icon 

(4) Acknowledge alarms

(5) Worst signal status

(6) Batch display

(7) At least one delay time effective at the block

(8) Maintenance request and release

(9) Memo display

(10) Open views of the block

(11) Back to block icon

(12) Pin faceplate

(13) Instance name of the block

(14) Help button

(1) Group display

The group display shows the information that is transferred from ALARM_8P of the block instance to WinCC.

- Alarms
- Warnings
- Tolerances
- Faults
- Operator prompts
- Process messages

Note

The message classes Alarm, Warning and Tolerance are not valid for user-configured message classes. For user-configured message classes, message types depend on the project-specific setting. Please take into consideration the validity of terms for user-configured message classes (Page 41).

(2) Lock/unlock alarms

You can use this button to lock or unlock block alarms.

The alarms are displayed again in the group view when you unlock the block's alarms. The block instance then resumes sending new alarms. Alarms generated in the locked phase are displayed when you enable the alarm function.

The operator authorization level for this button is set in the internal variable @LockMessageAuthLevel. This variable is set by the OS project editor.

For reasons of optimization, the operator authorization level set in @LockMessageAuthLevel also has to exist at an OperationLevelx of the block icon of the process tag (for example higher process control). Otherwise, operator control of the button "Disable messages" is not possible.

You can hide this button from specific users / user groups using the permissions in PCS 7-OS. Refer to the PCS 7-OS help system.

Note

Since as of V8.1 the disabling of block alarms must be confirmed via the PCS 7 system dialog, the operator can also enter a reason for the disabling in this dialog.

The inputs are always confirmed with "OK", even in 2 step operation.

(3) Suppress messages

Message suppression indicates whether or not the "Suppress process messages" function in the AS block is activated with the `MsgLock` parameter. If message suppression is activated, all messages in this block instance – except for process control messages – are suppressed.

(4) Acknowledge alarms

You can acknowledge all alarms from the block instance using this button.

You can hide this button from specific users / user groups using the permissions in PCS 7-OS. Refer to the *Process Control System PCS 7; OS Process Control* documentation for more on this.

(5) Display for worst signal status

This display shows the worst signal status currently present. You will find more detailed information in section Forming and outputting signal status for blocks (Page 108).

(6) Batch display

The batch display shows whether or not the block instance is in use by SIMATIC BATCH.

You will find more detailed information in section SIMATIC BATCH functionality (Page 67).

(7) At least one delay time effective at the block

	This display shows you the active delay time. You will find more detailed information in section Display of delay times (Page 250).
	The "dead band" function is temporarily deactivated.

(8) Maintenance request and release

This display shows you if a maintenance request or release has been made for this block.

You will find more detailed information in section Release for maintenance (Page 64).

(9) Memo display

This display shows you if a message has been left in the memo view for you. You will find more detailed information in section Memo view (Page 298).

(10) Open views of a block

You can use this field to open the various views of a block. Refer to the block description to learn of the available views. Left clicking shows the view in the same window. Right clicking opens a new window.

You can select from the following typical views here:

Icon	Identifier
	Standard view
	Message view
	Limit view (several limit views within a block are possible)
	Trend view
	Ramp view
	Parameter view (several parameter views within a block are possible)
	Preview
	Memo view

Icon	Identifier
	Batch view
	Setpoint view
	Parameter view 1 (continues to parameter view 4, see line below)
	Parameter view 4
	Quantity view
	Flow view
	Views (additional views within a block are possible)

Note

The buttons are unavailable when views cannot be selected.

(11) Back to block icon

Use this button to return to the block icon in the process image of the corresponding faceplate. You can use this function, for example, when you have pinned a block **(12)** and the process picture has changed in the meantime.

(12) Pin faceplate

You can pin the faceplate on top of the user interface using this button. This allows you to change to another picture or area without closing the faceplate.

Note

You can learn about additional operator controls in the descriptions of the individual blocks.

(13) Instance name of the block

This area displays the instance specific name of the block.

(14) Help button

Click this button to open the *APL Operator Guide* online help of the corresponding view of the faceplate.

2.3.2 Operator control permissions

Operator control permissions for blocks

The following conditions have an effect on operator control permissions:

- User management in the PCS 7 OS
- Local operating permission using the OpStations block
- Dependencies on operating modes in blocks
- Permissions via parameter assignment / interconnection of blocks
- Permissions via the `OS_Perm` input parameter at the block itself

User management in the PCS 7 OS

The following operator authorization levels from user management are used in the APL:

- Process control (for example, manual/automatic mode switchover, changing setpoints and manipulated variables).
  With this operator control permission, operations can be performed in the standard view of all blocks and input can be made in the ramp and memo views. The "Out of service" operating mode cannot be used with process controlling.
- Higher process controls (for example, changing limits, controller parameters and monitoring times).
  With this operator control permission, all operations in all views of all blocks are possible, including the operations for "Out of service" operating mode. Exception: The operations listed under "Highest process controlling".
- Highest process controls (simulate process values and release process tag for maintenance).
  With this operator control permission, simulation can be switched on and off in the parameter view and the process tag for maintenance work can be released.
- Extended operation 1
  Free project-specific operator authorization
- Extended operation 2
  Free project-specific operator authorization

Note

Exceptions to the uses described above are listed in the descriptions of the individual views.

Each operation is assigned with an operator authorization level in the faceplates. This fixed assignment can be changed for each instance at the "operator authorization level" property of an I/O in the AS block (for example, SP_Int with PIDConL). The following assignment applies:

Operator authorization level in the user management	Value "Operator authorization level" property
Process controls	1
Higher process controls	2

Operator authorization level in the user management	Value "Operator authorization level" property
Highest process controls	3
Extended operation 1	4
Extended operation 2	5

Note

- The free three assignments in the upper table can be changed in the block icon at the properties "OperationLevel1_backup", "OperationLevel2_backup" and "OperationLevel3_backup". Any operator authorization level from the user management can be assigned the values 1 to 3. This type of instance-specific configuration is still available only for reasons of downward compatibility and should no longer be used in new projects.
- The controls for the message system (e.g. acknowledge messages) and trend display (e.g. export) are fixed across the system and cannot be changed via the AS block.
  The "Lock messages" control can be changed system-wide via the internal variable "`@LockMessageAuthLevel`" with the value of the operator authorization level from the user management (for example, value "6" for "Higher process controls").

Local operating permission using the OpStations block

Local operating permission is an upstream operator control permission which is determined before the operator control permissions for user management and the release of the block, and is realized via the OpStations (Page 399) block.

If local operating permission is missing, operation of a block instance on an OS is usually blocked. Otherwise, when local operating permission is allowed, the operator control permission is normally determined through user management and the block.

Local operating permission can be set for each specific instance; in other words, block instances can be enabled or disabled for use on an operator station independently of one another.

You can find additional information for the use of local operating permission in the section Description of OpStations (Page 399).

Dependencies on operating modes in blocks

You can execute various functions depending on the block mode. These permissions are stored in the block algorithm and are determined dynamically in online mode.

Permissions via parameter assignment / interconnection of blocks

The block is either controlled by the operator or by the controller, depending on parameter settings or on the interconnection. An example for this is the switchover from manual to automatic mode by a higher-level controller or by the operator. These permissions are stored in the block algorithm and are determined dynamically in online mode.

Permissions via the OS_Perm input parameter at the block itself

Controllable blocks have the OS_Perm input parameter which allows you to implement individual operator control strategies by setting the operator control permissions. A pressure relief valve, for example, can only be opened by the master control system. The operator may only close the valve. These authorizations are defined during configuration. These operator control permissions are displayed in the preview view of the faceplate. For information about setting the individual operator control permissions (OS_Perm) refer to the description of the functions of the individual blocks.

The relevant operator controls are enabled if the operator has suitable permissions. The block algorithm processes the input data.

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

See also

Enabling local operator authorization (Page 157)

2.3.3 Display of delay times

Display of delay times

As soon as the immediate command output is delayed due to an active delay time at the "Large" blocks of the "Drives" family, this fact is signaled by a bit in the status word and a display in the faceplate.

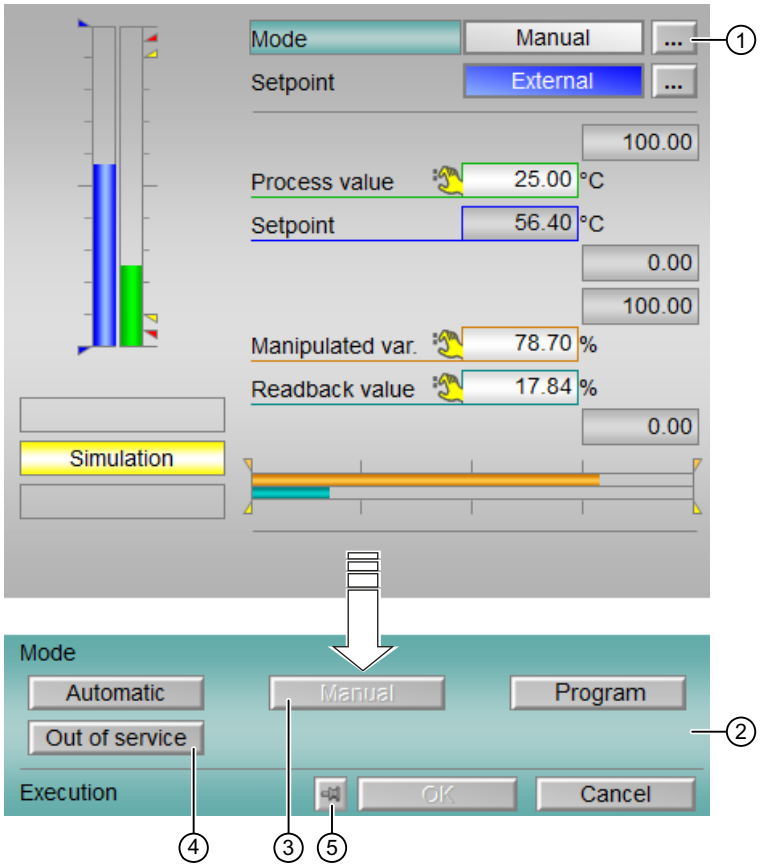
These times include:

- Pre-warning time
- On/Off delays
- Restart inhibit

2.3.4 Switching operating states and operating modes

Requirements

You can change the operating state, operating mode and other parameters if needed in faceplates if you have the corresponding operator control permission (`OS_Perm`). This operator control permission can be configured in the engineering system (ES).



(1) The mouse cursor changes when you place it over the following button:

The mouse pointer now looks as follows:

When you click on the button with the mouse pointer, the bottom of the faceplate expands. You now see the field for changing the operating mode, for example.

(2) Field for changing the operating mode, operating state etc. This example describes changing the operating mode.

If the indicators for the operating state is currently located in this field and you have configured text for specific instances, this text is also shown. You can find more information about this in the section Labeling of buttons and text (Page 205).

(3) The text on this button is gray. You cannot select this operating mode due to the following reasons:

- This operator control permission for this operating mode cannot be configured in the engineering system (ES).

or

- The operating mode is already selected at this time.

or

- Due to the technology, you cannot switch from the operating mode currently set and the desired operating mode.

(4) The text on this button is black. You can switch to this operating mode.

How to change the operating mode (using the PIDConL block in standard view as an example)

1. Click one of the selectable buttons in the operating mode field.
2. Confirm your selection by clicking "OK".
3. If you do not want to apply your selection, click "Cancel".

After clicking the "OK" or "Cancel" button, the faceplate is reduced again to its original form.

(5) Multiple operation

If the operating window is not to close after the confirmation of a command, it can be "attached". The following button is located below the operating window for this purpose:

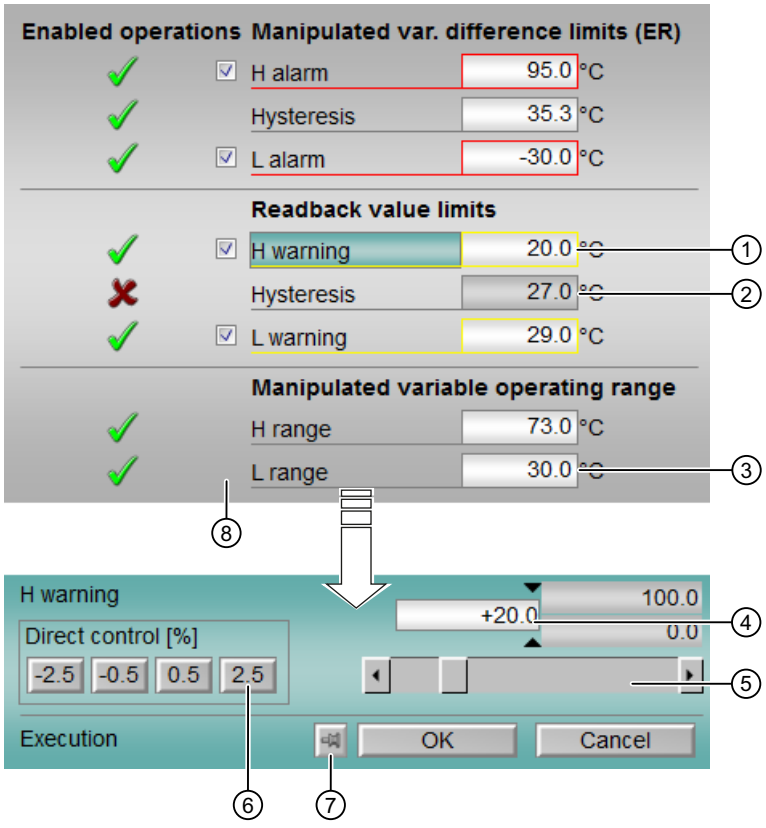

Operating window is closed after the value is applied


Operating window remains open after the value is applied

2.3.5 Changing values

Requirements

You can change the values in faceplates and in the block icons if you have the corresponding operator authorization (`OS_Perm`). This operator permission can be configured in the engineering system (ES). The following example shows how values are changed via the faceplate.



(1) The background color of the input box is white. You can change the value. The mouse pointer changes when you place it over the input box:

(2) The background color of the input box is gray. You cannot change the value.

(3) If you click on the input box, the bottom of the faceplate expands. You now see the field for changing values.

How to change values in faceplates

You have three options for changing values:

- "Direct control": Click on a button such as "2.5" in the box **(6)**. The value is immediately changed and applied. It is no longer necessary to confirm this value.
- Changing values using the slide control **(5)**: move the slide control until the desired value is shown in the box. Then confirm the value using the Enter key or by clicking "OK". Read the section "Setting the multiple step operation".
- Change the values in the input box **(4)**: Click the input box and enter the new value. If the new values are outside the limits, they are rejected and the old values are retained. Then confirm the value using the Enter key and clicking "OK". Read the section "Setting the multiple step operation".

Setting the multiple step operation

You can use the internal @APLCommandExecutionSteps tag in the Tag Management of the WinCC Explorer to specify if values are to be changed in two or three steps.

Follow the steps outlined below:

1. Double-click on the internal @APLCommandExecutionSteps tag
2. Change the start value to 2 or 3 in the Limits/Reporting tab.
   Start value = 2: It is no longer necessary to confirm the value in the faceplate by clicking "OK"; values are applied immediately.
   Start value = 3: Each value change in the faceplate (with the exception of those for direct control) needs to be confirmed with "OK".

Changing the values for "Direct control"

Specify the percentage values for the two inner keys for direct control with the `DirectOperationValue` property at the block icon. The outer two keys are automatically determined with `DirectOperationValue` times the factor 5.

If `DirectOperationValue` is not an integer but the values in the faceplate are integers, then `DirectOperationValue` is rounded up to the next integer.

If the rounded value is 0, 1 is used for `DirectOperationValue`.

The default value for `DirectOperationValue` is 0.5. With integer format, the "+/-1%" results for the inner keys and "+/-5%" for the outer keys.

(7) Multiple operation

If the operating window is not to close after the confirmation of a command, it can be "attached". The following button is located below the operating window for this purpose:


Operating window is closed after the value is applied

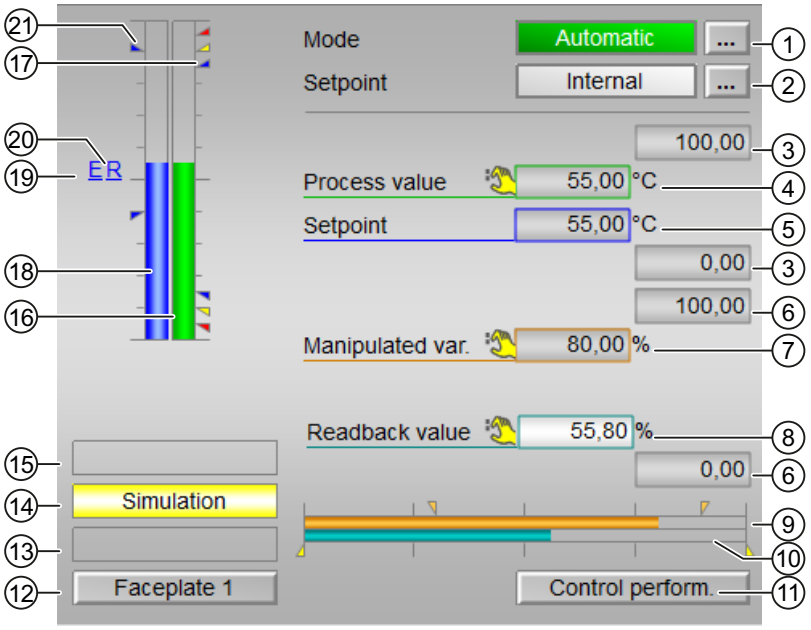

Operating window remains open after the value is applied

(8) Suppress messages

You can enable / disable messages by setting the check mark.

2.3.6 FM controllers standard view (analog)

Standard view (analog) of FM controllers



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 72)
- Automatic mode (Page 72)
- Program mode for controllers (Page 78)
- Out of service (Page 71)

Refer to chapter Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Display and switch the setpoint specification

This area shows how to specify the setpoint. The setpoint can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) chapter for information on switching the setpoint specification.

For more information on the setpoint specification, refer to the Setpoint specification - internal/ external (Page 127) chapter.

If text is configure for these commands, it is displayed as additional text and as button labels for command selection. You can find more information about this in chapter Labeling of buttons and text (Page 205)

(3) High and low scale range for the process value

These values provide information on the display range (`PV_OpScale`) for the bar graph of the process value. The scale range is defined in the Engineering System.

(4) Display of the process value including signal status

This area provides information on the current process value (`PV`) with the corresponding signal status.

(5) Display and change the setpoint including signal status

This area provides information on the current setpoint (`SP`) with the corresponding signal status.

Refer to the Changing values (Page 253) chapter for information on changing the setpoint. The setpoint specification **(2)** also needs to be set to "Internal" for this block.

(6) High and low scale range for the setpoint

This area is already set and cannot be changed.

(7) Display and change the manipulated variable including signal status

This area shows the current "Manipulated variable" (`MV`) with the corresponding signal status.

Refer to chapter Changing values (Page 253) for information on changing the manipulated variable. You can only make a change in manual mode.

(8) Display of the position feedback including signal status

This display is only visible when the corresponding block input is connected.

This area provides information on the current readback value of the manipulated variable with the corresponding signal status. This display is only available when the readback value in the box is interconnected to the `Rbk` input parameter.

(9) Bar graph for the "Manipulated variable"

This area shows the current "Manipulated variable" in the form of a bar graph (`MV_OpScale`). The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(10) Bar graph for position feedback

This display is only visible when the corresponding block input is connected.

This area shows the current position feedback in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(11) Navigation button for switching to the standard view of the ConPerMon block

Use this navigation button to reach the standard view of the ConPerMon block. The visibility of this navigation button depends on the configuration in the Engineering System (ES).

Refer also to chapter Opening additional faceplates (Page 203) for more on this.

(12) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the Engineering System. The visibility of this navigation button depends on the configuration in the Engineering System (ES).

Refer also to chapter Opening additional faceplates (Page 203) for more on this.

(13) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in chapter Release for maintenance (Page 64) Display area for block states.

(14) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in chapter Simulating signals (Page 58).

(15) Display area for block states

This area provides additional information on the operating state of the block (from high to low according to priority):

- "Fuzzy Optim." (FmCont only)
- "Tracking FB"
- "Tracking FM"
- "Safety mode FM"
- "Fuzzy control" (FmCont only)

(16) Limit display

These colored triangles indicate the configured limits in the respective bar graph:

- Red: Alarm
- Yellow: Warning
- Blue: Tolerance

Note

The symbols displayed are not valid for user-configured message classes. Please take into consideration the validity of terms for User-configured message classes (Page 41).

(17) Bar graph for the "Process value"

This area shows the current "Process value" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(18) Bar graph for the "Setpoint"

This area shows the current "Setpoint" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(19) Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

(20) Display for the target setpoint of the setpoint ramp

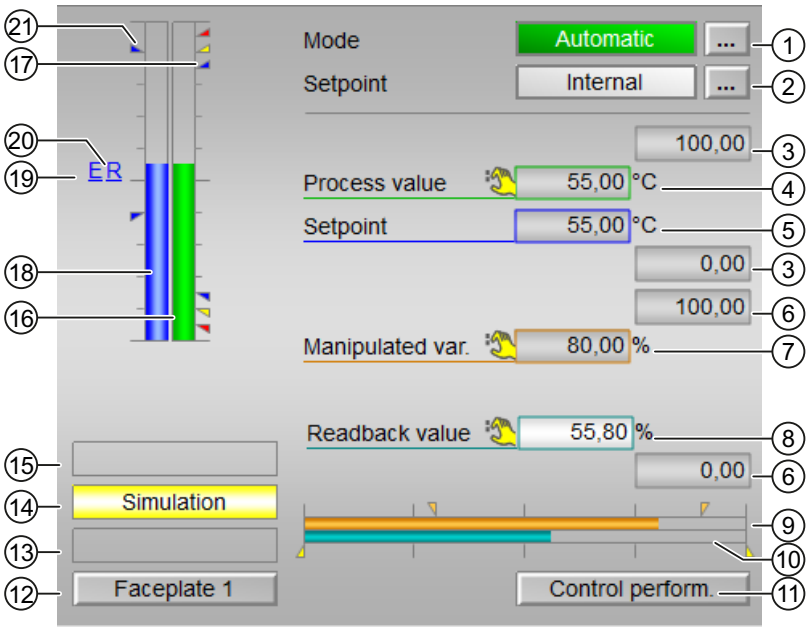
This display [R] shows the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

(21) Limit display for the setpoint

These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the ES.

2.3.7 FM controllers standard view (pulse controller)

Standard view (pulse) of FM controllers



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 72)
- Automatic mode (Page 72)
- Program mode for controllers (Page 78)
- Out of service (Page 71)

Refer to chapter Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Display and switch the setpoint specification

This area shows how to specify the setpoint. The setpoint can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) chapter for information on switching the setpoint specification.

For more information on the setpoint specification, refer to the Setpoint specification - internal/ external (Page 127) chapter.

(3) High and low scale range for the process value

These values provide information on the display range (PV_OpScale) for the bar graph of the process value. The scale range is defined in the Engineering System.

(4) Display of the process value including signal status

This area provides information on the current process value (PV) with the corresponding signal status.

(5) Display and change the setpoint including signal status

This area provides information on the current setpoint (SP) with the corresponding signal status.

Refer to the Changing values (Page 253) chapter for information on changing the setpoint. The setpoint specification **(2)** also needs to be set to "Internal" for this block.

(6) High and low scale range for the setpoint

This area is already set and cannot be changed.

(7) Display and change the manipulated variable including signal status

This area shows the current "Manipulated variable" (MV) with the corresponding signal status.

Refer to chapter Changing values (Page 253) for information on changing the manipulated variable. You can only make a change in manual mode.

(8) Display of the position feedback including signal status

This display is only visible when the corresponding block input is interconnected.

This area shows the current feedback of the manipulated variable with the corresponding signal status. This display is only available when the readback value in the box is interconnected to the Rbk input parameter.

(9) Bar graph for the "Manipulated variable"

This area shows the current "Manipulated variable" in the form of a bar graph (MV_OpScale). The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(10) Bar graph for position feedback

This display is only visible when the corresponding block input is interconnected.

This area shows the current position feedback in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(11) Button for switching to the standard view of the ConPerMon block

Use this button for the standard view of the ConPerMon block. The visibility of this button depends on the configuration in the Engineering System (ES).

Refer also to chapter Opening additional faceplates (Page 203) for more on this.

(12) Button for switching to the standard view of any faceplate

Use this button for the standard view of a block configured in the engineering system. The visibility of this button depends on the configuration in the Engineering System (ES).

Refer also to chapter Opening additional faceplates (Page 203) for more on this.

(13) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in chapter Release for maintenance (Page 64) Display area for block states.

(14) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in section Simulating signals (Page 58).

(15) Display area for block states

This area provides additional information on the operating state of the block (from high to low according to priority):

- "Fuzzy Optim." (FmCont only)
- "Tracking FB"
- "Tracking FM"
- "Safety mode FM"
- "Fuzzy control" (FmCont only)

(16) Bar graph for the "Process value"

This area shows the current "Process value" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(17) Limit display

These colored triangles indicate the configured limits in the respective bar graph:

- Red: Alarm
- Yellow: Warning
- Blue: Tolerance

Note

The symbols displayed are not valid for user-configured message classes. Please take into consideration the validity of terms for User-configured message classes (Page 41).

(18) Bar graph for the "Setpoint"

This area shows the current "Setpoint" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(19) Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

(20) Display for the target setpoint of the setpoint ramp

This display [R] shows the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

(21) Limit display for the setpoint

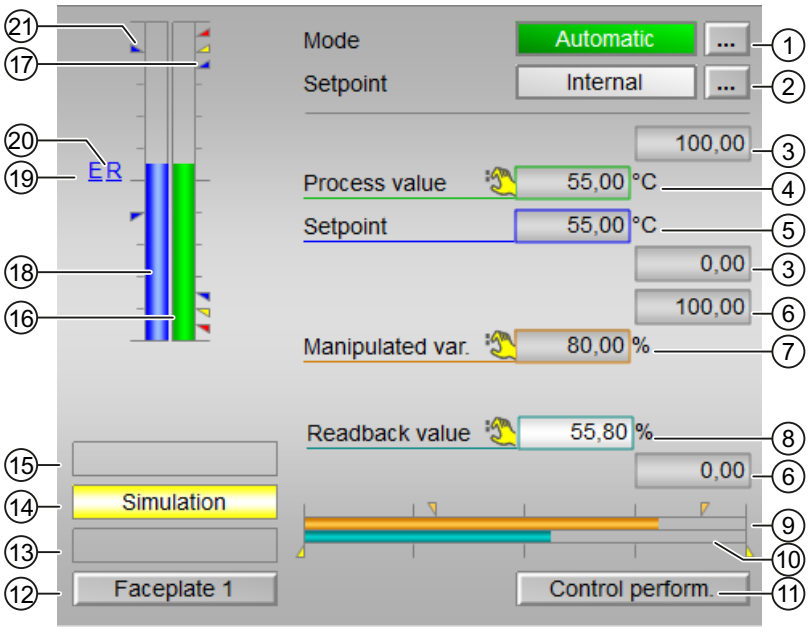
These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the ES.

See also

Labeling of buttons and text (Page 205)

2.3.8 FM controllers standard view (step controller with position feedback)

Standard view with position feedback of FM controllers



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 72)
- Automatic mode (Page 72)
- Program mode for controllers (Page 78)
- Out of service (Page 71)

Refer to chapter Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Display and switch the setpoint specification

This area shows how to specify the setpoint. The setpoint can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) chapter for information on switching the setpoint specification.

For more information on the setpoint specification, refer to the Setpoint specification - internal/ external (Page 127) chapter.

(3) High and low scale range for the process value

These values provide information on the display range (PV_OpScale) for the bar graph of the process value. The scale range is defined in the Engineering System.

(4) Display of the process value including signal status

This area provides information on the current process value (PV) with the corresponding signal status.

(5) Display and change the setpoint including signal status

This area provides information on the current setpoint (SP) with the corresponding signal status.

Refer to the Changing values (Page 253) chapter for information on changing the setpoint. The setpoint specification **(2)** also needs to be set to "Internal" for this block.

(6) High and low scale range for the setpoint

This area is already set and cannot be changed.

(7) Display and change the manipulated variable including signal status

This area shows the current "Manipulated variable" (MV) with the corresponding signal status.

Refer to chapter Changing values (Page 253) for information on changing the manipulated variable. You can only make a change in manual mode.

(8) Display of the position feedback including signal status

This display is only visible when the corresponding block input is interconnected.

This area shows the current feedback of the manipulated variable with the corresponding signal status. This display is only available when the readback value in the box is interconnected to the Rbk input parameter.

(9) Bar graph for the "Manipulated variable"

This area shows the current "Manipulated variable" in the form of a bar graph (MV_OpScale). The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(10) Bar graph for position feedback

This display is only visible when the corresponding block input is interconnected.

This area shows the current position feedback in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(11) Button for switching to the standard view of the ConPerMon block

Use this button for the standard view of the ConPerMon block. The visibility of this button depends on the configuration in the Engineering System (ES).

Refer also to chapter Opening additional faceplates (Page 203) for more on this.

(12) Button for switching to the standard view of any faceplate

Use this button for the standard view of a block configured in the Engineering System. The visibility of this button depends on the configuration in the Engineering System (ES).

Refer also to chapter Opening additional faceplates (Page 203) for more on this.

(13) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in chapter Release for maintenance (Page 64) Display area for block states.

(14) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in section Simulating signals (Page 58).

(15) Display area for block states

This area provides additional information on the operating state of the block (from high to low according to priority):

- "Fuzzy Optim." (FmCont only)
- "Tracking FB"
- "Tracking FM"
- "Safety mode FM"
- "Fuzzy control" (FmCont only)

(16) Bar graph for the "Process value"

This area shows the current "Process value" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(17) Limit display

These colored triangles indicate the configured limits in the respective bar graph:

- Red: Alarm
- Yellow: Warning
- Blue: Tolerance

Note

The symbols displayed are not valid for user-configured message classes. Please take into consideration the validity of terms for User-configured message classes (Page 41).

(18) Bar graph for the "Setpoint"

This area shows the current "Setpoint" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(19) Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

(20) Display for the target setpoint of the setpoint ramp

This display [R] shows the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

(21) Limit display for the setpoint

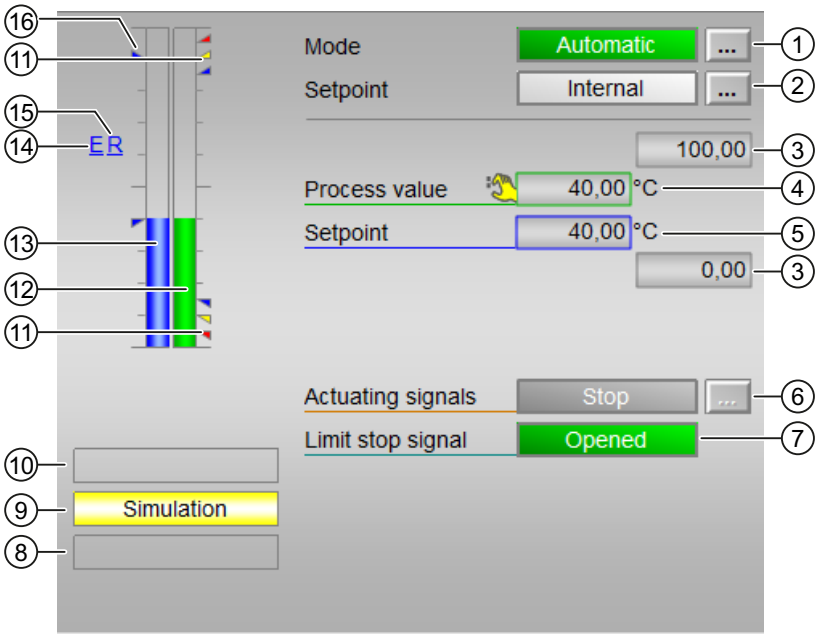
These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the ES.

See also

Labeling of buttons and text (Page 205)

2.3.9 FM controllers standard view (step controller without position feedback)

Standard view without position feedback of FM controllers



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 72)
- Automatic mode (Page 72)
- Program mode for controllers (Page 78)
- Out of service (Page 71)

Refer to chapter Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Display and switch the setpoint specification

This area shows how to specify the setpoint. The setpoint can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) chapter for information on switching the setpoint specification.

For more information on the setpoint specification, refer to the Setpoint specification - internal/external (Page 127) chapter.

(3) High and low scale range for the process value

These values provide information on the display range (PV_OpScale) for the bar graph of the process value. The scale range is defined in the Engineering System.

(4) Display of the process value including signal status

This area provides information on the current process value (PV) with the corresponding signal status.

(5) Display and change the setpoint including signal status

This area provides information on the current setpoint (SP) with the corresponding signal status.

Refer to the Changing values (Page 253) chapter for information on changing the setpoint. The setpoint specification **(2)** also needs to be set to "Internal" for this block.

(6) Operating and displaying the actuating signal

This area shows the current feedback of the actuating signal.

- "Open"
- "Stop"
- "Close"

The button is shown next to the display in manual mode. You can influence the actuating signal here. Refer to the Switching operating states and operating modes (Page 251) chapter for more on this.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(7) Display of the limit stop value including signal status

This area shows the limit stop signal with the corresponding signal status.

- "Open"
- "Closed"

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(8) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in chapter Release for maintenance (Page 64) Display area for block states.

(9) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in section Simulating signals (Page 58).

(10) Display area for block states

This area provides additional information on the operating state of the block (from high to low according to priority):

- "Fuzzy Optim." (FmCont only)
- "Tracking FB"
- "Tracking FM"
- "Safety mode FM"
- "Fuzzy control" (FmCont only)

(11) Limit display

These colored triangles indicate the configured limits in the respective bar graph:

- Red: Alarm
- Yellow: Warning
- Blue: Tolerance

Note

The symbols displayed are not valid for user-configured message classes. Please take into consideration the validity of terms for User-configured message classes (Page 41).

(12) Bar graph for the "Process value"

This area shows the current "Process value" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(13) Bar graph for the "Setpoint"

This area shows the current "Setpoint" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(14) Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

(15) Display for the target setpoint of the setpoint ramp

This display [R] shows the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

(16) Limit display for the setpoint

These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the ES.

See also

Opening additional faceplates (Page 203)

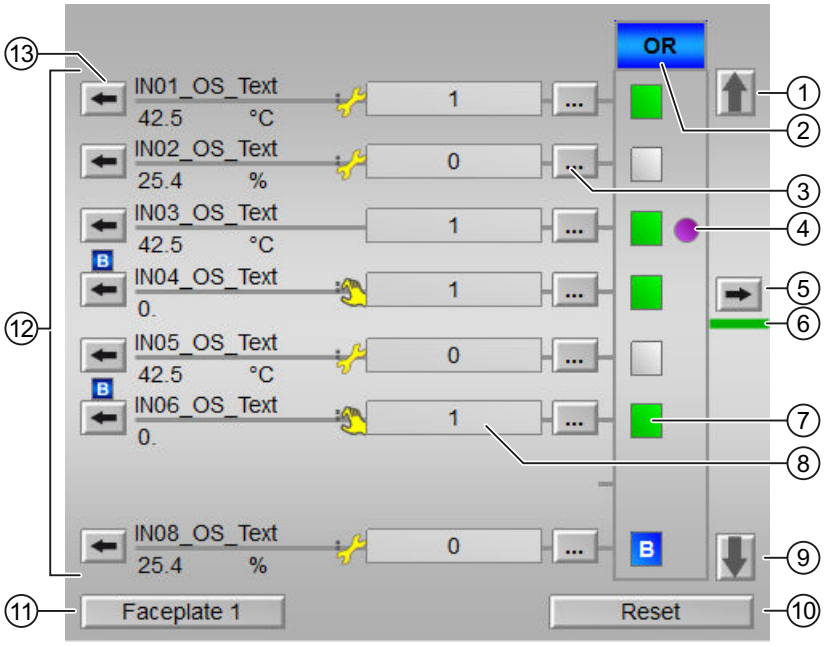
2.3.10 Interlock blocks standard view

Interlock blocks standard view Intlk02, Intlk04, Intlk08, Intlk16

The number of the displayed input values depends on the interlock block you have selected.

The operation and functions are identical for all interlock blocks and do not depend on the number of input values.

The **Intlk16** interlock block has two additional buttons for switching between the input values 1 to 8 and 9 to 16.



(1), (9) Switching between the input values 1 to 8 and 9 to 16 (for Intlk16 only)

The buttons (1) or (9) are displayed depending on the view you are in. These buttons are only available for the **Intlk16** block.

The **Intlk16** block provides two views:

- When you are in the first view, the input values 1 to 8 are available in the area (12). The button (9) is displayed. You switch to the second view by clicking on the button (9).
- When you are in the second view, the input values 9 to 16 are available in the area (12). The button (1) is displayed. You switch back to the first view by clicking on the button (1).

(2) Status of the output signal of the interlock block

This area (2) shows the status of the output signal of the interlock block (priority from high to low). You can configure the logic in the engineering system (ES).

	Logic	
Color of the field	**AND**	**OR**
Gray	Block is not used (only for display in the faceplate, interlock logic still enabled), set using the `NotUsed` = 1 parameter	
Blue	Excluded (bypass)	
Yellow	Simulated	
Red	Interlocked	
Green	Not interlocked	

(3) Exclude input values

You can use the button (3) to exclude input values from processing. Depending on the previous settings, you can "Set" or "Reset" this property.

If the input value has been excluded, the following symbol appears in the field (8):

For more information on the operation, refer to the section Switching operating states and operating modes (Page 251).

Note

This function can only be executed in the faceplate with "high-level operating permission".

Note

Operator permissions via OS_Perm do not depend on the Feature bit 5 setting. You can find additional information in section Activate OS_Perm bits (Page 156)

(4) "First in" status display

The following symbol is displayed next to an input value, if this input value has caused the last output signal change from 1 to 0 (good state to locked):

You can reset the first-in (initial) signal with the button (10).

Note

This function can only be executed in the faceplate with "process control" operating permission.

You can find additional information on this in the section Recording the first signal for interlock blocks (Page 52).

For more information on the operation, refer to the section Switching operating states and operating modes (Page 251).

(5) Open faceplate of the output value

When you press the button (5), you can open the faceplate associated with the output value. The function of this button depends on the configuration in the engineering system (ES). See also the section Opening additional faceplates (Page 203) for more on this.

(6) Status of the block output

The line color indicates the status of the block output:

Color of the line	Output status
Green	Output is enabled
White	Output is disabled

(7) Display the status for further processing

The symbol shows the status for further processing of the input values:

Icon	Further processing
	The input value is processed further with value 1.
	The input value is processed further with value 0.
	The input value is excluded from further processing.

(8) Display of input values (BOOL) with signal status (in front of the field)

These fields show the interlock information associated with the analog value (13) with a signal status:

- 1 = "Good" state
- 0 = "Locked"

Changing the display

You can change the displays for 0 and 1 in the CFC in the object properties of the Interlock block:

- Navigate to the I/Os (object properties).
- Change the default for the input parameters (`Inxx`) in the Text 0 and Text 1 columns to what you later want to see in runtime.

(9) Switching input values

Read point (1) for this.

(10) "Reset" the settings for further processing

When you press the button (10), you can "Reset" all input values:

- "Reset exclusions": the exclusions of the input values are reset.
- "Reset first-in": First-in detection / status display (4) is reset.

You can find additional information on this in the section Recording the first signal for interlock blocks (Page 52).

(11) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(12) Displaying analog input values

The interconnected analog input values (`AVxx`) are displayed in this area. Set a unit of measure (`AVxx_Unit` as shown in the picture with [Unit]) for each input in the engineering system (ES).

Changing the display

You can change the displays in the CFC in the object properties of the Interlock block:

- Navigate to the I/Os (object properties).
- In the “OS additional text” column, change the default setting for the input parameter (`INxx`) to what you want to see during runtime later.

- In the "Identifier" column, change the default setting for the input parameter (AVxx) to what you want to see during runtime later.
- The text is used as a label and is therefore always displayed, in other words, it is independent of the signal status of the corresponding AVxxx input.
- The font size is reduced during the runtime if the input text length is greater than the label width.

Note

If a text is added to the "OS additional text" field of the input parameter (INxx), this text will be displayed even if the "Identifier" text field of the input parameter (AVxx) is not empty. If the "OS additional text" field of the input parameter (INxx) is empty, the "Identifier" text of the input parameter (AVxx) is displayed.

The number of input values may vary depending on the selected interlock block:

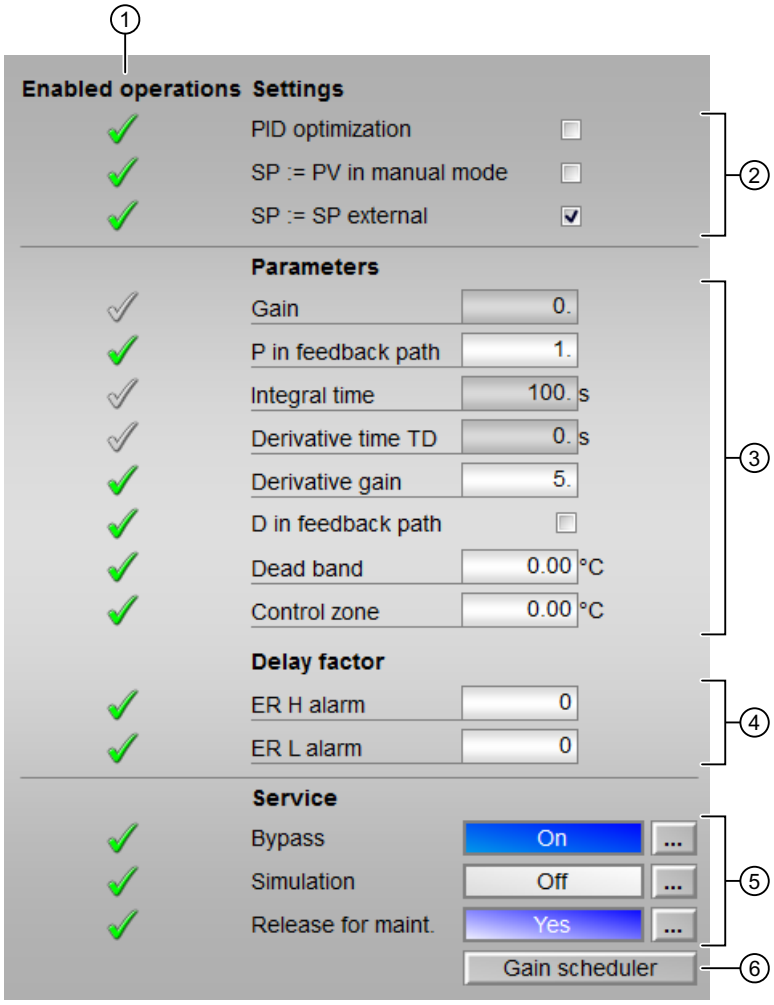
- **Intlk02:** the input values 1 and 2 are available.
- **Intlk04:** the input values 1 to 4 are available.
- **Intlk08:** the input values 1 to 8 are available.
- **Intlk16:** the input values 1 to 8 are available. The input values 9 to 16 become available by pressing the button (9). You can find additional information on this topic in the description for (1) and (9).

(13) Open faceplate of the input value

When you press the button (13), you can open the faceplate associated with each input value. The function of this button depends on the configuration in the engineering system (ES). See also the section Opening additional faceplates (Page 203) for more on this.

2.3.11 Parameter view of PID controllers

Parameter view of PID controllers



(1) "Enabled operation"

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (OS_Perm or OS1Perm).

(2) "Settings"

You can activate the following functions for the controller in this area:

- "PID optimization": ☑ activate controller optimization
- "SP := PV in manual mode": ☑ Bumpless switchover from manual mode to automatic mode
- "SP := SP external": ☑ Bumpless switchover of the setpoint for setpoint switchover from "external" to "internal". The internal setpoint is tracked to the external one.
  - With the PIDConR block, this area is only visible if you have set the `Feature` bit Switching operator controls for external setpoint to visible (Page 143) to 1.

(3) "Parameters"

In this area, you change parameters and therefore influence the controller. Refer to the Changing values (Page 253) section for more on this.

You can influence the following parameters:

- "Gain": Proportional gain
- "P in feedback path" : Proportional action to the feedback path (0 to 1), 0 = Proportional action is completely in the feedback path (only with PIDConL, PIDConR, and PIDStepL)
- "Integral time": Integral action time in [s]
- "Derivative time TD": Derivative action time in [s]
- "Derivative gain": Gain of the derivative action
- "D in feedback path": Derivative action is moved to the feedback path (only with PIDConL, PIDConR, and PIDStepL)
- "Dead band": Width of dead band
   Dead band is temporarily disabled
- "Control zone": Width of the control zone (only with PIDConL block)
- "Motor actuating time": Motor actuating time [s] (for PIDStepL block only)
- "Minimum pulse duration": Minimum pulse duration [s] (for PIDStepL block only)
- "Minimum break duration": Minimum break duration [s] (for PIDStepL block only)

(4) “Delay factor” (only for PIDConL and PIDConR)

In this area, you can change the following parameters:

- "ER H alarm": Delay factor at the positive setpoint step changes for incoming alarms at the control deviation monitoring `ER_AH_Lim`.
- "ER L alarm": Delay factor at the negative setpoint step changes for incoming alarms at the control deviation monitoring `ER_AL_Lim`.

(5) "Service"

You can select the following functions in this area:

- "Bypass" (only with PIDConL, PIDConR, and PIDStepL)
- "Simulation"
- "Release for maintenance" (with display for a maintenance request)

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Bypassing signals (Page 107) (for PIDConL, PIDConR, and PIDStepL)
- Simulating signals (Page 58)
- Release for maintenance (Page 64)

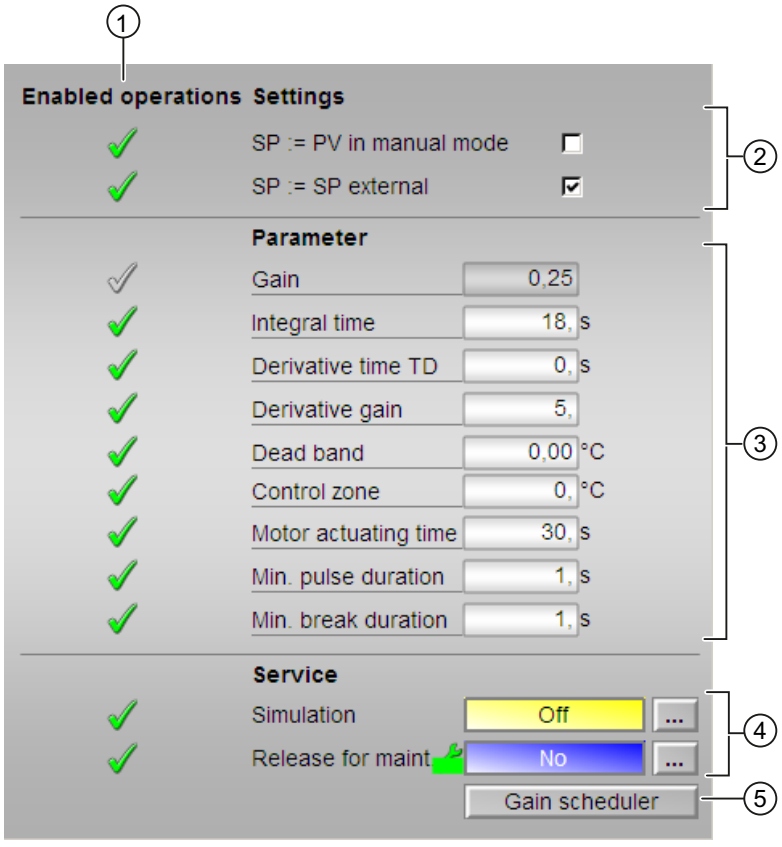
(6) Navigation button for the GainSched block

You can use this navigation button to reach the GainSched block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

2.3.12 Parameter view of FM controllers

Parameter view of FM controllers



(1) "Enabled operations"

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (OS_Perm or OS1Perm).

(2) "Settings"

You can activate the following functions for the controller in this area:

- "SP := PV in manual mode": ☑ Bumpless switchover from manual mode to automatic mode
- "SP := SP external": ☑ Bumpless switchover of the setpoint for setpoint switchover from "external" to "internal" The internal setpoint is tracked to the external one.

(3) "Parameters"

In this area, you change parameters and therefore influence the controller. Refer to the Changing values (Page 253) section for more on this.

You can influence the following parameters:

- "Gain": Proportional gain
- "Integral time" Integral action time in [s]
- "Derivative time TD": Derivative action time in [s]
- "Derivative gain": Gain of the derivative action
- "Dead band": Width of dead band
- "Control zone": Width of the control zone (for block FmTemp only)
- "Motor actuating time": Motor actuating time [s]
- "Minimum pulse duration": Minimum pulse duration [s]
- "Minimum break duration": Minimum break duration [s]

(4) "Service"

You can activate the following functions in this area:

- "Simulation"
- "Release for maintenance" (with display for a maintenance request)

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Simulating signals (Page 58)
- Release for maintenance (Page 64)

(5) Navigation button for the GainSched block

You can use this navigation button to reach the GainSched block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

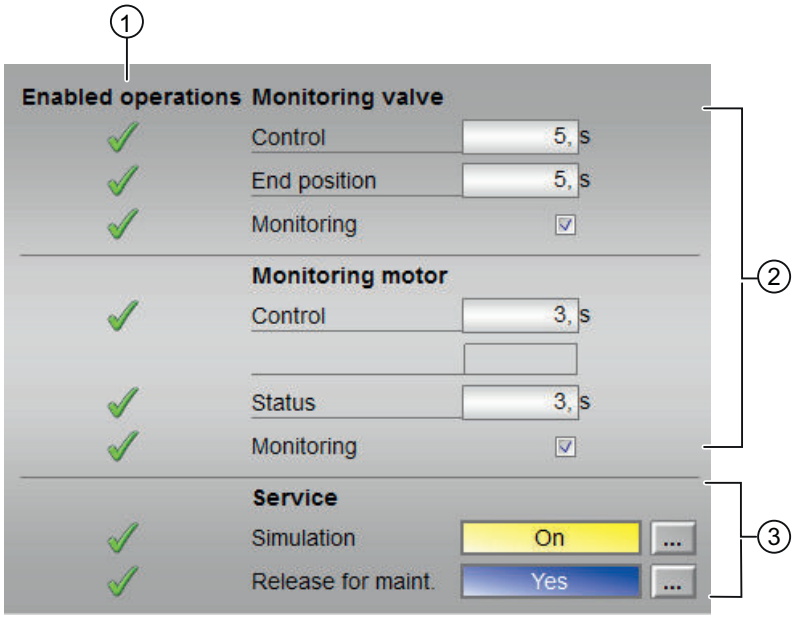
You can find additional information on this in the Opening additional faceplates (Page 203) section.

2.3.13 Parameter view for motors and valves

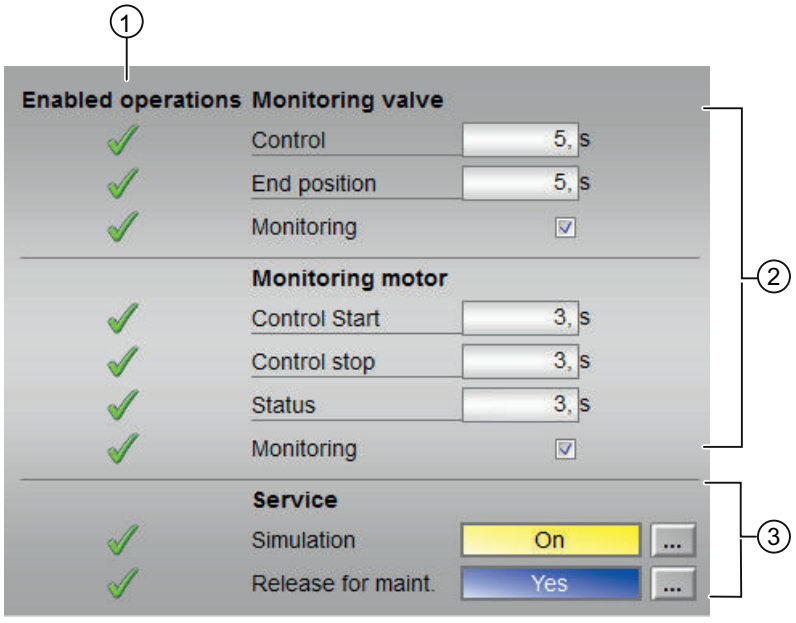
Parameter view for motors and valves

The following parameter view applies to the following bocks:

- MotL - Motor (Large) (Page 1059)
- MotRevL - Reversible motor (Page 1116)
- MotS - Motor (Small) (Page 1092)
- MotSpdL - Two-speed motor (Page 1210)
- VlvL - Valve (Large) (Page 1331)
- MotS - Motor (Small) (Page 1092)



Parameter view for MotL, MotRevL, MotSpdL and MotSpdCL with `Feature` bit 13 = 0.



Parameter view for MotL, MotRevL, MotSpdL and MotSpdCL with `Feature` bit 13 = 1.

(1) "Enabled operation"

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (OS_Perm or OS1Perm).

(2) "Monitoring"

In this area, you change parameters and therefore influence the motor. Refer to the Changing values (Page 253) section for more on this.

You can influence the following parameters:

- "Control": Monitoring time during startup and shutdown of the motor (dynamic)
- "Control stop": Monitoring time during shutdown of the motor (dynamic)
  only for MotL, MotRevL, MotSpdL and MotSpdCL with `Feature` bit 13 = 1
- "Control start": Monitoring time during startup of the motor (dynamic)
  only for MotL, MotRevL, MotSpdL and MotSpdCL with `Feature` bit 13 = 1
  "End position": Monitoring time during permanent operation of the motor (static)
- "Status": Monitoring time during permanent operation of the motor (static)
  The state is not displayed for small blocks.

Enabling "Monitoring"

You can enable monitoring by selecting the check box (☑)

You can find additional information on this in the section Monitoring the feedbacks (Page 97).

(3) "Service"

You can select the following functions in this area:

- "Simulation"
- "Release for maintenance" (with display for a maintenance request)

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Simulating signals (Page 58)
- Release for maintenance (Page 64)

See also

VlvS - Valve (small) (Page 1362)

2.3.14 Limit view of FM controllers

Limit view of FM controllers

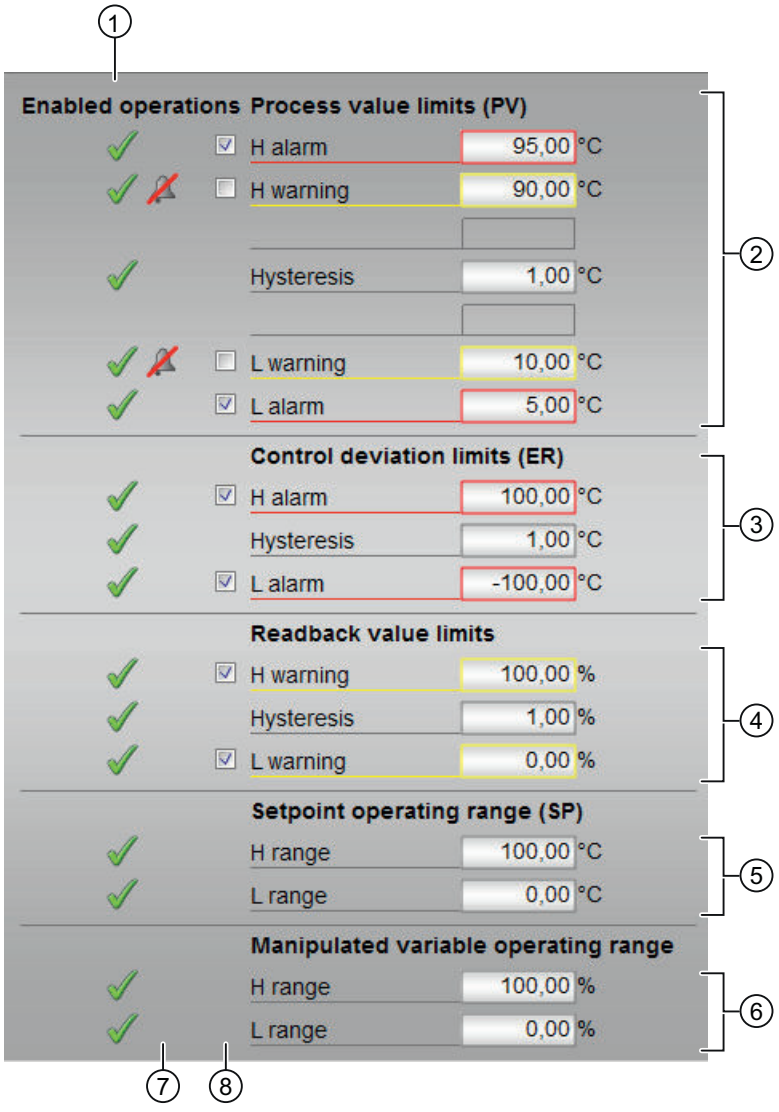
Several values are set in this view by default:

- Process value limits
- Error signal limits
- Readback value limits
- Setpoint operation range

The toolbars of the faceplate and the block icon indicate when the limits are reached or violated.

Note

The symbols displayed are not valid for user-configured message classes. Take into consideration the validity of terms for User-configured message classes (Page 41)



(1) "Enabled operation"

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (OS_Perm or OS1Perm)

(2) "Process value limits (PV)"

In this area, you can enter the limits for the process value. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "H warning": Warning high
- "H tolerance": Tolerance high
- "Hysteresis"
- "L tolerance": Tolerance low
- "L warning": Warning low
- "L alarm": Alarm low

(3) "Error signal limits (ER)"

In this area, you can enter the limits for the control deviation. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "Hysteresis"
- "L alarm": Alarm low

(4) "Readback value limits (RBK)"

In this area, you can enter the limits for the readback value. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H warning": Warning high
- "Hysteresis"
- "L warning": Warning low

(5) "Setpoint operation range (SP)"

In this area, you can enter the limits for the setpoint operation range. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H range": Range limit high
- "L range": Range limit low

(6) "Manipulated variable operating range"

In this area, you can enter the limits for the manipulated variable operation range. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H range": Range limit high
- "L range": Range limit low

(7) "Message suppression/delay"

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

Alarm delays are also displayed in this position; for more on this see section Area of application of the alarm delays (Page 195).

(8) Suppressing messages

You can enable / disable messages by setting the check mark.

2.3.15 Limit view of PID controllers

Limit view of PID controllers

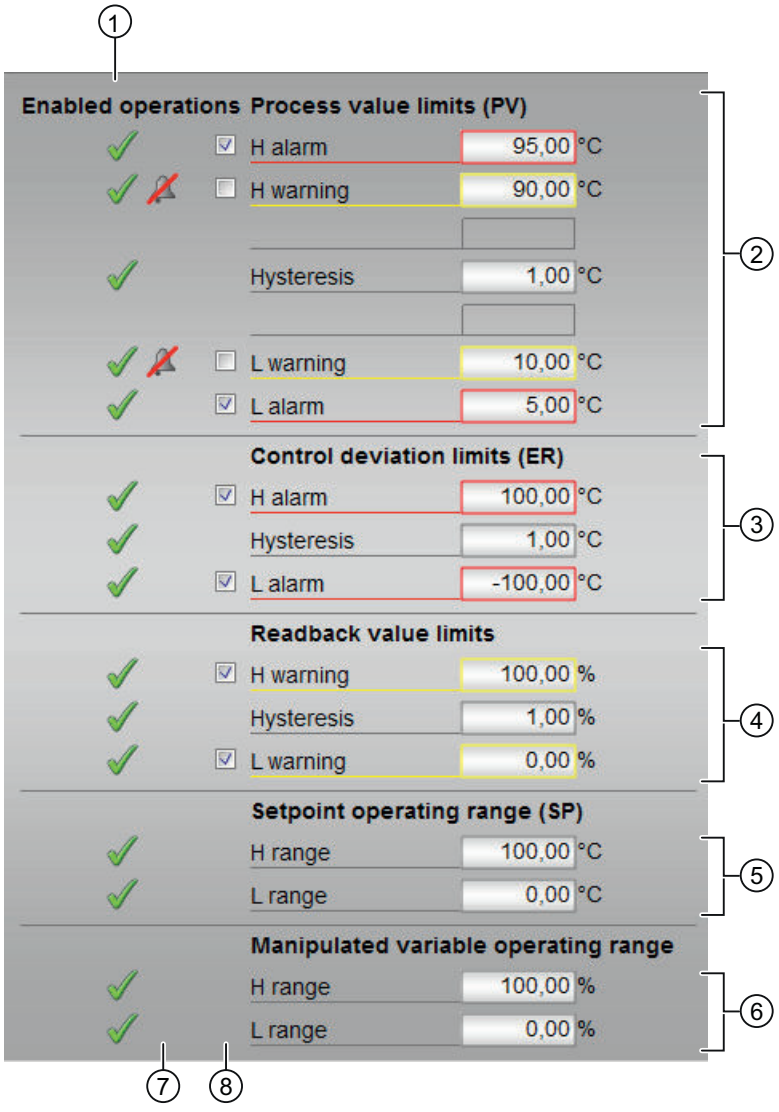
Several values are set in this view by default:

- Process value limits
- Error signal limits
- Readback value limits
- Setpoint operation range

The toolbars of the faceplate and the block icon indicate when the limits are reached or violated.

Note

The symbols displayed are not valid for user-configured message classes. Take into consideration the validity of terms for User-configured message classes (Page 41).



(1) "Enabled operation"

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (OS_Perm or OS1Perm)

(2) "Process value limits (PV)"

In this area, you can enter the limits for the process value. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "H warning": Warning high
- "H tolerance": Tolerance high (not with PIDConS)
- "Hysteresis"
- "L tolerance": Tolerance low (not with PIDConS)
- "L warning": Warning low
- "L alarm": Alarm low

(3) "Error signal limits (ER)" (not with PIDConS)

In this area, you can enter the limits for the control deviation. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "Hysteresis"
- "L alarm": Alarm low

(4) "Readback value limits (RBK)" (not with PIDConS)

In this area, you can enter the limits for the readback value. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H warning": Warning high
- "Hysteresis"
- "L warning": Warning low

(5) Setpoint operation range (SP)

In this area, you can enter the limits for the setpoint operation range. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H range": Range limit high
- "L range": Range limit low

(6) Manipulated variable operating range

In this area, you can enter the limits for the manipulated variable operation range. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H range": Range limit high
- "L range": Range limit low

(7) "Message suppression / delay"

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

Alarm delays are also displayed in this position; for more on this see section Area of application of the alarm delays (Page 195).

(8) Suppress messages

You can enable / disable messages by setting the check mark.

2.3.16 Limit view of motors

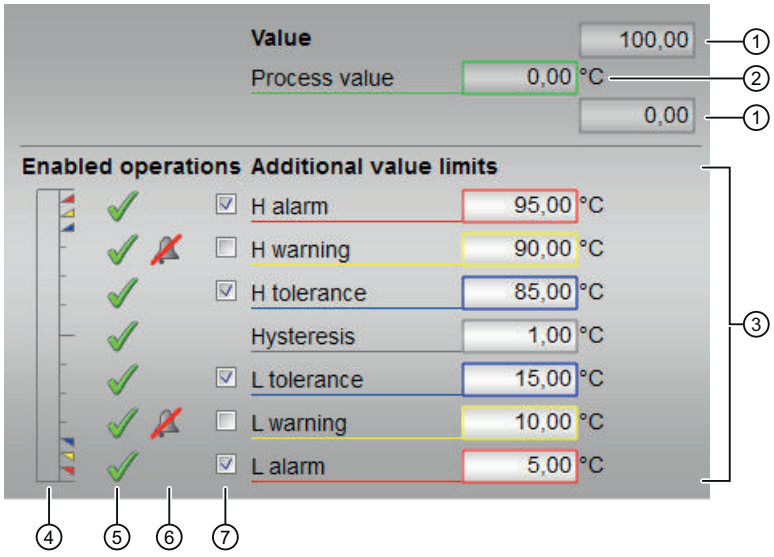
Limit view of motors

The limit view of motors is only available when an AV block has been interconnected to the motor.

The toolbars of the faceplate and the block icon indicate when the limits are reached or violated.

Note

The symbols displayed are not valid for user-configured message classes. Take into consideration the validity of terms for User-configured message classes (Page 41).



(1) High and low scale range for the additional value

These values provide information on the display range for the bar graph of the additional value. The scale range is defined in the engineering system.

(2) Display of the additional value including signal status

This area shows the current additional value with the corresponding signal status.

(3) "Limits for the additional value"

In this area, you can enter the limits for the additional value. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "H warning": Warning high
- "H tolerance": Tolerance high
- Hysteresis
- "L tolerance": Tolerance low
- "L warning": Warning low
- "L alarm": Alarm low

(4) Bar graph for the additional value

This area shows you the current additional value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(5) "Enabled operation"

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (OS_Perm or OS1Perm)

The colored triangles indicate the specified limits **(3)** for the additional value.

(6) Message suppression / delay

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

Alarm delays are also displayed in this position; for more on this see section Area of application of the alarm delays (Page 195).

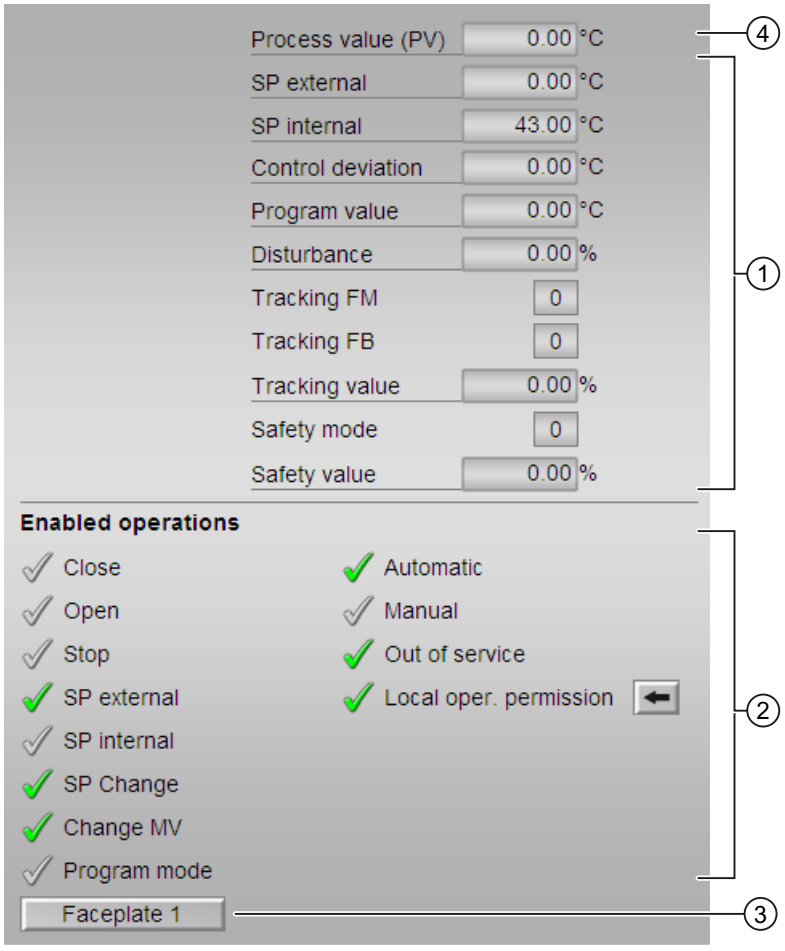
(7) Suppress messages

You can enable / disable messages by setting the check mark.

2.3.17 Preview of FM controllers

Preview of FM controllers

The preview shows you the parameters that you, as an OS operator, can control in the entire block. You cannot control anything in this view, however.



(1) Preview area

This area shows you a preview for the following values:

- "SP external": currently applicable external setpoint
- "SP internal": currently applicable internal setpoint
- "Control deviation": Current Control deviation
- "Program mode": Specified value for program mode
- "Disturbance variable": additive value for feedforward control
- "Tracking FM": track a manipulated variable in the FM module (value is 1)
- "Tracking FB": Track manipulated variable at the block (value is 1)

- "Tracking value": Effective manipulated variable for "Track manipulated variable at block"
- "Safety mode": safety mode in the FM module (value is 1)
- "Safety value": effective manipulated variable for "Safety mode"

(2) "Enabled operations"

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Close": You can select the manipulated variable "Close". If text is configured for this command, it is also displayed in brackets. You can find more information about this in the Section Labeling of buttons and text (Page 205).
- "Open": You can select the manipulated variable "Open". If text is configured for this command, it is also displayed in brackets. You can find more information about this in the Section Labeling of buttons and text (Page 205).
- "Stop": You can select the manipulated variable "Stop". If text is configured for this command, it is also displayed in brackets. You can find more information about this in the Section Labeling of buttons and text (Page 205).
- "SP external": You can feedforward the external setpoint.
- "SP internal": You can feedforward the internal setpoint.
- "Change SP": You can change the setpoint.
- "Change MV": You can change the manipulated variable.
- "Program mode": You can switch to program mode.
- "Automatic": You can switch to automatic mode.
- "Manual": You can switch to manual mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the "OpStations"" block. Additional information is available in the section Operator control permissions (Page 248).

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

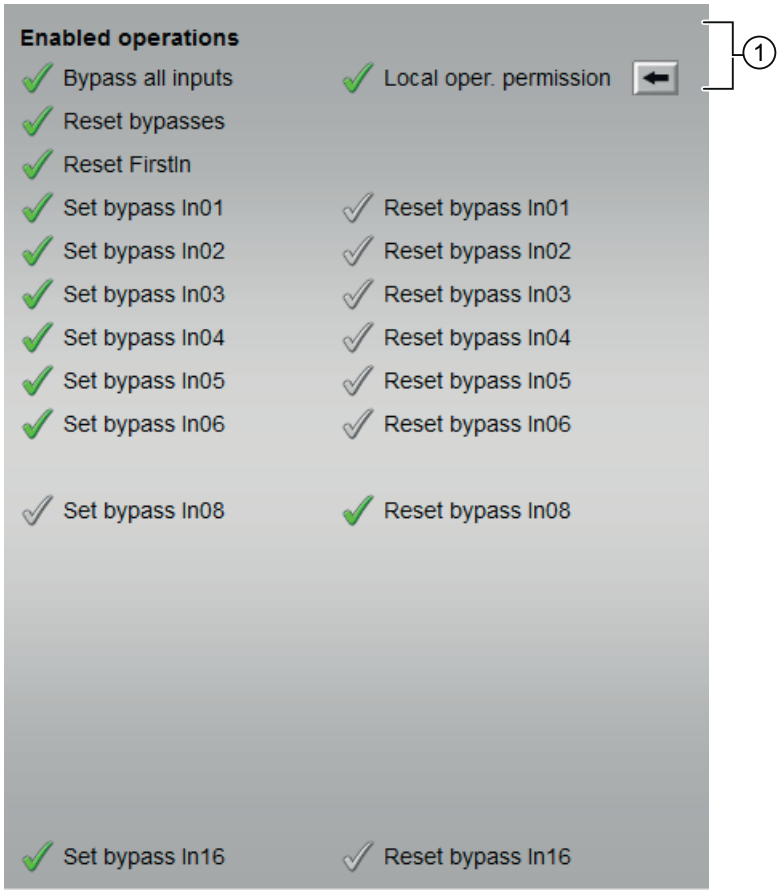
See also the Opening additional faceplates (Page 203) section for more on this.

(4) Process value

This area displays the real process value (PV).

2.3.18 Preview of interlock blocks

Preview of interlock blocks



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

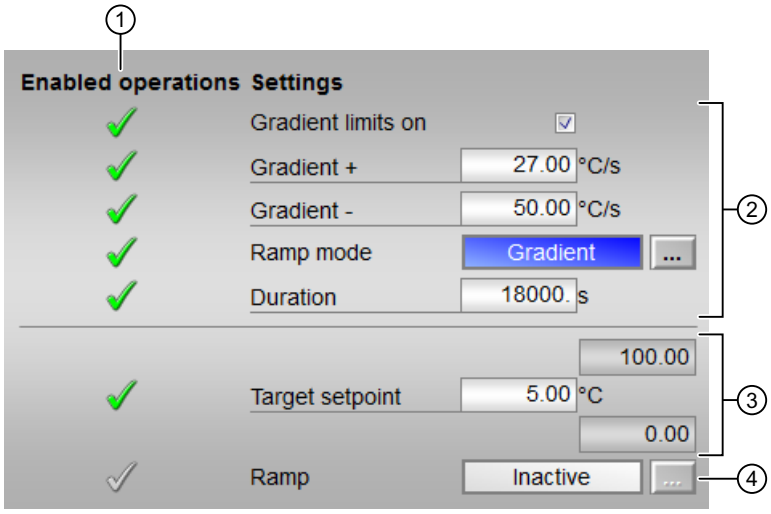
- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (OS_Perm or OS1Perm).

The following enabled operations are shown here:

- "Local operating permission": Use the <-- button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

2.3.19 Ramp view

Ramp view



(1) "Enabled operations"

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

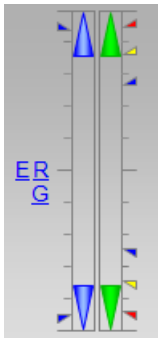
Symbols for enabled operations:

- Green check mark: the OS operator can control this parameter
- Gray check mark: the OS operator cannot control this parameter at this time due to the process
- Red cross: the OS operator cannot control this parameter due to the configured AS operator control permissions (OS_Perm or OS1Perm)

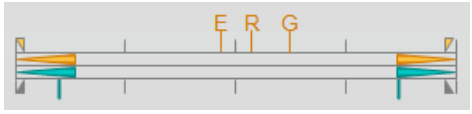
(2) Enable "gradient limit"

Use this check box to enable "gradient limit" for the setpoint. "Gradient limit" can be set separately for positive or negative setpoint changes ("Gradient +" or "Gradient -"). Refer to the Changing values (Page 253) section for more on this.

If there is a difference between target setpoint and currently effective setpoint, a blue "G" may be displayed at the bar in standard view of FmCont, FmTemp, PIDConL, PIDConR, PIDStepL, OpAnL and MotSpdCL with parameter assignment of `SP_RateTarget` (target setpoint for gradient limit).



If there is a difference between the target manipulated variable and the currently effective manipulated variable, an orange "G" may be displayed at the bar in the standard view of VlvAnl with parameter assignment of `MV_RateTarget` (target manipulated variable for gradient limit).



The gradient limitation includes the Ramp function (Page 123). You can set the ramp mode in the following two ways:

- Gradient
- Duration [s]

(3) "Target setpoint"

In this area, you can set the type of ramp function for the setpoint.

You can set the time duration and the target setpoint. Refer to the Changing values (Page 253) section for more on this.

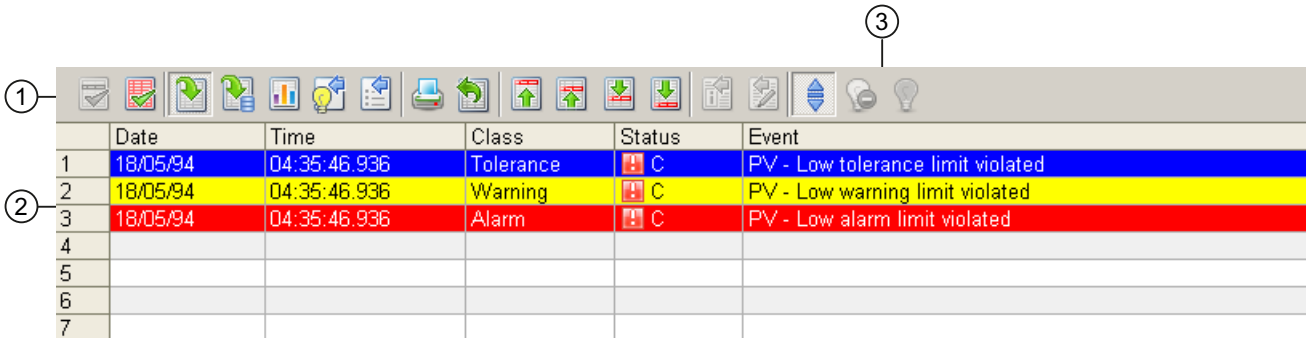
(4) Enable "Ramp"

You can use this control to enable or disable the configured function in the ramp function for the setpoint change.

You can only enable this when the setpoint specification is set to "Internal" in the standard view of the block. The enable is only valid for one setpoint change and is subsequently disabled again.

2.3.20 Alarm view

Message view



(1) Toolbar

If the short-term archive list is selected, a new button appears in the toolbar:

You can use this button to toggle between the "History" and "Operator messages" views.

You must be registered with the "Higher process control" operating permission in order to export and hide messages.

(2) Display area for alarms

For additional information about the alarm view, refer to the *WinCC Information System* Online Help.

(3) "Hide messages" button

Messages can be displayed and/or hidden with this button. The view of this button changes accordingly:

 Show messages

 Hide messages

"Higher process controlling" operating permission is required and manual hiding must be active. You can find additional information in the manual "Process Control System PCS 7 Operator Station".

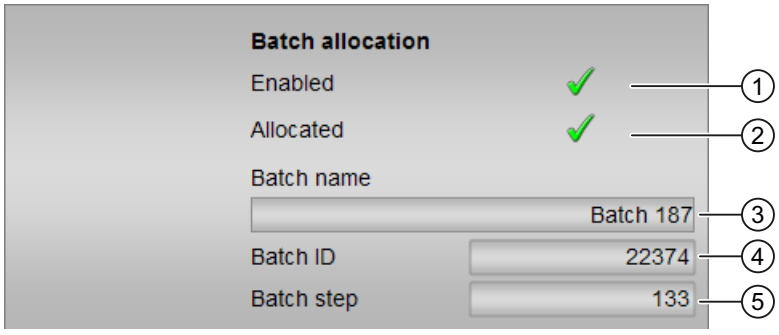
2.3.21 Batch view

Batch view

This area shows a display of the batch that is currently running (`Batchview`).

You can use the internal tag "@APLBatchEnable" to enable/disable the "Batch" button in the toolbar.

Value of @APLBatchEnable	Operating the "Batch" button in the toolbar	Icon of the "Batch" button in the toolbar
0	Disabled	
1	Enabled	



(1) "Enabled"

This area shows you if the block is enabled for operation via SIMATIC BATCH (`BatchEn = 1`).

(2) "Allocated"

This area shows if the block is currently in use by SIMATIC BATCH (Occupied = 1).

(3) "Batch name"

This area shows the name of the batch that is currently running (`Batchname`).

(4) "Batch ID"

This area shows the identification number of the batch that is currently running (`BatchID`).

The batch view is disabled if `BatchID` = 16#00000000.

(5) "Batch step"

This area shows the step number of the batch that is currently running (`StepNo`).

2.3.22 Memo view

Memo view

You can leave temporary messages for other OS operators in this view. Messages are entered in the text box, and saved and activated by selecting the check box in the lower right corner of the faceplate.

(1) Text box for notes

(2) Check box for activating the note

The next time the faceplate is opened or there is a process picture change, you can see in the status bar of the block icon and the faceplate that there is a new message for you.

Clearing the check box deletes the indicators in the status bars.

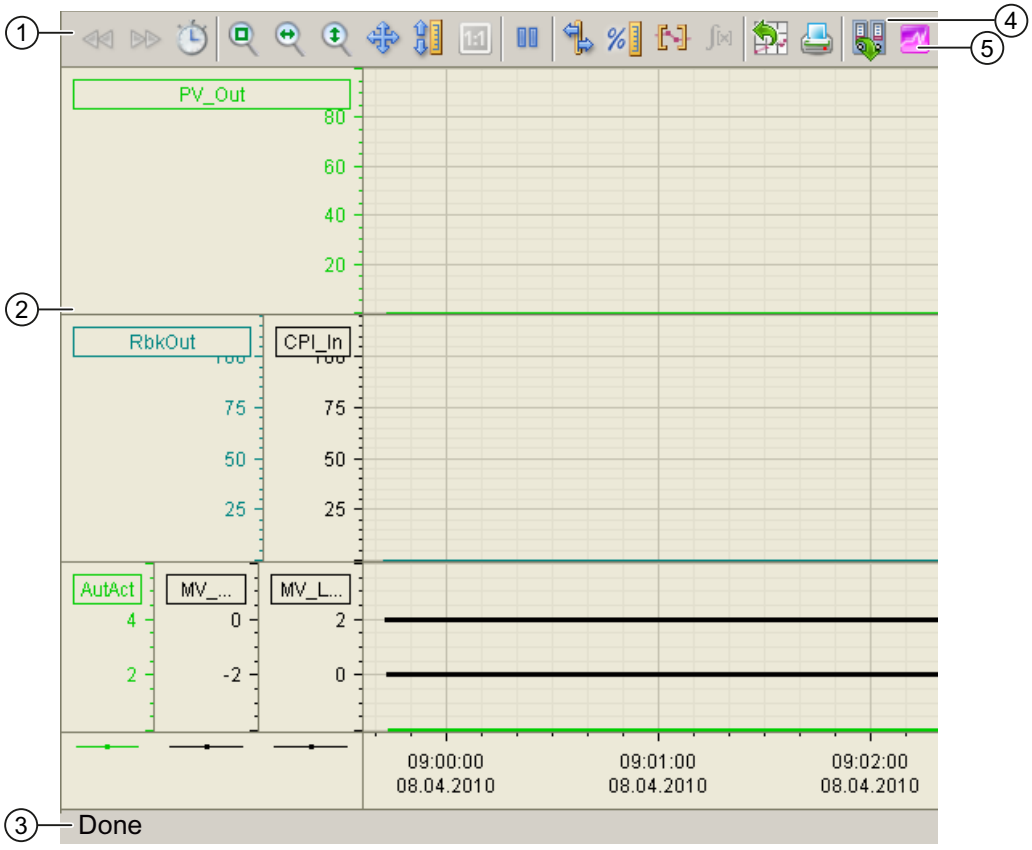
The message is not deleted automatically.

Note

The content of the memo view is cleared when you perform a full compilation and download of the OS.

2.3.23 Trend view

Trend view



(1) Toolbar

(2) Display area for trends

(3) Status bar

(4) Button for switching between archive tags and online tags. The status bar shows if the trend view is working with online data or archive data.

(5) Button for opening the "Scatter plot" window

The Export button is only visible and operable with the "Higher-level process control" operating permission.

For additional information about the trend view, refer to the *WinCC Information System* Online Help.

Configuration of the trend view

The trend view can be configured so that archive values are displayed immediately after opening. A prerequisite for this is that archive variables exist. Proceed as follows:

- A "1" is attached at the block icon in the "TrendPictureName" property. A semicolon is used as the separator to the name of the trend view.
  Example: @pg_apl_trendPID_Statistic.pdl;1

Special considerations for controllers

You can select two different representations for the display area:

1. Detailed display (default setting):

Display area consisting of three coordinate systems:

- Setpoint trend, actual value trend;
- Manipulated variable, control performance index trend;
- Binary trend via automatic/manual, manipulated variable at high or low limit

Open the scatterplot diagram with the user button (number 2) in the toolbar. It shows a coordinate system with the process value on the value axis and the manipulated variable or position feedback on the X axis. A new value pair is entered into the coordinate system with each cycle.

If you want to use the detail display, you need to enter the following in the block icon under Trends in the WinCC Graphics Designer:

TrendPictureName = @pg_apl_trendPID_Statistic.pdl

2. Simple display:

Display area consisting of two coordinate systems:

- Setpoint trend, actual value trend;
- Manipulated variable;

If you want to use the simple display, you need to enter the following in the block icon under Trends in the WinCC Graphics Designer:

TrendPictureName =@pg_apl_trendPID.pdl

Notes on step controllers with position feedback:

If you use a step controller with position feedback as the controller type, you need to enter the following in the block icon under Trends in the WinCC Graphics Designer:

TrendConfiguration5 = *.MV#Value;...

TrendConfiguration6 = .RbkOut#Value;...

The following applies to all other controller types (default setting):

TrendConfiguration5 = .MV#Value;...

TrendConfiguration6 = *.RbkOut#Value;...

2.3.24 APL Operator Trend Control (AOTC)

Opening the AOTC window

Press and hold the Ctrl key and left-click the value on the block icon to open the AOTC window. The first value is added to the Trend Control and the detailed information is displayed in the first row of the overview area.

Adding values to the AOTC window

To add values to the AOTC window, press and hold the Ctrl key and left-click the value on the block icon. The value is added to the Trend Control. You can add up to 8 values in the AOTC window. The following colors are assigned to the trends in the sequence they are added:

1. Black (HTML code 000000)
2. Red (HTML code FF0000)
3. Green (HTML code 00FF00)
4. Blue (HTML code 0000FF)
5. Yellow (HTML code FFFF00)
6. Turkish (HTML code 00FFFF)
7. Pink (HTML code FF00FF)
8. Orange (HTML code FFA100)

If the value is an archived value, the value is displayed from the archive. If the value is not archived, it is displayed as an online value. The default time range for the time axis is 1 minute.

Note

For the Web Navigator, when a new value is added, the Trend Control always displays only the online value. You can switch to display the archived values through the buttons "Monday", "Tuesday", and so on.

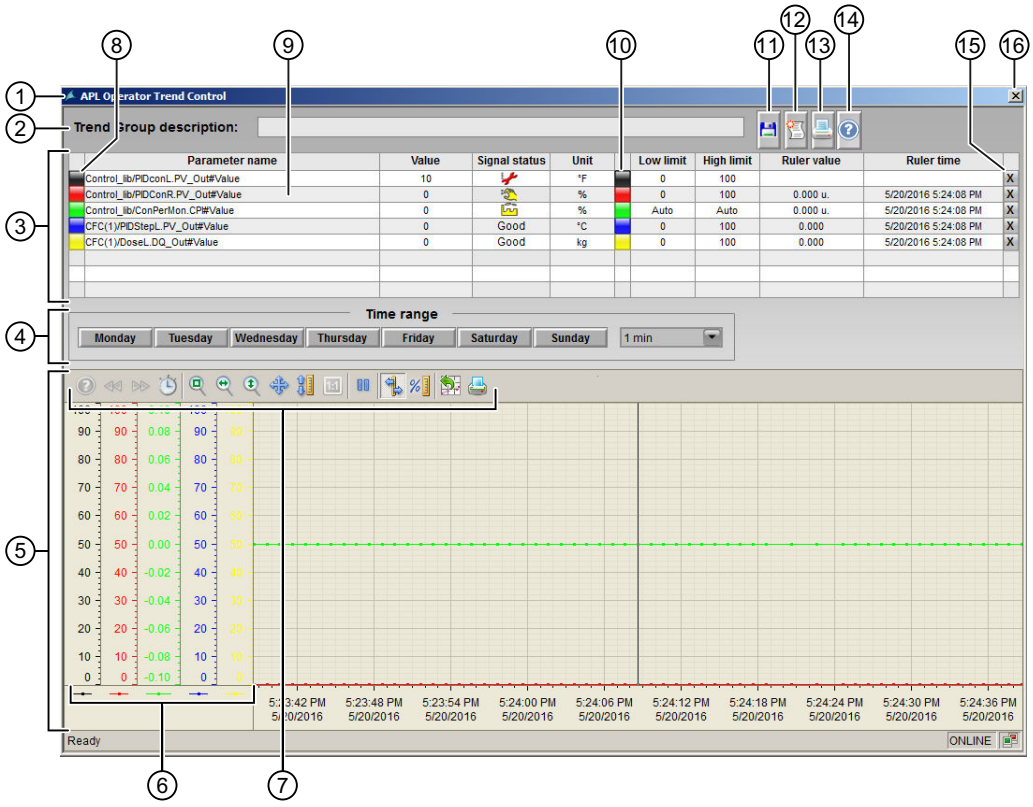
Adding digital values to the AOTC window

To add digital values to the AOTC window, press and hold the Ctrl key and left-click the value on the block icon. You can add digital values from the following blocks:

Family	Blocks
Drives	MotL, MotRevL, MotSpdCL, MotSpdL, MotS, VlvL, VlvS, Vlv2WayL, VlvAnL, VlvMotL, VlvPosL
Operate	OpDi01, OpTrig, OpDi03
Monitor	MonDiL, MonDiS, MonDi08

Example: If you perform Ctrl key + left-click operation on the MotL block icon, the AOTC window opens showing its current status taken from the parameter `FbkRunOut.Value`. After opening the AOTC window, if you perform Ctrl key + left+click operation on any other block then the parameter corresponding to that block is added to the AOTC window.

The AOTC window



1. Title
2. Trend group description
3. Tag row
4. Time range selection
5. WinCC online trend control
6. Value axis
7. Toolbar of the trend window
8. Enabling/disabling a trend
9. Opening a faceplate
10. Enabling/disabling a value axis
11. Saving a trend group
12. Opening a separate message window
13. Printing screenshot of the AOTC window
14. Help button
15. Deleting a trend
16. Closing the AOTC window

(1) Title

The title displays the name of the AOTC. If the trend group is not saved, the default title "APL Operator Trend Control" is displayed. If a saved trend group is opened, title displays the group name.

(2) Trend group description

This field displays the description of the trend group if it is saved with a description otherwise it is empty.

(3) Tag row

The tag row displays the following information about the trend added.

- Button to enable/disable the trend:
  Click this button to enable or disable the trend. For more information, refer to the description of **(8) Enabling/disabling a trend**.
- "Parameter name":
  This field displays the name of the trend.
- "Value":
  This field displays the current value of the trend.
- "Signal status":
  This field displays the signal status of the trend. The corresponding icon is displayed in case the quality is not good that is the quality code is not 0x80. In case of good quality (quality code 0x80), the text "Good" is displayed.
- "Unit":
  This field displays the unit of the trend. Units are calculated only while adding a trend to the AOTC window. If the unit of a trend changes after it has been added to the AOTC window, the new unit will not be updated in the AOTC window. For example, if `PV_Unit` changes, the trend must be removed and added again.
- Button to enable/disable the value axis:
  Click this button to enable or disable the value axis. For more information, refer to the description of **(10) Enabling/disabling a value axis**.
- "Low limit":
  This field displays the low limit of the trend. The low limit is taken from the tag "XX_OpScale#Low". If the tag is not present, the value is taken automatically by the Trend control and low limit field displays "Auto".
- "High limit":
  This field displays the high limit of the trend. The high limit is taken from the tag "XX_OpScale#High". If the tag is not present, the value is taken automatically by the Trend Control and the high limit field displays "Auto".
- "Ruler value":
  This field displays the ruler value of the trend where the ruler intersects with the trend in the Trend Control.

- "Ruler time":
  This field displays the ruler time of the trend where the ruler intersects with the trend in the Trend Control.
- Button to delete the trend:
  Click this button to delete the trend. For more information, refer to the description of **(14) Deleting a trend**.

(4) Time range selection

This area provides the option to select the time range based on the weekdays for the archived view and fixed time frame for the online view.

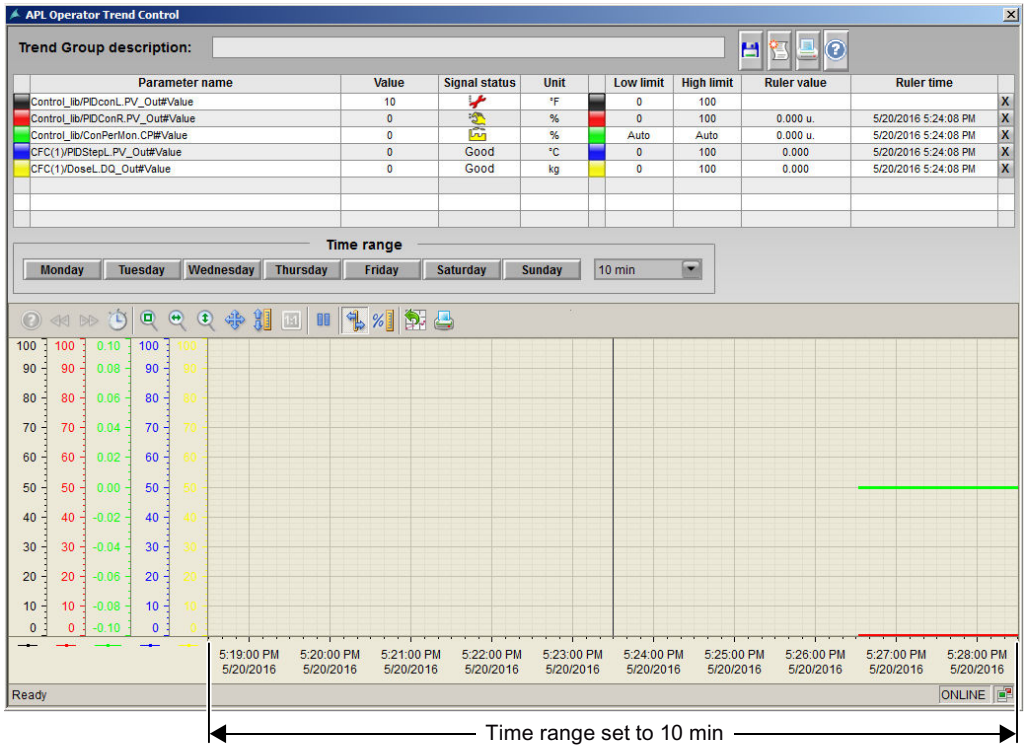
Press the buttons named "Monday" to "Sunday" to switch to the archive view for that day. For example, if today is Tuesday and the "Tuesday" button is pressed, the archived values from today will be displayed. If the "Sunday" button is pressed, the archived values from last Sunday will be displayed. The time range is 24 hours. If the trend is not archived, no value will be displayed.

Press the "Start/stop" button from the trend toolbar to switch back to the online view.

Use the drop-down list in the time range selection area to change the time range of the time axis. The following preconfigured time ranges are available:

- Hour/minute based
  - 1 min
  - 2 min
  - 5 min
  - 10 min
  - 15 min
  - 30 min
  - 1 h
  - 2 h
  - 3 h
  - 5 h
  - 12 h
  - 24 h

The following figure shows the AOTC window when the time range is selected to "10 min" from the drop-down list in "Time range" section:



(5) WinCC online trend control

The Trend Control displays the trends added to the AOTC window. Each trend added to the AOTC window will appear in different color.

The following colors have been defined default for the 8 trends:

1. Black
2. Red
3. Green
4. Blue
5. Yellow
6. Turkish
7. Pink
8. Orange

(6) Value axis

Each trend is shown in a separate value axis, therefore there can be maximum of 8 value axis. The value axis appears in the same color as the corresponding trend.

(7) Toolbar of the trend window

Use these button to operate the trend window.

For more information, refer to *WinCC Information System* Online Help.

(8) Enabling/disabling a trend

Press these buttons to disabled or enabled trends.

If a trend is enabled, the corresponding button will be displayed in the same color as the value trend.

If a trend is disabled, the corresponding button will be displayed in Grey color.

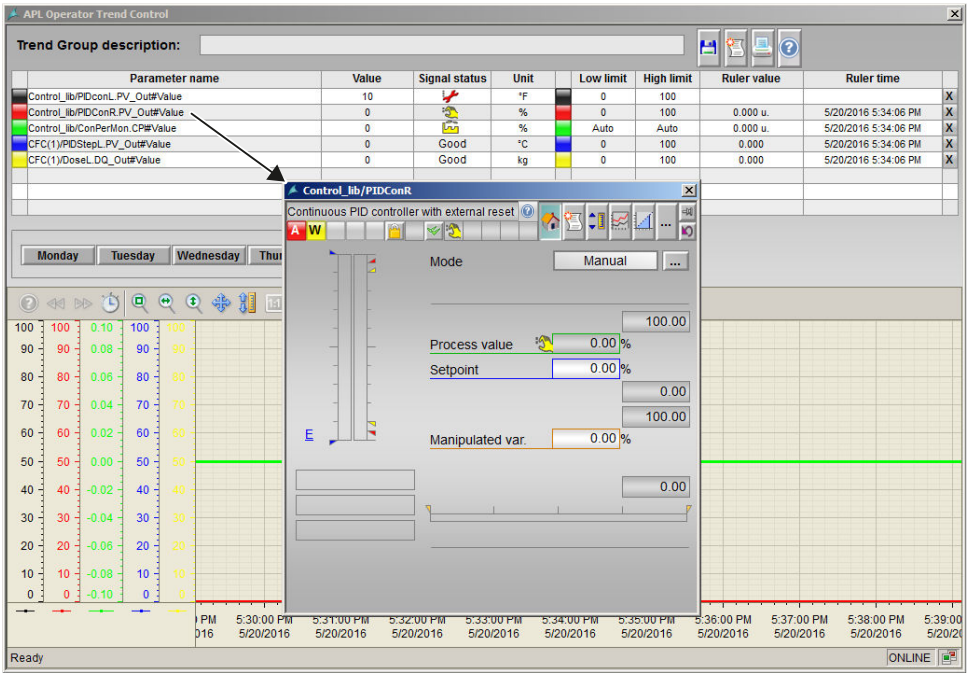
By default the trend added to the AOTC is enabled, therefore the corresponding button is visible and appears in the the same color as the value trend.

When the button is pressed the first time, the corresponding trend is hidden from the Trend Control. The color of the button changes to Grey. The corresponding tag row in the overview area is hidden except the "Parameter name".

On pressing the button again, the corresponding trend is visible again in the Trend Control. The color of the button changes to the same color as the value trend. The corresponding row of values is visible again.

(9) Opening a related faceplate

Click the parameter name in the tag row to open the corresponding faceplate.

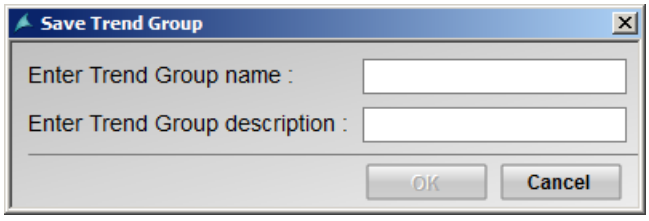


(10) Enabling/disabling a value axis

Click the button next to the I/O field of the unit to enable or disable the value axis. The value itself will stay in the Trend Control, only the value axis will be hidden from the Trend Control.

(11) Saving and reopening a trend group

Click the "Save" button to save the trend group. The "Save Trend Group" window appears to save the trend group:



Enter the following details in the "Save Trend Group" window:

- Trend group name
- Trend group description

In the "Save Trend Group" window, the "OK" button is enabled only if the trend group name is entered and the Enter key is pressed. The trend group descitption is optional.

The trend group name can be upto 32 characters long and the description can be upto 64 characters long.

Press the "OK" button to save the trend group. You can abort the operation by pressing the "Cancel" button. The dialog will close without saving the name or description.

Click the "Trend system" standard button at the bottom of the WinCC runtime window to open a saved trend:

Once the trend group is saved, further changes made in that trend group from the AOTC window will be saved automatically.

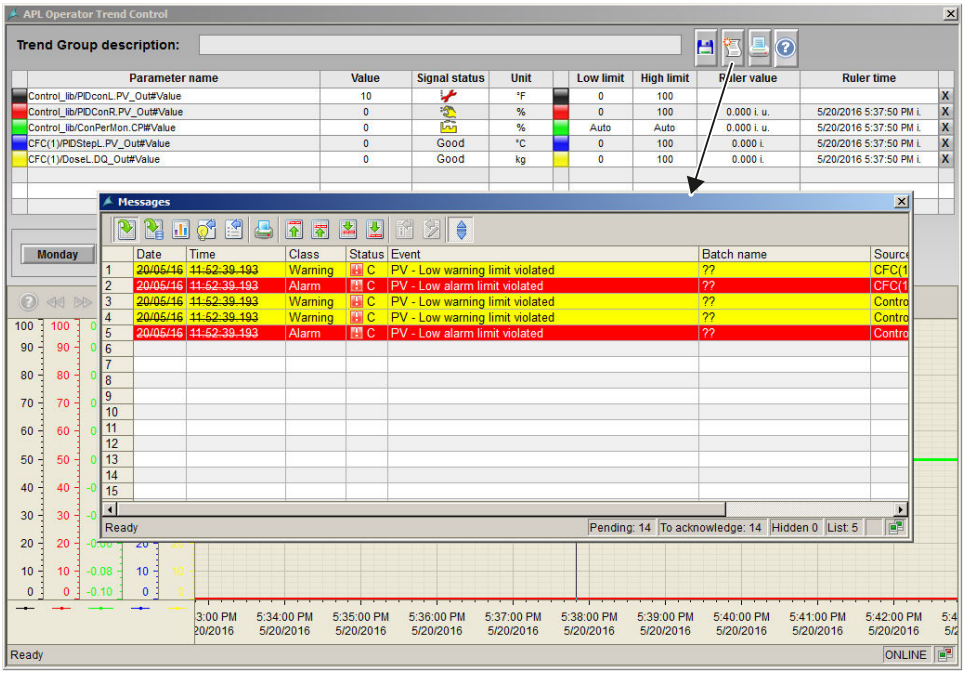
The trend group name should be different from the previously saved trend groups. You cannot save the trend group again with the same name.

The following special characters are not allowed in the trend group name:

' ' (space character), '.', ',', ';', ':', '/', '=', '\t', '\\', '"', '<', '>', '{', '}', '[', ']', '|', '?', '*'

(12) Opening a separate message window

Press the "Messages" button to open a new message window which displays the messages for the selected block types corresponding to the trends added in the AOTC window.



In the online mode, only the actual messages (all active messages, acknowledged and unacknowledged) will be displayed.

If you select a time range displaying values from the archive, the message window will display all the messages within this time range.

Messages only from the enabled trend will be displayed. If a trend is enabled or disabled and the message window is open, the corresponding messages will be displayed or not displayed accordingly. Similarly if a trend is deleted, the corresponding messages will disappear from the messages window.

If two or more trends of the same block type are added to the AOTC window, and one of them is disabled, the corresponding messages will not be removed from message window. All the trends should be disabled or deleted to remove the corresponding messages from the message window.

Only one message window can be opened at a time. If one more AOTC window is opened and you want to display the messages for the values in that window, the already opened message view should be closed and opened again from the specific AOTC window.

(13) Printing screenshot of the AOTC window

Press the "Print" button to print the screenshot of the complete AOTC window. WinCC runtime (also the Web Navigator) should be in full screen mode without scroll bar. As screenshot is taken from the process picture visible in runtime, the AOTC window should be visible completely in the runtime, otherwise the screenshot will show only the visible section of the AOTC faceplate.

(14) Help button

Click this button to open the AOTC chapter in the *APL Operator Guide* online help. You will find detailed information about the AOTC window and its functionalities in this online help.

(15) Deleting a trend

Click the "X" button at the end of the row in the overview area to delete existing values.

The corresponding trend is removed from the Trend Control. In the overview area, the rows below the deleted row are moved up and an empty row is added at the end.

New values are added to the first empty row.

(16) Closing the AOTC window

Click the "Close" button in the title bar to close the AOTC window.

Operator permissions

The operator permissions are similar to the existing online trend function.

The following fields are enabled only if the area authorization is available:

1. Delete buttons to delete trends.
2. Save button to open the pop-up window to save the trend group.
3. Field to enter the trend group name in the pop-up window.
4. Field to enter the trend group description in the pop-up window.
5. The "OK" button to save the trend in the pop-up window.

Note

The functions for example, disable/enable trend, disable/enable value axis, delete trend, open message window, print AOTC screenshot, need no operation area or quitting function. The feature "Save Trend Group" needs an operation area where only 3 step operation is possible. It does not support 2 step operation.

2.3.25 Limit operation and display in the faceplate

Limit operation and display in the faceplate

The limit value view of the faceplate can be used to modify limits and the hysteresis if the corresponding operating permission (higher process controlling) is available. The limits are displayed graphically in the standard view of the faceplate.

If the limits are reached or exceeded, an alarm, warning or tolerance class message is triggered. This is indicated graphically as follows:

Symbol	Meaning
	Alarm
	Warning
	Tolerance

Note

The symbols displayed are not valid for user-configured message classes. For user-configured message classes, the colors and abbreviations depend on the project-specific setting. Please take into consideration the validity of terms for user-configured message classes (Page 41).

See also

User-configured message classes (Page 41)

2.3.26 Central color management

Central color management

As of version 8.1, the colors in the faceplates and block icons can be changed centrally via the central color palette in WinCC. The central color palette is located in the "User interface and design" tab of the project properties.

The following colors cannot be changed via the central color palette:

- The colors (except trend colors) of WinCC Alarm Control, WinCC Online Trend Control, WinCC Ruler Control and WinCC Function Control.
- The window titles can only be changed by editing the Windows designs.
- The colors of check boxes cannot be changed.
- Alarm colors are changed via the PCS 7 message system.
- Trend colors are not automatically changed in the block icons in existing projects, because these properties are changeable instance-specific and not in the update mechanisms for the block icons.

The user can use a color spectrum of 0-199; the APL color spectrum begins with the color index 200.

Description of the APL color index:

Color index for block icons

The following color indexes can be created for block icons.

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Type block icons
200	1	0	B6B6B6	Gray	Background: Background color	BackgroundColor	Block icons with analog values
201	1	0	DADADA	Very light gray	Background: Fill pattern color	BackgroundFillPatternColor	Block icons with analog values
202	1	0	B6B6B6	Gray	Tag display: Background color	TagBackColor	All block icons with tag display
203	1	0	DADADA	Very light gray	Tag display: Fill pattern color	TagFillColor	All block icons with tag display
204	1	0	919191	Dark gray	Tag display: Border color	TagBorderColor	All block icons with tag display
205	1	0	000000	Black	Tag display: Font color	TagFontColor	All block icons with tag display
206	1	0	00FFFA	Pure cyan	Tag highlight: Background color	BlockIconHighlightBackColor	All block icons
207	1	0	5CC2BC	Medium cyan	Tag highlight: Fill pattern color	BlockIconHighlightFillColor	All block icons
208	1	0	FFFFFF	White	Tag highlight: Font color	BlockIconHighlightFontColor	All block icons
209	1	0	B6B6B6	Gray	Display of high limit: Background color	LimitObjectHighBackColor	Block icons with display of the high limit
210	1	0	DADADA	Very light gray	Display of high limit: Fill pattern color	LimitObjectHighFillColor	Block icons with display of the high limit
211	1	0	B6B6B6	Gray	Display of the low limit: Background color	LimitObjectLowBackColor	Block icons with display of the low limit
212	1	0	DADADA	Very light gray	Display of the low limit: Fill pattern color	LimitObjectLowFillColor	Block icons with display of the low limit
213	1	0	B6B6B6	Gray	Empty rectangle: Background color	EmptyRectBackColor"[1]	Block icons with status bar
214	1	0	DADADA	Very light gray	Empty rectangle: Fill pattern color	EmptyRectFillColor[1]	Block icons with status bar
215	1	0	808080	Dark gray	Empty rectangle: Border color	EmptyRectBorderColor[1]	Block icons with status bar
216	1	0	FFFFFF	White	Process value: Background color	Value1BackColorValue10BackColor	Block icons with analog value
217	1	0	000000	Black	Process value: Font color	Value1FontColorValue10FontColor	Block icons with analog value
218	1	100	C0C0C0	None (transparent)	Setpoint: Background color	Value2BackColor	Block icons with analog value

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Type block icons
219	1	0	0000FF	Pure blue	Setpoint: Font color	Value2FontColor	Block icons with analog value
220	2	100	C0C0C0	None (transparent)	Changed variable: Background color	Value3BackColor	Block icons with analog value
221	2	0	F24F00	Pure orange	Changed variable: Font color	Value3FontColor	Block icons with analog value
222	2	100	C0C0C0	None (transparent)	Read-back value: Background color	Value3BackColor Value4BackColor	Block icons with analog value
223	2	0	008080	Dark cyan	Read-back value: Font color	Value3FontColor Value4FontColor	Block icons with analog value
224	2	100	B6B6B6	None (transparent)	Unit², process value/ setpoint: Background color	UnitBackColor, Unit1BackColor-Unit4BackColor	Block icons with analog value
225	2	0	494949	Very dark gray	Unit, process value/ setpoint: Font color	UnitFontColor, Unit1FontColor-Unit4FontColor	Block icons with analog value
226	2	100	B6B6B6	None (transparent)	Unit², changed variable: Background color	Unit2BackColor	Block icons with analog value
227	2	0	494949	Very dark gray	Unit, changed variable: Font color	Unit2FontColor	Block icons with analog value
228	2	100	B6B6B6	None (transparent)	Unit², read-back value: Background color	Unit1BackColor, Unit2BackColor	Block icons with analog value
229	2	0	494949	Very dark gray	Unit, read-back value: Font color	Unit1FontColor, Unit2FontColor	Block icons with analog value
230	2	0	6D6D6D	Very dark gray	"Not used" state: background color	BackColorOn1	Interlock
231	2	0	919191	Dark gray	"Not used" state: fill pattern color	FillColorOn1	Interlock
232	2	0	0000FF	Pure blue	"Bypassed" state: background color	BackColorOn2	Interlock
233	2	0	00A2E8	Pure blue	"Bypassed" state: fill pattern color	FillColorOn2	Interlock
234	2	0	E4D400	Pure yellow	"Simulation" state: background color	BackColorOn3	Interlock
235	2	0	FFFF00	Pure yellow	"Simulation" state: fill pattern color	FillColorOn3	Interlock
236	2	0	00B500	Bright lime green	"Unlocked" state: background color	BackColorOn4	Interlock
237	2	0	00FF00	Pure lime green	"Unlocked" state: fill pattern color	FillColorOn4	Interlock
238	2	0	E60000	Pure red	"Locked" state: background color	BackColorOn5	Interlock

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Type block icons
239	2	0	FF0000	Pure red	"Locked" state: fill pattern color	FillColorOn5	Interlock
240	3	0	00B500	Bright lime green	"On" state: Background color	BackColorOn, OnBackColor	Family counter / MonDi08/ MonDiL-MonDiS (type 1 and 4) / OpDi03 / ShrdResS/ TimeTrig/ TotalL
241	3	0	00FF00	Pure lime green	"On" state: Fill pattern color	FillColorOn, OnFillColor	
242	3	0	00FF00	Pure lime green	"On" state: Background color	Value2BackColor	OpDi01, OpTrig
243	3	0	000000	Black	"On" state: Font color	Value2FontColor	OpDi01, OpTrig
244	3	0	FFFFFF	White	"On" state: Background color	Value2BackColor	MonDiL-MonDiS (types 2, 3 and 5)
245	3	0	000000	Black	"On" state: Font color	Value2FontColor	MonDiL-MonDiS (types 2, 3 and 5)
246	3	0	FFFFFF	White	"On" state: Font color	FontColorOn	MonDiL-MonDiS (type 4)
247	3	0	B6B6B6	Gray	"Off" state: Background color	BackColorOff, BackColorOn	
248	3	0	DADADA	Very light gray	"Off" state: Fill pattern color	FillColorOff, FillColorOn	
249	3	0	DADADA	Very light gray	"Off" state: Background color	BackColorOff	Family counter, MonDi08, TimeTrig
250	3	0	FFFFFF	White	"Off" state: Fill pattern color	FillColorOff	Family counter, MonDi08, TimeTrig
251	3	0	FFFFFF	White	"Off" state: Background color	Value1BackColor	MonDiLMonDiS (types 2, 3 and 5)
252	3	0	000000	Black	"Off" state: Font color	Value1FontColor	MonDiLMonDiS (types 2, 3 and 5)
253	3	0	000000	Black	"Off" state: Font color	FontColorOff	MonDiLMonDiS (type 4)
254	3	0	B6B6B6	Gray	"Not used" state: Background color	BackColorNA	MonDi08
255	3	0	000000	Black	"No operation" state: Font color	ForeColorValue	OpStations
256	3	0	C0C0C0	Light gray	Display of the selected analog value: Background color	Value2BackColor	SelA16In
257	3	0	6D6D6D	Very dark gray	Display of the selected analog value: Font color	Value2FontColor	SelA16In
258	3	0	494949	Very dark gray	ISO/DIN symbols: Line color	BorderColor	

[1] New property

[2] Note: The pattern in the block icon must be changed to "single color" for the color to take effect.

Color indexes for faceplates

The following color indexes can be created for faceplates.

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
259	3	0	AFAFAF	Gray	View: Background color	BackColor	Views in all faceplates
260	4	0	DEDEDE	Very light gray	View: Fill pattern color	FillColor	Views in all faceplates
261	4	0	62ACAA	Dark cyan	Operating area: Background color	BackColor	Total operating area
262	4	0	AACFCD	Gray cyan	Operating area: Fill pattern color	FillColor	Total operating area
263	4	0	000000	Black	Title or label of operating area: Font color	ForeColor	All static texts in the operating area
264	4	0	C0C0C0	Light gray	Slider operating area: Background color	BackColor, BackColorTop, BackColorBottom, ButtonColor, HighLimitColor, LowLimitColor	Operating area "@PG_APL_OA_Analogwithlimits100.PDL"
265	4	0	FFFFFF	White	Slider operating area: Border background color	BorderBackColor	Operating area "@PG_APL_OA_Analogwithlimits100.PDL"
266	4	0	000000	Black	Slider operating area: Border color	BorderColor	Operating area "@PG_APL_OA_Analogwithlimits100.PDL"
267	4	0	DADADA	Very light gray	Block comment: Fill pattern color	FillColor	Header in all faceplates
268	4	0	808080	Dark gray	Block comment: Border color	BorderColor	Header in all faceplates
269	4	0	000000	Black	Block comment: Font color	FontColor	Header in all faceplates
270	4	0	B6B6B6	Gray	Empty rectangle overview: Background color	BackColor	Header in all faceplates
271	4	0	DADADA	Very light gray	Empty rectangle overview: Fill pattern color	FillColor	Header in all faceplates
272	4	0	808080	Dark gray	Empty rectangle overview: Border color	BorderColor	Header in all faceplates
273	4	0	C0C0C0	Light gray	Group display overview: Background color	BackColor	Header in all faceplates [1]
274	4	0	C0C0C0	Light gray	Toolbar button: Background color	BackColor	Header in all faceplates

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
275	4	0	BABABA	Gray	Toolbar button: Fill pattern color	FillColor	Header in all faceplates
276	4	0	808080	Dark gray	Toolbar button: Border color	BorderColor	Header in all faceplates
277	4	0	000000	Black	Toolbar button: Font color	FontColor	Header in all faceplates [2]
278	4	0	C0C0C0	Light gray	Header button[3]: Background color	BackColor	Header in all faceplates
279	4	0	808080	Dark gray	Toolbar button: Border color	BorderColor 8	Header in all faceplates
280	5	0	000000	Black	Title or label in views: Font color	TextFontColor, FontColor, ForeColor, LeftHandTextFontColor[4]	Views with static text and APL objects with type ID=17-20, 22-24, 27, 29, 64, 120, 130
281	5	0	AFAFAF	Gray	Button in views/operating areas: Background color	BackColor	Views with APL objects with type ID=30-34, 50-52
282	5	0	CECECE	Light gray	Button in views/operating areas: Fill pattern color	FillColor	Views with APL objects with type ID=30-34, 50-52
283	5	0	808080	Dark gray	Button in views/operating areas: Border color	BorderColor	Views with APL objects with type ID=30-34, 50-52
284	5	0	FFFFFF	White	Button in views/operating areas: 3D border color	BorderColorTop	Views with APL objects with type ID=30-34, 50-52
285	5	0	808080	Dark gray	Button in views/operating areas: 3D shadow color	BorderColorBottom	Views with APL objects with type ID=30-34, 50-52
286	5	0	000000	Black	Button in views/operating areas: Font color	ForeColor	Views with APL objects with type ID=30-34, 50-52
287	5	0	DADADA	Very light gray	Analog value enabled in views/operating areas: Background color	EnabledBackColor, BackColor, BackColor_OK	Views with I/O fields (also extended) and APL objects with type ID=20, 22-24, 27, 28-29, 130
288	5	0	FFFFFF	White	Analog value enabled in views/operating areas: Fill pattern color	EnabledFillColor, FillColor, FillColor_OK	Views with I/O fields (also extended) and APL objects with type ID=20, 22-24, 27, 28-29, 130

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
289	5	0	B6B6B6	Gray	Analog value disabled in views/operating areas: Background color	DisabledBackColor, BackColor, BackColor_OK	Views with I/O fields (also extended) and APL objects with type ID=20, 22-24, 27, 28-29, 91[13],92,130
290	5	0	DADADA	Very light gray	Analog value disabled in views/operating areas: Fill pattern color	DisabledFillColor, FillColor, FillColor_OK	Views with I/O fields (also extended) and APL objects with type ID=20, 22-24, 27, 28-29, 91[13],92,130
291	5	0	000000	Black	Analog value or unit in views/operating areas: Font color	ValueFontColor, ForeColor, ForeColor_OK	Views with I/O fields (also extended) and APL objects with type ID=16, 20, 22-24, 27, 28-29, 91, 130
292	5	0	00B500	Bright lime green	Analog process value: Border color	LineColor, BorderColor	Views with I/O fields (also extended) and APL objects with type ID=20, 24
293	5	0	0000FF	Pure blue	Analog setpoint: Border color	LineColor, BorderColor	Views with I/O fields (also extended) and APL objects with type ID=20, 28
294	5	0	D27A00	Bright orange	Changed analog variable: Border color	LineColor, BorderColor	Views with I/O fields (also extended) and APL objects with type ID=20, 24, 28
295	5	0	008582	Dark cyan	Analog read-back value: Border color	LineColor, BorderColor	Views with I/O fields (also extended) and APL objects with type ID=20
296	5	0	808080	Dark gray	Analog parameter/ range value: border color	LineColor, BorderColor	Views with I/O fields (also extended) and APL objects with type ID=20, 22-24, 27, 28, 90, 92, 130

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
297	5	0	000000	Black	Analog limit (no alarm), input value, offset, gradient: border color	LineColor, BorderColor	In "@PG_CntOhSc_Limit.PDL", "@PG_MonAnL_Standard.PDL", "@PG_Ratio_Standard.PDL", "@PG_TotalL_Standard.PDL" APL objects with type ID=20, 24, 28, 29 and HMIStaticText
298	5	0	808080	Dark gray	Empty analog value displays	BorderColor	Views with APL objects with type ID=21, 25, 26
299	5	0	00B500	Bright lime green	Binary value on: Background color	OnBackColor, State1BackColor	Views with APL objects with type ID=17,18[13],120
300	6	0	00FF00	Pure lime green	Binary value on: Fill pattern color	OnFillColor, State1FillColor	Views with APL objects with type ID=17,18[13],120
301	6	0	FFFFFF	White	Binary value on: Font color	OnFontColor, State1FontColor	Views with APL objects with type ID=17,18[13],120
302	6	0	FFFFFF	White	Binary value off: Background color	OffBackColor, State2BackColor	Views with APL objects with type ID=17,18[13],120
303	6	0	DADADA	Very light gray	Binary value off: Fill pattern color	OffFillColor, State2FillColor	Views with APL objects with type ID=17,18[13],120
304	6	0	000000	Black	Binary value off: Font color	OffFontColor, State2FontColor	Views with APL objects with type ID=17,18[13],120
305	6	0	808080	Dark gray	Binary value: Border color	BorderColor	Views with APL objects with type ID=17,18[13],120
306	6	0	B6B6B6	Gray	Binary value preview: Background color	BackColor	Preview, APL objects with type ID=14
307	6	0	DADADA	Very light gray	Binary value preview: Fill pattern color	FillColor	Preview, APL objects with type ID=14
308	6	0	000000	Black	Binary value preview: Font color	FontColor	Preview, APL objects with type ID=14

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
309	6	0	808080	Dark gray	Binary value preview: Border color	BorderColor	Preview, APL objects with type ID=14
310	6	0	00B500	Bright lime green	Status display "On" mode: background color	State1BackColor… State7BackColor	Standard view, APL objects with type ID=11
311	6	0	00FF00	Pure lime green	Status display "On" mode: fill pattern color	State1FillColor… State7FillColor	Standard view, APL objects with type ID=11
312	6	0	FFFFFF	White	Status display "On" mode: font color	State1FontColor… State7FontColor	Standard view, APL objects with type ID=11
313	6	0	FFFFFF	White	Status display "Manual" mode: background color	State1BackColor… State7BackColor	Standard view, APL objects with type ID=11
314	6	0	CECECE	Light gray	Status display "Manual" mode: fill pattern color	State1FillColor… State7FillColor	Standard view, APL objects with type ID=11
315	6	0	000000	Black	Status display "Manual" mode: font color	State1FontColor… State7FontColor	Standard view, APL objects with type ID=11
316	6	0	00FF00	Pure lime green	Status display "Automatic" mode: background color	State1BackColor… State7BackColor	Standard view, APL objects with type ID=11
317	6	0	008500	Dark lime green	Status display "Automatic" mode: fill pattern color	State1FillColor… State7FillColor	Standard view, APL objects with type ID=11
318	6	0	FFFFFF	White	Status display "Automatic" mode: font color	State1FontColor… State7FontColor	Standard view, APL objects with type ID=11
319	6	0	00CECA	Bright cyan	Status display "Local" mode: background color	State1BackColor… State7BackColor	Standard view, APL objects with type ID=11
320	7	0	00FFFA	Pure cyan	Status display "Local" mode: fill pattern color	State1FillColor… State7FillColor	Standard view, APL objects with type ID=11
321	7	0	000000	Black	Status display "Local" mode: font color	State1FontColor… State7FontColor	Standard view, APL objects with type ID=11
322	7	0	0000FF	Pure blue	Status display "Programm MV/SP" mode: background color	State1BackColor… State7BackColor	Standard view, APL objects with type ID=11
323	7	0	98BDFF	Very light blue	Status display "Programm MV/SP" mode: fill pattern color	State1FillColor… State7FillColor	Standard view, APL objects with type ID=11
324	7	0	FFFFFF	White	Status display "Programm MV/SP" mode: font color	State1FontColor… State7FontColor	Standard view, APL objects with type ID=11

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
325	7	0	919191	Dark gray	Status display "Out of service" mode: background color	State1BackColor… State7BackColor	Standard view, APL objects with type ID=11
326	7	0	CECECE	Light gray	Status display "Out of service" mode: fill pattern color	State1FillColor… State7FillColor	Standard view, APL objects with type ID=11
327	7	0	000000	Black	Status display "Out of service" mode: font color	State1FontColor… State7FontColor	Standard view, APL objects with type ID=11
328	7	0	808080	Dark gray	Status display operating mode: Border color	BorderColor	Standard view, APL objects with type ID=11
329	7	0	FFFFFF	White	Status display control command "Stop"/"Close"/"Off"/"Pause"/"Cancel"/"Requested"/"No": background color	State1BackColor… State7BackColor	Mostly standard/ parameter view/ preview, APL objects with type ID=10, 11, 15
330	7	0	DADADA	Very light gray	Status display control command "Stop"/"Close"/"Off"/"Pause"/"Cancel"/"Requested"/"No": fill pattern color	State1FillColor… State7FillColor	Mostly standard/ parameter view/ preview, APL objects with type ID=10, 11, 15
331	7	0	000000	Black	Status display control command "Stop"/"Close"/"Off"/"Pause"/"Cancel"/"Requested"/"No": font color	State1FontColor… State7FontColor	Mostly standard/ parameter view/ preview, APL objects with type ID=10, 11, 15
332	7	0	00B500	Bright lime green	Status display control command "Start"/"Open"/"On"/"Continue"/"Active"/"ready"/"Yes": background color	State1BackColor… State7BackColor	Mostly standard/ parameter view/ preview, APL objects with type ID=10, 11, 15, 19
333	7	0	00FF00	Pure lime green	Status display control command "Start"/"Open"/"On"/"Continue"/"Active"/"ready"/"Yes": fill pattern color	State1FillColor… State7FillColor	Mostly standard/ parameter view/ preview, APL objects with type ID=10, 11, 15, 19
334	7	0	FFFFFF	White	Status display control command "Start"/"Open"/"On"/"Continue"/"Active"/"ready"/"Yes": font color	State1FontColor… State7FontColor	Mostly standard/ parameter view/ preview, APL objects with type ID=10, 11, 15, 19
335	7	0	FF0000	Pure red	Status display control command "Rapid stop": background color	State1BackColor… State7BackColor	Standard view, APL objects with type ID=10, 15

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
336	7	0	FF8296	Very light red	Status display control command "Rapid stop": fill pattern color	State1FillColor… State7FillColor	Standard view, APL objects with type ID=10, 15
337	7	0	000000	Black	Status display control command "Rapid stop": font color	State1FontColor… State7FontColor	Standard view, APL objects with type ID=10, 15
338	7	0	FFFFFF	White	Status display control command "Internal": background color	State1BackColor… State7BackColor	Standard view, APL objects with type ID=11
339	7	0	CECECE	Light gray	Status display control command "Internal": fill pattern color	State1FillColor… State7FillColor	Standard view, APL objects with type ID=11
340	8	0	000000	Black	Status display control command "Internal": font color	State1FontColor… State7FontColor	Standard view, APL objects with type ID=11
341	8	0	0000FF	Pure blue	Status display control command "External": background color	State1BackColor… State7BackColor	Standard view, APL objects with type ID=10, 11, 15
342	8	0	98BDFF	Very light blue	Status display control command "External": fill pattern color	State1FillColor… State7FillColor	Standard view, APL objects with type ID=10, 11, 15
343	8	0	FFFFFF	White	Status display control command "External": font color	State1FontColor… State7FontColor	Standard view, APL objects with type ID=10, 11, 15
344	8	0	808080	Dark gray	Status display control command: Border color	BorderColor	Standard view, APL objects with type ID=10, 11, 15
345	8	0	000000	Black	Status display error ("End position error", "Control error", "Invalid Signal", "Changeover error", "Motor protection", "Torque active", "External error", "CPI invalid", "Changeover error", "Fluttering"): Background color	BackColor, State1BackColor… State7BackColor	Standard view, mostly APL objects with type ID=12
346	8	0	919191	Dark gray	Status display error ("End position error", "Control error", "Invalid Signal", "Changeover error", "Motor protection", "Torque active", "External error", "CPI invalid", "Changeover error", "Fluttering"): Fill pattern color	FillColor, State1FillColor… State7FillColor	Standard view, mostly APL objects with type ID=12

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
347	8	0	FFFF00	Pure yellow	Status display error ("End position error", "Control error", "Invalid Signal", "Changeover error", "Motor protection", "Torque active", "External error", "CPI invalid", "Changeover error", "Fluttering"): Font color	FontColor, State1FontColor… State7FontColor	Standard view, mostly APL objects with type ID=12
348	8	0	919191	Dark gray	Status display AS block state ("Forced", "Request 0/1", "Tracking", "Optimizing", "Output …"[5], "Fuzzy...", "Safety mode", "End", "Taring", "Dribbling", "Paused", "Off"[6], "… Trigger", "Disabled", "SP/MV ramp active"): Background color	BackColor, State1BackColor… State7BackColor	Standard view, mostly APL objects with type ID=12, 13
349	8	0	DADADA	Very light gray	Status display AS block state ("Forced", "Request 0/1", "Tracking", "Optimizing", "Output …"[7], "Fuzzy...", "Safety mode", "End", "Taring", "Dribbling", "Paused", "Off"[8], "… Trigger", "Disabled", "SP/MV ramp active"): Fill pattern color	FillColor, State1FillColor… State7FillColor	Standard view, mostly APL objects with type ID=12, 13
350	8	0	000000	Black	Status display AS block state ("Forced", "Request 0/1", "Tracking", "Optimizing", "Output …"[9], "Fuzzy...", "Safety mode", "End", "Taring", "Dribbling", "Paused", "Off"[10], "… Trigger", "Disabled", "SP/MV ramp active"): Font color	FontColor, State1FontColor… State7FontColor	Standard view, mostly APL objects with type ID=12, 13
351	8	0	FF0000	Pure red	Status display alarm ("Flow", "Underdosed", "Overdosed"): Background color	BackColor, State1BackColor… State7BackColor	Standard view, mostly APL objects with type ID=12, 13
352	8	0	FFFFFF	White	Status display alarm ("Flow", "Underdosed", "Overdosed"): Fill pattern color	FillColor, State1FillColor… State7FillColor	Standard view, mostly APL objects with type ID=12, 13

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
353	8	0	000000	Black	Status display alarm ("Flow", "Underdosed", "Overdosed"): Font color	FontColor, State1FontColor… State7FontColor	Standard view, mostly APL objects with type ID=12, 13
354	8	0	00B500	Dark lime green	Status display enabled ("On"[11],"Enabled","… dosing"): Background color	BackColor, State1BackColor… State7BackColor	Standard view, mostly APL objects with type ID=12, 13, 15
355	8	0	00FF00	Pure lime green	Status display enabled ("On"[12],"Enabled","… dosing"): Fill pattern color	FillColor, State1FillColor… State7FillColor	Standard view, mostly APL objects with type ID=12, 13, 15
356	8	0	FFFFFF	White	Status display enabled ("On"[13],"Enabled","… dosing"): Font color	FontColor, State1FontColor… State7FontColor	Standard view, mostly APL objects with type ID=12, 13, 15
357	8	0	DADADA	Very light gray	Status display disabled ("Off"[14]): Background color	BackColor, State1BackColor… State7BackColor	Standard view, mostly APL objects with type ID=12, 13, 15
358	8	0	FFFFFF	White	Status display disabled ("Off"): Fill pattern color	FillColor, State1FillColor… State7FillColor	Standard view, mostly APL objects with type ID=12, 13, 15
359	8	0	000000	Black	Status display disabled ("Off"): Font color	FontColor, State1FontColor… State7FontColor	Standard view, mostly APL objects with type ID=12, 13, 15
360	9	0	FFFA00	Pure yellow	Status display important AS block information ("Simulation", "Process excitation", "Time delay"): Background color	BackColor, State1BackColor… State7BackColor	Standard/parameter view, mostly APL objects with type ID=12, 13
361	9	0	FFFFFF	White	Status display important AS block information ("Simulation", "Process excitation", "Time delay"): Fill pattern color	FillColor, State1FillColor… State7FillColor	Standard/parameter view, mostly APL objects with type ID=12, 13
362	9	0	000000	Black	Status display important AS block information ("Simulation", "Process excitation", "Time delay"): Font color	FontColor, State1FontColor… State7FontColor	Standard/parameter view, mostly APL objects with type ID=12, 13
363	9	0	0000FF	Pure blue	Status display maintenance ("Maintenance"): Background color	BackColor, State1BackColor… State7BackColor	Standard view, mostly APL objects with type ID=12, 13

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
364	9	0	96BBFF	Very light blue	Status display maintenance ("Maintenance"): Fill pattern color	FillColor, State1FillColor… State7FillColor	Standard view, mostly APL objects with type ID=12, 13
365	9	0	000000	Black	Status display maintenance ("Maintenance"): Font color	FontColor, State1FontColor… State7FontColor	Standard view, mostly APL objects with type ID=12, 13
366	9	0	808080	Dark gray	Status display error, ..., Maintenance: Border color	BorderColor	Standard view / preview, mostly APL objects with type ID=11-13, 15
367	9	0	DADADA	Very light gray	Interlock, display of input values (BOOL): Background color	BackColor	Standard view interlock, APL objects with type ID=100
368	9	0	CECECE	Light gray	Interlock, display of input values (BOOL): Fill pattern color	FillColor	Standard view interlock, APL objects with type ID=100
369	9	0	000000	Black	Interlock, display of input values (BOOL): Font color	FontColor	Standard view interlock, APL objects with type ID=100
370	9	0	808080	Dark gray	Interlock, display of input values (BOOL): Border color	BorderColor	Standard view interlock, APL objects with type ID=100
371	9	0	919191	Dark gray	Interlock, display of input values (BOOL): Background color separator	SeparatorBackColor	Standard view interlock, APL objects with type ID=100
372	9	0	00FF00	Pure lime green	Interlock, status "1" for further processing: Background color	SetBackColor	Standard view interlock, APL objects with type ID=101
373	9	0	00B500	Dark lime green	Interlock, status "1" for further processing: Fill pattern color	SetFillColor	Standard view interlock, APL objects with type ID=101
374	9	0	FFFFFF	White	Interlock, status "0" for further processing: Background color	ResetBackColor	Standard view interlock, APL objects with type ID=101
375	9	0	B6B6B6	Gray	Interlock, status "0" for further processing: Fill pattern color	ResetFillColor	Standard view interlock, APL objects with type ID=101

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
376	9	0	0000FF	Pure blue	Interlock, status "Bypassed" (excluded) for further processing: Background color	BypassBackColor	Standard view interlock, APL objects with type ID=101
377	9	0	00A2E8	Pure blue	Interlock, status "Bypassed" (excluded) for further processing: Fill pattern color	BypassFillColor	Standard view interlock, APL objects with type ID=101
378	9	0	FFFFFF	White	Interlock, status for further processing: Font color	ForeColor	Standard view interlock, APL objects with type ID=101
379	9	0	6D6D6D	Very dark gray	Interlock, status for further processing: Border color	BorderColor	Standard view interlock, APL objects with type ID=101
380	10	0	B6B6B6	Gray	Interlock, "Not used" status of the output signal of the interlock block: Background color	Set5BackColor	Standard view interlock, APL objects with type ID=103
381	10	0	DADADA	Very light gray	Interlock, "Not used" status of the output signal of the interlock block: Fill pattern color	Set5FillColor	Standard view interlock, APL objects with type ID=103
382	10	0	0000FF	Pure blue	Interlock, "Not used" status of the output signal of the interlock block: Background color	Set4BackColor	Standard view interlock, APL objects with type ID=103
383	10	0	00A2E8	Pure blue	Interlock, "Not used" status of the output signal of the interlock block: Fill pattern color	Set4FillColor	Standard view interlock, APL objects with type ID=103
384	10	0	E4D400	Pure yellow	Interlock, "Simulation" status of the output signal of the interlock block: Background color	Set2BackColor	Standard view interlock, APL objects with type ID=103
385	10	0	FFFF00	Pure yellow	Interlock, "Simulation" status of the output signal of the interlock block: Fill pattern color	Set2FillColor	Standard view interlock, APL objects with type ID=103
386	10	0	00B500	Bright lime green	Interlock, "Unlocked" status of the output signal of the interlock block: Background color	Set1BackColor	Standard view interlock, APL objects with type ID=103
387	10	0	00FF00	Pure lime green	Interlock, "Unlocked" status of the output signal of the interlock block: Fill pattern color	Set1FillColor	Standard view interlock, APL objects with type ID=103

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
388	10	0	E60000	Pure red	Interlock, "Locked" status of the output signal of the interlock block: Background color	Set3BackColor	Standard view interlock, APL objects with type ID=103
389	10	0	FF0000	Pure red	Interlock, "Locked" status of the output signal of the interlock block: Fill pattern color	Set3FillColor	Standard view interlock, APL objects with type ID=103
390	10	0	000000	Black	Interlock, status of the output signal of the interlock block: Font color	ForeColor	Standard view interlock, APL objects with type ID=103
391	10	0	919191	Dark gray	Interlock, status of the output signal of the interlock block: Border color	BorderColor	Standard view interlock, APL objects with type ID=103
392	10	0	850082	Dark magenta	Interlock, "First signal" status display: Background color	BackColor	Standard view interlock, APL objects with type ID=102
393	10	0	CD51FF	Light purple	Interlock, "First signal" status display: Fill pattern color	FillColor	Standard view interlock, APL objects with type ID=102
394	10	0	808080	Dark gray	Interlock, "First signal" status display: Border color	BorderColor	Standard view interlock, APL objects with type ID=102
395	10	0	00B500	Bright lime green	Interlock, "Unlocked" status of the block output: Line color	OnStateColor	Standard view interlock, APL objects with type ID=104
396	10	0	FFFFFF	White	Interlock, "Locked" status of the block output: Line color	OffStateColor	Standard view interlock, APL objects with type ID=104
397	10	0	B6B6B6	Gray	Interlock, block rectangle: Background color	BackgroundColor	Standard view interlock, HMIRectangle
398	10	0	DADADA	Very light gray	Interlock, block rectangle: Fill pattern color	FillPatternColor	Standard view interlock, HMIRectangle
399	10	0	919191	Dark gray	Interlock, block rectangle: Border color	BorderColor	Standard view interlock, HMIRectangle
400	11	0	00B500	Bright lime green	Bar chart, analog process value: Background color	ValueColor	Standard/setpoint view, APL objects with type ID=71

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
401	11	0	00FF00	Pure lime green	Bar chart, analog process value: Fill pattern color	ValueFillColor	Standard/setpoint view, APL objects with type ID=71
402	11	0	0000FF	Pure blue	Bar chart, analog setpoint: Background color, foreground color	ValueColor, FontColor	Standard/setpoint view, APL objects with type ID=70, 80, 81
403	11	0	98BDFF	Very light blue	Bar chart, analog setpoint: Fill pattern color	ValueFillColor	Standard/setpoint view, APL objects with type ID=70
404	11	0	D27A00	Bright orange	Bar chart, changed analog variable: Background color, foreground color	ValueColor, FontColor	Standard view, APL objects with type ID=72, 82, 83
405	11	0	FFC848	Light orange	Bar chart, changed analog variable: Fill pattern color	ValueFillColor	Standard view, APL objects with type ID=72
406	11	0	008582	Dark cyan	Bar chart, analog readback value: Background color	ValueColor	Standard view, APL objects with type ID=71, 73
407	11	0	00CECA	Bright cyan	Bar chart, analog readback value: Fill pattern color	ValueFillColor	Standard view, APL objects with type ID=71, 73
408	11	0	008582	Dark cyan	Bar chart, predicted value: Background color	ValueColor	Standard view, APL objects with type ID=75
409	11	0	00CECA	Bright cyan	Bar chart, predicted value: Fill pattern color	ValueFillColor	Standard view, APL objects with type ID=75
410	11	0	000000	Black	Bar chart, message gradient value or limit (without alarm): Background color	ValueColor, AlarmColor, WarningColor	Standard view, APL objects with type ID=74 or 71[15]
411	11	0	6D6D6D	Very dark gray	Bar chart, gradient value: Fill pattern color	ValueFillColor	Standard view, APL objects with type ID=74
412	11	0	808080	Dark gray	Bar chart: Border color	BorderColor	Standard/setpoint view, APL objects with type ID=7075
413	11	0	62ACAA	Mostly unsaturated dark cyan	Selection rectangle operation: Background color	BackgroundColor	All views with operation, HMIRectangle object name "rect_selectionBorder"
414	11	0	B1CFCD	Gray cyan	Selection rectangle operation: Fill pattern color	FillColor	All views with operation, HMIRectangle object name "rect_selectionBorder"

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Types views/ objects
415	11	0	808080	Dark gray	Selection rectangle operation: Border color	BorderColor	All views with operation, HMIRectangle object name "rect_selectionBorder"
416	11	0	808080	Dark gray	Dividing line: Border color	BorderColor	All views with dividing line, HMILine

[1] Note: Also types without messages have an empty group display

[2] Note: Only important for the "Next" button

[3] "Lock"/"Acknowledgement"/"Attach"/"Back to process picture" buttons; "Attach" button in the operating area

[4] In the standard view Ratio

[5] Used in type ID = 17, 18, 120

[6] In the standard view DoseL

[7] In the standard view Ratio

[8] In the standard view DoseL

[9] In the standard view Ratio

[10] In the standard view DoseL

[11] In the standard view OpDi01, OpDi03, display the confirmation of the command

[12] In the standard view OpDi01, OpDi03, display the confirmation of the command

[13] Not used, only in @PCS7ElementsAPL.pdl

[14] In the standard view OpDi01, OpDi03, display the confirmation of the command

[15] In @PG_CntOhSc_Standard.PDL

Color indexes for trends

The following color indexes can be created for trends:

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Type block icons
417	11	0	00CE00	Bright lime green	Actual value	TrendColor1 - TrendColor12	Trend view
418	11	0	0000FF	Pure blue	Setpoint	TrendColor1 - TrendColor12	Trend view
419	11	0	E69100	Pure orange	Changed variable	TrendColor1 - TrendColor12	Trend view
420	12	0	008582	Dark cyan	Read-back value	TrendColor1 - 1TrendColor12	Trend view

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Type block icons
421	12	0	00CE00	Bright lime green	Controller, closed loop	TrendColor8	Trend view controller
422	12	0	000000	Black	Controller, input for control performance index (CPI)	TrendColor7	Trend view controller
423	12	0	000000	Black	Controller, binary message "violated"[1]	TrendColor3, TrendColor4, TrendColor9, TrendColor10	Trend view controller
424	12	0	B100B5	Bright magenta	Controller, control performance index (CPI)	TrendColor1	Trend view ConPerMon
425	12	0	E69100	Pure orange	Ratio, analog input "In"	TrendColor1	Trend view Ratio
426	12	0	00FF00	Pure lime green	Ratio, input for process variable "InPV"	TrendColor2	Trend view Ratio
427	12	0	0000FF	Pure blue	Ratio, process value of the secondary component "SecComPV"	TrendColor3	Trend view Ratio
428	12	0	B100B5	Bright magenta	Ratio, output "Out"	TrendColor4	Trend view Ratio
429	12	0	FF0000	Pure red	Ratio, current ratio "RatioPV"	TrendColor5	Trend view Ratio
430	12	0	6D6D6D	Very dark gray	Ratio, applied ratio (RatioInt or RatioExt) "RatioOut"	TrendColor6	Trend view Ratio
431	12	0	009D00	Dark lime green	Dosing, dosing amount	TrendColor3	Trend view Dosing
432	12	0	FF0000	Pure red	Monitoring, gradient value	TrendColor2	Trend view MonAnL
433	12	0	E69100	Pure orange	Monitoring, output 1	TrendColor1	Trend view MonDi08, MonDiL, MonDiS
434	12	0	00CE00	Bright lime green	Monitoring, output 2	TrendColor2	Trend view MonDi08
435	12	0	0000FF	Pure blue	Monitoring, output 3	TrendColor3	Trend view MonDi08
436	12	0	B100B5	Bright magenta	Monitoring, output 4	TrendColor4	Trend view MonDi08
437	12	0	FF0000	Pure red	Monitoring, output 5	TrendColor5	Trend view MonDi08
438	12	0	494949	Very dark gray	Monitoring, output 6	TrendColor6	Trend view MonDi08
439	12	0	543500	Very dark orange	Monitoring, output 7	TrendColor7	Trend view MonDi08
440	13	0	005400	Very dark lime green	Monitoring, output 8	TrendColor8	Trend view MonDi08
441	13	0	00CE00	Bright lime green	Monitoring, flutter value	TrendColor2	Trend view MonDiL

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Type block icons
442	13	0	E69100	Pure orange	Operation, binary output / binary output 1	TrendColor1	Operation, trend view binary
443	13	0	494949	Very dark gray	Operation, binary output 2	TrendColor2	Operation, trend view binary
444	13	0	0000FF	Pure blue	Operation, binary output 3	TrendColor3	Operation, trend view binary
445	13	0	B100B5	Bright magenta	Operation, binary feedback 1	TrendColor4	Operation, trend view binary
446	13	0	FF0000	Pure red	Operation, binary feedback 2	TrendColor5	Operation, trend view binary
447	13	0	00CE00	Bright lime green	Operation, binary feedback / binary feedback 3	TrendColor2, TrendColor6	Operation, trend view binary
448	13	0	0000FF	Pure blue	Actuators, valve control output	TrendColor1 - TrendColor4, TrendColor6	Trend view Valve
449	13	0	000000	Black	Actuators, valve feedback open	TrendColor1	Trend view Valve
450	13	0	6D6D6D	Very dark gray	Actuators, valve feedback close	TrendColor2	Trend view Valve
451	13	0	005400	Very dark lime green	Actuators, valve feedback	TrendColor1, TrendColor3, TrendColor5, TrendColor7	Trend view Vlv2WayL
452	13	0	E69100	Pure orange	Actuators, open valve output	TrendColor1	Trend view VlvMotL
453	13	0	00CE00	Bright lime green	Actuators, close valve output	TrendColor2	Trend view VlvMotL
454	13	0	0000FF	Pure blue	Actuators, valve feedback open	TrendColor3	Trend view VlvMotL
455	13	0	B100B5	Bright magenta	Actuators, valve feedback close	TrendColor4	Trend view VlvMotL
456	13	0	FF0000	Pure red	Actuators, valve feedback open	TrendColor5	Trend view VlvMotL
457	13	0	005400	Very dark lime green	Actuators, valve feedback close	TrendColor6	Trend view VlvMotL
458	13	0	E69100	Pure orange	Actuators, motor start output	TrendColor1	Trend view MotL/MotS
459	13	0	00CE00	Bright lime green	Actuators, motor feedback	TrendColor2	Trend view MotL/MotS
460	14	0	E69100	Pure orange	Drives, motor start Speed1 / forward	TrendColor1	Trend view MotSpdL/MotRevL/ MotSpdCL
461	14	0	00CE00	Bright lime green	Drives, motor start Speed2 / reverse	TrendColor2	Trend view MotSpdL/MotRevL/ MotSpdCL

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Type block icons
462	14	0	0000FF	Pure blue	Drives, feedback Speed1 / forward	TrendColor3	Trend view MotSpdL/MotRevL
463	14	0	B100B5	Bright magenta	Drives, feedback Speed2 / reverse	TrendColor4	Trend view MotSpdL/MotRevL
464	14	0	00FFFA	Pure cyan	Drives, feedback forward	TrendColor3	Trend view MotSpdCL
465	14	0	B100B5	Bright magenta	Drives, feedback reverse	TrendColor4	Trend view MotSpdCL
466	14	0	E69100	Pure orange	Count: Output	TrendColor1	Trend view Counter
467	14	0	E69100	Pure orange	Count / time: Unit on	TrendColor1	CountOh / TimeTrig
468	14	0	005400	Very dark lime green	Count / time: Count type	TrendColor7	CountOh / TimeTrig
469	14	0	E69100	Pure orange	Count / time: Count type	TrendColor1	Trend view CntOhSc
470	14	0	00CE00	Bright lime green	Count / time: Unit on	TrendColor2	Trend view CntOhSc
471	14	0	005400	Very dark lime green	Count: Service life	TrendColor7	Trend view CntOhSc
472	14	0	6D6D6D	Very dark gray	Count: Count	TrendColor8	Trend view CntOhSc
473	14	0	0000FF	Pure blue	Statusdisplay operation command - Release for Maintenance” (“Yes”): background color	State1BackColor	Mostly In Parameter view APL objects with TypeID=11
474	14	0	FFFFFF	White	Statusdisplay operation command - Release for Maintenance” (“Yes”): filling pattern color	State1FillColor	Mostly In Parameter view APL objects with TypeID=11
475	14	0	FFFFFF	White	Statusdisplay operation command -Release for Maintenance (“Yes”): font color	State1FontColor	Mostly In Parameter view APL objects with TypeID=11
476	14	0	919191	Dark gray	Statusdisplay (“Not Ready”, “None”, “No”): background color	State1BackColor… State7BackColor	Mostly In ShrdResS Standard \Preview view APL objects with TypeID=11,19

Color index	Section	Transparency	HTML code (RGB)	Color name	Description	Property(ies) [attribute name]	Type block icons
477	14	0	DADADA	Very light gray	Statusdisplay (“Not Ready”, “None”,”No”): filling pattern color	State1FillColor… State7FillColor	Mostly In ShrdResS Standard \Preview view APL objects with TypeID=11,19
478	14	0	000000	Black	Statusdisplay (“Not Ready”, “None”,”No”): font color	State1FontColor… State7FontColor	Mostly In ShrdResS Standard \Preview view APL objects with TypeID=11,19

[1] For example, PIDConL "PV_ToleHi#Value", ".MV_HiAct#Value"

[2] ModPreCon=@PG_APL_TrendMPC.PDL; MPC10x10=@PG_APL_TrendMPC_L.PDL

Note

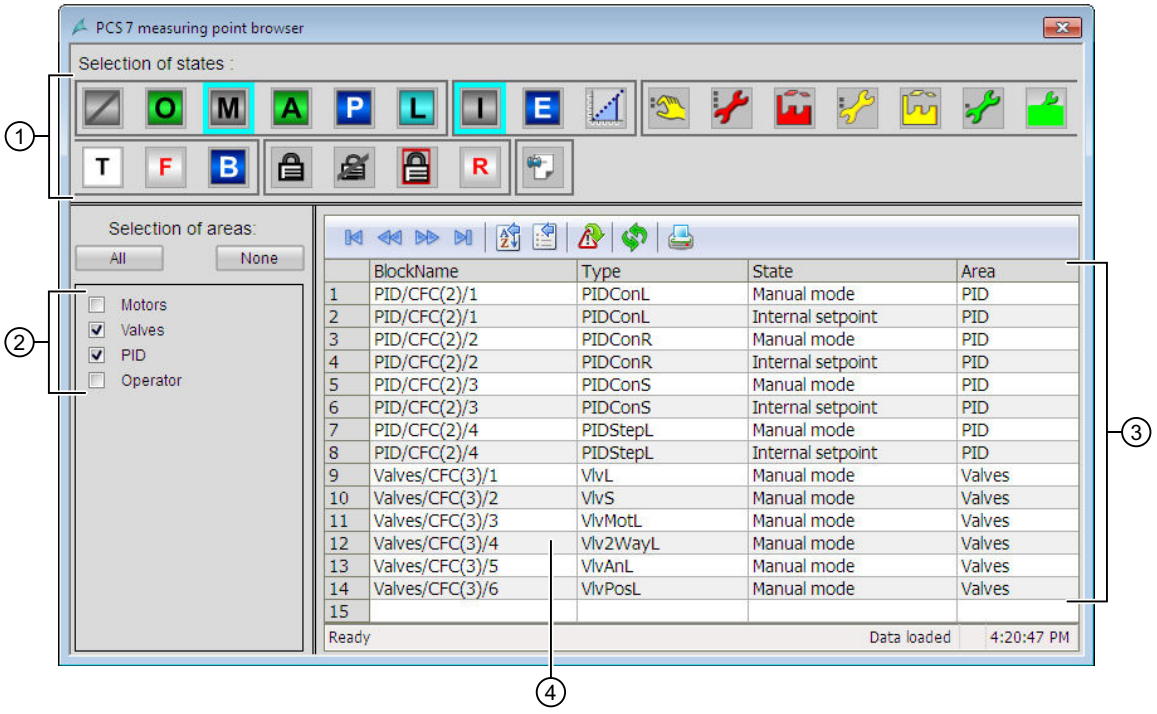
If these colors were changed only via the OS project editor, the default values can be assigned again.

2.4 PCS 7 measuring point browser

2.4.1 Overview of the "PCS 7 measuring point browser" window

Overview

The "PCS 7 measuring point browser" window displays all relevant status information of the APL function blocks.



(1) "Selection of states"

This area displays all possible states of APL function blocks. All state icons are shown in their respective status group frames. The group name is displayed in the tooltip when the mouse pointer is hovered in the area between the state icon and the group frame. You can select/ deselect one or more states from any status group by clicking the icon. The status information of the selected states is displayed in the status area.

The following states are displayed in this area:

Icon	State	Status group
	Out of service	Operating mode
	On mode	
	Manual mode	
	Automatic mode	
	Program mode	
	Local mode	
	Internal setpoint	Setpoint
	External setpoint	
	Ramp active	
	Manipulated value	Signal status
	Bad, device-related	
	Bad, process-related	
	Uncertain, device-related	
	Uncertain, process-related	
	Maintenance request	
	In progress	
	Tracking	Tracking, Forcing and Bypass
	Forcing	
	Bypass	

Icon	State	Status group
	Interlock active	Interlock
	Interlock disabled	
	Rapid stop	
	Reset request	
	Active memo	Memo

(2) "Selection of areas"

This area displays the top folders of the plant hierarchy. You can select/deselect one or more areas. The status information from the selected areas is displayed in the status area.

(3) Status area

This area shows the list of blocks with their corresponding "Type", "State", and "Area" depending on the "Selection of states" and "Selection of areas".

(4) Opening the faceplate

Double-click a row in the status area to open the corresponding faceplate.

Operator control blocks

3

3.1 Comparison of large & small blocks

3.1.1 OpAnL compared to OpAnS

Comparison of the OpAnL and OpAnS blocks

The following tables are intended to help you decide which block to use.

Memory and runtime savings of the small block compared to the large block

You save the following resources for each instance:

- Memory space: ~ 60%
- Runtime: ~ 30%

Block operating modes

	OpAnL	OpAnS
On (Page 71)	X	X
Out of service (Page 71)	X	X

Functions of the blocks

	OpAnL	OpAnS
Setpoint specification - internal/ external (Page 127)	X	X
Setpoint limiting for external set-points (Page 192)	X	
Using setpoint ramp (Page 123)	X	
Gradient limit of the setpoint (Page 124)	X	
Forming and outputting the signal status for technologic blocks (Page 108)	X	X
Selecting a unit of measure (Page 207)	X	X
Simulating signals (Page 58)	X	X
Operator control permissions (Page 248)	X	X

	OpAnL	OpAnS
Display and operator input area for process values and setpoints (Page 203)	X	
Opening additional faceplates (Page 203)	X	X
SIMATIC BATCH functionality (Page 67)	X	X
Generating instance-specific messages (Page 200)	X	
Suppressing messages using the MsgLock parameter (Page 201)	X	
Process value with separate scale range (Page 161)	X	

Configurable functions using the `Feature` parameter

Bit number	`Feature` bit function	OpAnL	OpAnS
0	Set startup characteristics (Page 137)	X	X
1	Reaction to the out of service mode (Page 176)	X	X
16	Process value with separate scale range (Page 161)	X	
22	Update acknowledgment and error status of the message call (Page 159)	X	
24	Enabling local operator authorization (Page 157)	X	X
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)	X	X

See also

Manual and automatic mode for control blocks (Page 72)

3.2 OpAnL - Check and output analog signals (large)

3.2.1 Description of OpAnL

Object name (type + number) and family

Type + number: FB 1865

Family: Operate

Area of application for OpAnL

The block is used for the following applications:

- Checking and transferring analog input values

How it works

The block checks incoming, internal (entered in the faceplate) or external (CFC/SFC) analog signals for their limits at the `SP_Int` or `SP_Ext` input and forwards them to the output `SP`, depending on the setting of the `SP_LiOp` input parameter.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

For a description of the individual parameters, see the section OpAnL I/Os (Page 344).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`
7	`SP_UpRaAct`, `SP_DnRaAct` limits enabled for gradient mode (`SP_RateOn = 1`)

Status bit	Parameter
8	SP_ExtAct.Value
9	SP_LoAct.Value
10	SP_HiAct.Value
11 - 13	Not used
14	SP_RmpOn
15	SP_RmpModTime
16 - 17	Not used
18	Feature Bit 16
19 - 21	Not used
22	1 = SP ramp active
23 - 31	Not used

See also

OpAnL functions (Page 339)

OpAnL messaging (Page 342)

OpAnL block diagram (Page 348)

OpAnL error handling (Page 342)

OpAnL modes (Page 338)

3.2.2 OpAnL modes

OpAnL modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

OpAnL I/Os (Page 344)

OpAnL messaging (Page 342)

OpAnL error handling (Page 342)
Description of OpAnL (Page 337)
OpAnL functions (Page 339)
OpAnL block diagram (Page 348)

3.2.3 OpAnL functions

Functions of OpAnL

The functions for this block are listed below.

Suppressing messages using the MsgLock parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201)

Internal or external setpoint selection

This block provides the standard function Setpoint specification - internal/external (Page 127).

Setpoint limitation

Use the `SP_HiLim` and `SP_LoLim` input parameters to limit the setpoint to maximum and minimum limits. If a limit is violated, the setpoint is limited to the limits you have set. If the limits are infringed, the output parameters `SP_HiAct` and `SP_LoAct` display 1.

Using setpoint ramp

This block provides the standard function Using setpoint ramp (Page 123).

Gradient limit of the setpoint

This block provides the standard function Gradient limit of the setpoint (Page 124).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `SP_Out.ST`
- `PV_In.ST`
- `SP_Ext.ST` (only if `Feature.Bit10` = 1)

The signal status of the `SP_Out` output parameter is always equivalent to the signal status of input parameter `SP_Ext` or `SP_Int`, depending on how the setpoint is specified. If the internal setpoint `SP_Int` is used, the signal status is always output as 16#80.

In case of `Feature.Bit10` = 1, `SP_Ext.ST` influences `ST_Worst` independent of setpoint specification.

If an external setpoint is used, the signal status 16#60 (external simulation) is suppressed because the block acts as a sink with external simulation.

Considering bad quality of automatic commands or external values

If the `Feature` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and the parameter `SP_Ext` has bad signal status (16#00 or 16#28), the block works with the last valid value of `SP_Ext` in the "Automatic" mode.

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Simulating signals

This block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
6	Ramp rate calculation (Page 178)
10	Considering bad quality of automatic commands or external values (Page 185)
12	Gradient limitation with time duration (Page 181)
16	Process value with separate scale range (Page 161)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode

Bit	Function
2	Not used
3	1 = Operator can switch to "Out of service" mode
4 - 9	Not used
10	1 = Operator can switch to internal
11	1 = Operator can switch to external
12	1 = Operator can enable bumpless switchover
13	1 = Operator can change `SP_Int`
14	1 = Operator can enable `SP_RateOn`
15	1 = Operator can change `SP_UpRaLim`
16	1 = Operator can change `SP_DnRaLim`
17	1 = Operator can activate the setpoint ramp (`SP_RmpOn`)
18	1 = Operator can switch between specification of the duration (`SP_RmpTime`) and gradient (`SP_DnRaLim`, `SP_UpRaLim`) for calculating the ramp slope
19	1 = Operator can change the time for the setpoint ramp (`SP_RmpTime`)
20	1 = Operator can change the target setpoint (`SP_RmpTarget`)
21 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Specifying the display area for process and setpoint values as well as operations

The block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Opening additional faceplates

The block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

The block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of OpAnL (Page 337)

OpAnL messaging (Page 342)

OpAnL I/Os (Page 344)

OpAnL block diagram (Page 348)

OpAnL error handling (Page 342)

OpAnL modes (Page 338)

3.2.4 OpAnL error handling

Error handling of OpAnL

Refer to the section Error handling (Page 119) in the basic instructions.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
51	Invalid signal `SP_LiOp` = 1 and `SP_ExtLi` = 1 and `SP_IntLi` = 1

See also

OpAnL block diagram (Page 348)

OpAnL I/Os (Page 344)

OpAnL messaging (Page 342)

OpAnL functions (Page 339)

OpAnL modes (Page 338)

Description of OpAnL (Page 337)

3.2.5 OpAnL messaging

Messaging

The following messages can be generated for this block:

- Process messages

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId	SIG 1	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 2	AS process control message - fault	$$BlockComment$$ External message 2
	SIG 3	AS process control message - fault	$$BlockComment$$ External message 3
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 4
	SIG 5	Reserved	$$BlockComment$$ Reserved
	SIG 6	Reserved	$$BlockComment$$ Reserved
	SIG 7	Reserved	$$BlockComment$$ Reserved
	SIG 8	Reserved	$$BlockComment$$ Reserved

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance **MsgEvId**

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	ExtVa104
5	ExtVa105
6	ExtVa106
7	ExtVa107

The associated values 4 ... 7 are allocated to the parameters ExtVa104 ... ExtVa107 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

OpAnL error handling (Page 342)

OpAnL modes (Page 338)

OpAnL block diagram (Page 348)

3.2.6 OpAnL I/Os

OpAnL I/Os

Input parameters

Parameter	Description	Type	Default
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
EN	1 = Called block will be processed	BOOL	1
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg4	Binary input for freely selectable message 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
Feature	I/O for additional functions (Page 339)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
MsgEvId	Message number (assigned automatically)	DWORD	16#00000000
Occupied	1 = Occupied by batch control	BOOL	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operating permission, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operating permissions (Page 339)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
PV_In	Process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Unit	Unit of measure for process value	INT	1001
PV_OpScale*	OS display range for process value	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SP_DnRaLim	Limit (low) for the gradient of the setpoint [SP_Unit/s]	REAL	100.0
SP_Ext	external setpoint - (to interconnection)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtLi	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOp*	1 = Select external setpoint (via operator)	BOOL	0
SP_HiLim	Limit (high) of setpoint	REAL	100.0
SP_LoLim	Limit (low) of setpoint	REAL	0.0
SP_Int*	Internal setpoint for operation	REAL	0.0
SP_IntLi	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_IntOp*	1 = Select internal setpoint (via operator)	BOOL	1

Parameter	Description	Type	Default
SP_LiOp	Select internal/external setpoint source: 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_OpScale	OS display range for setpoint	STRUCT • High: REAL • Low: REAL	- 100.0 0.0
SP_RateOn	1 = Activate limitation of setpoint gradients	BOOL	0
SP_RmpModTime	1 = Use time (SP_RmpTime) for setpoint ramp 0 = Use gradient	BOOL	0
SP_RmpOn*	1 = Activate setpoint ramp to target setpoint SP_RmpTarget	BOOL	0
SP_RmpTarget*	Target setpoint for setpoint ramp	REAL	0.0
SP_RmpTime*	Time for setpoint ramp [s] from current SP up to SP_RmpTarget	REAL	0.0
SP_TrkExt	1 = Bumpless switchover from external to internal setpoint active	BOOL	0
SP_Unit	Unit of measure for setpoint	INT	1001
SP_UpRaLim	Gradient limit (high) for the setpoint [SP_Unit/s]	REAL	100.0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
StepNo	Batch step number	DWORD	16#00000000
TimeFactor	Time unit: 0 = Seconds 1 = Minutes 2 = Hours	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see OpAnL error handling (Page 342)	INT	-1
MsgAckn	Message acknowledgment status (output ACK_STATE of ALARM_8P)	WORD	16#0000
MsgErr	1 = Message error (output ERROR of ALARM_8P)	BOOL	0
MsgStat	Message status (output STATUS of ALARM_8P)	WORD	16#0000
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature Bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
SP_Out	Setpoint used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_DnRaAct	1 = Negative gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtAct	1 = External setpoint is active 0 = Internal setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_HiAct	High limit for SP_Ext or SP_Int reached or exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
SP_LoAct	Low limit for SP_Ext or SP_Int reached or undershot	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_RateTarget	Target setpoint for the gradient limitation	REAL	0.0
SP_UpRaAct	1 = Positive gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_RemRT	Remaining ramp time of the setpoint	REAL	0.0
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word (Page 337)	DWORD	16#00000000

See also

OpAnL messaging (Page 342)

OpAnL block diagram (Page 348)

OpAnL modes (Page 338)

3.2.7 OpAnL block diagram

OpAnL block diagram

A block diagram is not provided for this block.

See also

OpAnL I/Os (Page 344)

OpAnL error handling (Page 342)

OpAnL functions (Page 339)

Description of OpAnL (Page 337)

OpAnL modes (Page 338)

OpAnL messaging (Page 342)

3.2.8 Operator control and monitoring

3.2.8.1 OpAnL views

Views of the OpAnL block

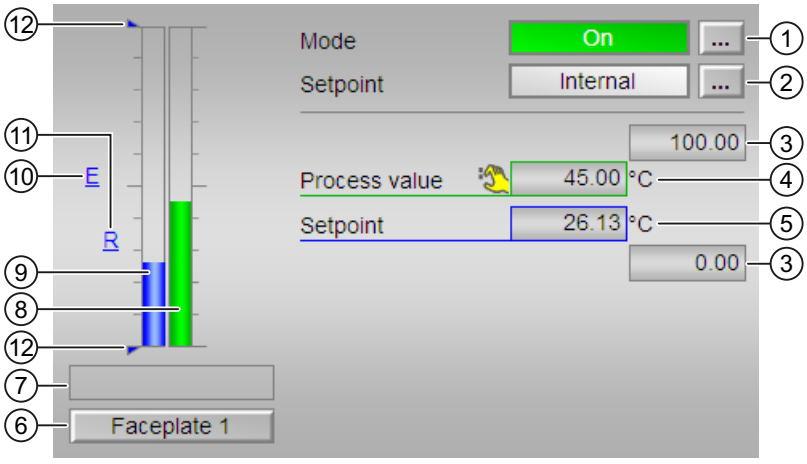
The block OpAnL provides the following views:

- OpAnL standard view (Page 349)
- Alarm view (Page 296)
- Trend view (Page 299)
- Ramp view (Page 294)
- OpAnL parameter view (Page 352)
- OpAnL preview (Page 354)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for OpAnL (Page 355)

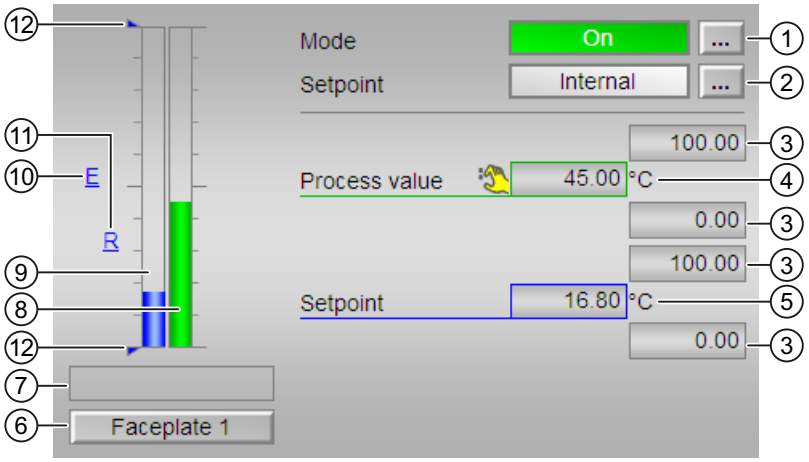
Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

3.2.8.2 OpAnL standard view

OpAnL standard view



The view is switched depending on the Feature Bit 16 = 1 (process value with separate scale range)



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to chapter Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Display and switch the setpoint specification

This area shows how to specify the setpoint. The setpoint can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) chapter for information on switching the setpoint specification.

You can find additional information on this in chapter Setpoint specification - internal/external (Page 127).

(3) High and low scale range for the process value and setpoint

These values provide information on the display range for the bar graph of the process value. The scale range is defined in the Engineering System.

(4) Display of the process value including signal status

This area shows the current process value with the corresponding signal status.

(5) Display and change the setpoint including signal status

This area shows the current setpoint with the corresponding signal status.

Refer to the Changing values (Page 253) chapter for information on changing the setpoint. The setpoint specification also needs to be set to "Internal" for this block.

(6) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the Engineering System (ES). The visibility of this navigation button depends on the configuration in the Engineering System (ES).

Refer also to chapter Opening additional faceplates (Page 203) for more on this.

(7) Display area for block states

This area provides additional information on the operating state of the block:

- "SP ramp active"

(8) Bar graph for the process value

This area shows the current process value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(9) Bar graph for the setpoint

This area shows the current setpoint in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(10) Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

(11) Display for the target setpoint of the setpoint ramp

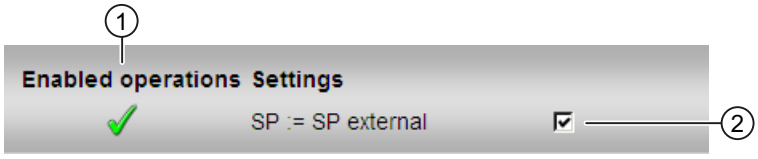
This display [R] shows the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

(12) Displaying the limits

These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the Engineering System (ES).

3.2.8.3 OpAnL parameter view

Parameter view of OpAnL



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

(2) Settings

You can select the following functions in this area:

- "SP := SP external": ☑ Bumpless switchover of setpoint from external to internal. The internal setpoint is tracked to the external one.

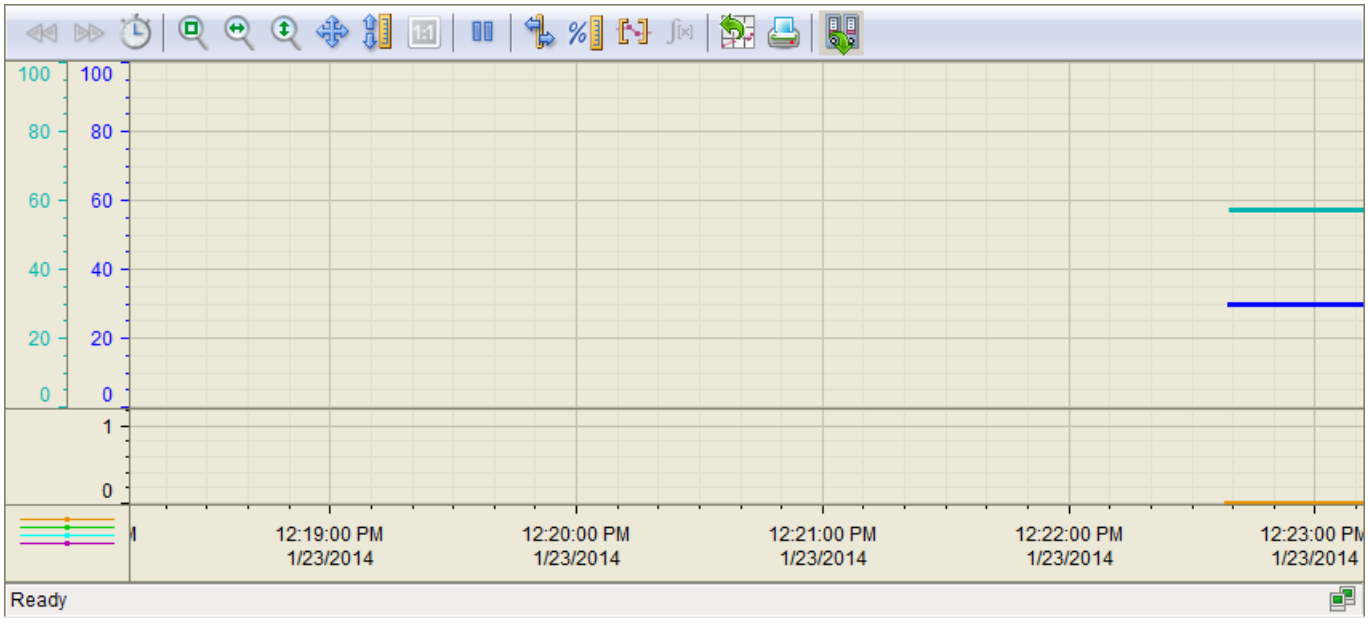
3.2.8.4 OpAnL trend view

OpAnL trend view

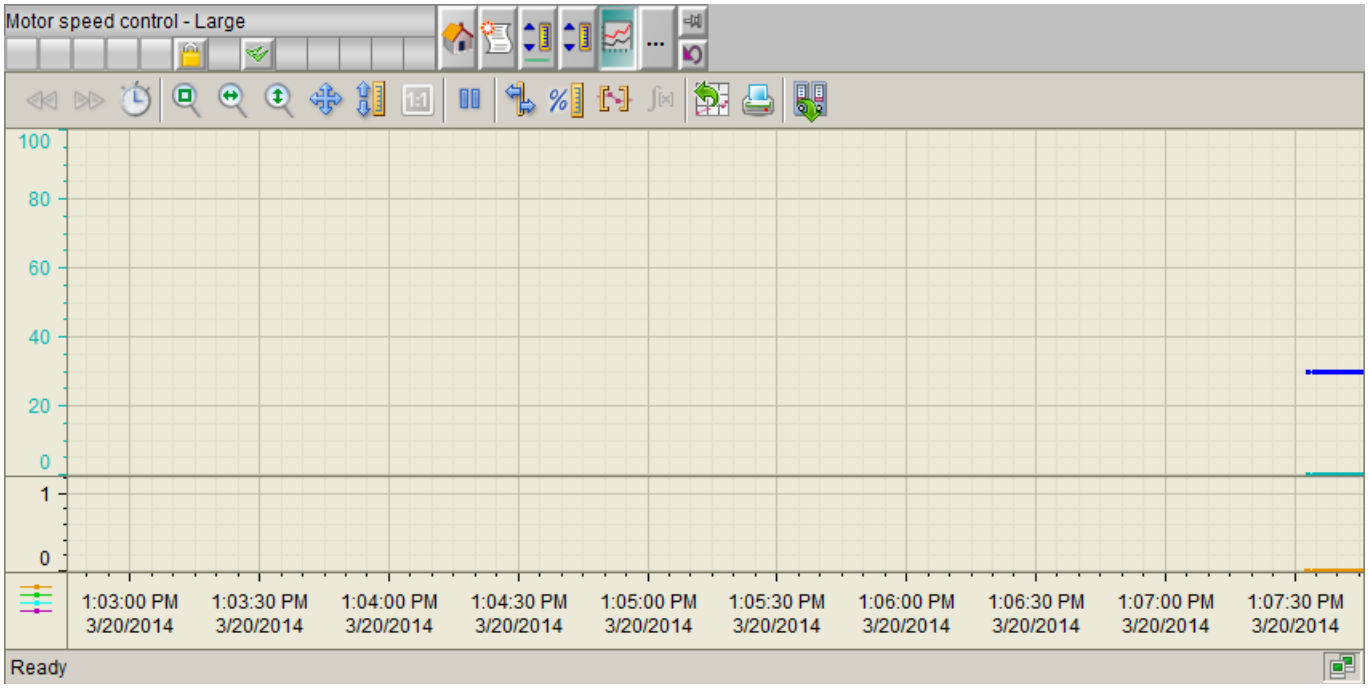
You can find general information in section Trend view (Page 299).

Depending on feature bit 16 (process value with separate scale range) either one or two value axes are shown in the trend view.

- Process value with separate scale range



- Process value without separate scale range

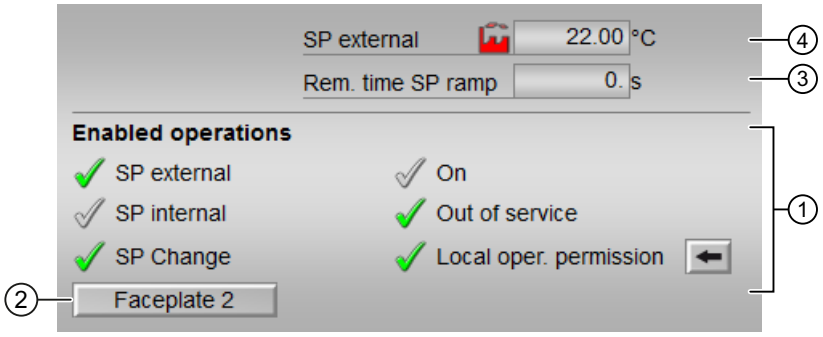


Note

This function only has an effect when `SP_Out` is configured in "TrendConfiguration1" and `PV_In` is configured in "TrendConfiguration2" in the block icon (default).

3.2.8.5 OpAnL preview

Preview of OpAnL



(1) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "SP external": You can feedforward the external setpoint.
- "SP internal": You can feedforward the internal setpoint.
- "Change SP": You can change the setpoint.
- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248).

(2) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

See also the section Opening additional faceplates (Page 203) for more on this.

(3) Remaining time to reach SP ramp

- "Rem. time SP ramp" : Remaining time to reach the ramp target value.

(4) SP external

Currently applicable external setpoint with the corresponding signal status.

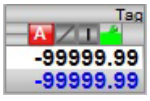
3.2.8.6 Block icon for OpAnL

Block icons for OpAnL

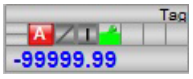
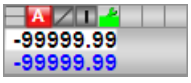
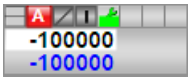
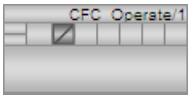
A variety of block icons are available with the following functions:

- Process tag type
- Operating modes
- Internal and external setpoint specification
- Signal status, release for maintenance
- Memo display
- Process value (black, with and without decimal places)
- Setpoint (blue, with and without decimal places)

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	
	4	

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

3.3 OpAnS - Check and output analog signals (small)

3.3.1 Description of OpAnS

Object name (type + number) and family

Type + number: FB 1915

Family: Operate

Area of application for OpAnS

The block is used for the following applications:

- Checking and transferring analog input values

How it works

The block forwards incoming, internal (entered in the faceplate) or external (CFC/SFC) analog signals at the `SP_Int` or `SP_Ext` input to the `SP` output, depending on the setting for the `SP_LiOp` input parameter.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status1` parameter

For a description of the individual parameters, see the chapter OpAnS I/Os (Page 362).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`
7	Not used

Status bit	Parameter
8	`SP_ExtAct.Value`
9 - 31	Not used

See also

OpAnS functions (Page 359)

OpAnS messaging (Page 362)

OpAnS block diagram (Page 365)

OpAnS error handling (Page 361)

OpAnS modes (Page 358)

3.3.2 OpAnS modes

OpAnS operating modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the chapter On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the chapter Out of service (Page 71).

See also

OpAnS I/Os (Page 362)

OpAnS messaging (Page 362)

OpAnS error handling (Page 361)

Description of OpAnS (Page 357)

OpAnS functions (Page 359)

OpAnS block diagram (Page 365)

3.3.3 OpAnS functions

Functions of OpAnS

The functions for this block are listed below.

Internal or external setpoint selection

This block provides the standard function Setpoint specification - internal/external (Page 127).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `SP_Out.ST`
- `PV_In.ST`
- `SP_Ext.ST` (only if `Feature.Bit10` = 1)

The signal status of the `SP_Out` output parameter is always equivalent to the signal status of input parameter `SP_Ext` or `SP_Int`, depending on how the setpoint is specified. If the internal setpoint `SP_Int` is used, the signal status is always output as 16#80.

In case of `Feature.Bit10` = 1, `SP_Ext.ST` influences `ST_Worst` independent of setpoint specification.

If an external setpoint is used, the signal status 16#60 (external simulation) is suppressed because the block acts as a sink with external simulation.

Considering bad quality of automatic commands or external values

If the `Feature` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and the parameter `SP_Ext` has bad signal status (16#00 or 16#28), the block works with the last valid value of `SP_Ext` in the "Automatic" mode.

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Simulating signals

This block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
10	Considering bad quality of automatic commands or external values (Page 185)
24	Enabling local operator authorization (Page 157)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4 - 9	Not used
10	1 = Operator can switch to internal
11	1 = Operator can switch to external
12	1 = Operator can enable bumpless switchover
13	1 = Operator can change `SP_Int`
14 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Specifying the display area for process and setpoint values as well as operations

The block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Opening additional faceplates

The block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

The block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of OpAnS (Page 357)

OpAnS messaging (Page 362)

OpAnS I/Os (Page 362)

OpAnS block diagram (Page 365)

OpAnS error handling (Page 361)

OpAnS modes (Page 358)

Using setpoint ramp (Page 123)

Gradient limit of the setpoint (Page 124)

Update acknowledgment and error status of the message call (Page 159)

3.3.4 OpAnS error handling

Error handling of OpAnS

Refer to the chapter Error handling (Page 119) in the basic instructions.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
51	Invalid signal `SP_LiOp` = 1 and `SP_ExtLi` = 1 and `SP_IntLi` = 1

See also

OpAnS block diagram (Page 365)

OpAnS I/Os (Page 362)

OpAnS messaging (Page 362)

OpAnS functions (Page 359)

OpAnS modes (Page 358)

Description of OpAnS (Page 357)

3.3.5 OpAnS messaging

Messaging

This block does not offer messaging.

See also

OpAnS error handling (Page 361)

OpAnS modes (Page 358)

OpAnS block diagram (Page 365)

3.3.6 OpAnS I/Os

I/Os of OpAnS

Input parameters

Parameter	Description	Type	Default
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 359)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Occupied	1 = Occupied by batch control	BOOL	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0

Parameter	Description	Type	Default
OpSt_In	Input parameter for local operating permission, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator control permissions (Page 359)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
PV_In	Process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Unit	Unit of measure for process value	INT	1001
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SP_Ext	external setpoint - (to interconnection)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtLi	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOp*	1 = Select external setpoint (via operator)	BOOL	0
SP_Int*	Internal setpoint for operation	REAL	0.0
SP_IntLi	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_IntOp*	1 = Select internal setpoint (via operator)	BOOL	1
SP_LiOp	Select internal/external setpoint source: 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_OpScale	OS display range for setpoint	STRUCT • High: REAL • Low: REAL	- 100.0 0.0
SP_TrkExt	1 = Bumpless switchover from external to internal setpoint active	BOOL	0
SP_Unit	Unit of measure for setpoint	INT	1001

Parameter	Description	Type	Default
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
StepNo	Batch step number	DWORD	16#00000000
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see OpAnS error handling (Page 361)	INT	-1
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
SP_Out	Setpoint used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word (Page 357)	DWORD	16#00000000

See also

OpAnS messaging (Page 362)

OpAnS block diagram (Page 365)

OpAnS modes (Page 358)

3.3.7 OpAnS block diagram

OpAnS block diagram

A block diagram is not provided for this block.

See also

OpAnS I/Os (Page 362)

OpAnS error handling (Page 361)

OpAnS functions (Page 359)

Description of OpAnS (Page 357)

OpAnS modes (Page 358)

OpAnS messaging (Page 362)

3.3.8 Operator control and monitoring

3.3.8.1 OpAnS views

Views of the OpAnS block

The block OpAnS provides the following views:

- OpAnS standard view (Page 366)
- Trend view (Page 299)
- OpAnS parameter view (Page 367)
- OpAnS preview (Page 368)
- Memo view (Page 298)
- Batch view (Page 296)
- OpAnS block icon (Page 369)

Refer to the chapters Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

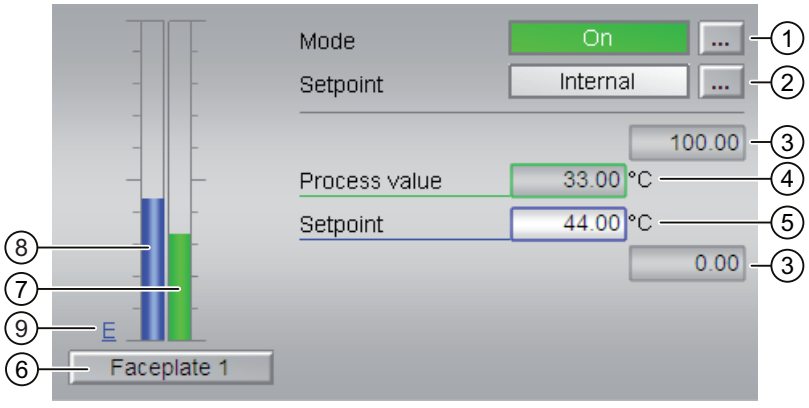
See also

Ramp view (Page 294)

Alarm view (Page 296)

3.3.8.2 OpAnS standard view

OpAnS standard view



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to chapter Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Display and switch the setpoint specification

This area shows how to specify the setpoint. The setpoint can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) chapter for information on switching the setpoint specification.

You can find additional information on this in chapter Setpoint specification - internal/external (Page 127).

(3) High and low scale range for the process value

These values provide information on the display range for the bar graph of the process value. The scale range is defined in the Engineering System.

(4) Display of the process value including signal status

This area shows the current process value with the corresponding signal status.

(5) Display and change the setpoint including signal status

This area shows the current setpoint with the corresponding signal status.

Refer to the Changing values (Page 253) chapter for information on changing the setpoint. The setpoint specification also needs to be set to "Internal" for this block.

(6) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the Engineering System (ES). The visibility of this navigation button depends on the configuration in the Engineering System (ES).

Refer also to chapter Opening additional faceplates (Page 203) for more on this.

(7) Bar graph for the process value

This area shows the current process value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(8) Bar graph for the setpoint

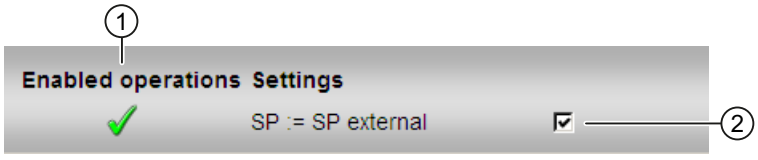
This area shows the current setpoint in the form of a bar graph. The visible area in the bar graph depends on the configuration in the Engineering System (ES).

(9) Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

3.3.8.3 OpAnS parameter view

Parameter view of OpAnS



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

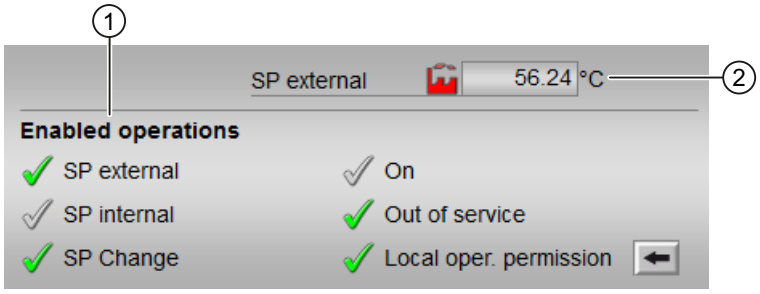
(2) Settings

You can select the following functions in this area:

- "SP := SP external": ☑ Bumpless switchover of setpoint from external to internal. The internal setpoint is tracked to the external one.

3.3.8.4 OpAnS preview

Preview of OpAnS



(1) Enabled operations

This area shows all operations for which special operating permissions are assigned. They depend on the configuration in the Engineering System (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter at all due to the configured AS operating permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "SP external": You can feedforward the external setpoint.
- "SP internal": You can feedforward the internal setpoint.
- "Change SP": You can change the setpoint.
- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in chapter Operator control permissions (Page 248).

(2) SP external

Currently applicable external setpoint with the corresponding signal status.

See also

Opening additional faceplates (Page 203)

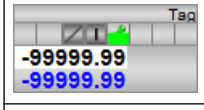
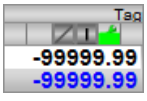
3.3.8.5 OpAnS block icon

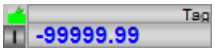
Block icons for OpAnS

A variety of block icons are available with the following functions:

- Process tag type
- Operating modes
- Internal and external setpoint specification
- Signal status, release for maintenance
- Memo display
- Process value (black, with and without decimal places)
- Setpoint (blue, with and without decimal places)

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	

Icons	Selection of the block icon in CFC	Special features
	3	
	4	
	5	Block icon in the full display
	6	Block icon in the full display

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

3.4 OpDi01 - Manipulating a digital value (2 pushbuttons)

3.4.1 Description of OpDi01

Object name (type + number) and family

Type + number: FB 1866

Family: Operate

Area of application for OpDi01

The block is used for the following applications:

- Manipulating a digital value

How it works

A digital value is manipulated by interconnection or via the faceplate.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

If `Feature.Bit0` = 0, the output parameter `Out` will reset to 0.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section OpDi01 I/Os (Page 376).

Status bit	Parameter
0 - 2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`
7	`Out.Value`
8	`LiOp`
9	`FbkIn.Value`
10 - 13	Not used
14	1 = Invalid signal status
15	Not used

Status bit	Parameter
16	0: Open padlock in the block icon 1: Closed padlock in the block icon
17	Hidden bypass signal in interlock
18	`Feature2` bit 2: Separate bypass signal
19 - 22	Not used
23	"Interlock" button is enabled
24 - 25	Not used
26	Bypass information from previous function block
27 - 31	Not used

See also

OpDi01 functions (Page 373)

OpDi01 messaging (Page 376)

OpDi01 block diagram (Page 378)

OpDi01 error handling (Page 375)

OpDi01 modes (Page 372)

3.4.2 OpDi01 modes

OpDi01 operating modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

OpDi01 block diagram (Page 378)

OpDi01 I/Os (Page 376)

OpDi01 messaging (Page 376)

OpDi01 error handling (Page 375)
OpDi01 functions (Page 373)
Description of OpDi01 (Page 371)

3.4.3 OpDi01 functions

Functions of OpDi01

The functions for this block are listed below.

Internal or external digital value

Use the `LiOp` input parameter to define whether the digital value is set (0 - 1, parameter `SetOp` or `SetLi`) or is reset (1 - 0, parameter `RstOp` or `RstLi`) via the faceplate or an interconnection.

`LiOp` = 0: Specification of digital value via faceplate (`SetOp` or `RstOp`)

`LiOp` = 1: Specification of digital value via interconnection (`SetLi` or `RstLi`)

Interlocks

Use the `Intl_En` = 1 and `Intlock.ST` ≠ 16#FF input parameters to activate the interlock function on this block.

An active interlock condition brings the block to the neutral position (`Intlock.Value` = 0 or `Intlock.ST` = 16#00 input). Output parameter `Out` is set to 0. When the interlocking condition no longer applies, the digital value currently valid is output again.

Input parameter for feedback value

This block has a `FbkIn` input parameter for displaying a feedback value in the faceplate.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Changing labels on buttons and text

This block provides the standard function Labeling of buttons and text (Page 205).

Instance-specific text can be configured for the following parameters:

- `Out`
- `SetOp`
- `RstOp`
- `FbkIn`

Forming the signal status for blocks

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `FbkIn.ST`
- `SetLi.ST` (only if `Feature.Bit10` = 1)
- `RstLi.ST` (only if `Feature.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position (`Out` = 0) in the "Automatic" mode:

- `SetLi.ST`
- `RstLi.ST`

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can set the digital value
5	1 = Operator can reset the digital value
6 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
5	Evaluation of the signal status of the interlock signals (Page 141)

Bit	Function
10	Considering bad quality of automatic commands or external values (Page 185)
24	Enabling local operator authorization (Page 157)

See also

Description of OpDi01 (Page 371)

OpDi01 messaging (Page 376)

OpDi01 I/Os (Page 376)

OpDi01 block diagram (Page 378)

OpDi01 error handling (Page 375)

OpDi01 modes (Page 372)

3.4.4 OpDi01 error handling

Error handling of OpDi01

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed
0	There is no error.
51	`SetLi` = 1 and `RstLi` = 1 and `LiOp` = 1

See also

OpDi01 block diagram (Page 378)

OpDi01 I/Os (Page 376)

OpDi01 messaging (Page 376)

OpDi01 functions (Page 373)

OpDi01 modes (Page 372)

Description of OpDi01 (Page 371)

3.4.5 OpDi01 messaging

Messaging

This block does not offer messaging.

See also

Description of OpDi01 (Page 371)
OpDi01 functions (Page 373)
OpDi01 I/Os (Page 376)
OpDi01 block diagram (Page 378)
OpDi01 error handling (Page 375)
OpDi01 modes (Page 372)

3.4.6 OpDi01 I/Os

I/Os of OpDi01

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`FbkIn`	Input for feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`Feature`	I/O for additional functions (Page 373)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`Intlock`	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
`Intl_En`	1 = Interlock without reset (interlock, parameter `Intlock`) can be used	BOOL	1
`LiOp`	Switchover of operating mode between: 1 = Interconnection 0 = Operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`OnOp*`	1 = "On" mode via operator	BOOL	0

Parameter	Description	Type	Default
OosLi	Edge transition (0-1) = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operating permission, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator control permissions (Page 373)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RstLi	Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	Reset by the operator	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
SetLi	Connected digital input	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SetOp*	Digital input for operator	BOOL	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see OpDi01 error handling (Page 375)	INT	-1
LockAct	1 = Interlock (Intlock) is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
Out	Digital output value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status	Status word (Page 371)	DWORD	16#00000000

See also

OpDi01 messaging (Page 376)

OpDi01 block diagram (Page 378)

OpDi01 modes (Page 372)

3.4.7 OpDi01 block diagram

OpDi01 block diagram

A block diagram is not provided for this block.

See also

OpDi01 I/Os (Page 376)

OpDi01 messaging (Page 376)

OpDi01 error handling (Page 375)

OpDi01 functions (Page 373)

OpDi01 modes (Page 372)

Description of OpDi01 (Page 371)

3.4.8 Operator control and monitoring

3.4.8.1 OpDi01 views

Views of the OpDi01 block

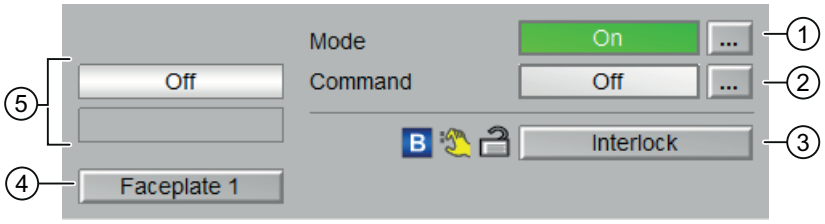
The block OpDi01 provides the following views:

- OpDi01 standard view (Page 379)
- Trend view (Page 299)
- OpDi01 preview (Page 381)
- Memo view (Page 298)
- Block icon for OpDi01 (Page 382)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

3.4.8.2 OpDi01 standard view

OpDi01 standard view



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

You can find additional information on this in section Switching operating states and operating modes (Page 251).

(2) Displaying and switching the command

This area shows you the current selection. You can output a continuous signal as follows:

- "On": Continuous signal is output
- "Off"

You can find additional information on this in section Switching operating states and operating modes (Page 251).

You can rename the display text as you please, as described in the section Labeling of buttons and text (Page 205).

Do this with the following parameters:

- Text for "Command": Parameter `SetOp#string_1`
- Text for "On/Off": Parameter `Out#string_0` / `Out#string_1`

(3) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is interconnected.

You can use this button to control the interlock functions of the block. You can find additional information on this in section OpDi01 functions (Page 373).

- Bypass information (see Forming the group status for interlock information (Page 104)):

(4) Button for switching to the standard view of any faceplate

Use this button for the standard view of a block configured in the engineering system. The visibility of this button depends on the configuration in the engineering system (ES).

You can find additional information on this in section Opening additional faceplates (Page 203).

(5) Displaying the feedback of the command

This area shows you the currently valid command. The following commands can be shown here:

- "On"
- "Off"
- "Invalid signal"

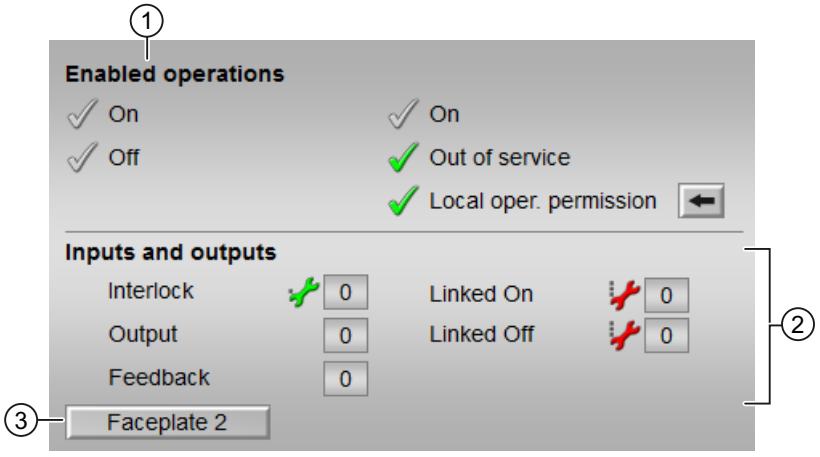
You can rename the display text as you please, as described in the section Labeling of buttons and text (Page 205).

Do this with the following parameter:

- Text for "On/Off": Parameter `FbkIn#string_0` / `FbkIn#string_1`

3.4.8.3 OpDi01 preview

Preview of OpDi01



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "On": You can set the digital value (0 - 1 edge).
  You can rename the display text as you please, as described in the section Labeling of buttons and text (Page 205).
  Do this with the following parameters:
  - Text for "On": Parameter `.Out#Value#string_1`
- "Off": You can set the digital value (1 - 0 edge).
  You can rename the display text as you please, as described in the section Labeling of buttons and text (Page 205).
  Do this with the following parameters:
  - Text for "Off": Parameter `.Out#Value#string_0`
- "On": You can switch to "On" operating mode.

- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

(2) Display of current inputs and outputs

This area shows the most important parameters for this block with the current selection:

- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Output": 1 = Digital output value set
- "Feedback": 1 = Feedback set
- "Linked On": 1 = Linked input on
- "Linked Off": 1 = Linked input off

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the Engineering System (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

Refer also to the Opening additional faceplates (Page 203) section for more on this.

3.4.8.4 Block icon for OpDi01

Block icons for OpDi01

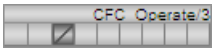
A variety of block icons are available with the following functions:

- Process tag type
- Operating modes
- Signal status, release for maintenance
- Bypass
- Interlocks
- Output signal
- Memo display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

3.5 OpDi03 - Manipulating a digital value (3 pushbuttons)

3.5.1 Description of OpDi03

Object name (type + number) and family

Type + number: FB 1867

Family: Operate

Area of application for OpDi03

The block is used for the following applications:

- Manipulating a digital value (3 pushbuttons)

How it works

A digital value is manipulated at three possible outputs by interconnection or via the faceplate.

If two or three input parameters are set for an interconnection (parameter `SetLix`), the input parameter with the highest index will be set to the corresponding output parameter. For example, if the `SetLi1` and `SetLi2` input parameters are set (= 1), then `Out2` will be set (= 1).

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

If `Feature.Bit0` = 0, the output parameters `Out1` to `Out3` will reset to 0.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section OpDi03 I/Os (Page 390).

Status bit	Parameter
0 - 2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`
7	`Out1.Value`
8	`Out2.Value`
9	`Out3.Value`

Status bit	Parameter
10	`LiOp.Value`
11	`Fbk1In.Value`
12	`Fbk2In.Value`
13	`Fbk3In.Value`
14	1 = Invalid signal status
15	Not used
16	0: Open padlock in the block icon 1: Closed padlock in the block icon
17	Hidden bypass signal in interlock
18	`Feature2` bit 2: Separate bypass signal
19 - 22	Not used
23	"Interlock" button is enabled
24 - 25	Not used
26	Bypass information from previous function block
27 - 31	Not used

See also

OpDi03 functions (Page 386)
OpDi03 messaging (Page 389)
OpDi03 block diagram (Page 393)
OpDi03 error handling (Page 389)
OpDi03 modes (Page 385)

3.5.2 OpDi03 modes

OpDi03 operating modes

The block can be operated using the following modes

- On (Page 71)
- Out of service (Page 71)

"On"

You can find general information about the "On" mode in the On (Page 71) section.

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

OpDi03 block diagram (Page 393)
OpDi03 I/Os (Page 390)
OpDi03 messaging (Page 389)
OpDi03 error handling (Page 389)
OpDi03 functions (Page 386)
Description of OpDi03 (Page 384)

3.5.3 OpDi03 functions

Functions of OpDi03

The functions for this block are listed below.

Internal or external digital value

Use the `LiOp` input parameter to define whether it is through the faceplate or an interconnection using a 1 out of 3 that the selection of the digital value is set (0 - 1) or reset (1 - 0, input parameter `RstOut`).

- `LiOp` = 0: Specification of digital value via faceplate. One of the input parameters `SetOp1`, `SetOp2` or `SetOp3` is now routed to the relevant output `Out1`, `Out2` or `Out3`. For example, if `SetOp2` = 1 , then `Out2` = 1.
- `LiOp` = 1: Specification of digital value via interconnection. One of the input parameters `SetLi1`, `SetLi2` or `SetLi3` is now routed to the relevant output `Out1`, `Out2` or `Out3`. For example, if `SetLi2` = 1 , then `Out2` = 1.

The reset (1 - 0) is always undertaken using the `RstOut` input parameter.

Interlocks

Use the `Intl_En` = 1 and `Intlock.ST` ≠ 16#FF input parameters to activate the interlock function on this block.

An active interlock condition brings the block to the neutral position (`Intlock.Value` = 0 or `Intlock.ST` = 16#00 input). Output parameter `Out` is set to 0. When the interlocking condition no longer applies, the digital value currently valid is output again.

Input parameter for feedback value

This block has three input parameters `Fbk1In`, `Fbk2In` and `Fbk3In` for displaying three feedback values in the faceplate.

Resetting all output values

Reset all output parameters (`Out1` ... `Out3`) by setting all interconnected input parameters for setting (`SetLi1` ... `SetLi3`) or enabled input parameters for setting (`SetOp1` ... `SetOp3`) to 0.

A 0 - 1 edge at the `RstOut` parameter then resets the three output parameters `Out1` ... `Out3`.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Changing labels on buttons and text

This block provides the standard function Labeling of buttons and text (Page 205).

Instance-specific text can be configured for the following parameters:

- `OutX`
- `SetOpX`
- `FbkXIn`

X = (1 ... 3)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `Fbk1In.ST`
- `Fbk2In.ST`
- `Fbk3In.ST`
- `SetLi1.ST` (only if `Feature.Bit10` = 1)
- `SetLi2.ST` (only if `Feature.Bit10` = 1)
- `SetLi3.ST` (only if `Feature.Bit10` = 1)
- `RstOut.ST` (only if `Feature.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position (`Out1` = 0, `Out2` = 0, and `Out3` = 0) in the "Automatic" mode:

- `SetLi1.ST`
- `SetLi2.ST`

- `SetLi3.ST`
- `RstOut.ST`

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can set the digital value `SetOp1`
5	1 = Operator can set the digital value `SetOp2`
6	1 = Operator can set the digital value `SetOp3`
7 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
5	Evaluation of the signal status of the interlock signals (Page 141)
10	Considering bad quality of automatic commands or external values (Page 185)
24	Enabling local operator authorization (Page 157)

See also

Description of OpDi03 (Page 384)

OpDi03 messaging (Page 389)

OpDi03 I/Os (Page 390)

OpDi03 block diagram (Page 393)

OpDi03 error handling (Page 389)
OpDi03 modes (Page 385)

3.5.4 OpDi03 error handling

Error handling of OpDi03

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed
0	There is no error.
51	The value 1 is set at two or more inputs (`SetLi1` = 1 AND `SetLi2` = 1) or (`SetLi2` = 1 AND `SetLi3` = 1) or (`SetLi1` = 1 AND SetLi3 = 1) and `LiOp` = 1

See also

OpDi03 block diagram (Page 393)
OpDi03 I/Os (Page 390)
OpDi03 messaging (Page 389)
OpDi03 functions (Page 386)
OpDi03 modes (Page 385)
Description of OpDi03 (Page 384)

3.5.5 OpDi03 messaging

Messaging

This block does not offer messaging.

See also

Description of OpDi03 (Page 384)
OpDi03 functions (Page 386)
OpDi03 I/Os (Page 390)
OpDi03 block diagram (Page 393)
OpDi03 error handling (Page 389)
OpDi03 modes (Page 385)

3.5.6 OpDi03 I/Os

I/Os of OpDi03

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`Fbk1In`	Feedback for input `In1`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`Fbk2In`	Feedback for input `In2`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`Fbk3In`	Feedback for input `In3`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`Feature`	I/O for additional functions (Page 386)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`Intlock`	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
`Intl_En`	1 = Interlock without reset (interlock, parameter `Intlock`) can be used	BOOL	1
`LiOp`	Switchover of operating mode between: 1 = Interconnection 0 = Operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`OnOp*`	1 = "On" mode via operator	BOOL	0

Parameter	Description	Type	Default
OosLi	1 = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operating permission, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator control permissions (Page 386)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RstOut	Reset by the operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
SetLi1	Connected digital input 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SetLi2	Connected digital input 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SetLi3	Connected digital input 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SetOp1*	Digital input 1 for operator	BOOL	0
SetOp2*	Digital input 2 for operator	BOOL	0
SetOp3*	Digital input 3 for operator	BOOL	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see OpDi03 error handling (Page 389)	INT	-1
LockAct	1 = Interlock (Intlock) is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
Out1	Digital output value 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Out2	Digital output value 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Out3	Digital output value 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status	Status word (Page 384)	DWORD	16#00000000

See also

OpDi03 messaging (Page 389)
OpDi03 block diagram (Page 393)
OpDi03 modes (Page 385)

3.5.7 OpDi03 block diagram

OpDi03 block diagram

A block diagram is not provided for this block.

See also

OpDi03 I/Os (Page 390)

OpDi03 messaging (Page 389)

OpDi03 error handling (Page 389)

OpDi03 functions (Page 386)

OpDi03 modes (Page 385)

Description of OpDi03 (Page 384)

3.5.8 Operator control and monitoring

3.5.8.1 OpDi03 view

Views of the OpDi03 block

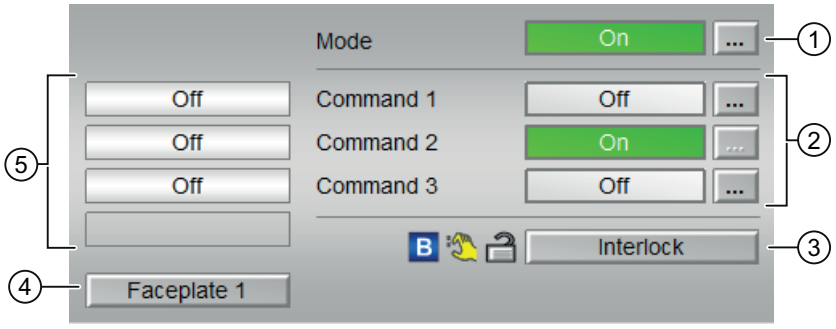
The block OpDi03 provides the following views:

- OpDi03 standard view (Page 394)
- Trend view (Page 299)
- OpDi03 preview (Page 396)
- Memo view (Page 298)
- Block icon for OpDi03 (Page 397)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

3.5.8.2 OpDi03 standard view

OpDi03 standard view



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

You can find additional information on this in section Switching operating states and operating modes (Page 251).

(2) Displaying and switching the command 1 to 3

This area shows you the current selection. You can output a continuous signal at the outputs `Out1` to `Out3` as follows:

- "On": Continuous signal is output
- "Off"

You can find additional information on this in section Switching operating states and operating modes (Page 251).

You can rename the display text as you please, as described in the section Labeling of buttons and text (Page 205).

Do this with the following parameters:

- Text for "Command 1/Command 2/Command 3": Parameter `SetOpX#string_1`, (X = 1 ... 3)
- Text for "On/Off": Parameter `OutX#string_0`, `OutX#string_1`, (X = 1 ... 3)

(3) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is interconnected.

You can use this button to control the interlock functions of the block. You can find additional information on this in section OpDi03 functions (Page 386).

- Bypass information (see Forming the group status for interlock information (Page 104)):

(4) Button for switching to the standard view of any faceplate

Use this button for the standard view of a block configured in the engineering system. The visibility of this button depends on the configuration in the engineering system (ES).

You can find additional information on this in section Opening additional faceplates (Page 203).

(5) Displaying the feedback of the command 1 to 3

This area shows you the current valid selection from `Out1` to `Out3`.

- "On"
- "Off"
- "Invalid signal"

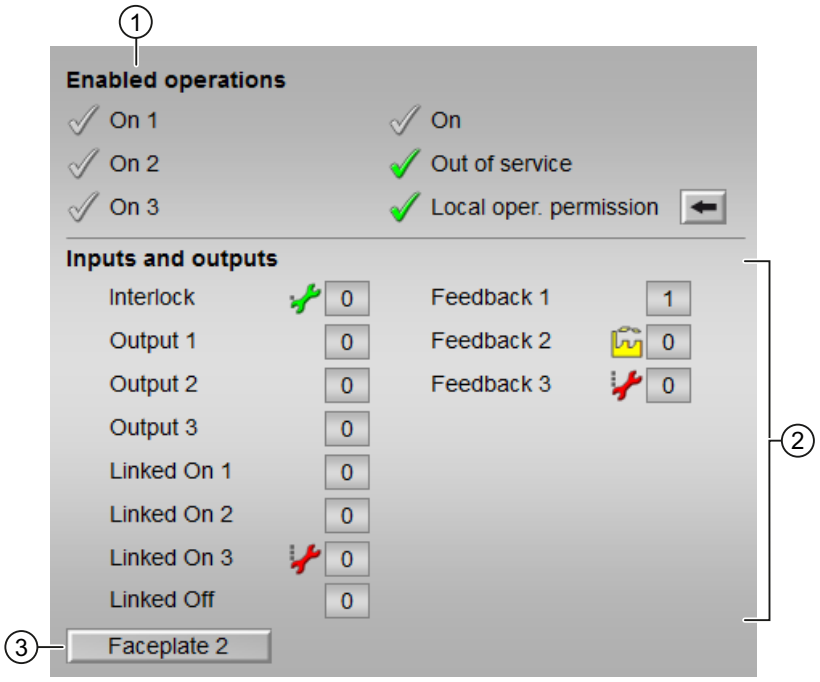
You can rename the display text as you please, as described in the section Labeling of buttons and text (Page 205).

Do this with the following parameters:

- Text for "On/Off": Parameter `FbkInX#string_1`, `FbkInX#string_0`, (X = 1 ... 3)

3.5.8.3 OpDi03 preview

Preview of OpDi03



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "On 1 to 3": You can now set the digital value (0 - 1 edge).
  You can rename the display text as you please, as described in the section Labeling of buttons and text (Page 205).
  Do this with the following parameters:
  - Text for "Command X": Parameter `OutX#string_1`, (X = 1 ... 3)
- "On": You can switch to "On" operating mode.

- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

(2) Display of current inputs and outputs

This area shows the most important parameters for this block with the current selection:

- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Output 1 to 3": 1 = Digital output value set
- "Feedback 1 to 3": 1 = Feedback set
- "Linked On 1 to 3": 1 = Linked input on
- "Linked Off": 1 = Linked input off

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

Refer also to the Opening additional faceplates (Page 203) section for more on this.

3.5.8.4 Block icon for OpDi03

Block icons for OpDi03

A variety of block icons are available with the following functions:

- Process tag type
- Operating modes
- Signal status, release for maintenance
- Bypass
- Interlocks
- Output signal
- Memo display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

3.6 OpStations - Configuration of the local operator authorization

3.6.1 Description of OpStations

Object name (type + number) and family

Type + number: FB 1901

Family: Operate

Area of application for OpStations

The block is used for the following applications:

- Configuration of the local operatorcontrol permission

How it works

The block converts the enabled operations or locks for up to 16 individual permissions in the bit-coded `Out` output.

Refer also to OpStations block diagram (Page 406).

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The OpStations block and the technologic block must be installed into the same cyclic interrupt OB.

Note

Install the blocks in a slower cyclic interrupt OB.

With the default, the local operatorcontrol permission for all block instances on the faceplate is disabled (= 0) and is configured as follows:

1. Assign each operator station a bit-coded ID (1, 2, 4, 8, 16, …), which you set in the internal APLOpStation variables as a start value. You can set up to 16 different operator stations. The variable is automatically created using the OS project editor and is located in the Split Screen Manager group.

   Note

   A redundant server pair can only be represented as an operator station, which also affects referenced client or single stations in the engineering station (ES). If a distinguishable operator station is needed, different groups of referenced clients or single stations must be created

2. Place the OpStations block in the chart.

   Note

   If you use the Opening additional faceplates (Page 203) function in multiple technologic blocks, you need to use one OpStations block for every technologic block. Otherwise, the call of the technologic block from the OpStations block is only possible for a technologic block.

3. Enable the function at the technologic block via `Feature` Bit 24(Enabling local operator authorization (Page 157)).
4. Interconnect the `OpSt_In` input parameter of the technologic block to the `Out` output parameter of the OpStations block.
5. Select the operator stations on which the technologic block will usually be used. You can select several at once. The selection is sent bit-coded to the `Out` output of the OpStations block.
6. Specify a list for the texts for the standard view in the OpStations faceplate in the shared declarations under the name APLOpStations in the SIMATIC Manager. Refer also to the section OpStations standard view (Page 408) for more information.

Once these configuration steps are completed, the local operatorcontrol permission is enabled. The enable for a technologic block is made when the bit-by-bit comparison between the `OpSt_Out` parameter and the operator station ID "APLOpStation" does not equal 0.

Note

The local operatorcontrol permission is not visualized in the "Enabled operation" icons of the block I/Os.

Startup characteristics

This block does not have any startup characteristics.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section OpStations I/Os (Page 404).

Status bit	Parameter
0	`In0`
1	`In1`
2	`In2`
3	`In3`
4	`In4`
5	`In5`
6	`In6`
7	`In7`
8	`In8`
9	`In9`
10	`In10`
11	`In11`
12	`In12`
13	`In13`
14	`In14`
15	`In15`
16 - 31	Not used

See also

OpStations operating modes (Page 401)

OpStations functions (Page 402)

OpStations error handling (Page 403)

OpStations messaging (Page 403)

3.6.2 OpStations operating modes

OpStations operating modes

This block does not have any modes.

See also

Description of OpStations (Page 399)

OpStations functions (Page 402)

OpStations error handling (Page 403)

OpStations messaging (Page 403)

OpStations I/Os (Page 404)
OpStations block diagram (Page 406)

3.6.3 OpStations functions

Functions of OpStations

The functions for this block are listed below.

Operator control permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can enable `In0`
1	1 = Operator can enable `In1`
2	1 = Operator can enable `In2`
3	1 = Operator can enable `In3`
4	1 = Operator can enable `In4`
5	1 = Operator can enable `In5`
6	1 = Operator can enable `In6`
7	1 = Operator can enable `In7`
8	1 = Operator can enable `In8`
9	1 = Operator can enable `In9`
10	1 = Operator can enable `In10`
11	1 = Operator can enable `In11`
12	1 = Operator can enable `In12`
13	1 = Operator can enable `In13`
14	1 = Operator can enable `In14`
15	1 = Operator can enable `In15`
16 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

See also

Description of OpStations (Page 399)
OpStations operating modes (Page 401)
OpStations error handling (Page 403)
OpStations messaging (Page 403)
OpStations I/Os (Page 404)
OpStations block diagram (Page 406)

3.6.4 OpStations error handling

OpStations error handling

This block does not have any error handling.

See also

Description of OpStations (Page 399)
OpStations operating modes (Page 401)
OpStations functions (Page 402)
OpStations messaging (Page 403)
OpStations I/Os (Page 404)
OpStations block diagram (Page 406)

3.6.5 OpStations messaging

OpStations messaging

This block does not offer messaging.

See also

Description of OpStations (Page 399)
OpStations operating modes (Page 401)
OpStations functions (Page 402)
OpStations error handling (Page 403)
OpStations I/Os (Page 404)
OpStations block diagram (Page 406)

3.6.6 OpStations I/Os

OpStations I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 402)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
In0	1 = Enable for operator station 1	BOOL	0
In1	1 = Enable for operator station 2	BOOL	0
In2	1 = Enable for operator station 3	BOOL	0
In3	1 = Enable for operator station 4	BOOL	0
In4	1 = Enable for operator station 5	BOOL	0
In5	1 = Enable for operator station 6	BOOL	0
In6	1 = Enable for operator station 7	BOOL	0
In7	1 = Enable for operator station 8	BOOL	0
In8	1 = Enable for operator station 9	BOOL	0
In9	1 = Enable for operator station 10	BOOL	0
In10	1 = Enable for operator station 11	BOOL	0
In11	1 = Enable for operator station 12	BOOL	0
In12	1 = Enable for operator station 13	BOOL	0
In13	1 = Enable for operator station 14	BOOL	0
In14	1 = Enable for operator station 15	BOOL	0
In15	1 = Enable for operator station 16	BOOL	0
OS_Perm	I/O for operator control permissions (Page 402)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF

Parameter	Description	Type	Default
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
Out	Output value: for additional interconnections to the OpSt_In input of a technologic block	DWORD	16#00000000
Status	Status word (Page 399)	DWORD	16#00000000

See also

OpStations operating modes (Page 401)

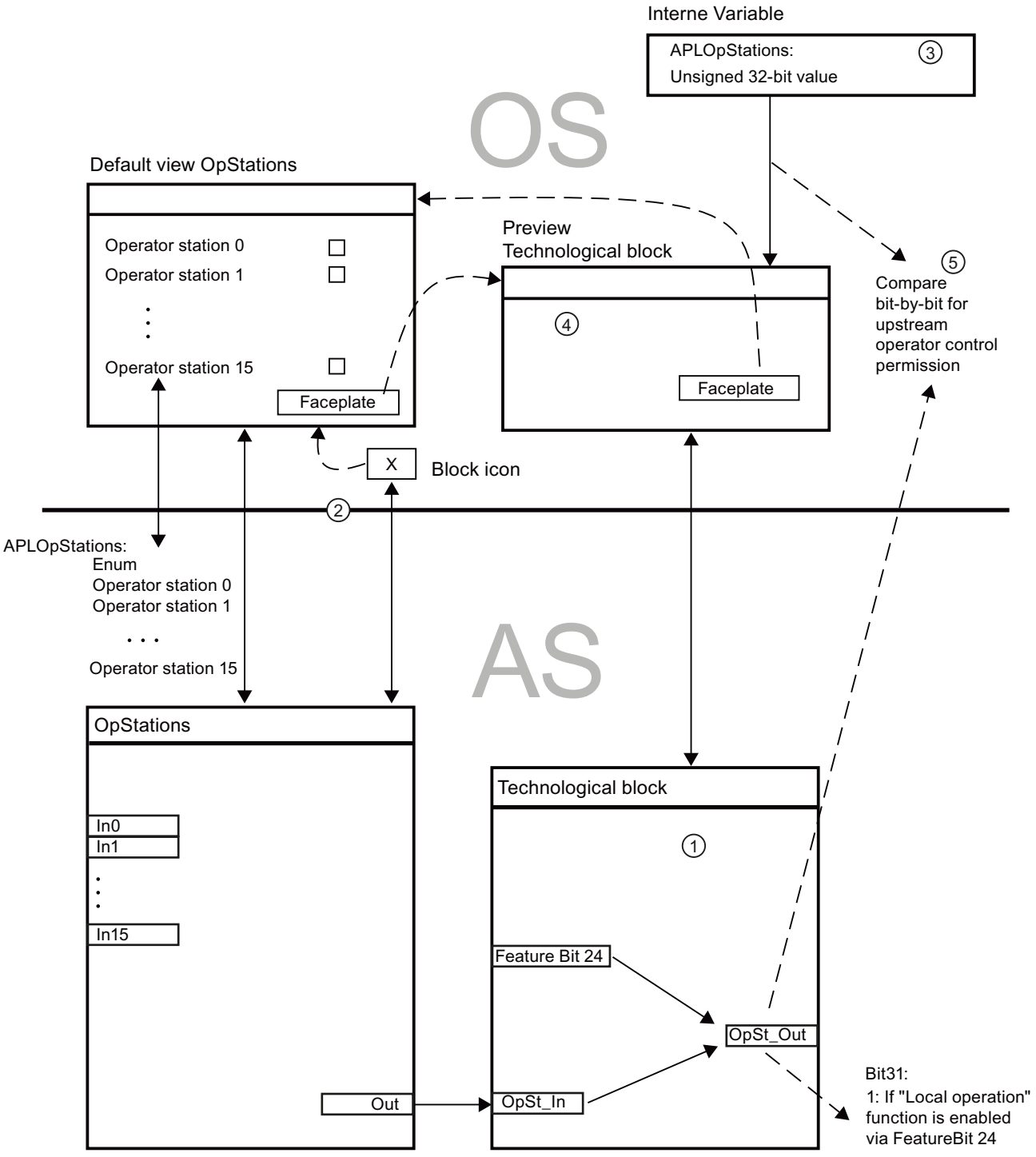
OpStations error handling (Page 403)

OpStations messaging (Page 403)

OpStations block diagram (Page 406)

3.6.7 OpStations block diagram

OpStations block diagram



See also

Description of OpStations (Page 399)

OpStations operating modes (Page 401)

OpStations functions (Page 402)

OpStations error handling (Page 403)

OpStations messaging (Page 403)

OpStations I/Os (Page 404)

3.6.8 Operator control and monitoring

3.6.8.1 OpStations views

Views of the OpStations block

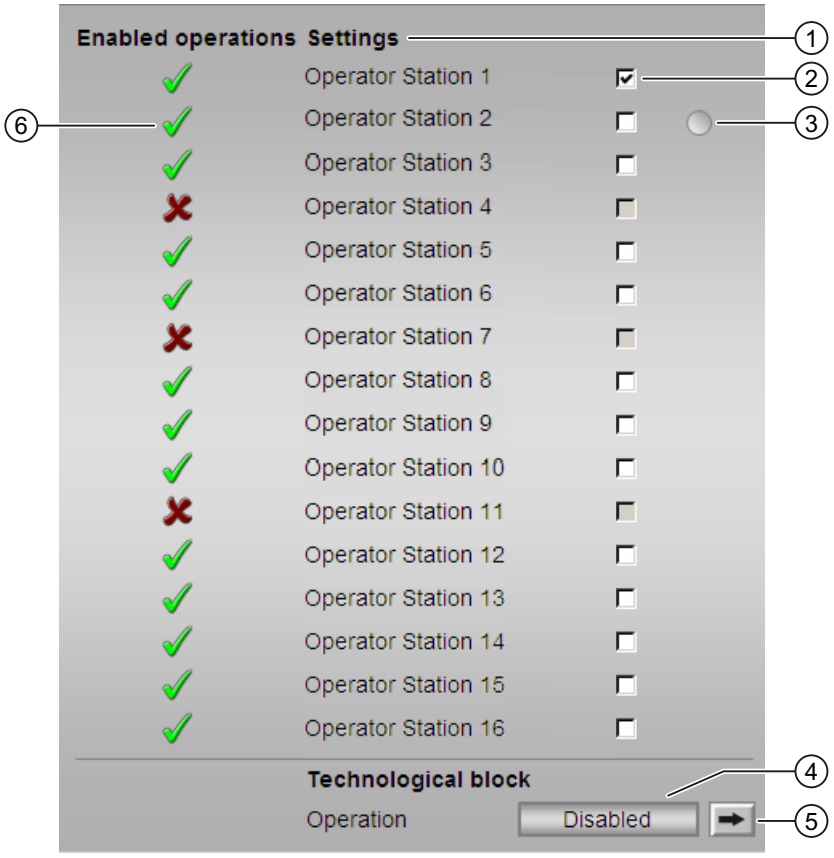
The `OpStations` block provides the following views:

- OpStations standard view (Page 408)
- Memo view (Page 298)
- Block icon of OpStations (Page 410)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

3.6.8.2 OpStations standard view

OpStations standard view



(1) Configurable display text under the settings

You yourself can define the text for operator station 1 to operator station 15 instead of the default text. Follow the steps outlined below:

- Create a list with the name "APLOpStations" in the SIMATIC Manager in the shared declarations. To learn more on this, refer to "How to save shared declarations" in the Process Control System PCS 7 Engineering System Configuration Manual.

Only values from 0 to 15 are permitted, other values will not be saved in the list.

The display names of the values are changed with the names of the operator stations. The display name corresponds to the value 0 of the display for the check box `In0` in the standard view etc.

If the text is not configured, the entire line is not displayed.

(2) Disabling or enabling operation for operator stations

In this area, you can disable the operation for an operator station or enable the operator station for the connected technologic block. In this case, the upper check box corresponds to the `In0` I/O and the lower check box to the `In15` I/O. Operation is only possible with the highest-level operator control permission (same as simulation) .

(3) Display for the current operator station

The value of the current operator station is displayed as a gray dot in the corresponding line.

(4) Display for operability

Display of the operability of the technologic block on the current operator station.

(5) Navigation button for switching to the standard view of the technologic block

Use this navigation button to reach the standard view of the technologic block. The visibility of this navigation button depends on the configuration in the engineering system (ES).

Additional information is available in the section Opening additional faceplates (Page 203).

(6) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

3.6.8.3 Block icon of OpStations

Block icons for OpStations

A variety of block icons are available with the following functions:

- Opening the faceplate

Icons	Selection of the block icon in CFC	Special features
Without icon	1	Default no block icon
	2	The block icon is transparent, if operation on the operator station is possible

Note

The block icon is the size of a field in the status bar of a technologic block icon and can be used as an add-on to the status bar. The layer of the block icon here should always be higher than the layer of the technologic block icon, otherwise the block icon may be hidden after an update.

3.7 OpTrig - Manipulating a digital value (1 pushbutton)

3.7.1 Description of OpTrig

Object name (type + number) and family

Type + number: FB 1868

Family: Operate

Area of application for OpTrig

The block is used for the following applications:

- Generation of a pulse signal (trigger)

How it works

Operator control block is used to implement single pushbutton control (comparable with RESET pushbutton).

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section OpTrig I/Os (Page 415).

Status bit	Parameter
0 - 2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`
7	`Out.Value`
8	`LiOpAct.Value`
9	`FbkIn.Value`
10 - 31	Not used

See also

OpTrig functions (Page 412)
OpTrig messaging (Page 415)
OpTrig block diagram (Page 417)
OpTrig error handling (Page 414)
OpTrig modes (Page 412)

3.7.2 OpTrig modes

OpTrig operating modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

OpTrig block diagram (Page 417)
OpTrig I/Os (Page 415)
OpTrig messaging (Page 415)
OpTrig functions (Page 412)
OpTrig error handling (Page 414)
Description of OpTrig (Page 411)

3.7.3 OpTrig functions

Functions of OpTrig

The functions for this block are listed below.

Issuing trigger signal internally or externally

Use the parameter LiOp to define whether the trigger signal is to be output by interconnection or by the OS operator:

LiOp = 0: Trigger signal by OS operator (input parameter InOp)

LiOp = 1: Trigger signal via interconnection (input parameter InLi)

Input parameter for feedback value

This block has a FbkIn input parameter for displaying a feedback value in the faceplate.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status ST_Worst for the block is formed from the following parameter:

- FbkIn.ST
- InLi.ST (only if Feature.Bit10 = 1)

Considering bad quality of automatic commands or external values

If the Feature bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and the parameter InLi.ST has bad signal status (16#00 or 16#28), the block goes to the neutral position (Out = 0) in the "Automatic" mode.

Operator control permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the OS_Perm parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can set the input parameter In
5 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Simulating signals

This block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
1	Reaction to the out of service mode (Page 176)
10	Considering bad quality of automatic commands or external values (Page 185)
24	Enabling local operator authorization (Page 157)

See also

Description of OpTrig (Page 411)

OpTrig messaging (Page 415)

OpTrig I/Os (Page 415)

OpTrig block diagram (Page 417)

OpTrig error handling (Page 414)

OpTrig modes (Page 412)

3.7.4 OpTrig error handling

OpTrig error handling

The block does not report any errors.

See also

OpTrig block diagram (Page 417)

OpTrig I/Os (Page 415)

OpTrig messaging (Page 415)

OpTrig functions (Page 412)

OpTrig modes (Page 412)

Description of OpTrig (Page 411)

3.7.5 OpTrig messaging

Messaging

This block does not offer messaging.

See also

Description of OpTrig (Page 411)

OpTrig functions (Page 412)

OpTrig I/Os (Page 415)

OpTrig block diagram (Page 417)

OpTrig error handling (Page 414)

OpTrig modes (Page 412)

3.7.6 OpTrig I/Os

OpTrig I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
FbkIn	Input for feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Feature	I/O for additional functions (Page 412)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
InLi	Interconnectable binary input	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
InOp*	Binary input for operator	BOOL	0

Parameter	Description	Type	Default
LiOp	Switchover of operating mode between: 1 = Interconnection 0 = Operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operating permission, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator control permissions (Page 412)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
SelFp1	Call a block saved in this parameter as additional faceplate (Page 203) in standard view	ANY	-
SelFp2	Call a block saved in this parameter as additional faceplate (Page 203) in the preview	ANY	-
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Reserved	INT	-1
LiOpAct	Operator/interconnection is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80

Parameter	Description	Type	Default
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status	Status word (Page 411)	DWORD	16#00000000

See also

OpTrig messaging (Page 415)

OpTrig block diagram (Page 417)

OpTrig error handling (Page 414)

OpTrig modes (Page 412)

3.7.7 OpTrig block diagram

OpTrig block diagram

A block diagram is not provided for this block.

See also

OpTrig I/Os (Page 415)

OpTrig messaging (Page 415)

OpTrig functions (Page 412)

OpTrig error handling (Page 414)

OpTrig modes (Page 412)

Description of OpTrig (Page 411)

3.7.8 Operator control and monitoring

3.7.8.1 OpTrig views

Views of the OpTrig block

The block OpTrig provides the following views:

- OpTrig standard view (Page 418)
- OpTrig preview (Page 419)
- Memo view (Page 298)
- Block icon for OpTrig (Page 420)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

3.7.8.2 OpTrig standard view

OpTrig standard view

(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

You can find additional information on this in the Switching operating states and operating modes (Page 251) section.

(2) Set

Clicking "Set", outputs a pulse signal with the length of the cycle time at the `Out` output.

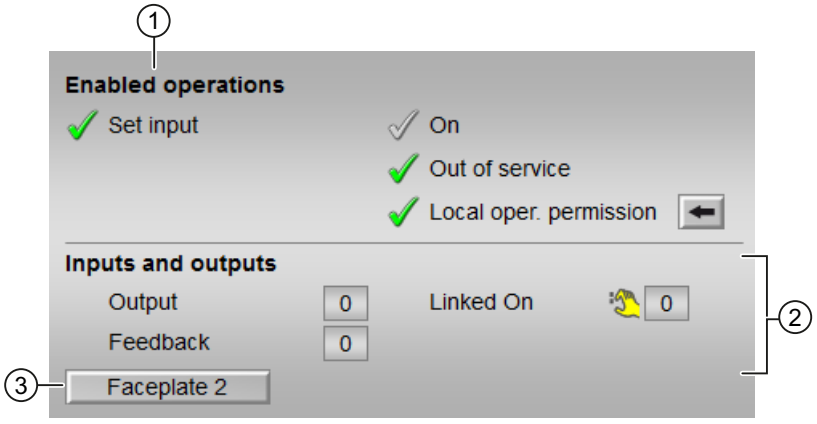
(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the Engineering System (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the Opening additional faceplates (Page 203) section.

3.7.8.3 OpTrig preview

Preview of OpTrig



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter.
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process.
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

The following enabled operations are shown here:

- "Set input": You can set the input.
- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

(2) Display of current inputs and outputs

This area shows the most important parameters for this block with the current selection:

- "Output": 1 = Digital output value set
- "Feedback": 1 = Feedback set
- "Linked On": 1 = Linked input on

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

Refer also to the Opening additional faceplates (Page 203) section for more on this.

3.7.8.4 Block icon for OpTrig

Block icons for OpTrig

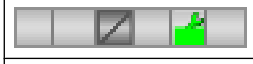
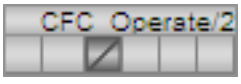
A variety of block icons are available with the following functions:

- Process tag type
- Operating modes
- Signal status, release for maintenance
- Memo display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Monitoring blocks

4

4.1 Comparison of large & small blocks

4.1.1 MonAnL compared to MonAnS

Comparison of the MonAnL and MonAnS blocks

The following tables are intended to help you decide which block to use.

Memory and runtime savings of the small block compared to the large block

You save the following resources for each instance:

- Memory space: ~ 45%
- Runtime: ~ 30%

Operating modes of the blocks

	MonAnL	MonAnS
On (Page 71)	X	X
Out of service (Page 71)	X	X

Functions of the blocks

	MonAnL	MonAnS
Alarm delays with two time values per limit pair (Page 197)	X	
Alarm delays with one time value per limit pair (Page 196)		X
Limit monitoring (Page 86)	X	X
Suppressing messages using the MsgLock parameter (Page 201)	X	X
Gradient monitoring (Page 440)	X	
Displaying auxiliary values (Page 207)	X	
Forming and outputting the signal status for technologic blocks (Page 108)	X	X

	MonAnL	MonAnS
Dead band (Page 61)	X	X
Release for maintenance (Page 64)	X	X
Simulating signals (Page 58)	X	X
Selecting a unit of measure (Page 207)	X	X
Operator control permissions (Page 248)	X	X
Generating instance-specific messages (Page 200)	X	X
Display and operator input area for process values and setpoints (Page 203)	X	X
Opening additional faceplates (Page 203)	X	X
Time stamp (Page 1634)	X	
SIMATIC BATCH functionality (Page 67)	X	X
Providing PV limit at the output (Page 440)	x	

Configurable functions using the `Feature` parameter

Bit number	`Feature` bit function	MonAnL	MonAnS
0	Set startup characteristics (Page 137)	X	X
1	Reaction to the out of service mode (Page 176)	X	X
22	Update acknowledgment and error status of the message call (Page 159)	X	
24	Enabling local operator authorization (Page 157)	X	X
25	Suppression of all messages (Page 173)	X	X
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)	X	X
28	Disabling operating points (Page 144)	X	
29	Signaling limit violation (Page 169)	X	

4.1.2 MonDiL compared to MonDiS

Comparison of the MonDiL and MonDiS blocks

The following tables are intended to help you decide which block to use.

Memory and runtime savings of the small block compared to the large block

You save the following resources for each instance:

- Memory space: ~ 55%
- Runtime: ~ 12%

Operating modes of the blocks

	MonDiL	MonDiS
On (Page 71)	x	x
Out of service (Page 71)	x	x

Functions of the blocks

	MonDiL	MonDiS
Suppression and and reporting of signal flutter (Page 491)	x	
Delaying the on and off function (Page 491)	x	
Delaying the on function (Page 515)		x
Specifying the status display for the block icon (Page 491)	x	x
Displaying auxiliary values (Page 207)	x	
Labeling of buttons and text (Page 205)	x	x
Generating instance-specific messages (Page 200)	x	x
Suppressing messages using the MsgLock parameter (Page 201)	x	x
Forming and outputting the signal status for technologic blocks (Page 108)	x	x
Release for maintenance (Page 64)	x	x
Simulating signals (Page 58)	x	x

	MonDiL	MonDiS
Operator control permissions (Page 248)	x	x
Opening additional faceplates (Page 203)	x	x
Time stamp (Page 1634)	x	
SIMATIC BATCH functionality (Page 67)	x	x
Group display for limit monitoring, CSF and ExtMsgx (Page 85)	x	x

Configurable functions using the `Feature` parameter

Bit number	`Feature` bit function	MonDiL	MonDiS
0	Set startup characteristics (Page 137)	x	x
1	Reaction to the out of service mode (Page 176)	x	x
22	Update acknowledgment and error status of the message call (Page 159)	x	
24	Enabling local operator authorization (Page 157)	x	x
25	Suppression of all messages (Page 173)	x	x

4.2 AV - Displaying and monitoring additional value

4.2.1 Description of AV

Object name (type + number) and family

Type + number: FB 1903

Family: Monitor

Area of application for AV

The block is used for the following applications:

- Monitoring an additional analog value at a technologic block (for example, motor, valve).

How it works

The block must be connected to a channel block and monitors an additional analog value. The messages of the AV block appear in the alarm view of the technologic block connected to it. Monitoring limits are configured and controlled at the technologic block.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100). The AV block and the technologic block must be installed in the same cyclic interrupt OB.

Interconnect the `AV_Tech` output parameter of the AV block to the input parameter `AV` of the technologic block in the CFC.

Note

Interconnection of the block to multiple technologic blocks is not permitted.

For the AV block, the Advanced Process Library contains process tag type templates; these serve as examples by providing various application scenarios for this block.

Examples of process tag types:

- Motor with an additional analog value and time-stamped signals (Motor_AV_EventTs) (Page 2339)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section AV I/Os (Page 433).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`MsgLock`
4	`AV_AH_Act.Value`
5	`AV_WH_Act.Value`
6	`AV_TH_Act.Value`
7	`AV_TL_Act.Value`
8	`AV_WL_Act.Value`
9	`AV_AL_Act.Value`
10	`AV_AH_En`
11	`AV_WH_En`
12	`AV_TH_En`
13	`AV_TL_En`
14	`AV_WL_En`
15	`AV_AL_En`
16	`AV_AH_MsgEn`
17	`AV_WH_MsgEn`
18	`AV_TH_MsgEn`
19	`AV_TL_MsgEn`
20	`AV_WL_MsgEn`
21	`AV_AL_MsgEn`
22 - 31	Not used

See also

AV modes (Page 428)
AV functions (Page 429)
AV error handling (Page 431)
AV messaging (Page 431)
AV block diagram (Page 436)

4.2.2 AV modes

AV operating modes

The block does not have any of its own operating modes.

If the interconnected technologic block is set to the "Out of service" operating mode, the AV block is also set to the "Out of service" mode. In this case `AV_Out` = `AV`.

See also

Description of AV (Page 427)

AV functions (Page 429)

AV error handling (Page 431)

AV messaging (Page 431)

AV I/Os (Page 433)

AV block diagram (Page 436)

4.2.3 AV functions

Functions of AV

The functions for this block are listed below.

Alarm delays with two time values per limit pair

This block includes the standard function alarm delay for Two time values per limit pair (Page 197).

Limit monitoring of an additional analog value

This block provides the standard function Limit monitoring of an additional analog value (Page 91).

Limit monitoring with hysteresis

This block provides the standard function Limit monitoring with hysteresis (Page 97). It is performed via the input parameter `AV_Hyst`.

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Forming the signal status for blocks

The worst signal status `ST_Worst` for the block is formed from the following parameter:

- `AV_Out.ST`

Simulating signals

This block provides the standard function Simulating signals (Page 58).

The simulation in the block AV is activated at the technological block (`SimOn` = 1). The simulation value `SimAV` for the block AV is also specified there. If internal simulation of the technological block is switched on, the output `AV_Out` has the value of `SimAV` and the status 16#60.

Release for maintenance

The release for maintenance in the AV block is activated at the technological block (`MS_Release` = 1).

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) chapter. The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
22	Update acknowledgment and error status of the message call (Page 159)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175) (transferred to AV via AV_Tech.Mode.Bit)
29	Signaling limit violation (Page 169) (transferred to AV via AV_Tech.Mode.Bit)

See also

AV modes (Page 428)

AV error handling (Page 431)

AV messaging (Page 431)

AV I/Os (Page 433)

AV block diagram (Page 436)

Description of AV (Page 427)

4.2.4 AV error handling

Error handling of AV

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error

Overview of error numbers

The `ErrorNum` I/O can be used to output various error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed
0	There is no error.
30	The value of `AV` can no longer be displayed in the REAL number field.

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

See also

Description of AV (Page 427)

AV modes (Page 428)

AV functions (Page 429)

AV messaging (Page 431)

AV I/Os (Page 433)

AV block diagram (Page 436)

4.2.5 AV messaging

Messaging

The following messages can be generated for this block:

- Process messages

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	Alarm - high	$$BlockComment$$ AV - high alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ AV - high warning limit violated
	SIG 3	Tolerance - high	$$BlockComment$$ AV - high tolerance limit violated
	SIG 4	Tolerance - low	$$BlockComment$$ AV - low tolerance limit violated
	SIG 5	Warning - low	$$BlockComment$$ AV - Low warning limit violated
	SIG 6	Alarm - low	$$BlockComment$$ AV - low alarm limit violated

Explanation

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance **MsgEvId1**

Associated value	Block parameters
1	Reserved
2	Reserved
3	Reserved
4	AV_Out
5	ExtVa105
6	ExtVa106
7	ExtVa107
8	Reserved
9	Reserved
10	Reserved

The associated values 5 ... 7 are allocated to the parameters ExtVa105 ... ExtVa107 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of AV (Page 427)

AV modes (Page 428)

AV functions (Page 429)

AV error handling (Page 431)

AV I/Os (Page 433)
AV block diagram (Page 436)

4.2.6 AV I/Os

I/Os of AV

Input parameters

Parameter	Description	Type	Default
AV	Analog value	STRUCT • Value:REAL • ST:BYTE	- • 0.0 • 16#80
AV_Unit	Unit of measure for the analog value	INT	1001
AV_A_DC*	Delay time for incoming alarms [s]	REAL	0.0
AV_A_DG*	Delay time for outgoing alarms [s]	REAL	0.0
AV_AH_En	1 = Enable high alarm	BOOL	1
AV_AH_MsgEn	1 = Enable high alarm message	BOOL	1
AV_AL_En	1 = Enable low alarm	BOOL	1
AV_AL_MsgEn	1 = Enable low alarm message	BOOL	1
AV_OpScale	Limit for scale in bar graph of faceplate for the analog value	STRUCT • High:REAL • Low:REAL	- • 100.0 • 0.0
AV_T_DC*	Delay time for incoming tolerances [s]	REAL	0.0
AV_T_DG*	Delay time for outgoing tolerances [s]	REAL	0.0
AV_TH_En	1 = Enable high tolerance	BOOL	0
AV_TH_MsgEn	1 = Enable high tolerance message	BOOL	1
AV_TL_En	1 = Enable low tolerance	BOOL	0
AV_TL_MsgEn	1 = Enable low tolerance message	BOOL	1
AV_W_DC*	Delay time for incoming warnings [s]	REAL	0.0
AV_W_DG*	Delay time for outgoing warnings [s]	REAL	0.0
AV_WH_En	1 = Enable high warning	BOOL	1
AV_WH_MsgEn	1 = Enable high warning message	BOOL	1
AV_WL_En	1 = Enable low warning	BOOL	1
AV_WL_MsgEn	1 = Enable low warning message	BOOL	1
EN	1 = Called block will be processed	BOOL	1
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	

Parameter	Description	Type	Default
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
Feature	I/O for additional functions (Page 429)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AV_AH_Act	1 = High alarm active. You can change the reaction for this parameter with Feature Bit 28 (Disabling operating points (Page 144)) and with Feature Bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AV_AL_Act	1 = Low alarm active. You can change the reaction for this parameter with Feature Bit 28 (Disabling operating points (Page 144)) and with Feature Bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AV_Out	Analog value output	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
AV_Tech	Output parameter with which the input parameter AV of the technologic block has to be connected. This includes tags which are passed on to the technologic block for additional processing.	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#00
AV_TH_Act	1 = High tolerance active. You can change the reaction for this parameter with Feature Bit 28 (Disabling operating points (Page 144)) and with Feature Bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
AV_TL_Act	1 = Low tolerance active. You can change the reaction for this parameter with Feature Bit 28 (Disabling operating points (Page 144)) and with Feature Bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AV_WH_Act	1 = High warning active. You can change the reaction for this parameter with Feature Bit 28 (Disabling operating points (Page 144)) and with Feature Bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AV_WL_Act	1 = Low warning active. You can change the reaction for this parameter with Feature Bit 28 (Disabling operating points (Page 144)) and with Feature Bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see AV error handling (Page 431)	INT	-1
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 427)	DWORD	16#00000000

See also

AV modes (Page 428)

AV messaging (Page 431)

AV block diagram (Page 436)

4.2.7 AV block diagram

AV block diagram

A block diagram is not provided for this block.

See also

Description of AV (Page 427)
AV modes (Page 428)
AV functions (Page 429)
AV error handling (Page 431)
AV messaging (Page 431)
AV I/Os (Page 433)

4.3 MonAnL - Monitoring of an analog process tag (Large)

4.3.1 Description of MonAnL

Object name (type+number) and family

Type + number: FB 1845

Family: Monitor

Area of application for MonAnL

The block is used for the following fields of applications:

- Monitoring an analog process value
- Monitoring of the gradient of an analog process value

Note

This block is also available as a small block. A comparison of the MonAnL and MonAnS blocks is available in the section: MonAnL compared to MonAnS (Page 423)

How it works

The MonAnL block is used to monitor an analog process tag and the corresponding limits. It also monitors the gradient of these signals. The block generates and outputs corresponding messages if limits are violated or if a signal gradient does not meet requirements.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the MonAnL block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Monitoring an analog process tag (AnalogMonitoring) (Page 2330)
- Monitoring of an analog process tag for PA/FF devices (AnalogMonitoring_Fb) (Page 2331)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

The output parameters PV_HysOut, PV_AH_Out, PV_WH_Out, PV_TH_Out, PV_AL_Out, PV_WL_Out and PV_TL_Out are written by the corresponding input parameters PV_Hys, PV_AH_Lim, PV_WH_Lim, PV_TH_Lim, PV_AL_Lim, PV_WL_Lim and PV_TL_Lim.

Status word allocation for Status1 parameter

You can find a description for each parameter in section MonAnL I/Os (Page 450).

Status bit	Parameter
0	Occupied
1	BatchEn
2	SimOn
3	OosAct.Value
4	OosLi.Value
5	Not used
6	OnAct.Value
7	BypassAct.Value
8	Display of BypassAct.Value in faceplate (display and operator controls) and block icon
9	Not used
10	SimLiOp.Value
11	Delay of the PV_AH_Lim message
12	Delay of the PV_WH_Lim message
13	Delay of the PV_TH_Lim message
14	Delay of the PV_TL_Lim message
15	Delay of the PV_WL_Lim message
16	Delay of the PV_AL_Lim message
17	Collection of message delays
18 - 29	Not used
30	Auxiliary value 1 visible
31	Auxiliary value 2 visible

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock
1	PV_AH_Act.Value
2	PV_WH_Act.Value
3	PV_TH_Act.Value
4	PV_TL_Act.Value
5	PV_WL_Act.Value
6	PV_AL_Act.Value
7	PV_AH_En
8	PV_WH_En

Status bit	Parameter
9	PV_TH_En
10	PV_TL_En
11	PV_WL_En
12	PV_AL_En
13	PV_AH_MsgEn
14	PV_WH_MsgEn
15	PV_TH_MsgEn
16	PV_TL_MsgEn
17	PV_WL_MsgEn
18	PV_AL_MsgEn
19	GradHUpAct.Value
20	GradHDnAct.Value
21	GradLAct.Value
22	GradHUpEn
23	GradHDnEn
24	GradLEn
25	GradHUpMsgEn
26	GradHDnMsgEn
27	GradLMsgEn
28	0 = falling measured value 1 = rising measured value
29	Not used
30	Not used
31	MS_RelOp

Status word allocation for `Status3` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via EventTsIn
1	Effective signal 2 of the message block connected via EventTsIn
2	Effective signal 3 of the message block connected via EventTsIn
3	Effective signal 4 of the message block connected via EventTsIn
4	Effective signal 5 of the message block connected via EventTsIn
5	Effective signal 6 of the message block connected via EventTsIn
6	Effective signal 7 of the message block connected via EventTsIn
7	Effective signal 8 of the message block connected via EventTsIn
8 - 23	Effective signal 9..16 of the message block connected via EventTsIn
24 - 31	Not used

See also

MonAnL functions (Page 440)

MonAnL messaging (Page 448)

MonAnL block diagram (Page 457)

MonAnL error handling (Page 447)

MonAnL modes (Page 440)

4.3.2 MonAnL modes

MonAnL operating modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

You can find general information about the "On" mode in the On (Page 71) section.

"Out of service"

You can find general information about the "Out of service" mode in the Out of service (Page 71) section.

See also

MonAnL block diagram (Page 457)

MonAnL I/Os (Page 450)

MonAnL messaging (Page 448)

MonAnL error handling (Page 447)

MonAnL functions (Page 440)

Description of MonAnL (Page 437)

4.3.3 MonAnL functions

Functions of MonAnL

The functions for this block are listed below.

Alarm delays with two time values

With the Separate delay times for each alarm (Page 169) function (Feature bit 8), separate input parameters can be set for the alarm delay on the high and low limits.

Feature bit **8 = 0**

This block includes the standard function alarm delay for Two time values per limit pair (Page 197).

Feature bit **8 = 1**

This block includes the standard function alarm delay for Two time values for each individual limit (Page 198).

Limit monitoring of the process value

This block provides the standard function Limit monitoring of the process value (Page 86).

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Gradient monitoring

The `PV_Grad` gradient is calculated with a delay function, `LagTime`. This setting smoothes the jumps of the PV input value with gradient calculation.

The gradient peak values are output at the output parameters `PV_GradNP` (negative slope) and `PV_GradPP` (positive slope). They are reset as soon as the operator issues the reset command.

The slope of the `PV_Grad` gradient can be monitored for the following limits:

- Limit (high) for positive gradients (`GradHUpLim`)
- Limit (high) for negative gradients (`GradHDnLim`)
- Limit (low) for absolute gradients (`GradLLim`)

The individual monitoring functions are activated at the corresponding "Enable" parameters, e.g. `GradHUpEn` for activating the high gradient limit for positive gradients (`GradHUpLim`).

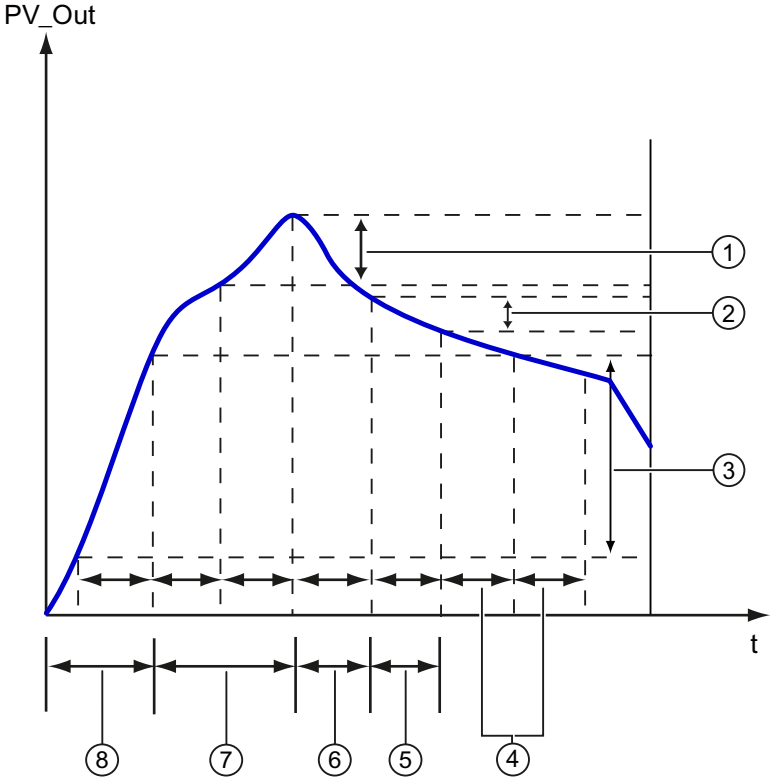
When the values you have defined are reached or exceeded, this is indicated at the corresponding "Active" output parameters, e.g. with `GradHUpAct` = 1 for the limit (high) for positive gradients.

Messages can be output for these alarms. You activate these as follows:

- Message for alarms (high) for positive gradients: `GradHUpMsgEn` = 1
- Message for alarms (high) for negative gradients: `GradHDnMsgEn` = 1
- Message for alarms (low) for absolute gradients: `GradLMsgEn` = 1

Example of generating alarms for gradient monitoring

The following example shows how alarms for gradient monitoring are generated.



Number	Description
1	Absolute gradient difference ≥ `GradHDnLim`
2	Absolute gradient difference ≤ `GradLLim`
3	Gradient difference ≥ `GradHUpLim`
4	Sampling time (`SampleTime`)
5	Alarm (low) for absolute gradients
6	Alarm (high) for negative gradients
7	No alarm
8	Alarm (high) for positive gradients

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

Providing PV limit at the output

The following input parameters are also available at the output for making additional interconnections to other blocks:

• PV_HysOut	:= PV_Hyst
• PV_AH_Out	:= PV_AH_Lim
• PV_WH_Out	:= PV_WH_Lim
• PV_TH_Out	:= PV_TH_Lim
• PV_TL_Out	:= PV_TL_Lim
• PV_WL_Out	:= PV_WL_Lim
• PV_AL_Out	:= PV_AL_Lim

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `PV_Out.ST`

Dead band

To suppress values that fluctuate around the zero point, this block has the standard function Dead band (Page 61).

Smoothing PV

The block smoothes the input value `PV` using a first order time delay. This delay time can be configured with the parameter `SmoothTi`. The input of `SmoothTi` is limited to 0 to 999999 sec. The block writes back the limits if the input value is outside the limits.

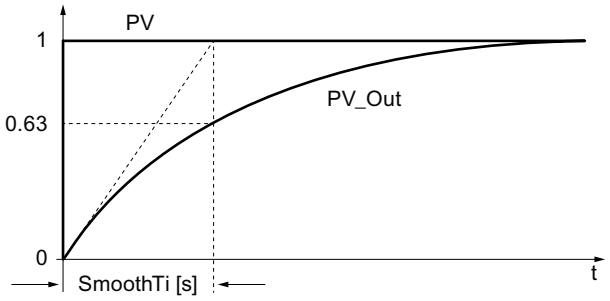
The block works according to the following formula:

$PV\_Out_{(n)} = PV_{(n)} + (PV\_Out_{(n-1)} - PV_{(n)})*e^{(-SampleTime/SmoothTi)}$

Where:

- `PV_Out` = Output value
- `SmoothTi` = Smooth time
- `SampleTime` = Sampling time
- `PV` = Input value

The formula is valid only if `SmoothTi` > 0. If `SmoothTi` = 0, the input is passed directly to the output. If the input value is outside the REAL range limits, the calculation is stopped. If the input value is inside the range limits again, the calculation is resumed automatically.



Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Simulating signals

This block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Process value (`SimPV`, `SimPV_Li`)

Bypass function

This block provides the standard function Bypassing signals (Page 107).

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
8	Separate delay times for each alarm (Page 169)
9	Substitution value is active if the block is in bypass (Page 184)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can activate bypass functionality
5	1 = Operator can deactivate bypass functionality
6	1 = Operator can change the simulation value `SimPV`
7	1 = Operator can reset the maximum values
8	1 = Operator can activate / deactivate messages via `GradHUpMsgEn`
9	1 = Operator can activate / deactivate messages via `GradHDnMsgEn`
10	1 = Operator can activate / deactivate messages via `GradLMsgEn`
11	1 = Operator can activate the Simulation function
12	1 = Operator can activate the Release for maintenance function
13	1 = Operator can change the limit (`PV`) for high alarm
14	1 = Operator can change the limit (`PV`) for high warning
15	1 = Operator can change the limit (`PV`) for high tolerance
16	1 = Operator can change the limit (`PV`) for hysteresis
17	1 = Operator can change the limit (`PV`) for low alarm
18	1 = Operator can change the limit (`PV`) for low warning
19	1 = Operator can change the limit (`PV`) for low tolerance
20	1 = Operator can change the value for the high gradient limit for positive slopes (`GradHUpLim`)
21	1 = Operator can change the value for high gradient limit for negative slopes (`GradHDnLim`)
22	1 = Operator can change the value for the low gradient limit for positive and negative slopes (`GradLLim`)
23	1 = Operator can change the dead band parameter `DeadBand`
24	1 = Operator can activate / deactivate messages via `PV_AH_MsgEn`
25	1 = Operator can activate / deactivate messages via `PV_WH_MsgEn`
26	1 = Operator can activate / deactivate messages via `PV_TH_MsgEn`
27	1 = Operator can activate / deactivate messages via `PV_TL_MsgEn`
28	1 = Operator can activate / deactivate messages via `PV_WL_MsgEn`
29	1 = Operator can activate / deactivate messages via `PV_AL_MsgEn`
30	1 = Operator can enter `SmoothTi`
31	Reserved

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Specifying the display area for process and setpoint values as well as operations

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of MonAnL (Page 437)

MonAnL messaging (Page 448)

MonAnL I/Os (Page 450)

MonAnL block diagram (Page 457)

MonAnL error handling (Page 447)

MonAnL modes (Page 440)

EventTs functions (Page 1634)

Error handling (Page 119)

Sealing the valve (Page 175)

4.3.4 MonAnL error handling

Error handling of MonAnL

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following error message can be generated at this block:

- Error numbers
- Control system fault (CSF)

Overview of error numbers

The ErrorNum I/O can be used to output various error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.
11	SmoothTi < 0 or SmoothTi > 999999: Due to the internal limitation to 0 or 999999 by the block, the error number has the value 11 only for one cycle. If SmoothTi is connected and is written in every cycle, the value 11 will be retained.
30	The value of PV can no longer be displayed in the REAL number field.
43	TimeFactor < 0 or TimeFactor > 2

Control system fault (CSF)

An external signal can be activated via the CSF input. If this signal changes to = 1, a control system fault is triggered. Refer to the Error handling (Page 119) section for more on this.

See also

MonAnL block diagram (Page 457)

MonAnL I/Os (Page 450)

MonAnL messaging (Page 448)

MonAnL functions (Page 440)

MonAnL modes (Page 440)

Description of MonAnL (Page 437)

4.3.5 MonAnL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a process control fault is triggered (`MsgEvId2`, SIG 2).

Process messages

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 1	Alarm - high	$$BlockComment$$ `PV` - high alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ `PV` - high warning limit violated
	SIG 3	Tolerance - high	$$BlockComment$$ `PV` - high tolerance limit violated
	SIG 4	Tolerance - low	$$BlockComment$$ `PV` - low tolerance limit violated
	SIG 5	Warning - low	$$BlockComment$$ `PV` - low warning limit violated
	SIG 6	Alarm - low	$$BlockComment$$ `PV` - low alarm limit violated
	SIG 7	Alarm - high	$$BlockComment$$ Limit (high) for positive gradients violated
	SIG 8	Alarm - high	$$BlockComment$$ Limit (high) for negative gradients violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You can use up to three instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 1	Alarm - low	$$BlockComment$$ Limit (low) for absolute gradients violated
	SIG 2	AS process control message - fault	$$BlockComment$$ External error has occurred
	SIG 3	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 2
	SIG 5	AS process control message - fault	$$BlockComment$$ External message 3

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance MsgEvId1

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	ExtVa104
5	ExtVa105
6	ExtVa106
7	ExtVa107
8	ExtVa108
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters ExtVa104 ... ExtVa108 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

Associated values for message instance MsgEvId2

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	ExtVa204
5	ExtVa205

Associated value	Block parameters
6	ExtVa206
7	ExtVa207
8	ExtVa208
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters ExtVa204 ... ExtVa208 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of MonAnL (Page 437)
MonAnL functions (Page 440)
MonAnL I/Os (Page 450)
MonAnL block diagram (Page 457)
MonAnL error handling (Page 447)
MonAnL modes (Page 440)

4.3.6 MonAnL I/Os

MonAnL I/Os

Input parameters

Parameter	Description	Type	Default
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypLiOp	1 = Bypass commands via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypLock	1 = Bypass activation or deactivation is locked for operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypPV	Substitution value if block is in bypass	REAL	0.0
BypPVLi	1 = Select bypass PV (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypPVOp	1 = Select bypass PV (via operator)	BOOL	0

Parameter	Description	Type	Default
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DeadBand	Width of dead band	REAL	0.0
EN	1 = Called block will be processed	BOOL	1
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
ExtVa204	Associated value 4 for messages (MsgEvID2)	ANY	
ExtVa205	Associated value 5 for messages (MsgEvID2)	ANY	
ExtVa206	Associated value 6 for messages (MsgEvID2)	ANY	
ExtVa207	Associated value 7 for messages (MsgEvID2)	ANY	
ExtVa208	Associated value 8 for messages (MsgEvID2)	ANY	
Feature	I/O for additional functions (Page 440)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
GradHUpLim	Gradient monitoring: Limit (high) for positive gradients	REAL	10.0
GradHDnLim	Gradient monitoring: Limit (high) for negative gradients	REAL	10.0
GradLLim	Gradient monitoring: Limit (low) for absolute gradients	REAL	1.0

Parameter	Description	Type	Default
GradHUpEn	1 = Activate gradient monitoring (high) for positive changes	BOOL	1
GradHDnEn	1 = Activate gradient monitoring (high) for negative changes	BOOL	1
GradLEn	1 = Activate gradient monitoring (low)	BOOL	0
GradHUpMsgEn	1 = Activate gradient (high) message for positive changes	BOOL	1
GradHDnMsgEn	1 = Activate gradient (high) message for negative changes	BOOL	1
GradLMsgEn	1 = Activate gradient (low) message	BOOL	1
LagTime	Delay time [s]	REAL	1.0
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgEvId2	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = Occupied by batch control	BOOL	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 440)	STRUCT • Bit 0: BOOL • Bit 10: BOOL • Bit 31: BOOL	- • 1 • 1 • 1
PV*	Process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_A_DC*	Delay time for incoming PV alarms for overshoot/undershoot or only for undershot process values [s]	REAL	0.0
PV_A_DG*	Delay time for outgoing PV alarms for overshoot/undershoot or only for undershot process values [s]	REAL	0.0
PV_AH_DC*	Delay time for incoming alarms for overshot process values [s]	REAL	0.0
PV_AH_DG*	Delay time for outgoing alarms for overshot process values [s]	REAL	0.0
PV_AH_En	1 = Enable PV alarm limit (high)	BOOL	1

Parameter	Description	Type	Default
PV_AH_Lim	Limit PV alarm (high)	REAL	95.0
PV_AH_MsgEn	1 = Enable PV alarm (high) message	BOOL	1
PV_AL_En	1 = Enable PV alarm limit (low)	BOOL	1
PV_AL_Lim	PV alarm limit (low)	REAL	5.0
PV_AL_MsgEn	1 = Enable PV alarm (low) message	BOOL	1
PV_Hyst*	Hysteresis for PV alarm, warning and tolerance limits	REAL	0.0
PV_OpScale	Limit for scale in PV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
PV_T_DC*	Delay time for incoming PV tolerance messages for overshoot/undershoot or only for undershot process values [s]	REAL	0.0
PV_T_DG*	Delay time for outgoing PV tolerance messages for overshoot/undershoot or only for undershot process values [s]	REAL	0.0
PV_TH_DC*	Delay time for incoming tolerance messages for overshot process values [s]	REAL	0.0
PV_TH_DG*	Delay time for outgoing tolerance messages for overshot process values [s]	REAL	0.0
PV_TH_En	1 = Enable PV tolerance limit (high)	BOOL	0
PV_TH_Lim	Limit PV tolerance message (high)	REAL	85.0
PV_TH_MsgEn	1 = Enable message for PV tolerance message (high)	BOOL	1
PV_TL_En	1 = Enable PV tolerance limit (low)	BOOL	0
PV_TL_Lim	Limit PV tolerance message (low)	REAL	15.0
PV_TL_MsgEn	1 = Activate message for PV tolerance message (low)	BOOL	1
PV_Unit	Unit of measure for process value	INT	1001
PV_W_DC*	Delay time for incoming PV warnings for overshoot/undershoot or only for undershot process values [s]	REAL	0.0
PV_W_DG*	Delay time for outgoing PV warnings for overshoot/undershoot or only for undershot process values [s]	REAL	0.0
PV_WH_DC*	Delay time for incoming warnings for overshot process values [s]	REAL	0.0
PV_WH_DG*	Delay time for outgoing warnings for overshot process values [s]	REAL	0.0
PV_WH_En	1 = Enable PV warning limit (high)	BOOL	1
PV_WH_Lim	Limit PV warning (high)	REAL	90.0
PV_WH_MsgEn	1 = Enable PV warning (high) message	BOOL	1
PV_WL_En	1 = Enable PV warning limit (low)	BOOL	1
PV_WL_Lim	Limit PV warning (low)	REAL	10.0
PV_WL_MsgEn	1 = Enable PV warning (low) message	BOOL	1

Parameter	Description	Type	Default
RstBypLi	1 = Reset bypass PV (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstBypOp	1 = Reset bypass PV (via operator)	BOOL	0
RstOp*	1 = Operator reset of the gradient's peak values	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimPV*	Process value that is used for SimOn = 1	REAL	0.0
SimPV_Li	Process value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
StepNo	Batch step number	DWORD	16#00000000
SmoothTi*	Smooth time [s] (limited to 0 to 999999)	REAL	0.0
TimeFactor	Time unit: 0 = Seconds 1 = Minutes 2 = Hours	INT	0
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UA2unit	Unit of measure for analog auxiliary value 2	INT	0
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
BypassAct	1 = Bypass is activated in this block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see MonAnL error handling (Page 447)	INT	-1
GradHUpAct	1 = Gradient alarm (high) for positive changes. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GradHDnAct	1 = Gradient alarm (high) for negative changes. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GradLAct	1 = Low gradient alarm. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgAckn2	Message acknowledgement status 2 (output ACK_STATE of second ALARM_8P)	WORD	16#0000
MsgErr1	1 = Alarm error 1 (output ERROR of the first ALARM_8P)	BOOL	0
MsgErr2	1 = Alarm error 2 (output ERROR of the second ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
MsgStat2	Message status 2 (output STATUS of second ALARM_8P)	WORD	16#0000
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
OpSt_Out	Value of the OpSt_In input parameter, for feed-forwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
PV_AH_Act	1 = PV alarm (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AH_Out	PV - alarm limit (high) output	REAL	0.0
PV_AL_Act	1 = PV alarm (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AL_Out	PV - alarm limit (low) output	REAL	0.0
PV_Grad	Gradient value.	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_GradPP	Gradient maximum peak value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_GradNP	Gradient minimum peak value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_HysOut	Hysteresis for PV alarm, output	REAL	0.0
PV_Out	Output for process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_TH_Act	1 = PV tolerance message (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_TH_Out	PV tolerance limit (high) output	REAL	0.0
PV_TL_Act	1 = PV tolerance message (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_TL_Out	PV tolerance limit (low) output	REAL	0.0

Parameter	Description	Type	Default
PV_WH_Act	1 = PV warning (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_WH_Out	Limit PV warning (high) output	REAL	0.0
PV_WL_Act	1 = PV warning (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_WL_Out	Limit PV warning (low) output	REAL	0.0
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 437)	DWORD	16#00000000
Status2	Status word 2 (Page 437)	DWORD	16#00000000
Status3	Status word 3 (Page 437)	DWORD	16#00000000
SumMsgAct	1 = Group message is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

MonAnL messaging (Page 448)

MonAnL block diagram (Page 457)

MonAnL modes (Page 440)

4.3.7 MonAnL block diagram

MonAnL block diagram

A block diagram is not provided for this block.

See also

MonAnL I/Os (Page 450)

MonAnL messaging (Page 448)

MonAnL error handling (Page 447)

MonAnL functions (Page 440)

MonAnL modes (Page 440)

Description of MonAnL (Page 437)

4.3.8 Operator control and monitoring

4.3.8.1 MonAnL views

Views of the MonAnL block

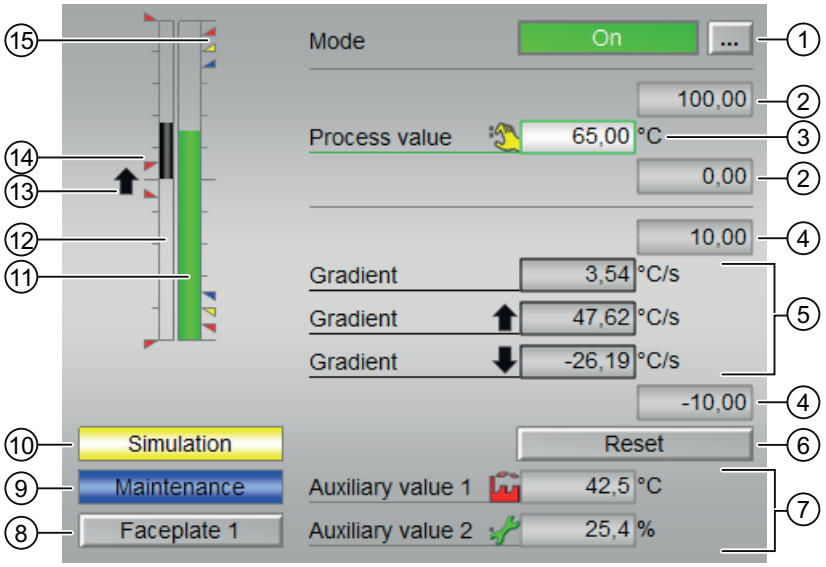
The block MonAnL provides the following views:

- MonAnL standard view (Page 458)
- Alarm view (Page 296)
- MonAnL limit view (Page 461)
- Trend view (Page 299)
- MonAnL parameter view (Page 463)
- MonAnL preview (Page 464)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MonAnL (Page 465)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

4.3.8.2 MonAnL standard view

MonAnL standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) High and low scale range for the process value

These values provide information on the display range for the bar graph of the process value. The scale range is defined in the engineering system.

(3) Display of the process value including signal status

This area shows the current process value with the corresponding signal status.

If text is configured for this command, it is displayed as additional text and button label for command selection. Additional information is available in the section Labeling of buttons and text (Page 205).

You can change the text for the process value with the `PV_Out` parameter.

(4) High and low scale range for the gradient value

These values provide information on the display range for the bar graph of the gradient. The scale range corresponds to 10% of the scale range for the process value: For example, once you have specified a process value scale range of 0 to 100, the scale range of the gradient will be automatically set to a value between -10 and 10.

The current gradient value is displayed when one of the following monitoring functions is activated:

- Gradient monitoring for positive changes `(GradHUpEn = 1)`
- Gradient monitoring for negative changes `(GradHDnEn = 1)`
- Gradient monitoring `(GradLEn = 1)`

(5) Display of the gradient

This area shows the current, minimum and maximum gradient value and the rise and fall of the value. This display of the minimum and maximum gradient value functions like a min/max pointer.

The current gradient value is displayed when one of the following monitoring functions is activated:

- Gradient monitoring for positive changes `(GradHUpEn = 1)`
- Gradient monitoring for negative changes `(GradHDnEn = 1)`
- Gradient monitoring `(GradLEn = 1)`

The maximum peak gradient value is displayed when the gradient monitoring is activated for positive changes `(GradHUpEn = 1)`

The minimum peak gradient value is displayed when the gradient monitoring is activated for negative changes `(GradHDnEn = 1)`

(6) Resetting the peak values of the gradient

You can use this button to reset the maximum or minimum peak value of the gradient (`PV_GradPP` and `PV_GradNP` output parameters).

The button is displayed when gradient monitoring is activated for positive `(GradHUpEn = 1)` or negative changes `(GradHDnEn = 1)`.

(7) Display of auxiliary values

You can use this area to display two auxiliary values that have been configured in the engineering system. You can find additional information on this in the Displaying auxiliary values (Page 207) section.

(8) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the Opening additional faceplates (Page 203) section.

(9) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

(10) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

(11) Bar graph for the "process value"

This area shows the current "Process value" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(12) Bar graph for the gradient

This area shows the current gradient value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

The bar graph is displayed when gradient monitoring is activated for positive `(GradHUpEn = 1)` or negative changes `(GradHDnEn = 1)`.

(13) Display of the gradient

This display indicates the movement of the gradient up (↑) or down (↓).

Gradient monitoring is displayed when the gradient value PV_Grad ≠ 0 and one of the following monitoring functions is activated:

- Gradient monitoring for positive changes `(GradHUpEn = 1)`
- Gradient monitoring for negative changes `(GradHDnEn = 1)`
- Gradient monitoring `(GradLEn = 1)`

(14) Display of limits in the bar graph

This area shows you the specified limits. Refer to the MonAnL limit view (Page 461) section for more on this.

The display only appears when the bar for the gradients is also displayed.

(15) Limit display

These colored triangles show you the specified limits in the respective bar graph.

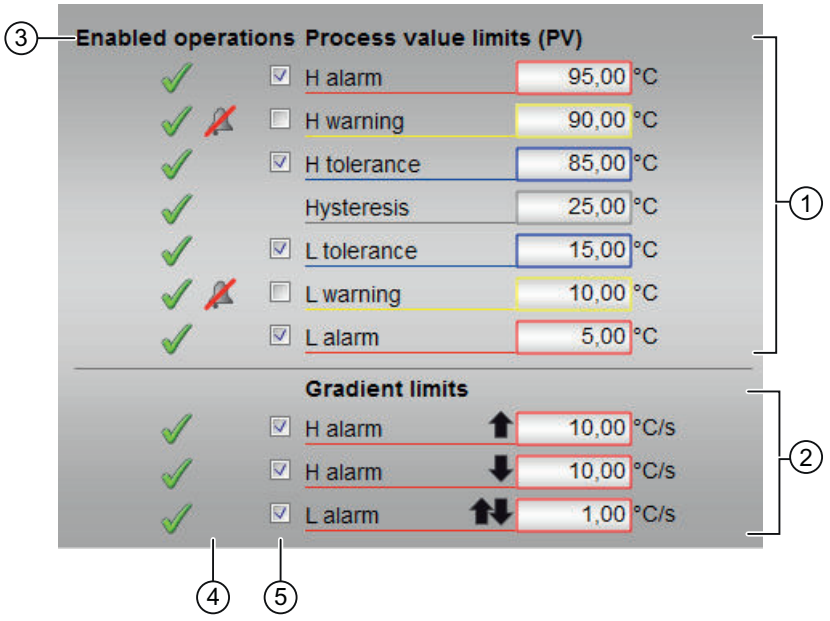
4.3.8.3 MonAnL limit view

Limit view of MonAnL

Several values are set in this view by default:

- Process value limits
- Gradient limits

The toolbars of the faceplate and the block icon indicate when the limits are reached or violated.



(1) Process value limits

In this area, you can enter the limits for the process value. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "H warning": Warning high
- "H tolerance": Tolerance high
- "Hysteresis"
- "L tolerance": Tolerance low
- "L warning": Warning low
- "L alarm": Alarm low

(2) Gradient limits

You can enter the gradient limits in this area. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm ⇑": Gradient for the high slope for positive changes
- "H alarm ⇓": Gradient for the high slope for negative changes
- "L alarm ⇑⇓": Gradient for the low slope (absolute)

(3) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

(4) Message suppression / delays

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

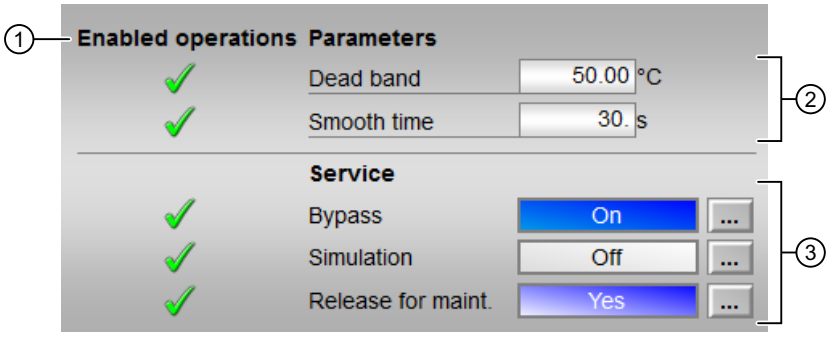
Alarm delays are also displayed in this position; for more on this see section Area of application for alarm delays (Page 195).

(5) Suppress messages

You can enable / disable messages by setting the check mark.

4.3.8.4 MonAnL parameter view

Parameter view of MonAnL



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

(2) Parameter

You can change the following parameter in this area:

- "Dead band"
- "Smooth time"

You can find more information about this in the section Changing values (Page 253).

(3) Service

You can select the following functions in this area:

- "Bypass"
- "Simulation"
- "Release for maintenance"

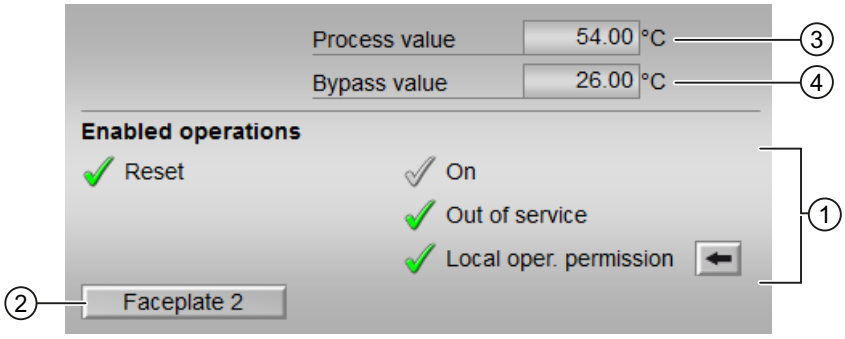
Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Bypassing signals (Page 107)
- Simulating signals (Page 58)
- Release for maintenance (Page 64)

4.3.8.5 MonAnL preview

Preview of MonAnL



(1) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- Reset: You can reset the peak value of the gradient.
- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248) .

(2) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

(3) Process value

This area displays the real process value (`PV`).

(4) Bypass value

This area displays the bypass value (`BypPV`).

4.3.8.6 Block icon for MonAnL

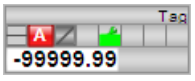
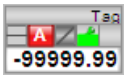
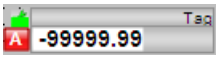
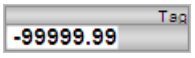
Block symbols for MonAnL

A variety of block symbols are available with the following functions:

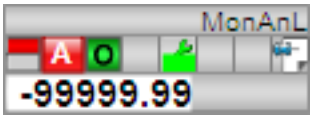
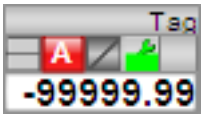
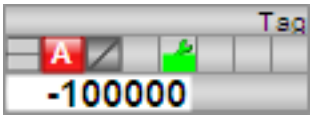
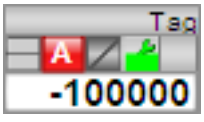
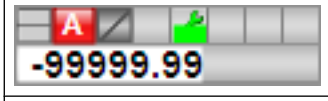
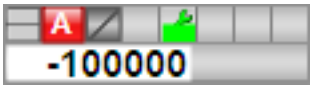
- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes

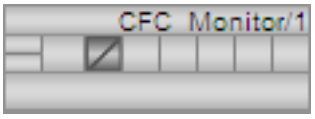
- Signal status, release for maintenance
- Memo display
- Process value

The block symbols from template @TemplateAPLV8.PDL:

Symbols	Selection of the block symbol in CFC	Special features
	1	
	2	
	3	
	4	

The block symbols from template @TemplateAPLV7.PDL:

Symbols	Selection of the block symbol in CFC	Special features
	1	Block symbol in the full display
	2	
	3	
	4	
	5	
	6	
	7	

Symbols	Selection of the block symbol in CFC	Special features
	8	
	-	Block symbol in "Out of service" mode (example with block symbol type 1)

Additional information on the block symbol and the control options in the block symbol is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

4.4 MonAnS - Monitoring of an analog process tag (Small)

4.4.1 Description of MonAnS

Object name (type + number)

Type + number: FB 1912

Family: Monitor

Area of application for MonAnS

The block is used for the following fields of application

- Monitoring an analog process value

Note

This block is also available as a large block. A comparison of the MonAnL and MonAnS blocks is available in the section: MonAnL compared to MonAnS (Page 423)

How it works

The MonAnS block is used to monitor an analog process tag and the corresponding limits. The block generates and outputs messages if any violation of limits is detected.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section MonAnS I/Os (Page 476).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`

Status bit	Parameter
3	OosAct.Value
4	OosLi.Value
5	Not used
6	OnAct.Value
7 - 10	Not used
11	Delay of the PV_AH_Lim message
12	Delay of the PV_WH_Lim message
13 - 14	Not used
15	Delay of the PV_WL_Lim message
16	Delay of the PV_AL_Lim message
17	Collection of message delays
18 - 31	Not used

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock
1	PV_AH_Act.Value
2	PV_WH_Act.Value
3 - 4	Not used
5	PV_WL_Act.Value
6	PV_AL_Act.Value
7	PV_AH_En
8	PV_WH_En
9 - 10	Not used
11	PV_WL_En
12	PV_AL_En
13	PV_AH_MsgEn
14	PV_WH_MsgEn
15 - 16	Not used
17	PV_WL_MsgEn
18	PV_AL_MsgEn
19 - 30	Not used
31	MS_RelOp

See also

MonAnS operating modes (Page 470)

MonAnS functions (Page 470)

MonAnS error handling (Page 473)

MonAnS messaging (Page 474)

MonAnS block diagram (Page 479)

4.4.2 MonAnS operating modes

MonAnS operating modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of MonAnS (Page 468)

MonAnS functions (Page 470)

MonAnS error handling (Page 473)

MonAnS messaging (Page 474)

MonAnS I/Os (Page 476)

MonAnS block diagram (Page 479)

4.4.3 MonAnS functions

Functions of MonAnS

The functions for this block are listed below.

Alarm delays with one time value per limit pair

This block includes the standard function alarm delay for One time value per limit pair (Page 196).

Limit monitoring of the process value

This block provides the standard function Limit monitoring of the process value (Page 86).

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block has the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `PV_Out.ST`

Dead band

To suppress values that fluctuate around the zero point, this block has the standard function Dead band (Page 61).

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Simulating signals

This block provides the standard function Simulating signals (Page 58).

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)

Bit	Function
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4 - 9	Not used
10	1 = Operator can change the simulation value `SimPV`
11	1 = Operator can activate the Simulation function
12	1 = Operator can activate the Release for maintenance function
13	1 = Operator can change the limit (`PV`) for high alarm
14	1 = Operator can change the limit (`PV`) for high warning
15	Not used
16	1 = Operator can change the limit (`PV`) for hysteresis
17	1 = Operator can change the limit (`PV`) for low alarm
18	1 = Operator can change the limit (`PV`) for low warning
19 - 22	Not used
23	1 = Operator can change the dead band parameter `DeadBand`
24	1 = Operator can activate / deactivate messages via `PV_AH_MsgEn`
25	1 = Operator can activate / deactivate messages via `PV_AL_MsgEn`
26 -27	Not used
28	1 = Operator can activate / deactivate messages via `PV_WH_MsgEn`
29	1 = Operator can activate / deactivate messages via `PV_WL_MsgEn`
30 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Specifying the display area for process and setpoint values as well as operations

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of MonAnS (Page 468)

MonAnS operating modes (Page 470)

MonAnS error handling (Page 473)

MonAnS messaging (Page 474)

MonAnS I/Os (Page 476)

MonAnS block diagram (Page 479)

4.4.4 MonAnS error handling

Error handling of MonAnS

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following error message can be generated at this block:

- Error numbers
- Control system fault (`CSF`)

Overview of error numbers

The `ErrorNum` I/O can be used to output various error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.
30	The value of `PV` can no longer be displayed in the REAL number field.

Control system fault (CSF)

An external signal can be activated via the CSF input. If this signal changes to = 1, a control system fault is triggered. Refer to the Error handling (Page 119) section for more on this.

See also

Description of MonAnS (Page 468)

MonAnS operating modes (Page 470)

MonAnS functions (Page 470)

MonAnS messaging (Page 474)

MonAnS I/Os (Page 476)

MonAnS block diagram (Page 479)

4.4.5 MonAnS messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

You can interconnect an external fault (signal) to input parameter CSF. If it changes to CSF = 1, a process control fault is triggered (MsgEvId1, SIG 5).

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	Alarm - high	$$BlockComment$$ PV - high alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ PV - high warning limit violated
	SIG 3	Warning - low	$$BlockComment$$ PV - low warning limit violated
	SIG 4	Alarm - low	$$BlockComment$$ PV - low alarm limit violated
	SIG 5	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You have the option to use two instance-specific messages for this block.

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 6	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 7	AS process control message - fault	$$BlockComment$$ External message 2

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance **MsgEvId1**

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	ExtVa104
5	ExtVa105
6	Reserved
7	Reserved
8	Reserved
9	Reserved
10	Reserved

The associated values 4 ... 5 are allocated to the parameters `ExtVa104` ... `ExtVa105` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of MonAnS (Page 468)
MonAnS operating modes (Page 470)
MonAnS functions (Page 470)
MonAnS error handling (Page 473)
MonAnS I/Os (Page 476)
MonAnS block diagram (Page 479)

4.4.6 MonAnS I/Os

MonAnS I/Os

Input parameters

Parameter	Description	Type	Default
`BatchEn`	1 = Enable allocation	BOOL	0
`BatchID`	Batch ID	DWORD	16#00000000
`BatchName`	Batch name	S7-String	
`CSF`	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`DeadBand`	Width of dead band	REAL	0.0
`EN`	1 = Called block will be processed	BOOL	1
`ExtMsg1`	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ExtMsg2`	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ExtVa104`	Associated value 4 for messages (`MsgEvID1`)	ANY	
`ExtVa105`	Associated value 5 for messages (`MsgEvID1`)	ANY	
`Feature`	I/O for additional MonAnS functions (Page 470)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`MS_RelOp*`	1 = Release for maintenance by OS operator	BOOL	0

Parameter	Description	Type	Default
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = Occupied by batch control	BOOL	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, Description of OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for MonAnS functions (Page 470)	STRUCT • Bit 0: BOOL • Bit 10: BOOL • Bit 31: BOOL	- • 1 • 1 • 1
PV*	Process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_A_DC*	Delay time for incoming PV alarms [s]	REAL	0.0
PV_AH_En	1 = Enable PV alarm limit (high)	BOOL	1
PV_AH_Lim	Limit PV alarm (high)	REAL	95.0
PV_AH_MsgEn	1 = Enable PV alarm (high) message	BOOL	1
PV_AL_En	1 = Enable PV alarm limit (low)	BOOL	1
PV_AL_Lim	PV alarm limit (low)	REAL	5.0
PV_AL_MsgEn	1 = Enable PV alarm (low) message	BOOL	1
PV_Hyst*	Hysteresis for PV alarm, warning and tolerance limits	REAL	0.0
PV_OpScale	Limit for scale in PV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
PV_Unit	Unit of measure for process value	INT	1001
PV_W_DC*	Delay time for incoming PV warnings [s]	REAL	0.0
PV_WH_En	1 = Enable PV warning limit (high)	BOOL	1
PV_WH_Lim	Limit PV warning (high)	REAL	90.0
PV_WH_MsgEn	1 = Enable PV warning (high) message	BOOL	1
PV_WL_En	1 = Enable PV warning limit (low)	BOOL	1
PV_WL_Lim	Limit PV warning (low)	REAL	10.0
PV_WL_MsgEn	1 = Enable PV warning (low) message	BOOL	1

Parameter	Description	Type	Default
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an Opening additional faceplates (Page 203) in the standard view	ANY	-
SimOn	1 = Simulation on	BOOL	0
SimPV*	Process value used for SimOn = 1	REAL	0.0
StepNo	Batch step number	DWORD	16#00000000
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see MonAnS error handling (Page 473)	INT	-1
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
PV_AH_Act	1 = PV alarm (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AL_Act	1 = PV alarm (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
PV_Out	Output for process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_WH_Act	1 = PV warning (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_WL_Act	1 = PV warning (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Description of MonAnS (Page 468)	DWORD	16#00000000
Status2	Description of MonAnS (Page 468)	DWORD	16#00000000
SumMsgAct	1 = Group message is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

MonAnS operating modes (Page 470)

MonAnS messaging (Page 474)

MonAnS block diagram (Page 479)

4.4.7 MonAnS block diagram

MonAnS block diagram

A block diagram is not provided for this block.

See also

Description of MonAnS (Page 468)

MonAnS operating modes (Page 470)

MonAnS functions (Page 470)

MonAnS error handling (Page 473)

MonAnS messaging (Page 474)

MonAnS I/Os (Page 476)

4.4.8 Operator control and monitoring

4.4.8.1 MonAnS views

Views of the MonAnS block

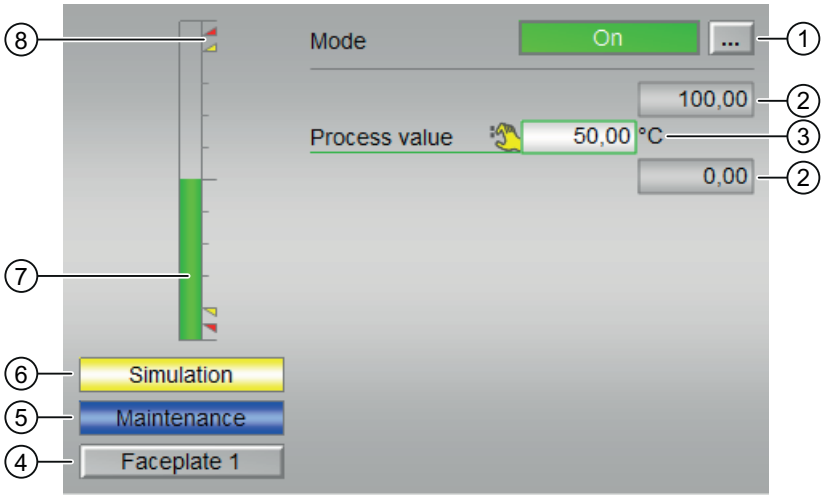
The block MonAnS provides the following views:

- MonAnS standard view (Page 480)
- Alarm view (Page 296)
- MonAnS limit view (Page 482)
- Trend view (Page 299)
- MonAnS parameter view (Page 483)
- MonAnS preview (Page 484)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MonAnS (Page 485)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

4.4.8.2 MonAnS standard view

MonAnS standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) High and low scale range for the process value

These values provide information on the display range for the bar graph of the process value. The scale range is defined in the engineering system.

(3) Display of the process value including signal status

This area shows the current process value with the corresponding signal status.

If text is configured for this command, it is displayed as additional text and button label for command selection. You can find more information about this in the section Labeling of buttons and text (Page 205).

You can change the text for the process value with the `PV_Out` parameter.

(4) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

Additional information is available in the section Opening additional faceplates (Page 203).

(5) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

(6) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

(7) Bar graph for the "process value"

This area shows the current "Process value" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(8) Limit display

These colored triangles show you the configured limits in the respective bar graph.

See also

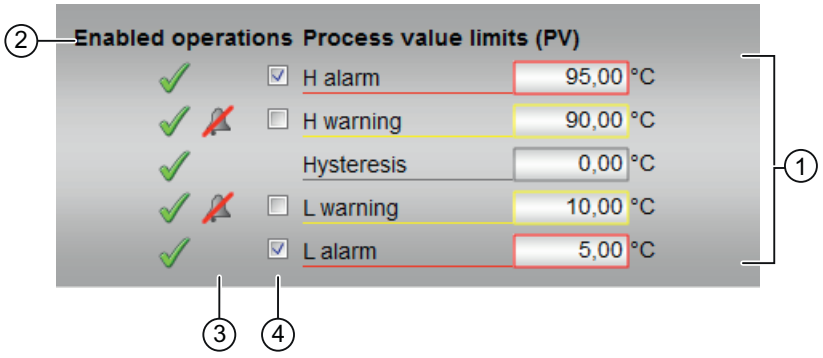
Displaying auxiliary values (Page 207)

4.4.8.3 MonAnS limit view

Limit view of MonAnS

You can specify the process value limits in this view:

The toolbars of the faceplate and the block icon indicate when the limits are reached or violated.



(1) Process value limits

In this area, you can enter the limits for the process value. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "H warning": Warning high
- "Hysteresis"
- "L warning": Warning low
- "L alarm": Alarm low

(2) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

(3) Message suppression / delay

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

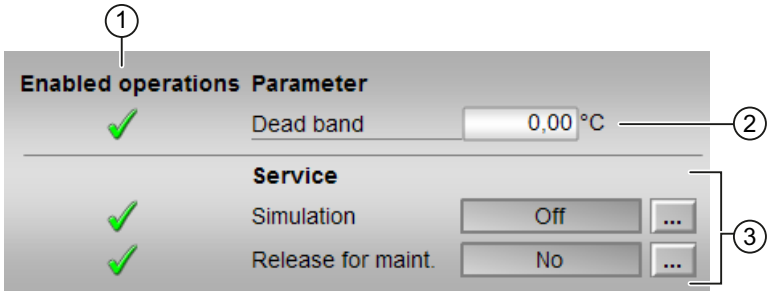
Alarm delays are also displayed in this position; for more on this see section Area of application for alarm delays (Page 195).

(4) Suppress messages

You can enable / disable messages by setting the check mark.

4.4.8.4 MonAnS parameter view

Parameter view of MonAnS



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

(2)Parameter

You can change parameters in this area. Refer to the section Changing values (Page 253).

You can influence the following parameters:

- "Dead band": Width of dead band

(3)Service

You can select the following functions in this area:

- "Simulation"
- "Release for maintenance"

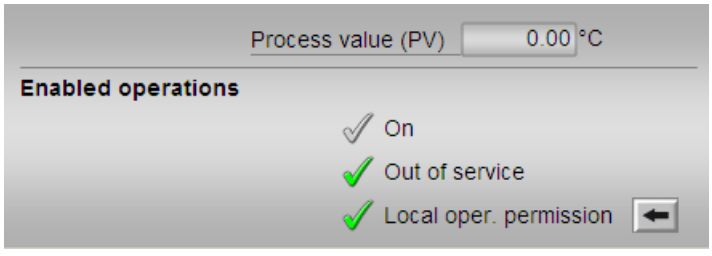
Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Simulating signals (Page 58)
- Release for maintenance (Page 64)

4.4.8.5 MonAnS preview

Preview of MonAnS



Process value

This area displays the real process value (PV).

Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

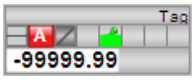
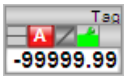
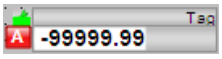
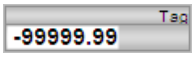
4.4.8.6 Block icon for MonAnS

Block symbols for MonAnS

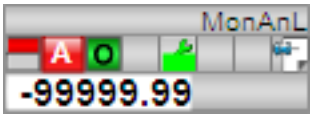
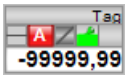
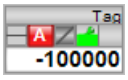
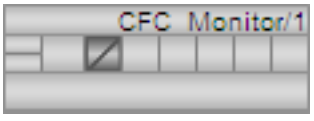
A variety of block symbols are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes
- Signal status, release for maintenance
- Memo display
- Process value

The block symbols from template @TemplateAPLV8.PDL:

Symbols	Selection of the block symbol in CFC	Special features
	1	
	2	
	3	
	4	

The block symbols from template @TemplateAPLV7.PDL:

Symbols	Selection of the block symbol in CFC	Special features
	1	Block symbol in the full display
	2	
	3	
	4	
	-	Block symbol in "Out of service" mode (example with type 1 block symbol)

Additional information on the block symbol and the control options in the block symbol is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

4.5 MonDiL - Monitoring of a digital process tag (Large)

4.5.1 Description of MonDiL

Object name (type + number) and family

Type + number: FB 1848

Family: Monitor

Area of application for MonDiL

The block is used for the following applications:

- Monitoring a digital process tag

Note

This block is also available as a small block. A comparison of the MonDiL and MonDiS blocks is available in the section: MonDiL compared to MonDiS (Page 425)

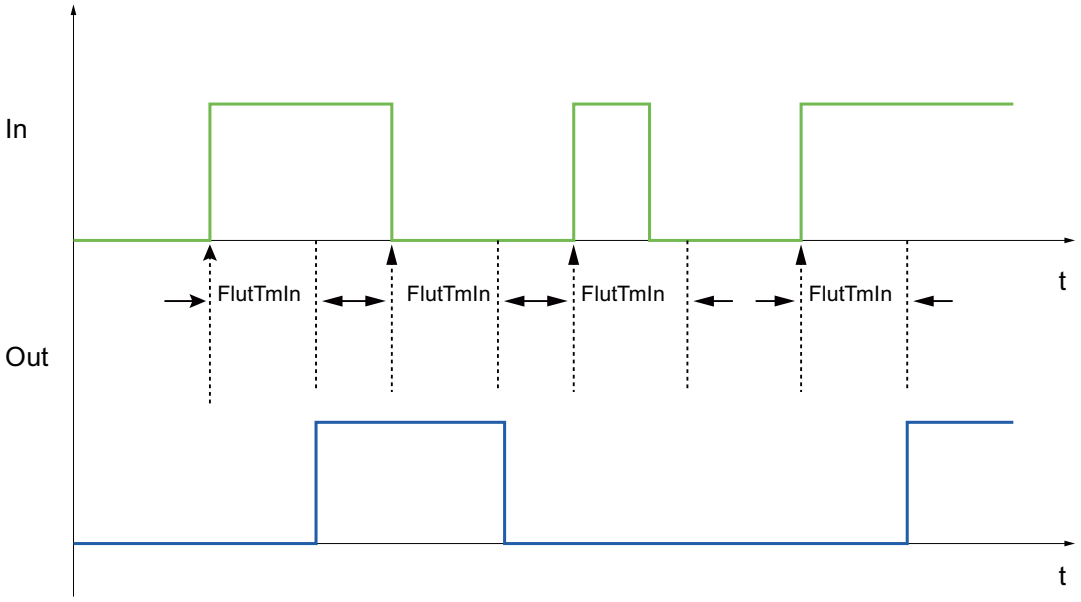
How it works

The MonDiL block is used to monitor a digital process tag with flutter suppression. The block reports excess flutter signals which are generated within a defined period.

The digital value to be monitored is interconnected to the `In` input parameter. Every time there is a signal change (1 - 0 or 0 - 1) the configurable timer (`FlutTmIn`) for flutter suppression is started as shown in the figure below.

When the time you have specified expires and no single change occurs, the input signal is written to the `Out` output parameter.

Set the time in the timer (`FlutTmIn`) to 0 seconds; the input signal is written directly to the output.



Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the MonDiL block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Monitoring of a digital process tag (DigitalMonitoring) (Page 2327)
- Monitoring a digital process tag for PA/FF devices (DigitalMonitoring_Fb) (Page 2328)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block. The `Out` and `FlutAct` parameters are affected.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section MonDiL I/Os (Page 498).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`BypassAct.Value`

Status bit	Parameter
6	OnAct.Value
7	Out.Value
8	AlmMsgEn
9	Display of BypassAct.Value in faceplate (display and operator controls) and block icon
10	SimLiOp.Value
11 - 29	Not used
30	Auxiliary value 1 is visible
31	Auxiliary value 2 is visible

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock.Value
1	FlutAct.Value
2	FlutEn
3	FlutMsgEn
4 - 30	Not used
31	MS_RelOp

Status word allocation for Status3 parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via EventTsIn
1	Effective signal 2 of the message block connected via EventTsIn
2	Effective signal 3 of the message block connected via EventTsIn
3	Effective signal 4 of the message block connected via EventTsIn
4	Effective signal 5 of the message block connected via EventTsIn
5	Effective signal 6 of the message block connected via EventTsIn
6	Effective signal 7 of the message block connected via EventTsIn
7	Effective signal 8 of the message block connected via EventTsIn
8 - 13	Not used
14	Color = 11 (Background color is derived directly from EventState, without message classes process message and operator prompt), only for block icon type 5
15	Color = 10 (Background color of the message instance MsgEvId1 - Message identifier SIG1)
16-19	Not used
20	Color = 6 (Violet, operator prompt) and Out = 1
21	Not used
22	Color = 5 (Pastel green, process) and Out = 1
23 - 24	Not used
25	Color = 4 (Black, control system) and Out = 1
26	Color = 9 (UCMC 6) and Out = 1

Status bit	Parameter
27	Color = 3 (Blue, tolerance / UCMC 5) and `Out = 1`
28	Color = 8 (UCMC 4) and `Out = 1`
29	Color = 2 (Yellow, warning UCMC 3) and `Out = 1`
30	Color = 7 (UCMC 2) and `Out = 1`
31	Color = 1 (Red, alarm / UCMC 1) and `Out = 1`

Explanation:
UCMC = user-configured messages classes

Note

The user-configured message classes 5, 3 and 1 are displayed in the existing status bits 27, 29 and 31.

Status word allocation for `Status4` parameter

Status bit	Parameter
0 - 15	Effective signal 9..16 of the message block connected via `EventTsIn`
16 - 31	Not used

See also

MonDiL functions (Page 491)
MonDiL messaging (Page 497)
MonDiL block diagram (Page 503)
MonDiL error handling (Page 496)
MonDiL modes (Page 490)

4.5.2 MonDiL modes

MonDiL operating modes

This block provides the following modes.

- On (Page 71)
- Out of service (Page 71)

"On"

You can find general information about the "On" mode in the On (Page 71) section.

Out of service

You can find general information about the "Out of service" mode in the Out of service (Page 71) section.

See also

MonDiL block diagram (Page 503)

MonDiL I/Os (Page 498)

MonDiL messaging (Page 497)

MonDiL error handling (Page 496)

MonDiL functions (Page 491)

Description of MonDiL (Page 487)

4.5.3 MonDiL functions

Functions of MonDiL

The functions for this block are listed below.

Suppression and reporting of signal flutter

The block is operated as "flutter filter". The block receives digital signals at the `In` input parameter and ideally these do not develop any flutter. The block monitors this if you activate the function via the `FlutEn` = 1 input parameter.

Use the `FlutTmIn` input parameter to set how long a continuous signal should last in order to be transferred to the process without flutter.

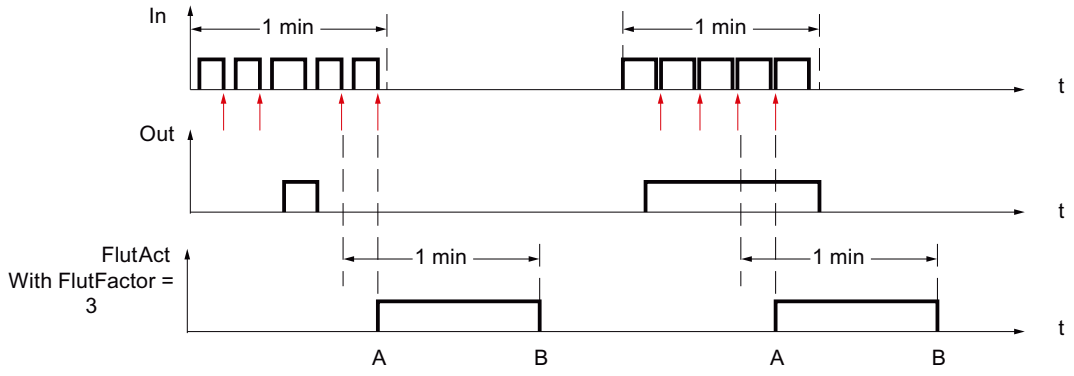
The preprocessed signal is sent to the process via the `Out` output parameter.

The block can be set up to report signal flutter. To do this, set parameter `FlutMsgEn` = 1. Monitoring starts at the next 0 - 1 - 0 edge transition at the input signal `In`.

Use the `FlutFactor` input parameter to specify the maximum number of signals to be filtered per minute by the block. If this maximum value is exceeded, this is indicated at the output parameter `FlutAct` by a 1. The message (`FlutAct` = 0). is cleared after the maximum value has dropped again by more than half.

Example of the flutter suppression

You can use the `FlutFactor` input parameter to limit flutter signals to 3 per minute.



Case A:

Four flutter signals have occurred within a minute. `FlutFactor` is set to three, in other words, a maximum of three flutter signal per minute are allowed, the `FlutAct` output is set to one.

Case B:

Only one flutter signal has occurred within a minute, which corresponds to less than half the `FlutFactor`. The `FlutAct` output is reset.

Delaying the on and off switching functions

You set delay times for setting the output using the input parameter `Out_DC` or `Out_DG`:

- `Out_DC`: delay time [s] for rising edges (0 - 1 edge)
- `Out_DG`: delay time [s] for falling edge (1 - 0 edge)

The `Out` output parameter is after expiration of the delay time.

You disable this function if you set the value of the respective parameter to 0 seconds.

Adapting the color representation in the configured message class

You can set the background color of instance-specific texts for the block icons 3 to 6 and thereby adapt them to the configured message class. This background color is displayed when the output parameter `Out` = 1.

You set the color coding with the `Color` parameter:

Value `Color`	Color
0	Old reaction
1	Red (alarm)
2	Yellow (warning)
3	Blue (tolerance)
4	Black (control system)
5	Pastel green (process)
6	Violet (operator prompt)

Value `Color`	Color
7...9	Old reaction
10	Background color of the message instance MsgEvId1 - Message identifier SIG1
11	For block icon type 5 only, background color is derived directly from EventState without message classes process message and operator prompt [1]

[1] This also takes into consideration the CSF input

The following colors are shown for user-configured message classes

Value `Color`	Color
0	Old reaction
1	User-configured message classes 1
2	User-configured message classes 3
3	User-configured message classes 5
4	Black (control system)
5	Pastel green (process)
6	Violet (operator prompt)
7	User-configured message classes 2
8	User-configured message classes 4
9	User-configured message classes 6
10	Background color of the message instance MsgEvId1 - Message identifier SIG1
11	For block icon type 5 only, background color is derived directly from EventState without message classes process message and operator prompt [1]

[1] This also takes into consideration the CSF input

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

Changing labels on buttons and text

This block provides the standard function Labeling of buttons and text (Page 205).

You can change the label for "Process value" for this block as you please. The change is made with the `FlutTmIn` parameter.

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Group display `SumMsgAct` for limit monitoring, `CSF` and `ExtMsgx`

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `In.ST`

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Simulating signals

This block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Value (`SimIn`, `SimInLi`)

Bypass function

This block provides the standard function Bypassing signals (Page 107).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
9	Substitution value is active if the block is in bypass (Page 184)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the OS_Perm parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can change the delay time for rising edges (Out_DC)
5	1 = Operator can change the delay time for falling edges (Out_DG)
6 - 7	Not used
8	1 = Operator can activate bypass functionality
9	1 = Operator can deactivate bypass functionality
10	1 =Operator can change the simulation value SimIn
11	1 = Operator can activate the Simulation function
12	1 = Operator can activate the Release for maintenance function
13	Not used
14	1 = The user can input the duration for flutter suppression.
15	1 = The user can specify the number of flutter signals that should be suppressed.
16 - 31	Not used

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter EventTsOut of the block EventTs or Event16Ts is connected to the input parameter EventTsIn of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of MonDiL (Page 487)

MonDiL messaging (Page 497)

MonDiL I/Os (Page 498)

MonDiL block diagram (Page 503)

MonDiL error handling (Page 496)

MonDiL modes (Page 490)

EventTs functions (Page 1634)

Reaction of the switching points in the "Out of service" operating mode (Page 175)

User-configured message classes (Page 41)

4.5.4 MonDiL error handling

Error handling of MonDiL

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Control system fault (CSF)
- Flutter alarm

Overview of error numbers

The ErrorNum I/O can be used to output various error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.
11	FlutTmIn < 0
16	FlutFactor is -ve or > 100

Control system fault (CSF)

An external signal can be activated via the CSF input. If this signal changes to = 1, a control system fault is triggered. Refer to the Error handling (Page 119) section for more on this.

Flutter alarm

An alarm is output at output FlutAct with 1 if signal flutter is detected. Refer to the functions of the block > Error handling (Page 119).

See also

MonDiL block diagram (Page 503)
MonDiL I/Os (Page 498)
MonDiL messaging (Page 497)
MonDiL functions (Page 491)
MonDiL modes (Page 490)
Description of MonDiL (Page 487)

4.5.5 MonDiL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a process control fault is triggered (`MsgEvId1`, SIG 3).

Process messages

You can use up to four instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 1	Alarm - high	$$BlockComment$$ `Out` - Binary value set
	SIG 2	Warning - high	$$BlockComment$$ Flutter limits violated
	SIG 3	AS process control message - fault	$$BlockComment$$ External error has occurred
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 5	AS process control message - fault	$$BlockComment$$ External message 2
	SIG 6	AS process control message - fault	$$BlockComment$$ External message 3

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`ExtVa104`
5	`ExtVa105`
6	`ExtVa106`
7	`ExtVa107`
8	`ExtVa108`
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters `ExtVa104` ... `ExtVa108` and can be used. Additional information is available in the "Process Control System PCS7 - Engineering System" manual.

See also

Description of MonDiL (Page 487)
MonDiL functions (Page 491)
MonDiL I/Os (Page 498)
MonDiL block diagram (Page 503)
MonDiL error handling (Page 496)
MonDiL modes (Page 490)
Time stamp (Page 201)

4.5.6 MonDiL I/Os

I/Os of MonDiL

Input parameters

Parameter	Description	Type	Default
`AlmMsgEn`	1 = Alarms are output	BOOL	1
`BatchEn`	1 = Enable allocation	BOOL	0
`BatchID`	Batch ID	DWORD	16#00000000
`BatchName`	Batch name	S7-String	

Parameter	Description	Type	Default
BypLiOp	1 = Bypass commands via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypLock	1 = Bypass activation or deactivation is locked for operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypIn	Substitution value if block is in bypass	REAL	0.0
BypInLi	1 = Select bypass In (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypInOp	1 = Select bypass In (via operator)	BOOL	0
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Color	Display for status display in the block icon: 0 = Default setting 1 = Red (alarm) 2 = Yellow (warning) 3 = Blue (tolerance) 4 = Black (control system message) 5 = Pastel green (process message) 6 = Violet (operator prompt)	BYTE	16#00
EN	1 = Called block will be processed	BOOL	1
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	

Parameter	Description	Type	Default
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
Feature	I/O for additional functions (Page 491)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
FlutEn	1 = Flutter suppression on	BOOL	1
FlutFactor	Number of flutter signals that can be suppressed	INT	2
FlutMsgEn	1 = Signal flutter is reported if the period of the signal flutter is < FlutTmIn.	BOOL	1
FlutTmIn	Period during which signal flutter is suppressed.	REAL	0.0
In	Digital input value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = Occupied by batch control	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnOp*	1 = "On" mode via operator	BOOL	0
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 491)	STRUCT • Bit 0: BOOL • Bit 10: BOOL • Bit 31: BOOL	- • 1 • 1 • 1
Out_DC	Delay time for setting the output parameter Out for rising edges	REAL	0.0
Out_DG	Delay time for setting the output parameter Out for falling edges	REAL	0.0
RstBypLi	1 = Reset bypass In (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstBypOp	1 = Reset bypass In (via operator)	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3

Parameter	Description	Type	Default
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimIn*	Value used for SimOn = 1	BOOL	0
SimInLi	Value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StepNo	Batch step number	DWORD	16#00000000
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA2unit	Unit of measure for analog auxiliary value 2	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
BypassAct	1 = Bypass is activated in this block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see MonDiL error handling (Page 496)	INT	-1
FlutAct	1 = Suppresses flutter signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	1 = Alarm error 1 (output ERROR of the first ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feed-forwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 487)	DWORD	16#00000000
Status2	Status word 2 (Page 487)	DWORD	16#00000000
Status3	Status word 3 (Page 487)	DWORD	16#00000000
SumMsgAct	1 = Group message is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

MonDiL messaging (Page 497)

MonDiL block diagram (Page 503)

MonDiL modes (Page 490)

4.5.7 MonDiL block diagram

MonDiL block diagram

A block diagram is not provided for this block.

See also

MonDiL I/Os (Page 498)

MonDiL messaging (Page 497)

MonDiL error handling (Page 496)

MonDiL functions (Page 491)

MonDiL modes (Page 490)

Description of MonDiL (Page 487)

4.5.8 Operator control and monitoring

4.5.8.1 MonDiL views

Views of the MonDiL block

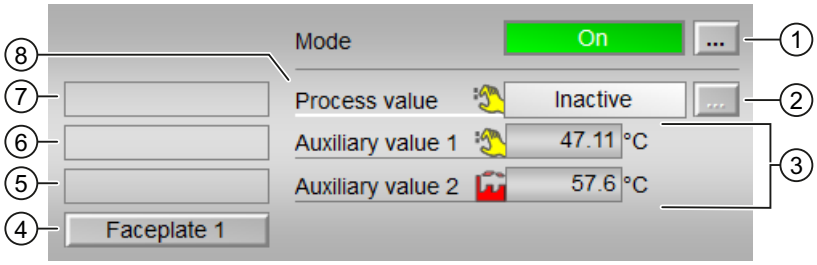
The block MonDiL provides the following views:

- MonDiL standard view (Page 504)
- Alarm view (Page 296)
- Trend view (Page 299)
- MonDiL parameter view (Page 506)
- MonDiL preview (Page 507)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MonDiL (Page 508)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

4.5.8.2 MonDiL standard view

MonDiL standard view



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to chapter Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Display of process value enabled/disabled

This area shows you the status of the individual connected parameters.

The identifier can be changed using Text 0 / Text 1 at the output parameter `Out`.

If the block is in simulation, you can enable or disable the process value. To do this, click on the display to open the operator input area.

If text is configure for these commands, it is displayed as additional text and as button labels for command selection. You can find additional information on this in chapter Labeling of buttons and text (Page 205).

You can change the text for the process value with the `FlutTmIn` parameter.

The background color of the display can be changed using the "Color" parameter; see MonDiL functions (Page 491) under "Adapting the color representation in the configured message class".

(3) Display of auxiliary values

You can use this area to display two auxiliary values that have been configured in the Engineering System. You can find additional information on this in chapter Displaying auxiliary values (Page 207).

(4) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the Engineering System (ES). The visibility of this navigation button depends on the configuration in the Engineering System (ES).

You can find additional information on this in chapter Opening additional faceplates (Page 203).

(5) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

(6) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

(7) Display area for block states

This area provides additional information on the operating state of the block:

- "Fluttering"

(8) Displaying input values

Changing the display:

Follow these steps to change the displays:

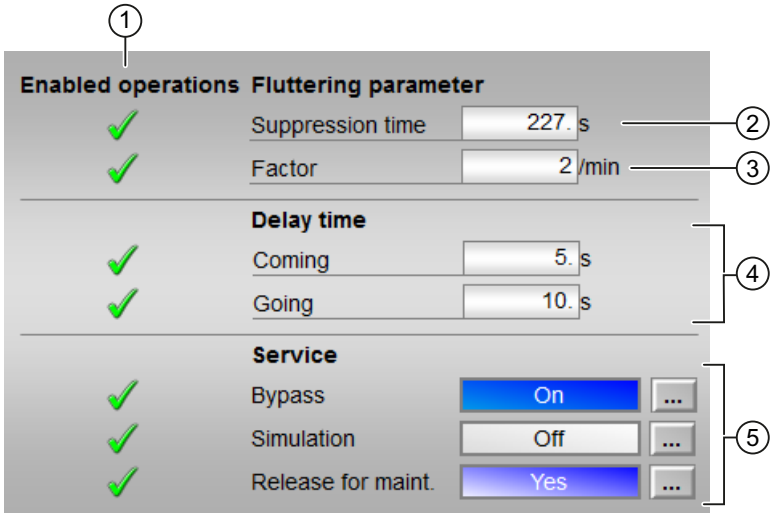
1. In the CFC, right-click the MonDiL block and select "Object Properties" from the context menu.
2. Select "I/Os" tab in the properties window.
3. In the “OS additional text” column, change the default setting for the input parameter (`FlutTmIn`) to what you want to see during runtime later.
4. In the "Identifier" column, change the default setting for the input parameter (`FlutTmIn`) to what you want to see during runtime later.
5. If both “OS additional text” and “Identifier” columns are empty, the default value is displayed in the runtime.
6. The text is used as a label and is therefore always displayed, which means it is independent of the signal status of the corresponding input (`FlutTmInx`).
7. The font size is reduced during the runtime if the input text length is greater than the label width.

Note

If a text is added to the "OS additional text" field of the input parameter (`FlutTmIn`), this text is displayed even if the "Identifier" text field of the same input parameter is not empty. If the "OS additional text" field of the input parameter (`FlutTmIn`) is empty, the "Identifier" text of the same input parameter is displayed.

4.5.8.3 MonDiL parameter view

Parameter view of MonDiL



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

(2) Suppression time

Enter the time period during which signal flutter is suppressed. You can find additional information on this in the Changing values (Page 253) section.

(3) Factor

Enter the number of flutter signals that can be suppressed. You can find additional information on this in the Changing values (Page 253) section.

(4) Delay time

Enter here the delay time by which the output should be set. Enter delay times here for positive ("incoming", 0 - 1 edge) and negative edges ("outgoing", 1 - 0 edge). You can find additional information on this in the Changing values (Page 253) section.

(5) Service

You can select the following functions in this area:

- "Bypass"
- "Simulation"
- "Release for maintenance"

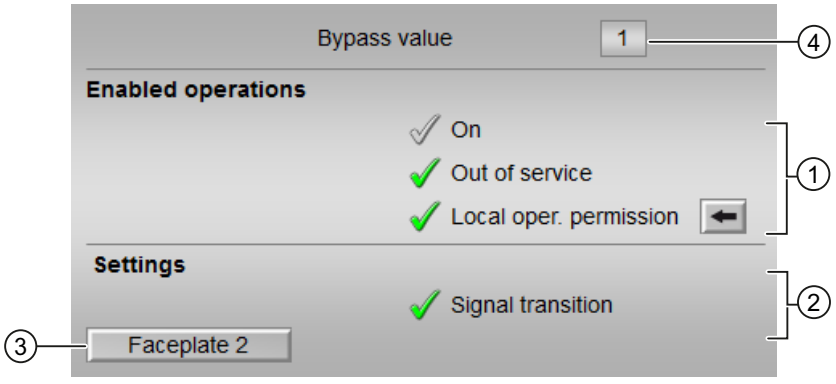
Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Bypassing signals (Page 107)
- Simulating signals (Page 58)
- Release for maintenance (Page 64)

4.5.8.4 MonDiL preview

Preview of MonDiL



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

(2) Settings

- "Signal transition":
  - Activated: A message is generated with a "0 → 1" signal transition at the monitored input.
  - Deactivated: No message is generated.

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the Opening additional faceplates (Page 203) section.

(4) Bypass value

This area displays the bypass value (`BypIn`).

4.5.8.5 Block icon for MonDiL

Block icons for MonDiL

A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes
- Signal status, release for maintenance

- Memo display
- Display of the output signal
- Display configured instance-specific text for the process value (only for block icons 3 to 6). The instance-specific text for the process value can be changed using Text 0 / Text 1 at the output parameter `Out`.

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	The text background color for output `Out = 1` depends on the parameter assignment of the input "`Color`".
	3	See 2
	4	The text background color for output `Out = 1` depends on the parameter assignment of the input "`Color`". Text in block icon can be configured via Text 1 at the `Out` output parameter, a maximum of two characters are supported.
	5	The text background color for output `Out = 1` depends on the parameter assignment of the input "`Color`". For `Color = 11`, the text background color is directly derived from EventState (but without the message classes process message and operator prompt) which also takes into consideration the CSF input. `Color = 11` is only supported by type 5.

Note

Text 1 is also used in the standard view of the faceplate in the display process value active/inactive. Long static texts must therefore be displayed on the "Process value" text using the "`FlutTmIn`" parameter (see MonDiL standard view (Page 504))

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	The text background color for output `Out = 1` depends on the parameter assignment of the input "`Color`".
	4	See 3
	5	See 3
	6	See 3
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

4.6 MonDiS - Monitoring of a digital process tag (Small)

4.6.1 Description of MonDiS

Object name (type + number) and family

Type + number: FB 1913

Family: Monitor

Area of application for MonDiS

The block is used for the following applications:

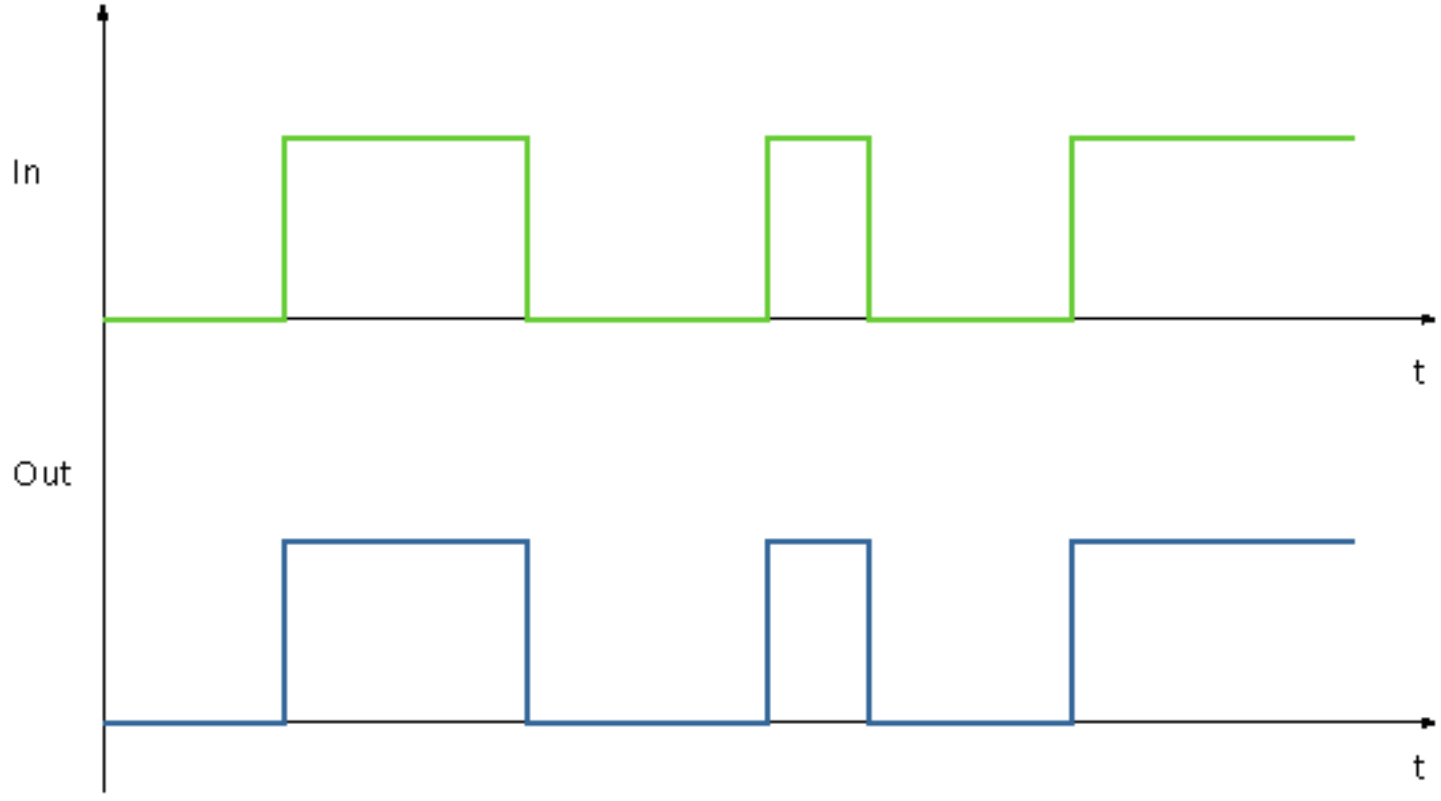
- Monitoring a digital process tag

Note

This block is also available as a large block. A comparison of the MonDiL and MonDiS blocks is available in the section: MonDiL compared to MonDiS (Page 425)

How it works

The MonDiS is used to monitor a digital process tag. The digital value to be monitored is interconnected to the `In` input parameter. The input signal is written directly to the `Out` output parameter.



Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block. The `Out` parameter is affected.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section MonDiS I/Os (Page 520).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`

Status bit	Parameter
2	SimOn
3	OosAct.Value
4	OosLi.Value
5	Not used
6	OnAct.Value
7	Out.Value
8	AlmMsgEn
9 - 31	Not used

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock.Value
1 - 30	Not used
31	MS_RelOp

Status word allocation for Status3 parameter

Status bit	Parameter
0 - 13	Not used
14	Color = 11 (Background color is derived directly from EventState, without message classes process message and operator prompt), only for block icon type 5
15	Color = 10 (Background color of the message instance MsgEvId1 - Message identifier SIG1)
16-19	Not used
20	Color = 6 (Violet, operator prompt) and Out = 1
21	Not used
22	Color = 5 (Pastel green, process) and Out = 1
23 - 24	Not used
25	Color = 4 (Black, control system) and Out = 1
26	Color = 9 (UCMC 6) and Out = 1
27	Color = 3 (Blue, tolerance / UCMC 5) and Out = 1
28	Color = 8 (UCMC 4) and Out = 1
29	Color = 2 (Yellow, warning / UCMC 3) and Out = 1
30	Color = 7 (UCMC 2) and Out = 1
31	Color = 1 (Red, alarm / UCMC 1) and Out = 1

Explanation:

UCMC = user-configured messages classes

Note

The user-configured message classes 5, 3 and 1 are displayed in the existing status bits 27, 29 and 31.

See also

MonDiS operating modes (Page 514)

MonDiS functions (Page 515)

MonDiS error handling (Page 518)

MonDiS messaging (Page 519)

MonDiS block diagram (Page 523)

4.6.2 MonDiS operating modes

MonDiS operating modes

This block provides the following modes.

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

Out of service

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of MonDiS (Page 511)

MonDiS functions (Page 515)

MonDiS error handling (Page 518)

MonDiS messaging (Page 519)

MonDiS I/Os (Page 520)

MonDiS block diagram (Page 523)

4.6.3 MonDiS functions

Functions of MonDiS

The functions for this block are listed below.

Delaying the on function

You set delay times for setting the output using the input parameter `Out_DC`:

- `Out_DC`: delay time [s] for rising edges (0 - 1 edge)

The `Out` output parameter is after expiration of the delay time.

You disable this function if you set the value of the respective parameter to 0 seconds.

Adapting the color representation in the configured message class

You can set the background color of instance-specific texts for the block icons 2 and 3 and thereby adapt them to the configured message class. This background color is displayed when the output parameter `Out` = 1.

You set the color coding with the `Color` parameter:

Value `Color`	Color
0	Old reaction
1	Red (alarm)
2	Yellow (warning)
3	Blue (tolerance)
4	Black (control system)
5	Pastel green (process)
6	Violet (operator prompt)
7...9	Old reaction
10	Background color of the configured message class SIG1 ("Out - setting binary output values")
11	For block icon type 5 only, background color is derived directly from EventState without message classes process message and operator prompt [1]

[1] This also takes into consideration the CSF input

The following colors are shown for user-configured message classes

Value `Color`	Color
0	Old reaction
1	User-configured message classes 1
2	User-configured message classes 3
3	User-configured message classes 5
4	Black (control system)
5	Pastel green (process)
6	Violet (operator prompt)

Value `Color`	Color
7	User-configured message classes 2
8	User-configured message classes 4
9	User-configured message classes 6
10	Background color of the configured message class SIG1 ("Out - setting binary output values")
11	For block icon type 5 only, background color is derived directly from EventState without message classes process message and operator prompt [1]

[1] This also takes into consideration the CSF input

Changing labels on buttons and text

This block provides the standard function Labeling of buttons and text (Page 205).

You can change the label for "Process value" for this block as you please. The change is made with the `OnOp` parameter.

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Group display `SumMsgAct` for limit monitoring, `CSF` and `ExtMsgx`

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `In.ST`

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Simulating signals

This block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can change the delay time for rising edges (`Out_DC`)
5 - 9	Not used
10	1 =Operator can change the simulation value `SimIn`
11	1 = Operator can activate the Simulation function
12	1 = Operator can activate the Release for maintenance function
13 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of MonDiS (Page 511)

MonDiS operating modes (Page 514)

MonDiS error handling (Page 518)
MonDiS messaging (Page 519)
MonDiS I/Os (Page 520)
MonDiS block diagram (Page 523)
User-configured message classes (Page 41)

4.6.4 MonDiS error handling

Error handling of MonDiS

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Control system fault (CSF)

Overview of error numbers

The ErrorNum I/O can be used to output various error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.

Control system fault (CSF)

An external signal can be activated via the CSF input. If this signal changes to = 1, a control system fault is triggered. Refer to the Error handling (Page 119) section for more on this.

See also

Description of MonDiS (Page 511)
MonDiS operating modes (Page 514)
MonDiS functions (Page 515)
MonDiS messaging (Page 519)
MonDiS I/Os (Page 520)
MonDiS block diagram (Page 523)

4.6.5 MonDiS messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a process control fault is triggered (`MsgEvId1`, SIG 3).

Process messages

You can use up to four instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 1	Alarm - high	$$BlockComment$$ `Out` - Binary value set
	SIG 2	Not used	
	SIG 3	AS process control message - fault	$$BlockComment$$ External error has occurred
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 5	AS process control message - fault	$$BlockComment$$ External message 2

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`ExtVa104`
5	`ExtVa105`
6	Reserved
7	Reserved
8	Reserved

Associated value	Block parameters
9	Reserved
10	Reserved

The associated values 4 and 5 are allocated to the parameters `ExtVa104` and `ExtVa105` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of MonDiS (Page 511)
MonDiS operating modes (Page 514)
MonDiS functions (Page 515)
MonDiS error handling (Page 518)
MonDiS block diagram (Page 523)

4.6.6 MonDiS I/Os

I/Os of MonDiS

Input parameters

Parameter	Description	Type	Default
`AlmMsgEn`	1 = Alarms are output	BOOL	1
`BatchEn`	1 = Enable allocation	BOOL	0
`BatchID`	Batch ID	DWORD	16#00000000
`BatchName`	Batch name	S7-String	
`CSF`	1 = External error (control system fault) Error handling (Page 518)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`Color`	Color for status display in the block icon: 0 = Default setting 1 = Red (alarm) 2 = Yellow (warning) 3 = Blue (tolerance) 4 = Black (control system message) 5 = Pastel green (process message) 6 = Violet (operator prompt)	BYTE	16#00
`EN`	1 = Called block will be processed	BOOL	1
`ExtMsg1`	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
Feature	I/O for additional functions (Page 515)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
In	Digital input value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = Occupied by batch control	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnOp*	1 = "On" mode via operator	BOOL	0
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 515)	STRUCT • Bit 0: BOOL • Bit 10: BOOL • Bit 31: BOOL	- • 1 • 1 • 1
Out_DC	Delay time for setting the output parameter Out for rising edges	REAL	0.0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SimOn	1 = Simulation on	BOOL	0
SimIn*	Value used for SimOn = 1	BOOL	0
StepNo	Batch step number	DWORD	16#00000000
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see MonDiS error handling (Page 518)	INT	-1
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feed-forwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 511)	DWORD	16#00000000
Status2	Status word 2 (Page 511)	DWORD	16#00000000
Status3	Status word 3 (Page 511)	DWORD	16#00000000
SumMsgAct	1 = Group message is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

MonDiS operating modes (Page 514)

MonDiS block diagram (Page 523)

Error handling (Page 119)

4.6.7 MonDiS block diagram

MonDiS block diagram

A block diagram is not provided for this block.

See also

Description of MonDiS (Page 511)

MonDiS operating modes (Page 514)

MonDiS functions (Page 515)

MonDiS messaging (Page 519)

MonDiS I/Os (Page 520)

MonDiS error handling (Page 518)

4.6.8 Operator control and monitoring

4.6.8.1 MonDiS views

Views of the MonDiS block

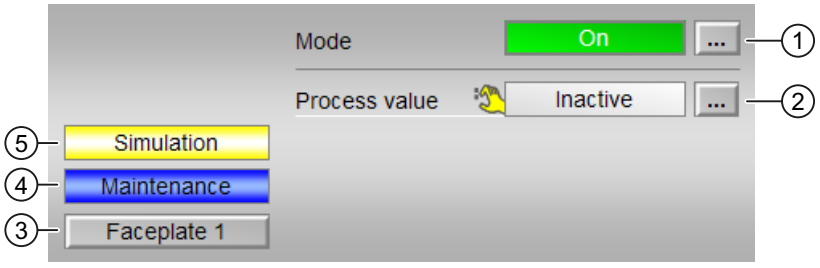
The block MonDiS provides the following views:

- MonDiS standard view (Page 524)
- Alarm view (Page 296)
- Trend view (Page 299)
- MonDiS parameter view (Page 525)
- MonDiS preview (Page 526)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MonDiS (Page 527)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

4.6.8.2 MonDiS standard view

MonDiS standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) Display of process value enabled/disabled

This area shows you the status of the individual connected parameters.

The identifier can be changed using Text 0 / Text 1 at the output parameter `Out`.

If the block is in simulation, you can enable or disable the process value. To do this, click on the display to open the operator input area.

If text is configure for these commands, it is displayed as additional text and as button labels for command selection. Additional information is available in the section Labeling of buttons and text (Page 205).

You can change the text for the process value with the `OnOp` parameter.

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

Additional information is available in the section Opening additional faceplates (Page 203).

(4) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

(5) Display area for block states

This area provides additional information on the operating state of the block:

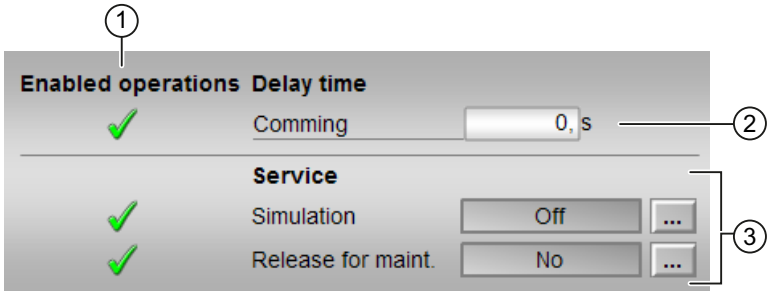
- "Simulation"

See also

Displaying auxiliary values (Page 207)

4.6.8.3 MonDiS parameter view

Parameter view of MonDiS



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

(2) Delay time

Enter here the delay time by which the output should be set. Enter delay times here for positive ("incoming", 0 - 1 edge) edges. Additional information is available in the section Switching operating states and operating modes (Page 251).

(3) Service

You can select the following functions in this area:

- "Simulation"
- "Release for maintenance"

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

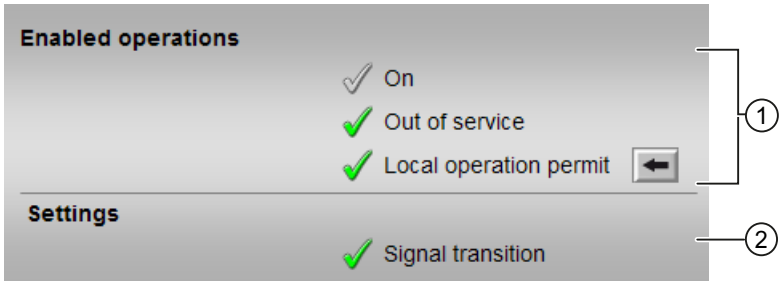
- Simulating signals (Page 58)
- Release for maintenance (Page 64)

See also

Changing values (Page 253)

4.6.8.4 MonDiS preview

Preview of MonDiS



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

(2) Settings

- "Signal transition":
  - Activated: A message is generated with a "0 → 1" signal transition at the monitored input.
  - Deactivated: No message is generated.

4.6.8.5 Block icon for MonDiS

Block icons for MonDiS

A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes
- Signal status, release for maintenance
- Memo display
- Display of the output signal
- Display configured instance-specific text for the process value (only for block icons 2 and 3). The instance-specific text for the process value can be changed using Text 0 / Text 1 at the output parameter `Out`.

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	The text background color for output `Out = 1` depends on the parameter assignment of the input "`Color`".
	3	See 2

Icons	Selection of the block icon in CFC	Special features
	4	The text background color for output `Out = 1` depends on the parameter assignment of the input "`Color`". Text in block icon can be configured via Text 1 at the `Out` output parameter, a maximum of two characters are supported.
	5	The text background color for output `Out = 1` depends on the parameter assignment of the input "Color". For `Color = 11`, the text background color is directly derived from EventState (but without the message classes process message and operator prompt) which also takes into consideration the CSF input.`Color = 11` is only supported by type 5.

Note

Text 1 is also used in the standard view of the faceplate in the display process value active/inactive. Long static texts must therefore be displayed on the "Process value" text using the "`FlutTmIn`" parameter (see MonDiS standard view (Page 524))

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	The text background color for output `Out = 1` depends on the parameter assignment of the input "`Color`".
	3	See 2
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

4.7 MonDi08 - Monitoring 8 digital process tags

4.7.1 Description of MonDi08

Object name (type + number) and family

Type + number: FB 1847

Family: Monitor

Area of application for MonDi08

The block is used for the following applications:

- Monitoring of up to eight digital process tags

How it works

The MonDi08 block is used to monitor up to eight digital process tags with flutter suppression.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the MonDi08 block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Monitoring eight digital process tags (Digital8Monitoring) (Page 2329)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section MonDi08 I/Os (Page 538).

Status bit	Parameter
0 - 2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`

Status bit	Parameter
7	Out1.Value
8	Out2.Value
9	Out3.Value
10	Out4.Value
11	Out5.Value
12	Out6.Value
13	Out7.Value
14	Out8.Value
15 - 19	Not used
20	In1 is used
21	In2 is used
22	In3 is used
23	In4 is used
24	In5 is used
25	In6 is used
26	In7 is used
27	In8 is used
28 - 31	Not used

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock.Value
1	Alm1MsgEn
2	Alm2MsgEn
3	Alm3MsgEn
4	Alm4MsgEn
5	Alm5MsgEn
6	Alm6MsgEn
7	Alm7MsgEn
8	Alm8MsgEn
9 - 30	Not used
31	MS_RelOp

Status word allocation for Status3 parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via EventTsIn
1	Effective signal 2 of the message block connected via EventTsIn
2	Effective signal 3 of the message block connected via EventTsIn
3	Effective signal 4 of the message block connected via EventTsIn
4	Effective signal 5 of the message block connected via EventTsIn

Status bit	Parameter
5	Effective signal 6 of the message block connected via `EventTsIn`
6	Effective signal 7 of the message block connected via `EventTsIn`
7	Effective signal 8 of the message block connected via `EventTsIn`
8 - 23	Effective signal 9..16 of the message block connected via `EventTsIn`
24 - 31	Not used

See also

MonDi08 messaging (Page 536)

MonDi08 functions (Page 533)

MonDi08 block diagram (Page 542)

MonDi08 error handling (Page 535)

MonDi08 modes (Page 532)

4.7.2 MonDi08 modes

MonDi08 operating modes

This block provides the following modes.

- On (Page 71)
- Out of service (Page 71)

"On"

You can find general information about the "On" mode in the On (Page 71) section.

Out of service

You can find general information about the "Out of service" mode in the Out of service (Page 71) section.

See also

MonDi08 block diagram (Page 542)

MonDi08 I/Os (Page 538)

MonDi08 messaging (Page 536)

MonDi08 error handling (Page 535)

MonDi08 functions (Page 533)

Description of MonDi08 (Page 530)

4.7.3 MonDi08 functions

Functions of MonDi08

The functions for this block are listed below.

Monitoring and output of digital signals

The block is operated as "flutter filter". The block receives digital signals at the input `Inx` (x = 1 ... 8) which ideally do not develop any flutter. The block monitors these signals. Use the `FlutTmInx` (x = 1 ... 8) input to determine the duration of a continuous signal in order to transfer it to the process without flutter.

The preprocessed signal is sent to the process via the `Outx` (x = 1 ... 8) output.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `In1.ST`
- `In2.ST`
- `In3.ST`
- `In4.ST`
- `In5.ST`
- `In6.ST`
- `In7.ST`
- `In8.ST`

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Simulating signals

This block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) section. The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)

Operator control permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4 - 11	Not used
12	1 = Operator can activate the Release for maintenance function
13	1 = Operator can change the time for flutter suppression at the `In1` input
14	1 = Operator can change the time for flutter suppression at the `In2` input
15	1 = Operator can change the time for flutter suppression at the `In3` input
16	1 = Operator can change the time for flutter suppression at the `In4` input
17	1 = Operator can change the time for flutter suppression at the `In5` input
18	1 = Operator can change the time for flutter suppression at the `In6` input
19	1 = Operator can change the time for flutter suppression at the `In7` input
20	1 = Operator can change the time for flutter suppression at the `In8` input
21 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Changing labels on buttons and text

This block provides the standard function Labeling of buttons and text (Page 205).

You can change the label for "Process value" for this block as you please. The change is made with the `FlutXTmIn` parameter.

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

MonDi08 messaging (Page 536)

MonDi08 I/Os (Page 538)

Description of MonDi08 (Page 530)

MonDi08 block diagram (Page 542)

MonDi08 error handling (Page 535)

MonDi08 modes (Page 532)

EventTs functions (Page 1634)

4.7.4 MonDi08 error handling

Error handling of MonDi08

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Flutter alarm

Overview of error numbers

The `ErrorNum` I/O can be used to output various error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
11	Fluttering time < 0

Flutter alarm

An alarm is output at output `FlutAct` with 1 if signal flutter is detected. Refer to the functions of the block > Monitoring and reporting flutter signals (Page 533).

See also

MonDi08 block diagram (Page 542)

MonDi08 I/Os (Page 538)

MonDi08 messaging (Page 536)

MonDi08 modes (Page 532)

Description of MonDi08 (Page 530)

4.7.5 MonDi08 messaging

Messaging

The following messages can be generated for this block:

- Process messages
- Instance-specific messages

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId 1	SIG 1	Alarm - high	$$BlockComment$$ Signal change Signal 1 has occurred
	SIG 2	Alarm - high	$$BlockComment$$ Signal change Signal 2 has occurred
	SIG 3	Alarm - high	$$BlockComment$$ Signal change Signal 3 has occurred
	SIG 4	Alarm - high	$$BlockComment$$ Signal change Signal 4 has occurred
	SIG 5	Alarm - high	$$BlockComment$$ Signal change Signal 5 has occurred
	SIG 6	Alarm - high	$$BlockComment$$ Signal change Signal 6 has occurred
	SIG 7	Alarm - high	$$BlockComment$$ Signal change Signal 7 has occurred
	SIG 8	Alarm - high	$$BlockComment$$ Signal change Signal 8 has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	Out1.ST
2	Out2.ST
3	Out3.ST
4	Out4.ST
5	Out5.ST
6	Out6.ST
7	Out7.ST
8	Out8.ST

See also

Description of MonDi08 (Page 530)
MonDi08 functions (Page 533)
MonDi08 I/Os (Page 538)
MonDi08 block diagram (Page 542)
MonDi08 error handling (Page 535)
MonDi08 modes (Page 532)
Time stamp (Page 201)

4.7.6 MonDi08 I/Os

I/Os of MonDi08

Input parameters

Parameter	Description	Type	Default
Alm1MsgEn	1 = Activate error message	BOOL	1
Alm2MsgEn	1 = Activate error message	BOOL	1
Alm3MsgEn	1 = Activate error message	BOOL	1
Alm4MsgEn	1 = Activate error message	BOOL	1
Alm5MsgEn	1 = Activate error message	BOOL	1
Alm6MsgEn	1 = Activate error message	BOOL	1
Alm7MsgEn	1 = Activate error message	BOOL	1
Alm8MsgEn	1 = Activate error message	BOOL	1
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
EN	1 = Called block will be processed	BOOL	1
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	

Parameter	Description	Type	Default
ExtVal08	Associated value 8 for messages (MsgEvID1)	ANY	
Feature	I/O for additional functions (Page 533)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Flut1TmIn	Flutter time [s]	REAL	0.0
Flut2TmIn	Flutter time [s]	REAL	0.0
Flut3TmIn	Flutter time [s]	REAL	0.0
Flut4TmIn	Flutter time [s]	REAL	0.0
Flut5TmIn	Flutter time [s]	REAL	0.0
Flut6TmIn	Flutter time [s]	REAL	0.0
Flut7TmIn	Flutter time [s]	REAL	0.0
Flut8TmIn	Flutter time [s]	REAL	0.0
In1	Binary input In1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In2	Binary input In2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In3	Binary input In3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In4	Binary input In4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In5	Binary input In5	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In6	Binary input In6	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In7	Binary input In7	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In8	Binary input In8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000

Parameter	Description	Type	Default
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = Occupied by batch control	BOOL	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 533)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
StepNo	Batch step number	DWORD	16#00000000
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see MonDi08 error handling (Page 535)	INT	-1
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	1 = Alarm error 1 (output ERROR of the first ALARM_8P)	BOOL	0

Parameter	Description	Type	Default
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
Out1	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Out2	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Out3	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Out4	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Out5	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Out6	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Out7	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Out8	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 530)	DWORD	16#00000000
Status2	Status word 2 (Page 530)	DWORD	16#00000000
Status3	Status word 3 (Page 530)	DWORD	16#00000000

See also

MonDi08 messaging (Page 536)
MonDi08 block diagram (Page 542)
MonDi08 modes (Page 532)

4.7.7 MonDi08 block diagram

MonDi08 block diagram

A block diagram is not provided for this block.

See also

MonDi08 I/Os (Page 538)
MonDi08 messaging (Page 536)
MonDi08 error handling (Page 535)
MonDi08 functions (Page 533)
MonDi08 modes (Page 532)
Description of MonDi08 (Page 530)

4.7.8 Operator control and monitoring

4.7.8.1 MonDi08 views

Views of the MonDi08 block

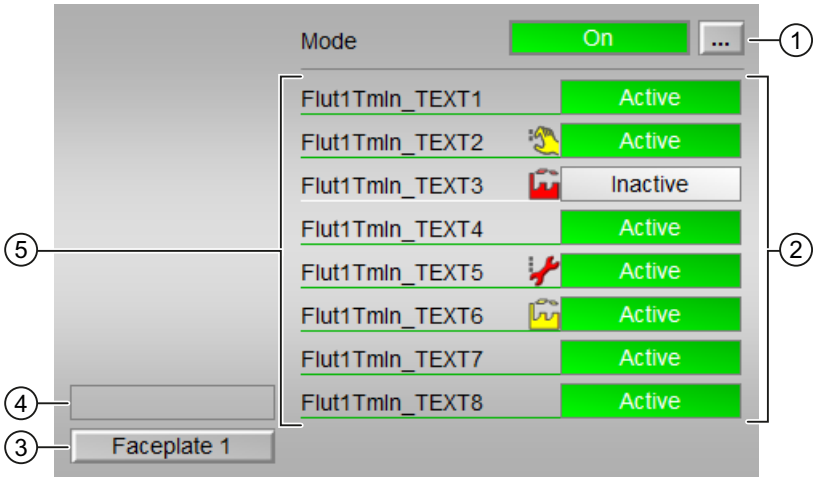
The block MonDi08 provides the following views:

- MonDi08 standard view (Page 543)
- Alarm view (Page 296)
- Trend view (Page 299)
- MonDi08 parameter view (Page 545)
- MonDi08 preview (Page 546)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MonDi08 (Page 547)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

4.7.8.2 MonDi08 standard view

MonDi08 standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) Display of the status for each parameter

This display is only visible when the corresponding block input is connected.

This area shows you the status of the individual parameters available.

You can determine the names for the connected parameters using the `S7_String` attribute at the corresponding input parameter. A default text is displayed if you enter nothing here.

You can change the text for value 1 ... 8 with the `FlutXTmIn` parameter.

Additional information is available in the section Labeling of buttons and text (Page 205).

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the Opening additional faceplates (Page 203) section.

(4) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

(5) Displaying input values

Changing the display:

Follow these steps to change the displays:

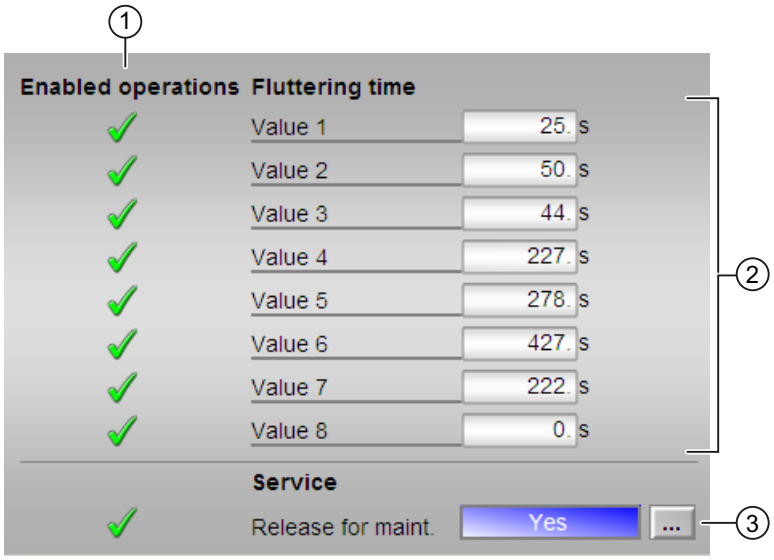
1. In the CFC, right-click the MonDi08 block and select "Object Properties" from the context menu.
2. Select "I/Os" tab in the properties window.
3. In the “OS additional text” column, change the default setting for the input parameter (`FlutXTmIn`, where x = 1...8) to what you want to see during runtime later.
4. In the "Identifier" column, change the default setting for the input parameter (`FlutXTmIn`, where x = 1...8) to what you want to see during runtime later.
5. If both “OS additional text” and “Identifier” columns are empty, the default value is displayed in the runtime.
6. The text is used as a label and is therefore always displayed, which means it is independent of the signal status of the corresponding input (`FlutXTmIn`, where x = 1...8).
7. The font size is reduced during the runtime if the input text length is greater than the label width.

Note

If a text is added to the "OS additional text" field of the input parameter (`FlutXTmIn`, where x = 1...8), this text is displayed even if the "Identifier" text field of the same input parameter is not empty. If the "OS additional text" field of the input parameter (`FlutXTmIn`, where x = 1...8) is empty, the "Identifier" text of the same input parameter is displayed.

4.7.8.3 MonDi08 parameter view

Parameter view of MonDi08



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

(2) Area for entering the flutter time

Use this area to set the time period to determine how long a continuous signal should last in order for it to be transferred to the process without flutter.

Refer to the Changing values (Page 253) section for more on this.

You can determine the names for the connected parameters using the `S7_String` attribute at the corresponding input parameter. A default text is displayed if you enter nothing here.

You can change the text for value 1 ... 8 with the `FlutXTmIn` parameter.

Additional information is available in the section Labeling of buttons and text (Page 205).

(3) Service

You can select the following function in this area:

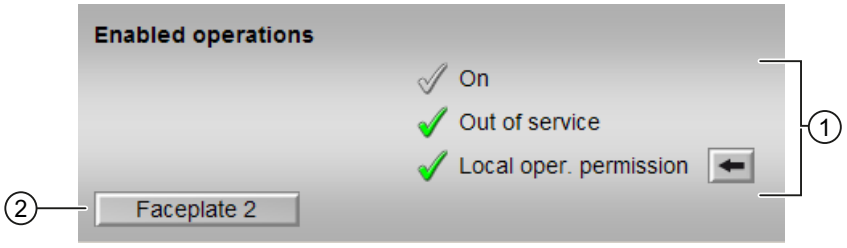
- "Release for maintenance"

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the Release for maintenance (Page 64) section.

4.7.8.4 MonDi08 preview

Preview of MonDi08



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

(2) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the Opening additional faceplates (Page 203) section.

4.7.8.5 Block icon for MonDi08

Block symbols for MonDi08

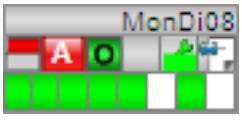
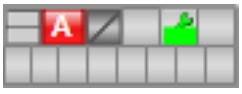
A variety of block symbols are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes
- Signal status, release for maintenance
- Memo display
- Display of the output signal

The block symbols from template @TemplateAPLV8.PDL:

Symbols	Selection of the block symbol in CFC	Special features
	1	

The block symbols from template @TemplateAPLV7.PDL:

Symbols	Selection of the block symbol in CFC	Special features
	1	Block symbol in the full display
	2	
	-	Block symbol in "Out of service" mode (example with type 1 block symbol)

Additional information on the block symbol and the control options in the block symbol is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

Controller blocks

5

5.1 Comparison of large & small blocks

5.1.1 PIDConL compared to PIDConS

Comparison of the PIDConL and PIDConS blocks

The following tables are intended to help you decide which block to use.

Memory and runtime savings of the small block compared to the large block

You save the following resources for each instance:

- Memory space: ~ 45%
- Runtime: ~ 40%

Block operating modes

	PIDConL	PIDConS
Manual mode (Page 72)	X	X
Automatic mode (Page 72)	X	X
Program mode for controllers (Page 78)	X	
Out of service (Page 71)	X	X

Functions of the blocks

	PIDConL	PIDConS
Generating and limiting the manipulated variable	X	X
Tracking and limiting a manipulated variable (Page 192)	X	X
Tracking and limiting a manipulated variable (Page 192)	X	X
Neutral position for motors, valves and controllers (Page 48)	X	X
Outputting group errors (Page 122)	X	X
Outputting a signal for start readiness (Page 53)	X	X

	PIDConL	PIDConS
"Actuator active" information	X	X
Limit monitoring of the feedback (Page 94)	X	
Setpoint specification - internal/ external (Page 127)	X	X
Setpoint limiting for external set-points (Page 192)	X	X
Gradient limit of the setpoint (Page 124)	X	
Using setpoint ramp (Page 123)	X	
Tracking setpoint in manual mode (Page 192)	X	X
Simulating signals (Page 58)	X	X
Limit monitoring of the process value (Page 86)	X	X
Alarm delays with two time val-ues per limit pair		
Alarm delay with one time value per limit pair	X	X
Error signal generation and dead band (Page 188)	X	X
Delay alarm for control deviation at setpoint step changes (Page 186)	X	
Limit monitoring of error signal	X	
Inverting control direction (Page 188)	X	
Physical standardization of set-point, manipulated variable and process value	X	
Selecting a unit of measure (Page 207)	X	X
PID algorithm	X	X
Structure segmentation at con-trollers (Page 194)	X	
Anti-windup (Page 727)	X	X
Feedforwarding and limiting dis-turbance variables (Page 193)	X	
Using control zones (Page 190)	X	
Forming and outputting the sig-nal status for technologic blocks (Page 108)	X	X
Operator control permissions (Page 248)	X	X
Release for maintenance (Page 64)	X	X
Generating instance-specific messages (Page 200)	X	X

	PIDConL	PIDConS
Suppressing messages using the MsgLock parameter (Page 201)	X	X
Display and operator input area for process values and setpoints (Page 203)	X	X
Opening additional faceplates (Page 203)	X	X
SIMATIC BATCH functionality (Page 67)	X	X
Time stamp	X	
Dead band is temporarily disabled (Page 140)	X	
Labeling of buttons and text (Page 205)	X	
Displaying current control signals from output channel block	X	X
Interlocks (Page 99)	X	
Disabling interlocks (Page 103)	X	

Configurable functions using the `Feature` parameter

Bit number	`Feature` bit function	PIDConL	PIDConS
0	Set startup characteristics (Page 137)	X	X
1	Reaction to the out of service mode (Page 176)	X	X
2	Resetting the commands for changing the mode (Page 160)	X	X
4	Setting switch or button mode (Page 166)	X	
12	Control zone with specified I component (Page 159)	X	
13	Control zone with frozen I component (Page 159)	X	
14	External control deviation (Page 150)	X	
15	Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165)	X	X
16	Neutral position manipulated variable takes effect at startup (Page 165)	X	X
18	Disabling bumpless switchover to automatic mode for controllers (Page 172)	X	
22	Update acknowledgment and error status of the message call (Page 159)	X	
24	Enabling local operator authorization (Page 157)	X	X

Bit number	`Feature` bit function	PIDConL	PIDConS
25	Suppression of all messages (Page 173)	X	X
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)	X	X
28	Disabling operating points (Page 144)	X	
29	Signaling limit violation (Page 169)	X	
30	Dead band is temporarily disabled (Page 140)	X	

5.2 ConPerMon - Monitoring of the control performance of control loops

5.2.1 Description of ConPerMon

Object name (type + number) and family

Type + number: FB 1805

Family: Control

Area of application for ConPerMon

The block is used for the following applications:

- Permanent monitoring of control performance of control loops for early detection of problems as they develop

The block calculates:

- Stochastic characteristics of the control performance with the process in a steady state
  - Mean value, variance and standard deviation of controlled variable
  - Mean value of the manipulated variable and control deviation
  - Control performance index
  - Estimated steady state process gain
- Deterministic characteristics of the control performance with step changes in the setpoint
  - Response time and settling time and the settling ratio
  - Overshoot absolute and relative to the step height

Other statistical and graphic evaluations of the signals in the control loop over longer, freely selectable periods are available in the faceplate of the ConPerMon block.

In an overview representation of a plant or unit, you can obtain a clear picture of the status of all control loops based on ConPerMon block icons (indicator light function).

The aim is to detect problems as they develop and to focus the attention of the user on the control loops in a plant that are no longer operating correctly.

How it works

The ConPerMon block evaluates the setpoint and process value signals and the manipulated variable of the PID controller in a sliding time window. The mode of the controller is also taken into account.

With the process in a steady state, the detected stochastic characteristics are compared with the reference values obtained during commissioning. If there is a step change in the setpoint, the stochastic characteristics are by definition irrelevant and are temporarily frozen. Instead, the monitoring of the deterministic characteristics is automatically activated.

If the control performance falls below a defined limit a message is generated. This is also the case when a defined limit for overshoot is exceeded when there is a step change in the setpoint.

Configuration

Each PID controller has a ConPerMon block assigned to it that is installed in the same CFC chart and interconnected with the controller. This already takes place with the corresponding process tag types.

You can open the standard view for the ConPerMon block from the standard view of a controller (for example PIDConL). Additional information on this topic is available in the section Opening additional faceplates (Page 203).

After successful commissioning and optimization of the PID controller to be monitored, the ConPerMon block is initialized while the process is in a steady state and it stores the corresponding characteristic values as reference values.

Follow the steps outlined below:

- Change the PID controller you want to monitor to automatic mode and set the setpoint to the typical operating point. This operating status is intended to represent the normal operation of the process; in other words, the entire plant/unit should be running under production conditions. Monitor the process with a trend writer (CFC trend in the Engineering-System or WinCC Online-Trend-Control on the Operator Station) and wait until the process has settled
- To specify the length `TimeWindow` of the sliding time window, monitor the `PV_Variance` block output of the ConPerMonblock in a trend. The time window should be long enough to keep the variance fairly constant in the relevant decimal places. If the selected time window is too short in relation to the time constants in the control loop and the disturbance signal spectrum, the variance will have too much noise and no useful information.
  If the selected time window is too long, it takes longer before any deterioration of the control performance is detected by the ConPerMon block. It also takes longer following a step change in the setpoint before the monitoring of the stochastic characteristics can be resumed. A good starting value for the `TimeWindow` parameter is 10 times as long as the longest process time constant or 20 times as long as the reset time of the PID controller.
- If the controller
  - is set perfectly,
  - has achieved a steady state,
  - the time window has been defined and filled with values from the steady state,

  the ConPerMon block can be initialized. You do this by clicking the "Initialize" button in the parameter view of the ConPerMon faceplate or by setting the `InitRefVar` = 1 parameter in the CFC block. This saves the `PV_Variance` parameter in the current time window as a reference value for calculating the control performance in the block along with reference values for manipulated variable and process variable.

The **C**ontrol **P**erformance **I**ndex CPI should now be approximately at 100% and therefore indicate that the control loop is operating correctly. Due to stochastic fluctuations, the CPI can also temporarily exceed the 100% mark. If, however, the CPI drops by a significant amount over a longer period, this indicates deterioration of the control performance.

For more detailed information on interpreting the calculation results of the block, refer to the section ConPerMon functions (Page 557).

Note

If the length of the time window is changed during runtime, the CPI will temporarily deviate considerably from its old value and then gradually settle to the new steady value. It is advisable to reinitialize the ConPerMon block after the CPI value has settled to a constant level.

The ConPerMon faceplate is opened from the faceplate of the assigned PID controller so that the ConPerMonicon pictures do not need to be installed separately in each OS picture. It is, in fact, advisable to group all ConPerMon block icons of a plant or unit in one overview picture at the appropriate hierarchy level.

You can expand this overview picture with the trend display of the control performance of all control loops over a longer time to allow you to recognize gradual deterioration (for example reflecting wear and tear). You can also display a further view of the message archive (WinCC AlarmLogging Control) as a hit list sorted according to the frequency in which they occur. In this list, the control loops that caused the most alarms will be shown at the top.

For the ConPerMon block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303)
- PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301)
- PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298)
- PIDConR with safety logic and control loop monitoring (PIDConR_ConPerMon) (Page 2299)
- Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312)
- Ratio control with PIDConR (RatioR) (Page 2315)
- PID controller with Smith predictor (SmithPredictorControl) (Page 2306)
- Step controller with direct access to the actuator and without position feedback (StepControlDirect) (Page 2307)

Startup characteristics

When the CPU starts up, the block is reinitialized but the stored reference values are retained. The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section ConPerMon I/Os (Page 571)

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	`PID_AutAct.Value`
6	`OnAct.Value`
7 - 14	Not used
15	`CPI_Suppress`
16 - 31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	`OvsAH_Act.Value`
2	`OvsWH_Act.Value`
3 - 4	Not used
5	`CPI_WL_Act.Value`
6	`CPI_AL_Act.Value`
7	`OvsAH_En`
8	`OvsWH_En`
9 - 10	Not used
11	`CPI_WL_En`
12	`CPI_AL_En`
13	`OvsAH_MsgEn`
14	`OvsWH_MsgEn`
15 - 16	Not used
17	`CPI_WL_MsgEn`
18	`CPI_AL_MsgEn`
19	Delay of the `CPI_WL_Lim` message
20	Delay of the `CPI_AL_Lim` message
21	Collection of message delays
22 - 31	Not used

See also

ConPerMon messaging (Page 569)

ConPerMon block diagram (Page 577)

ConPerMon error handling (Page 568)

ConPerMon modes (Page 557)

5.2.2 ConPerMon modes

ConPerMon operating modes

The block can be operated using the following modes

- On (Page 71)
- Out of service (Page 71)

"On"

You can find general information about the "On" mode in the On (Page 71) section.

"Out of service"

You can find general information about the "Out of service" mode in the Out of service (Page 71) section.

See also

ConPerMon block diagram (Page 577)

ConPerMon I/Os (Page 571)

ConPerMon messaging (Page 569)

ConPerMon error handling (Page 568)

ConPerMon functions (Page 557)

Description of ConPerMon (Page 553)

5.2.3 ConPerMon functions

Functions of ConPerMon

The functions for this block are listed below.

Monitoring of stochastic characteristics of the control performance

The mean value of a variable relating to an ergodic stochastic process (Page 2362) can be determined from a sliding time window with the length n = `TimeWindow` / `SampleTime`, for example, for the controlled variable y = `PV`:

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y(i)$$

A recursive formulation of this calculation is included in the MeanTime block that is called by the ConPerMonblock. Most steady-state time series can be considered as being ergodic so that the expected value can be estimated by averaging over a window of finite length.

The mean control deviation is `ER_Mean` = `SP` - `PV_Mean`. A mean steady-state control deviation ≠ 0 at a constant setpoint is an indication of problems in the control loop if the controller has I component. You should then check the following potential causes:

- The actuator does not have sufficient capacity. As a result, the controller's actuating signal constantly approaches its limit. This can be caused by unsuitably dimensioned actuators or may simply by wear and tear.
- The manipulated variable demanded by the controller does not take effect in the process, for example because the actuator is defective.

If a steady-state reference operating point (`MV_Ref`, PV_Ref) is known, this can be used to estimate the current mean steady state gain of a linear process model if it is assumed that only disturbances without mean value have an effect:

$$\mathrm{StatGain} = \frac{\mathrm{PV}_{\mathrm{Mean}} - \mathrm{PV}_{\mathrm{Ref}}}{\mathrm{MV}_{\mathrm{Mean}} - \mathrm{MV}_{\mathrm{Ref}}}$$

Normally the reference operating point is obtained during the initialization of the ConPerMon block. Estimation of the steady state gain is then, however, impossible precisely at this operating point. As an alternative, you can also enter the reference values `PV_Ref` and `MV_Ref` manually at the appropriate block inputs. Typical steady-state operating points are often known in advance, for example

- Flow control: `PV` = 0 for `MV` = 0, in other words, valve closed,
- Temperature control: `PV` = `PV_Ambient` for `MV` = 0, in other words, ambient temperature

If the steady state gain changes gradually as time progresses, this is an indication of wear phenomena in the process, such as deposits on heat exchangers, valves or shutters, failing efficiency of process plant, etc.

If, for example, a temperature regulation circuit is closed by a heat exchanger and a deposit forms on the exchanger surfaces, the heat transfer coefficient, and consequently the process gain, is reduced. Within certain limits, this can be compensated by a closed control loop (so that the controller initially disregards the problem). Although the original control loop dynamics can be restored (to a certain extent) by suitable increase of the controller gain as the pollution increases, it is advisable to eliminate the cause of the problem; in other words, to clean the heat exchanger.

If the estimated steady state gain changes suddenly and temporarily, this tends to point to an external disturbance. This may be a normal occurrence in the operation of the process. If, however, these occurrences become more frequent, it is worth finding out the cause.

Due to the approach, the variance `PV_Variance` as second moment requires the calculation of differences of each current measured value from (constant !) mean value:

$$\sigma_y^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(y(i)-\overline{y}\right)^2 = \frac{1}{n}\left(\sum_{i=1}^{n} y^2(i)\right) - \overline{y}^2$$

Within the function block, however, a variant of the calculation is used that saves computing time. The standard deviation

$$PV\_StdDev = \sigma_y = \sqrt{\sigma_y^2}$$

as the square root of the variance is easier to interpret because it has the same physical unit as the measured value.

The control performance index CPI (**C**ontrol **P**erformance **I**ndex) in the unit [%] describes the current variance of the controlled variable relative to a reference variance (benchmark). It is defined as

$$\varsigma = \frac{\sigma_{ref}^2}{\sigma_y^2} 100\%$$

The CPI moves in the $0 < \varsigma \leq 100\%$ range. If the current variance corresponds to the benchmark, the index reaches the value 100. If, by contrast, the current variance becomes larger, the control performance index decreases accordingly. Ideally, the benchmark is obtained in a defined good state of the control loop and stored when the ConPerMon block is initialized. It does not matter if the CPI temporarily reaches values somewhat higher than 100%. A CPI > 100% only means that the variance of the controlled variable is currently somewhat lower than in the reference state. Other alternatives for determining the benchmark will be explained in a separate .

If you consider the calculated CPI signal to be too strongly affected by noise, you can smooth it using the integrated low pass filter (parameter `CPI_FiltFactor`) with the filter time constant `TimeWindow · CPI_FiltFactor`.

The disadvantage of these stochastic characteristics is that they assume an ergodic (Page 2362) or steady state in the process - at least in a statistical sense. Each step change in the standpoint in a controller is an elementary violation of this requirement and leads temporarily to incorrect statements of the stochastic characteristics, for example variances increasing too much. The basic principle of the combined approach implemented in the ConPerMonblock is to use both stochastic and deterministic characteristics for the control performance and to select the suitable characteristics automatically depending on the operating state.

If a step change in the setpoint is detected in a control loop, the ConPerMon block freezes the CPIvalue and automatically suppresses all messages relating to this. As a user, you can also force the suppression of the messages manually via the `ManSupprCPI` = 1 binary input. This setting is useful to avoid false alarms when known disturbances occur, for example at a load change in a Conti process (Page 2363) or a dosing procedure in a Batch process (Page 2357). In such cases the variance of the controlled variable usually rises momentarily. This should not be interpreted as a worsening of the control performance.

Monitoring of deterministic characteristics of the control performance

Assessment of the control performance based on the response to a step change in the setpoint is relatively simple. In the sense of automatic monitoring, the ConPerMon block is capable of determining the essential characteristics of the control performance directly from the signal changes so that when necessary a message or an alarm can be generated automatically by the system.

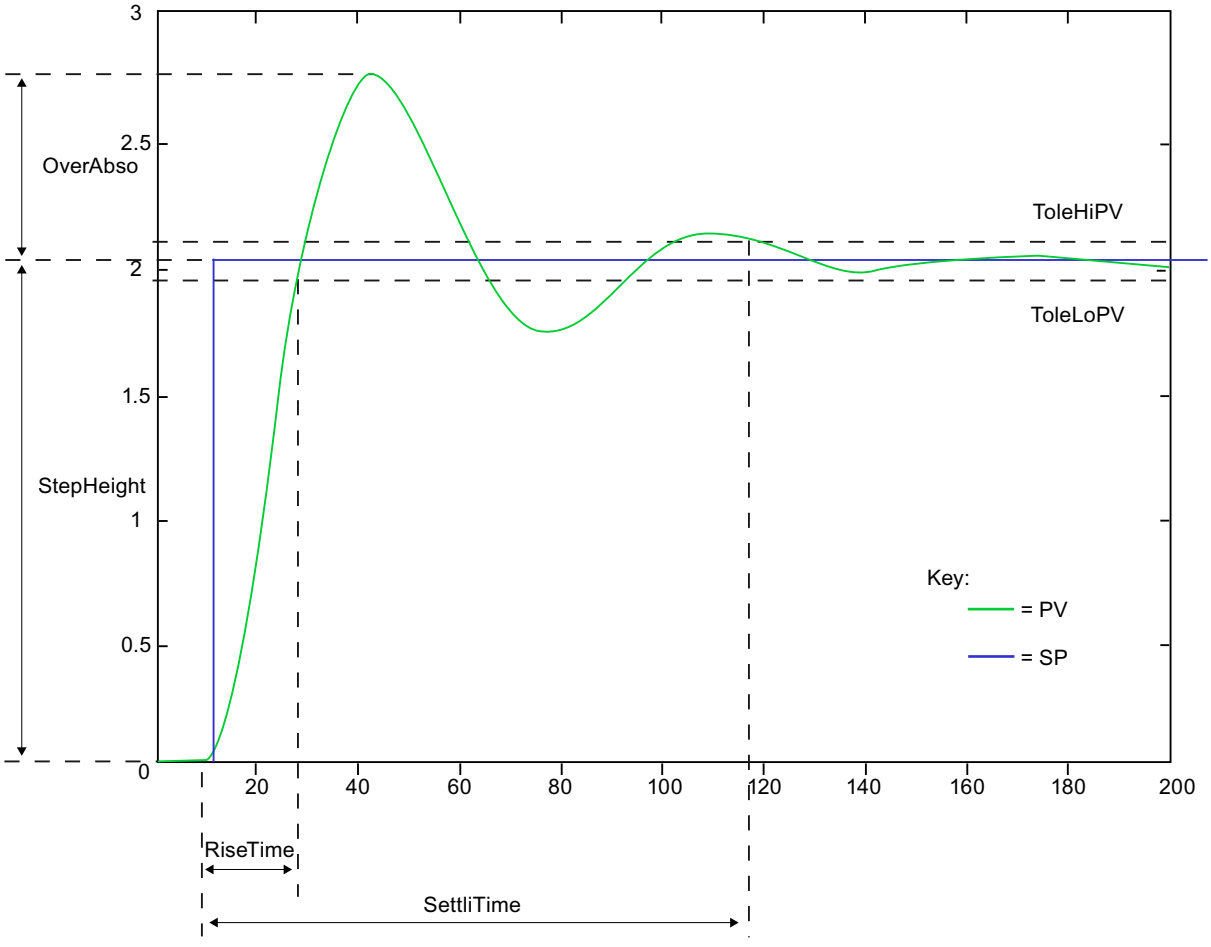
The first thing to look for is always the overshoot if it is present and clearly distinct from the noise level. For a positive step response,

`OverAbso` = max(`PV`) - `SP` > 0

is output where is for a negative step response (step response down), and negative values

`OverAbso` = min(`PV`) - `SP` < 0

are also output. For normalization, the absolute overshoot is related to the height of the step change in the setpoint and is therefore always positive. The relative overshoot (`Overshoot`) as a percentage is a measure of the damping of the control loop. If this is more than 20 or 30%, the loop gain (gain of the controller multiplied by the gain of the control system) is generally too high either because the controller was badly set from the beginning or because the properties of the control system have changed over the course of time. If overshoot is significantly too high, the control loop is generating weakly damped oscillations in the plant. The block sends a message to this effect if the relative overshoot is above a specified limit.



In every control loop, there is a general correlation between overshoot and phase reserve: The higher the overshoot, the lower the phase reserve. If the response of the closed control loop can be described approximately by a 2nd order transfer function

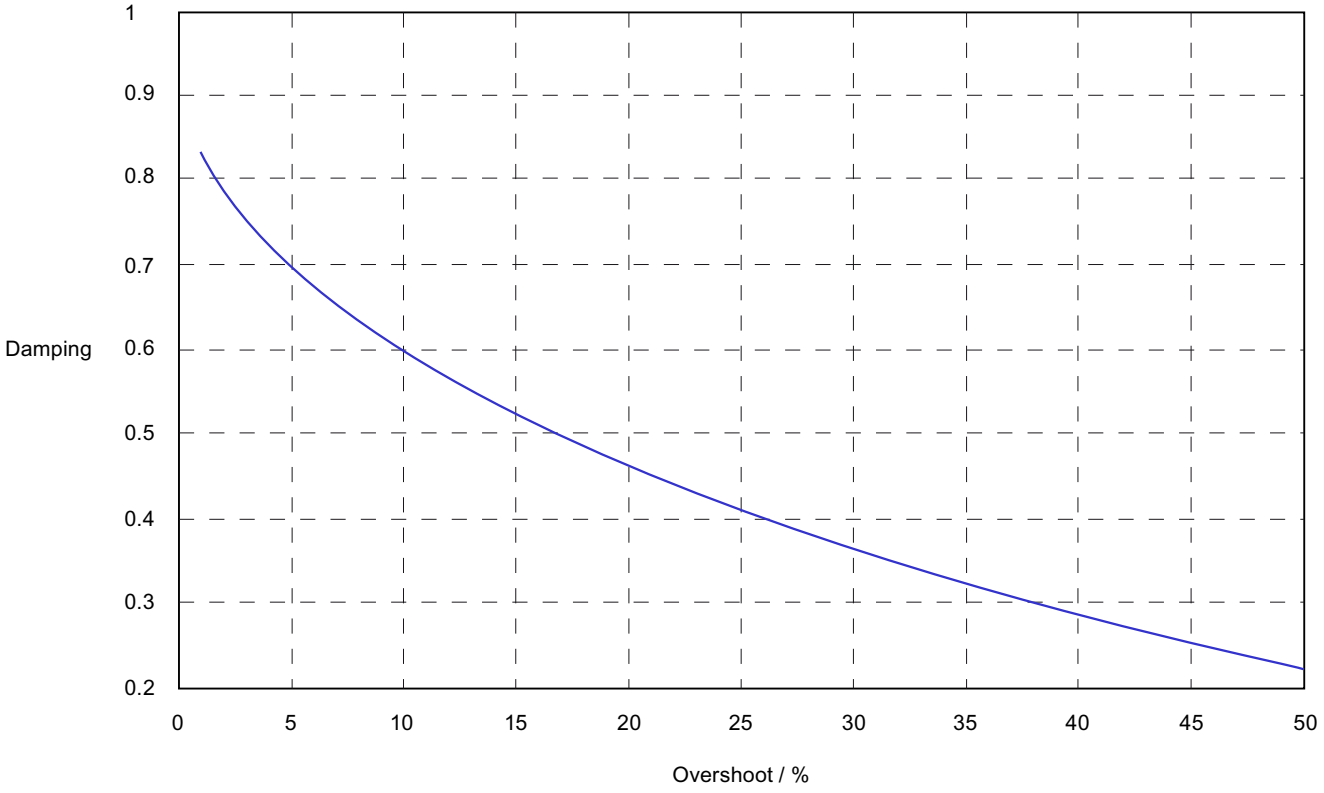
$$g_{cl}(s) = \frac{PV(s)}{SP(s)} = \frac{1}{\frac{1}{\omega_0^2}s^2 + 2\frac{\delta}{\omega_0}s + 1}$$

the following relationships are known:

- If $\delta \geq 1$, the overshoot is equal to zero and the settling response is asymptotic.
- If $\delta < 1$, overshoot and oscillations occur.

The damping of the closed loop can be determined approximately from the overshoot:

$$\delta = \frac{-\ln\left(\frac{\text{Overshoot}}{100\%}\right)}{\sqrt{\ln^2\left(\frac{\text{Overshoot}}{100\%}\right) + \pi^2}}$$



An optimum controller setting typically aims for overshoot between 5 and 25%, which means damping between 0.7 and 0.4.

If overshoot is too high, it is often helpful to reduce the gain of the controller.

While overshoot primarily serves to check controller gain, there is a further characteristic that provides information on the setting of the I component: If the setting of the reset time is unsuitable, the process value will creep towards the new setpoint following a step change in the setpoint. The ratio to the `RiseTime` is relevant, and not the absolute value of the `SettliTime`. If the settling ratio, in other words the quotient of the rise time and settling time, is less than approximately 25%, it can generally be assumed that the reset time of the controller is too slow. To determine the rise time and settling time, a 3σ tolerance band is placed around the setpoint and is also displayed in the faceplate of the ConPerMon block. The absolute values of the settling time and rise time can be assessed in terms of the concrete requirements of the process control for a specific application.

During a step change in the setpoint, larger mathematical variances of the controlled variable are bound to occur compared with the steady state so that generating alarms due to the variance limits being exceeded needs to be suppressed until the settling process has neared completion following the step change in the setpoint. The calculated deterministic characteristics are then output and the stochastic evaluation is reactivated.

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Alternatives for determining the benchmark

During planned commissioning of a plant with integrated ConPerMon, following controller optimization, the ConPerMon block is initialized for every control loop and the calculated variance stored as the benchmark for calculating the CPI.

As an alternative, a benchmark can be set via the `RefVarExt` input parameter by setting the `RefVarExtOn` = 1 input parameter. There are various ways of obtaining numeric values for the benchmark:

- Take the lowest variance that was ever measured in this control loop since the initialization of the ConPerMon block. This is displayed at the `PV_VarMin` output parameter. This value is only useful when the control loop has been in a stable and desirable operating state for a longer period of time at least once since the initialization of the ConPerMon block.
- Take the variance of the control loop with a theoretical minimum variance controller as can be obtained based on archived data using another supplier's CPM application. This depends only on the process dead time and the disturbance model. This form of CPI is known as the Harris index and represents a lower barrier that can generally not be reached by a PID controller which is why CPI seldom reaches the value 100% even by well tuned controllers. Low CPI values provide the first indication that the controller settings could be improved. You should, however, bear in mind that the minimum variance is only a theoretically achievable value and that the minimum variance controller has characteristics that are not desired in the real application, for example extremely high manipulated variable amplitudes. With minimum variance-based CPI, therefore, it is not worth making every effort to bring this as close as possible to 100%

Cascade control

In a cascade control, you should only use the ConPerMon block for the primary controller and not for the secondary controller. The ConPerMon block cannot make any useful statements about the control performance of the secondary controller because

- the variance of the process value in the secondary control loop depends directly on the variance of the setpoint that is set as the manipulated variable by the primary controller,
- there are neither operating phases with a constant setpoint nor defined step changes in the setpoint.

Apart from this, from the perspective of process control, the primary control loop is, of course, the one whose control performance should be monitored while the control performance of the secondary loop is of secondary importance. It is nevertheless advisable to set the secondary controller carefully before optimization and monitoring of the primary controller is started because a poor response by the secondary controller cannot be compensated by the primary controller.

For additional information, read about the process tag template Cascade control with control loop monitoring through ConPerMon (CascadeControl) (Page 2316).

Split-range control

The split-range function block contains two separate (static) characteristics for both actuators. Any significant difference between the two actuators in terms of performance (in other words, different steady state gains for heating and cooling) can be compensated by setting different gradients for the characteristics, so that the controller is presented with a linear process response (regardless of the sign) as far as possible. It this does not work, the control performance will differ slightly in the two areas. The initialization of the ConPerMon block should then be performed in the worse area to avoid error alarms.

For additional information, read about the process tag template Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl) (Page 2309).

PID controller with gain scheduler

The aim of gain scheduling is to achieve consistent control performance over the entire operating range. If this does not work perfectly, the initialization of the ConPerMon block should be performed in an operating point with worse control performance to avoid error alarms. We recommend that you expand the alarm limits somewhat at the ConPerMon block: permit lower `CPI`s and higher overshoot.

For additional information, read about the process tag template PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301).

Override control

For change-over control, different controllers are active depending on the process state; their control performances differ, of course. We recommend using control loop monitoring only for the primary controller, and to suppress it using the `ManSuprCPI` input parameter if limit controlling is activated.

For additional information, read about the process tag template Override control (Page 2322).

Feedforward control

The task of feedforward control is to avoid or at least to reduce degradation of the control performance caused by a measurable disturbance variable. Control loop monitoring therefore can basically be used as it is used for simple control loop. When the disturbance variable is quiet for a time and then acts up for a brief period, the resulting fluctuations of the control performance cannot be ruled out. The reason behind it is that feedforward control represents a model-based intervention, and a model is never a perfect reflection of reality.

For additional information, read about the process tag template PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303).

Smith predictor

The Smith predictor enables higher control performance than a simple PI controller in control loops with dead time. Control loop monitoring therefore can basically be used as it is used for simple control loop. If the dead time changes during ongoing operation, control performance will most likely go down.

For additional information, read about the process tag template PID controller with Smith predictor (SmithPredictorControl) (Page 2306).

Ratio control

With ratio controlling, the control loop monitoring should only be used in the primary control loop if the setpoints are to be determined for combined components from the actual value of the primary component. In this case, you can expect continuous setpoint changes in the control loops for the combined components - similar to sequential control loop of a cascade. If the setpoints for combined components are to be determined from the setpoint of the primary component, the lower-level control loops can be monitored as well.

For additional information, read about the process tag template Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312).

Multivariable controller

The mathematical concept of the ConPerMon block is intended for monovariable control loops. If the variance in a control loop is found to be too high, the block cannot determine whether the actual cause is within the control loop or whether influences are being brought in due to interactions from the outside. If, therefore, you notice strong interactions between various control loops in your plant or even use multivariable controllers, the information provided by the ConPerMon block should be treated with caution.

It can nevertheless make sense to equip a multivariable controller such as the ModPreCon block with control loop monitoring to establish whether the control performance achieved during commissioning of the controller is also retained in runtime. In this case, each controller channel of the multivariable controller has a separate ConPerMon block. Several additional logic functions need to be configured upstream from the `ManSuprCPI` input parameter as

shown in the corresponding sample project Predictive control of a 2x2 multi-variable controlled system (ModPreConSim) (Page 2355):

- If one or more other channels for the multivariable controller is in a non-steady state (for example, step change in the setpoint) indicated by the `CPI_SupRoot` = 1 output parameter, the temporarily increased variance cannot be avoided in this controller channel and should not cause a CPI message.
- If one or more other channels of the multivariable controller have higher variances (poor control performance) indicated by the corresponding output `CPI_WrnAct` = 1, due to the interaction, these variances also cause a higher variance in this controller channel that cannot be avoided and should not lead to a CPI warning. It is possible to find the actual cause of a disturbance in a multivariable system as follows: The channel that first detects higher variances, set the alarm while subsequent alarms in adjacent channels are suppressed.

  Note

  In the case of multivariables, the estimated steady state gains from the monovariable observation are irrelevant. By setting the input parameter `StGainValid` = 0, this status is also displayed in the faceplate as "Uncertain, process related".

If a PID controllers is remotely controlled in program mode (Page 78), it should be treated similar to a secondary controller for a cascade connection in regard to control performance monitoring, i.e. monitoring is usually impractical in this case.

If program mode is the typical operating mode of the controller involved, the corresponding ConPerMon block can be completely removed. If the controller involved is often used in automatic mode, however, monitoring can be temporarily disabled during program mode by connecting the output parameter `AdvCoAct` of the PIDConL block to the input parameter `ManSupprCPI` of the ConPerMon block.

Selecting a unit of measure

The block provides the standard function Selecting a unit of measure (Page 207).

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status for the block is formed from the following parameters:

- `SP_Mon.ST`
- `PV_Mon.ST`
- `MV_Mon.ST`
- `ER_Mean.ST`
- In addition, the ConPerMon block has the following special functions for determining status values:

- If you use a step controller without position feedback, there is no manipulated variable that you could interconnect with the input parameter `MV_Mon`. Unlike most other input parameters `MV_Mon`does not have the preset signal status "Uncertain, process related" (16#78). If there is no interconnected value, this status is transferred to the calculated output parameters `MV_Mean` and `StatGain`.
- `StGainValid`: Set this input to 0 if you use a multivariable controller or observe strong interactions between neighboring control loops. This gives the calculated output parameter `StatGain` the signal status "Uncertain, process related". If known disturbances affect your process, for example dosing procedures in a batch process, you can also set this input temporarily using the recipe control.
- Under normal circumstances the output parameter `StatGain` accepts the worse signal status of `PV_Mon` and `MV_Mon`. Other possible causes of uncertain status for `StatGain` are:
  - the process is currently very close to the reference operating point, or
  - the process is currently in transition, e.g. step change in the setpoint.
- The signal status of the CPI output parameter is dependent on output parameter `CPI_Suppress`: If `CPI_Suppress` = 1, the control performance index CPI is uncertain. Apart from this, the CPI can also become uncertain in occasional situations when there are numeric problems in the calculation of the variance. Under normal circumstances the CPI signal status is the same as the `PV_Mon` signal status.
- The signal status of the `OverAbso` output parameter is set to invalid when step changes in the setpoint are evaluated whose step change height is too low in relation to the noise level.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
1	Reaction to the out of service mode (Page 176)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)

Operator permissions

The block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the OS_Perm parameter:

Bit	Function
0 - 1	Not used
2	1 = Operator can switch to "Out of service" mode
3 - 27	Not used
28	1 = Operator can initialize the block
29	1 = Operator may input a value for the time window, the reference value for the controlled variable and the reference value of the manipulated variable
30	1 = Operator can abort the evaluation of the step response
31	Not used

The block has the following permissions for the OS1Perm parameter:

Bit	Function
0	1 = Operator can change the limit (overshoot) for high alarm
1	1 = Operator can change the limit (process value) for the high warning
2	Not used
3	1 = Operator may change a value for the CPI hysteresis.
4	Not used
5	1 = Operator can change the limit (control performance index CPI) for the low warning
6	1 = Operator can change the limit (control performance index CPI) for the low alarm
7	1 = Operator can activate / deactivate messages via CPI_WL_MsgEn
8	1 = Operator can activate / deactivate messages via CPI_AL_MsgEn
9	1 = Operator can activate / deactivate messages via OvsAH_MsgEn
10	1 = Operator can activate / deactivate messages via OvsWH_MsgEn
7 - 31	Not used

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

Alarm delays with a time value for all limits

The block provides the standard function One time value for all limits (Page 195).

This function is used only for monitoring the control performance index CPI.

Limit operation and display in the faceplate

This block provides the standard function Limit operation and display in the faceplate (Page 309).

Generating instance-specific messages

The block provides the standard function Generating instance-specific messages (Page 200) without the time stamp function in the I/O.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of ConPerMon (Page 553)

ConPerMon messaging (Page 569)

ConPerMon I/Os (Page 571)

ConPerMon block diagram (Page 577)

ConPerMon error handling (Page 568)

ConPerMon modes (Page 557)

5.2.4 ConPerMon error handling

Error handling of ConPerMon

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.

Error number	Meaning of the error number
2	`SampleTime` < 0.001
10	`TimeWindow` < 20 · `SampleTime`

See also

ConPerMon block diagram (Page 577)

ConPerMon I/Os (Page 571)

ConPerMon messaging (Page 569)

ConPerMon functions (Page 557)

ConPerMon modes (Page 557)

Description of ConPerMon (Page 553)

5.2.5 ConPerMon messaging

Messaging

If the control performance falls below a defined limit a message is generated. This is also the case when a defined limit for overshoot is exceeded when there is a step change in the setpoint.

If the CPI temporarily falls below the configured warning and alarm limits, it is not necessary to trigger an alarm immediately. The main aim of the control loop monitoring is to signal the need for maintenance or optimization measures in individual control loops. With the alarm delay, you can make sure that an alarm is triggered only after the cause exists for longer than a configured period `AlmDelay`.

The following messages can be generated for this block:

- Process messages
- Instance-specific messages

Process messages

Message instance	Message identifier	Message class	Event
MsgEvID	SIG 1	Alarm - high	$$BlockComment$$ Overshoot - high alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ Overshoot - high warning limit violated
	SIG 3	Warning - low	$$BlockComment$$ CPI - low warning limit violated
	SIG 4	Alarm - low	$$BlockComment$$ CPI - low alarm limit violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You can use up to three instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvID	SIG 5	AS process control message - fault	$$BlockComment$$ External message 1 Status 16#@5%x@
	SIG 6	AS process control message - fault	$$BlockComment$$ External message 2 Status 16#@6%x@
	SIG 7	AS process control message - fault	$$BlockComment$$ External message 3 Status 16#@7%x@

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance **MsgEvID**

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	CPI(control performance index)
5	ExtMsg1.ST
6	ExtMsg2.ST
7	ExtMsg3.ST

Associated value	Block parameters
8	ExtVa108
9	ExtVa109
10	Reserved

The associated values 8 ... 9 are allocated to the parameters ExtVa108 ... ExtVa109 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of ConPerMon (Page 553)
ConPerMon functions (Page 557)
ConPerMon I/Os (Page 571)
ConPerMon block diagram (Page 577)
ConPerMon error handling (Page 568)
ConPerMon modes (Page 557)

5.2.6 ConPerMon I/Os

I/Os of ConPerMon

Input parameters

Parameter	Description	Type	Default
AlmDelay	Alarm delay time [s] for monitoring the control performance index CPI 0 = Alarm delay deactivated	REAL	0.0
BatchEn	1 = Enable allocation for batch control	BOOL	0
BatchID	Current batch ID	DWORD	16#00000000
BatchName	Current batch designation	S7-String	
BreakSuppress	1 = Manually cancel suppression of the control performance alarm during a step response	BOOL	0
CPI_AlmHyst	Alarm hysteresis of the control performance index [%]	REAL	5.0
CPI_AL_En	1 = Activate alarm (low) for monitoring of control performance	BOOL	1
CPI_AL_Lim	Low alarm limit for control performance [%]	REAL	30.0
CPI_FiltFactor	Low pass filter for CPI, filter time constant = TimeWindow · CPI_FiltFactor	REAL	10.0

Parameter	Description	Type	Default
CPI_WL_En	1 = Activate warning (low) for monitoring of control performance	BOOL	1
CPI_WL_Lim	Low warning limit for control performance [%]	REAL	50.0
CPI_AL_MsgEn	1 = Activate alarm message for: Low limit for control performance	BOOL	0
CPI_WL_MsgEn	1 = Activate warning message for: Low limit for control performance	BOOL	0
EN	1 = Called block will be processed	BOOL	1
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa108	Associated value 8 for messages (MsgEvID)	ANY	
ExtVa109	Associated value 9 for messages (MsgEvID)	ANY	
Feature	I/O for additional functions (Page 557)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
InitRefVar	1 = Initialization of the block. The benchmark of the controlled variable variance and the reference values of the controlled variable and manipulated variable are measured in the steady state.	BOOL	0
LoopClosed	1 = Control loop is closed	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManSupprCPI	1 = Manual suppression of the CPI calculation and message, e.g. during known disturbances	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgEvID	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MV_Mon	Manipulated variable of the controller for monitoring	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#78
MV_Ref	Reference value of the manipulated variable	REAL	0.0
MV_Unit	Unit of measure for manipulated variable	INT	1342
Occupied	1 = Occupied by batch control	BOOL	0
OnOp*	1 = "On" mode via operator	BOOL	1
OosLi	1 = Edge transition (0-1) = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- 0 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 557)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OS1Perm	I/O for operator permissions (Page 557)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OvsAH_En	1 = Activate alarm (high) for overshoot	BOOL	1
OvsAH_Lim	Overshoot alarm limit [%]	REAL	25.0
OvsAH_MsgEn	1 = Enable alarm message for overshoot	BOOL	0
OvsWH_En	1 = Activate warning (high) for overshoot	BOOL	1
OvsWH_Lim	Overshoot warning limit [%]	REAL	20.0
OvsWH_MsgEn	1 = Enable warning message for overshoot	BOOL	0
PV_Mon	Controlled variable for monitoring	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Ref	Reference value for controlled variable	REAL	0.0
PV_Unit	Unit of measure for process value	INT	1001
RefVarExt	Reference value for PV_Variance in control loop "good" status	REAL	0.0
ReVaExOn	1 = Use the external reference value of RefVarExt	BOOL	0
RefVariance	Variance of controlled variable in control loop "good" status	REAL	100.0

Parameter	Description	Type	Default
`RunUpCyc`	Number of start cycles in which messages are suppressed	INT	32000
`SampleTime`	Sampling time [s] (assigned automatically)	REAL	0.1
`SelFp1`	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
`SelFp2`	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
`SettlingTimer`	Settling time for the adjustment of the dead band	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SP_Mon`	Setpoint of the corresponding controller for monitoring	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`StepNo`	Batch step number	DWORD	16#00000000
`StGainValid`	0 = Output parameter `StatGain` systematically invalid, e.g. for multi-variable processes	BOOL	1
`TimeWindow`	Width of the sliding time window [s] for statistical evaluations	REAL	120.0
`UserStatus`	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
`CPI`	Control performance index	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`CPI_AL_Act`	1 = Alarm due to control performance is active. You can change the reaction for this parameter with `Feature` bit 28 (Disabling operating points (Page 144)) and with `Feature` bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`CPI_Suppress`	1 = Message for the control performance index is suppressed; retain last valid `CPI` value	BOOL	1

Parameter	Description	Type	Default
CPI_SuRoot	1= CPI message suppression was triggered in this control loop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CPI_WL_Act	1 = Warning due to control performance is active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ER_Mean	Mean value of the error signal in the time window [PV_Unit]	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see ConPerMon error handling (Page 568)	INT	-1
MsgAckn	Message acknowledgment status (output ACK_STATE of ALARM_8P)	WORD	16#0000
MsgErr	1 = Message error (output ERROR of ALARM_8P)	BOOL	0
MsgStatus	Message status (output STATUS of ALARM_8P)	WORD	16#0000
MV_Mean	Mean value of the manipulated variable in the time window [MV_Unit]	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
O_MS_Ext	Reserved	DWORD	0
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS1PermOut	Display of OS1Perm	DWORD	16#FFFFFFFF
OS1PermLog	Display of OS1Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF

Parameter	Description	Type	Default
OverAbso	Absolute overshoot of the step response [PV_Unit]	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Overshoot	Relative overshoot of the step response with respect to the step change height [%]	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
OvsAH_Act	1 = Alarm due to overshoot is active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OvsWH_Act	1 = Warning due to overshoot is active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Mean	Mean value of the controlled variable in the time window [PV_Unit]	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_StdDev	Standard deviation of the controlled variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_ToleHi	Limit (high) of the 3σ band around the setpoint	REAL	0.0
PV_ToleLo	Limit (low) of the 3σ band around the setpoint	REAL	0.0
PV_Variance	Variance of the controlled variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_VarMin	Minimum observed value of the process variance (slave pointer)	REAL	10000.0
RefStdDev	Standard deviation of controlled variable in control loop "good" status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RiseTime	Rise time of the step response [s]	REAL	0.0
SettliDeadBand	Settling time for the adjustment of the dead band	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SettlRatio	Ratio = Rise time/settling time · 100%	REAL	0.0
SettliTime	Settling time of the step response [s]	REAL	0.0

Parameter	Description	Type	Default
StatGain	Steady-state process gain [PV_Unit / MV_Unit]	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 553)	DWORD	16#00000000
Status2	Status word 2 (Page 553)	DWORD	16#00000000
StepPhase	Phase of the step response: 0 = Waiting 1 = Rising 2 = Overshoot 3 = Settled	INT	0

See also

ConPerMon messaging (Page 569)

ConPerMon block diagram (Page 577)

ConPerMon modes (Page 557)

5.2.7 ConPerMon block diagram

ConPerMon block diagram

A block diagram is not provided for this block.

See also

ConPerMon I/Os (Page 571)

ConPerMon messaging (Page 569)

ConPerMon error handling (Page 568)

ConPerMon functions (Page 557)

ConPerMon modes (Page 557)

Description of ConPerMon (Page 553)

5.2.8 Operator control and monitoring

5.2.8.1 ConPerMon views

Views of the ConPerMon block

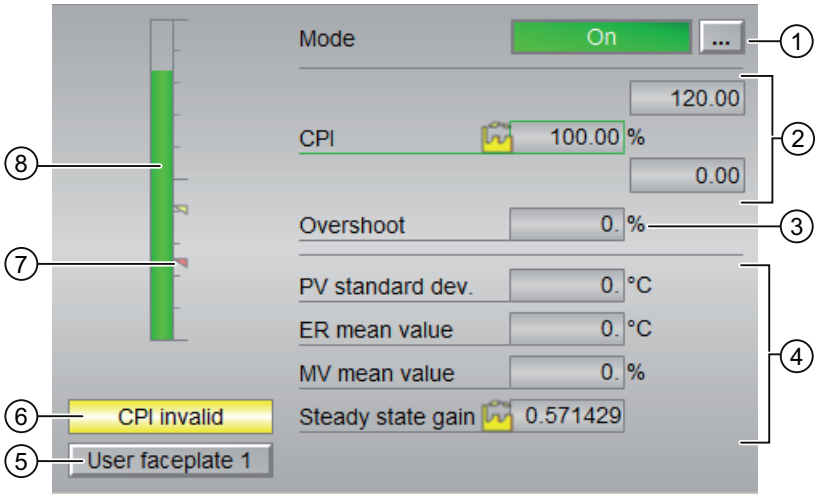
The block ConPerMon provides the following views:

- ConPerMon standard view (Page 578)
- Alarm view (Page 296)
- ConPerMon limit view (Page 580)
- Trend view (Page 299)
- ConPerMon parameter view (Page 581)
- ConPerMon preview (Page 582)
- Memo view (Page 298)
- Batch view (Page 296)
- ConPerMon setpoint view (Page 583)
- Block icon for ConPerMon (Page 585)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

5.2.8.2 ConPerMon standard view

ConPerMon standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) Display area for control performance

This area shows the current control performance index.

(3) Display area for the overshoot

This area shows you the relative overshoot based on a step change [%].

(4) Display area for the static evaluation of the current time window (`TimeWindow`)

This area shows you the statistical evaluation of the current time window. The following values are evaluated:

- "PV standard dev.": Standard deviation of the controlled variable
- "ER mean value": Mean value of the control deviation
- "MV mean value": Mean value of the manipulated variable
- "Steady state gain": Steady-state process gain

(5) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

See also the Opening additional faceplates (Page 203) section for more on this.

(6) Display for CPI valid / CPI invalid

This area shows you if the control performance index is valid or invalid:

- "CPI valid": Control performance is valid
- "CPI invalid": Control performance is invalid

You set the limits for the control performance index in the limits views, depending on the configuration in the engineering system (ES).

(7) Limit display

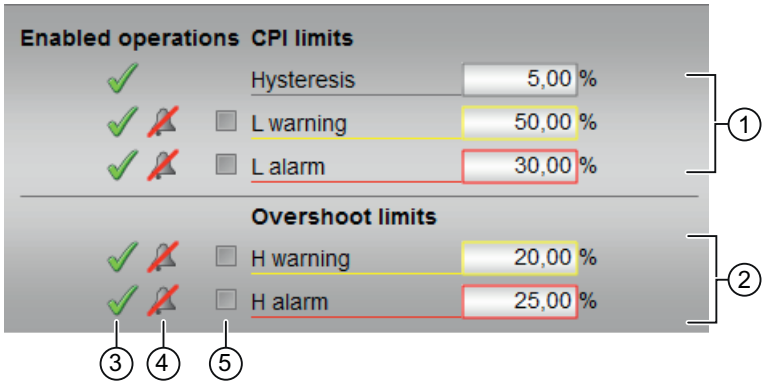
These colored triangles show you the configured limits in the respective bar graph.

(8) Bar graph for control performance index

This area shows you the current CPI control performance index in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

5.2.8.3 ConPerMon limit view

Limit view of ConPerMon



(1) CPI limits

In this area, you can enter the limits for the CPI control performance index. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "Hysteresis"
- "L warning": Warning low
- "L alarm": Alarm low

(2) Overshoot limits

In this area, you can enter the limits for the overshoot. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H warning": Warning high
- "H alarm": Alarm high

(3) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

(4) "Message suppression/delay"

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

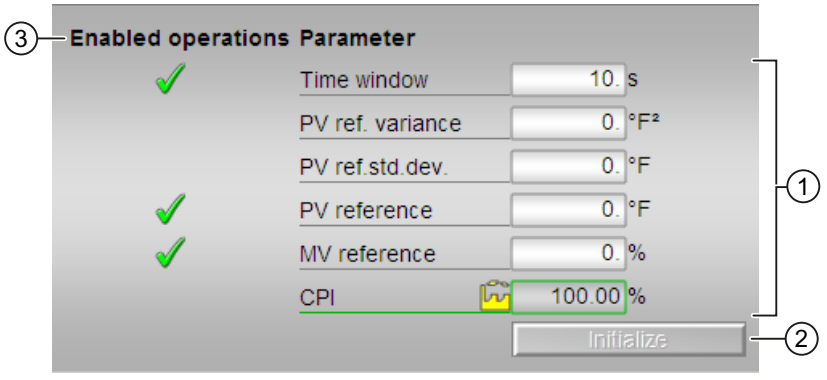
Alarm delays are also displayed in this position; for more on this see section Area of application of the alarm delays (Page 195).

(5) Suppress messages

You can enable / disable messages by setting the check mark.

5.2.8.4 ConPerMon parameter view

Parameter view of ConPerMon



(1) Parameter

In this area, you change parameters and therefore influence the controller. Refer to the Changing values (Page 253) section for more on this.

You can influence the following parameters:

- "Time window": Set the time window here, in which the statical evaluation for the following values is to be performed:
  - Standard deviation of the controlled variable
  - Mean value of the control deviation
  - Mean value of the manipulated variable
  - Steady-state process gain
- "PV reference": Reference value for controlled variable
- "MV reference": Reference value of the manipulated variable

(2) Initialize button

Clicking this button initializes the block. The benchmark of the controlled variable variance and the reference values of the controlled variable and manipulated variable are measured in the steady state.

(3) Enabled operations

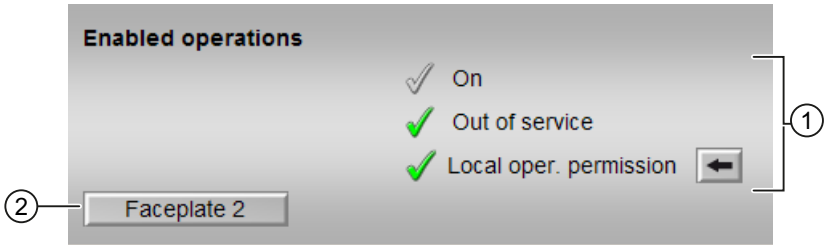
This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

5.2.8.5 ConPerMon preview

Preview of ConPerMon



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. You can find more information about this in the section titled: Operator control permissions (Page 248)

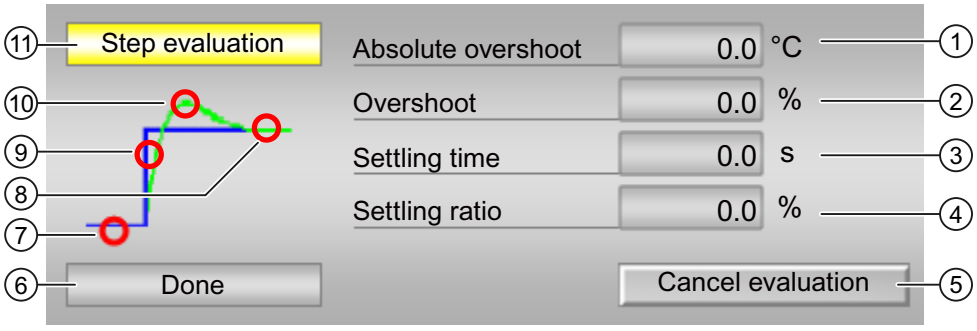
(2) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

Additional information is available in the section Opening additional faceplates (Page 203).

5.2.8.6 ConPerMon setpoint view

Setpoint view of ConPerMon



(1) Absolute overshoot

The absolute overshoot is given in the physical unit of the actual value.

(2) Overshoot

Output of the relative overshoot base on a step change.

(3) Settling time

Settling time of the step response in seconds.

(4) Settling time ratio

The settling time ratio is formed from the ramp time by the settling time.

(5) Cancel evaluation button

You can use the button to show the evaluation of the step response.

Note

The "Cancel evaluation" button is operable when **all** of the following conditions are met:

- Operator permission level = 2 (Higher-level process control)
- Parameter `OS_Perm Bit30` = 1 (Operator can abort the evaluation of the step response)

(6), (7), (8), (9) and (10): Status of the step response

The following states are shown here:

- (6) Textual display of the states
- (7) "Idle": steady state
- (8) "Steady-state": i.e. the actual value is located within the tolerance band of the setpoint
- (9) "Rising phase": from the initial state to the first time the setpoint is reached
- (10) "Overshoot"

(11) Display: Evaluation of the step response in progress:

- "Step evaluation"
- "Constant PV"

See also

ConPerMon standard view (Page 578)
ConPerMon limit view (Page 580)
ConPerMon parameter view (Page 581)

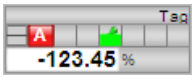
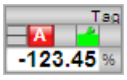
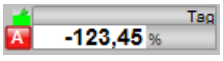
5.2.8.7 Block icon for ConPerMon

Block icons for ConPerMon

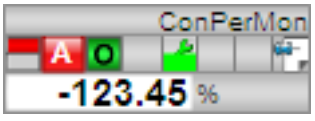
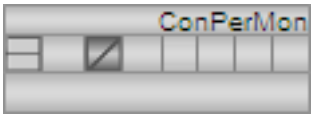
A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits
- Operating modes
- Signal status, release for maintenance
- Memo display
- Process value (black, with and without decimal places)

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

5.3 FmCont - Interface to module FM 355

5.3.1 Description of FmCont

Object name (type + number) and family

Type + number: FB 1818

Family: Control

Area of application for FmCont

The block is used for the following applications:

- Fixed setpoint control
- Cascade control
- Ratio control
- Split-range control

How it works

Block FmCont is used to interface the `FM 355` controller modules.

FmCont can be used for the C (continuous controllers) and S (step and pulse controllers) module types. It contains the algorithms of the setpoint ramp, the setpoint rise limitation, and the limit monitoring of the process value, the control deviation, and the position feedback. Limit monitoring is not used on the module. The control function itself (e.g. PID algorithm) is processed on the module.

You can use the FmCont block to monitor all relevant process values and to change all relevant controller parameters.

Application examples of the FM 355 and detailed descriptions of the associated input and output parameters can be found in the manual for the FM 355. controller module.

Process values such as temperatures, levels and flows can be controlled. However, pressure processes which are not excessively fast are also possible.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100). Set the input parameter `LogAddr` to the module address from HW Config and the input parameter `Channel` to the desired controller channel (1 ... 4).

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The parameter `CoordNo` is set
- The in/out parameter `Mode` is interconnected to the corresponding `OMODE_xx` output parameter of the MOD block.

- The parameter `FM 355` is set in accordance with the module type C/S
- The in/out parameter `EnCoord` is interconnected to the output `EN_CO_x` of the FM_CO block of the Basis Library (x = number of the rack)
- The output parameter `EnCoNum` is interconnected to the input `ENCOx_yy` of the `FmCont` block (x = number of the rack, yy = coordination number).

For the control loop monitoring to work as planned in the trend view of the controller faceplates, the

`S7_xarchive:='Value,shortterm;'`

attributes in the process tag types for control loops at the controller function block must be set for the following tags:

- Input parameters:
  - `CPI_In`
- Output parameters
  - `MV`
  - `MV_HiAct`
  - `MV_LoAct`
  - `LoopClosed`
  - `SP`
  - `PV_Out`
  - `PV_ToleHi`
  - `PV_ToleLo`

Startup characteristics

Use the `Feature` startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

For a description of the individual parameters, see the section I/Os of FmCont (Page 606)

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutAct.Value`
6	Not used
7	`ManAct.Value`

Status bit	Parameter
8	SP_ExtAct.Value
9	MV_SafeOn.Value AND NOT OosAct.Value
10	MV_TrkOn.Value AND NOT (ManAct.Value OR OosAct.Value OR MV_SafeAct.Value)
11	MV.Value > ManLoLim for continuous or pulse controller NOT FbkClosed.Value for step controller with/without position feedback
12	Open.Value
13	Close.Value
14	Stop.Value
15	FbkOpnOut.Value
16	FbkClsOut.Value
17	SimOn AND ManAct
18	MV_SafeOn.Value
19	AdvCoAct.Value
20	1 = Input parameter Rbk is not interconnected (RbkOut.ST = 16#FF)
21	NegGain.Value
22	MV_FmTrkAct.Value AND NOT (OosAct.Value OR MV_SafeAct.Value)
23 - 24	Not used
25	MV_TrkOn.Value
26	MV_FmTrkOn.Value
27	AdvCoModSP
28	1 = Analog controller (FM 355 = 1)
29	1 = Pulse controller (FM 355 = 0 AND StepCon = 0)
30	1 = Step controller with position feedback (FM 355 = 0 AND StepCon = 1 AND WithRbk = 1)
31	1=Step controller without position feedback (FM 355 = 0 AND StepCon = 1 AND WithRbk = 0)

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock
1	PV_AH_Act.Value
2	PV_WH_Act.Value
3	PV_TH_Act.Value
4	PV_TL_Act.Value
5	PV_WL_Act.Value
6	PV_AL_Act.Value
7	PV_AH_En
8	PV_WH_En
9	PV_TH_En
10	PV_TL_En
11	PV_WL_En

Status bit	Parameter
12	PV_AL_En
13	PV_AH_MsgEn
14	PV_WH_MsgEn
15	PV_TH_MsgEn
16	PV_TL_MsgEn
17	PV_WL_MsgEn
18	PV_AL_MsgEn
19	ER_AH_Act.Value
20	ER_AL_Act.Value
21	ER_AH_En
22	ER_AL_En
23	ER_AH_MsgEn
24	ER_AL_MsgEn
25	RbkWH_Act.Value
26	RbkWL_Act.Value
27	RbkWH_En
28	RbkWL_En
29	RbkWH_MsgEn
30	RbkWL_MsgEn
31	MS_RelOp

Status word allocation for `Status3` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via EventTsIn
1	Effective signal 2 of the message block connected via EventTsIn
2	Effective signal 3 of the message block connected via EventTsIn
3	Effective signal 4 of the message block connected via EventTsIn
4	Effective signal 5 of the message block connected via EventTsIn
5	Effective signal 6 of the message block connected via EventTsIn
6	Effective signal 7 of the message block connected via EventTsIn
7	Effective signal 8 of the message block connected via EventTsIn
8	Delay of the PV_AH_Lim message
9	Delay of the PV_WH_Lim message
10	Delay of the PV_TH_Lim message
11	Delay of the PV_TL_Lim message
12	Delay of the PV_WL_Lim message
13	Delay of the PV_AL_Lim message
14	Delay of the ER_AH_Lim message
15	Delay of the ER_AL_Lim message
16	Collection of message delays
17 - 26	Not used

Status bit	Parameter
27	SP_UpRaAct, SP_DnRaAct limits enabled for gradient mode (SP_RateOn = 1)
28	GrpErr.Value
29	RdyToStart.Value
30	SimLiOp.Value
31	Not used

Status word allocation for Status4 parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via EventTsIn
8 - 31	Not used

See also

FmCont messaging (Page 604)

FmCont block diagram (Page 622)

FmCont modes (Page 591)

FmCont error handling (Page 602)

FmCont functions (Page 592)

5.3.2 FmCont modes

FmCont operating modes

The block can be operated using the following modes:

- Automatic mode (Page 72)
- Manual mode (Page 72)
- Program mode for controllers (Page 78)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and bumpless switchover in the Manual and automatic mode for control blocks (Page 72) section.

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the Manual and automatic mode for control blocks (Page 72) section.

"Program mode for controllers"

General information on "Program mode for controllers" is available in the section Program mode for controllers (Page 78).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

FmCont block diagram (Page 622)

FmCont I/Os (Page 606)

Description of FmCont (Page 587)

FmCont functions (Page 592)

FmCont error handling (Page 602)

FmCont messaging (Page 604)

5.3.3 FmCont functions

Functions of FmCont

The functions for this block are listed below.

Module types

FmCont can be used for the C (continuous controllers) and S (step controllers with and without position feedback and pulse controllers) module types. You can use the following parameters to identify which module type and controller type has been set:

FM 355	StepCon	WithRbk	Module type, controller type
1 or C	-	-	FM 355 C: Continuous controller
0 or S	1	1	FM 355 S: Step controller with position feedback
0 or S	1	0	FM 355 S: Step controller without position feedback
0 or S	0	-	FM 355 S: Pulse controller

You must set input parameter `StepCon` if you want to set the step controller with/without position feedback as the controller type.

Generating manipulated variables for continuous controllers, step controllers with position feedback, or pulse controllers

The manipulated variable MV and the actuating signals Open, Close and Stop are generated as follows

MV_Safe-On	MV_FmTrk On	ManAct	MV_TrkOn	AdvCoAct AND NOT AdvCo-ModSP	MV =	Manipu-lated varia-ble limit	State	Open, Close, Stop
1	-	-	-	-	MV_Safe	MV_HiLim MV_LoLim	Tracking to safety value	**Cont. controller:** Open, Close, Stop = 0 **Step controller with position feedback:** Depending on Rbk and MV, the output signals Open, Close and Stop are generated using the algorithm of a positioner. **Pulse controller:** Depending on MV, the output signals Open and Close are generated using the algorithm of a pulse controller (Stop = 0).
0	1	-	-	-	Prepared FM analog input	MV_HiLim MV_LoLim	Tracking to an FM analog input	
0	0	1	-	-	Man	ManHiLim ManLoLim	Manual mode, set by the operator	
0	0	0	1	-	MV_Trk	MV_HiLim MV_LoLim	Tracking to block input MV_Trk	
0	0	0	0	1	AdvCoMV	MV_HiLim MV_LoLim	Higher-level program mode	
0	0	0	0	0	P_Part + I_Part + D_Part + FFwd	MV_HiLim MV_LoLim	Automatic mode (PID algorithm)	

Generating actuating signals for step controllers without position feedback (WithRbk = 0)

The manipulated variable signals Open, Close and Stop can be generated as follows:

ManAct	Open, Close, Stop	State
1	The output signals are generated using input signals OpenOp/Li, CloseOp/Li or StopOp/Li	Manual mode, set by the operator
0	The output signals are generated using PID output variables P_Part, I_Part, D_Part and FFwd	Automatic mode (PID algorithm)

Tracking and limiting a manipulated variable (cont. controller, step controller with position feedback and pulse controller)

The block provides the standard function Tracking and limiting a manipulated variable (Page 192).

Neutral position

The controller modules have their own mechanism for feedforwarding a safety value (see manual for Temperature Controller FM 355-2 or manual for Controller Module FM 355).

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `ModErr`
- `ParFM_Err`
- `PerACCErr`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

"Actuator active" information

For continuous and pulse controllers: If the manipulated variable `MV` is greater than the minimum manual limit `ManLoLim`, this is recognized as actuator active.

For step controllers: If the parameter `FbkClosed` = 0 , this is recognized as "Actuator active".

This status can be used to indicate a customized symbol in the process image, for example, and is saved in the status word (see Status word section in Description of FmCont (Page 587)).

Limit monitoring of position feedback (cont. controller, step controller with position feedback and pulse controller)

The block provides the standard function Limit monitoring of the feedback (Page 94).

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

External/internal setpoint specification

The block provides the standard function Setpoint specification - internal/external (Page 127).

Setpoint limiting for external setpoints

The block provides the standard function Setpoint limiting for external setpoints (Page 192).

Gradient limit of the setpoint

The block provides the standard function Gradient limit of the setpoint (Page 124).

Using setpoint ramp

The block provides the standard function Using setpoint ramp (Page 123).

Tracking setpoint in manual mode

The block provides the standard function Tracking setpoint in manual mode (Page 192).

Simulating signals

The block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Process value (`SimPV`, `SimPV_Li`)
- Position feedback (`SimRbk`, `SimRbkLi`)

Note

The simulated process value `SimPV` only affects alarm processing and not the PID algorithm in the control module.

Limit monitoring of the process value

The block provides the standard function Limit monitoring of the process value (Page 86).

Error signal generation and dead band

The block provides the standard function Error signal generation and dead band (Page 188).

Limit monitoring of error signal

The block provides the standard function Limit monitoring of setpoint, manipulated variable and control deviation (Page 95).

Inverting control direction

The block provides the standard function Inverting control direction (Page 188).

Physical standardization of setpoint, manipulated variable and process value

Controller gain `Gain` is entered either using a physical variable or as standardized value.

- `Gain` as a physical variable:
  The standardized variables retain their default values:
  - `NormPV.High` = 100 and `NormPV.Low` = 0
  - `NormMV.High` = 100 and `NormMV.Low` = 0

  For step controllers with/without position feedback and pulse controllers, the values of `NormMV.High` and `NormMV.Low` are not taken into account. The algorithm uses default values 0 and 100 for internal calculations.
  The effective gain is: `GainEff` = `Gain`
- Entering a standardized `Gain` (dimensionless):
  Change the standardized variables to the actual range of the process values and manipulated variables.
  - Internal and external setpoints; the process value and corresponding parameters are entered according to the physical measuring range of the process value.

  Continuous controller, pulse controller:
  - The manual value, the tracking value of the manipulated variable, feedforward control and the corresponding parameters are set according to the physical measuring range of the manipulated variable.

  Step controller with position feedback:
  - The manual parameter, the tracking value of the manipulated variable, feedforward control and the corresponding parameters are entered as a percentage 0 ... 100.

  Step controller without position feedback:
  - No physical measuring range available.

  The effective gain is:
  - Step controller with/without position feedback:
    `GainEff` = 100.0 / (`NormPV.High` - `NormPV.Low`) · `Gain`
  - Continuous controller, pulse controller:
    `GainEff` = (`NormMV.High` - `NormMV.Low`) / (`NormPV.High` - `NormPV.Low`) · `Gain`

Selecting a unit of measure

The block provides the standard function Selecting a unit of measure (Page 207).

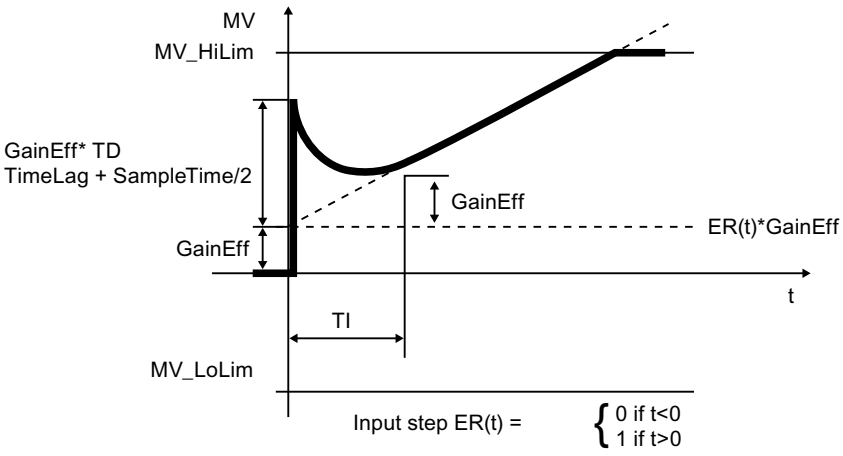
PID algorithm

The manipulated variable is generated in automatic mode according to the following algorithm:

`MV` = `GainEff` · (1 + 1 / (`TI` · s) + (`TD` · s) / (1 + `TD` / `DiffGain` · s)) · `ER`

Where: s = Complex number

The following step response occurs:



Note

This formula describes a standard application where the P, I and D components are activated and the P and D components are not in the feedback circuit (`PropSel` = 1, `TI` ≠ 0, `D_InSel` = 0 and `P_FbkSel` = 0).

The D component delay is derived from `TD` / `DiffGain`.

- The P component can be shut down by `PropSel` = 0 .
- The I component can be shut down by `TI` = 0 .
- The D component can be shut down by `TD` = 0 .

Structure segmentation at controllers

The PID controller algorithm of FM 355 features structure segmentation. It is activated via the `P_FbkSel` and `D_InSel` parameters. The precise functionality is described in the FM 355 manual.

Anti-windup

The PID control algorithm of FM 355 has an anti-windup function. The I component is frozen or tracked after the manipulated variable has reached its limits (`MV_HiLim` or `MV_LoLim`).

Feedforwarding and limiting disturbance variables

The block provides a function for activating the disturbance variable feedforward. The precise functionality is described in the FM 355 manual.

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

- Signal status for the setpoint value `SP`:
  The signal status of the SP output parameter is always equivalent to the signal status of input parameter `SP_Ext` or `SP_Int`, depending on how the setpoint is specified. If the internal setpoint `SP_Int` is used, the signal status is always output as `16#80`.
- Signal status for `PV_Out`, `RbkOut`, `Open`, `Close`, `Stop`:
  The signal status is always `16#60` when simulation is activated.
  When the `ModErr.Value, ChFM_Err, ParFM_Err` module error occurs, the signal status of `PV_Out` is always `16#0`. `RbkOut` is always `16#0` for step controllers with position feedback
  Otherwise, the following applies:
  ```
  PV_Out.ST: 16#80
  Step controller: RbkOut.ST: = 16#80
  Continuous controller or pulse controller: RbkOut.ST: = Rbk.ST
  Open.ST := 16#80;
  Close.ST := 16#80;
  Stop.ST := 16#80;
  ```
- Signal status of the error signal `ER`:
  The signal status of output parameter `ER` is obtained from the worst signal status of the two output parameters `PV_Out` and `SP` and is output. The signal status `16#60` (external simulation) is suppressed because the block acts as a sink with external simulation.
  Signal status for `FbkOpnOut, FbkClsOut:`
  ```
  FbkOpnOut.ST := FbkOpened.ST;
  FbkClsOut.ST := FbkClosed.ST;
  ```
- Signal status for the manipulated variable `MV`:
  The status signal from the output parameter `MV` is always set to `16#80` in "manual mode" and for step controllers without position feedback.
  In "automatic mode", the signal status for continuous controllers or pulse controllers is formed from the following parameters:
  `RbkOut.STFFwdOut.STER.ST`With step controllers, the `FbkOpnOut.STFbkClsOut.ST`parameters are also included. The signal status `16#60` (external simulation) is suppressed because the block acts as a sink with external simulation.
- Worst signal status:
  The worst signal status `ST_Worst` for the block is formed from:
  - `PV_Out.ST`;
  - `SP:ST`;
  - `FFwdOut.ST`;
  - `RbkOut.ST`;

  With step controllers (FM355 = 0, StepCon = 1), the following are also included:
  - `FbkOpnOut.ST`;
  - `FbkClsOut.ST`;

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
4	Setting switch or button mode (Page 166)
6	Ramp rate calculation (Page 178)
10	Condition monitoring information at the channel blocks (Page 144)
22	Update acknowledgment and error status of the message call (Page 159)
23	SP following PV in open loop has no priority over SP_Ext and SP limits (Page 178)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)

Operator permissions

The block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode" `AutModOp`
1	1 = Operator can switch to "manual mode" `ManModOp`
2	1 = Operator can switch to "Out of service" mode `OosOp`
3	1 = Operator can switch to "program mode" `AdvCoEn`
4	1 = Operator can switch the setpoint to "external" `SP_ExtOp`
5	1 = Operator can switch the setpoint to "internal" `SP_IntOp`
6	1 = Operator can change the internal setpoint `SP_Int`
7	Continuous controllers, pulse controllers or step controllers with position feedback: 1 = Operator can change the manual parameter `Man` Step controller without position feedback: 1 = Operator can change the manual operation signals `OpenOp`, `StopOp`, `CloseOp`
8	1 = Operator can change operation high limit of the setpoint `SP_InHiLim`
9	1 = Operator can change operation low limit of the setpoint `SP_InLoLim`
10	1 = Operator can change the operation high limit of the manipulated variable `ManHiLim`
11	1 = Operator can change the operation low limit of the manipulated variable `ManLoLim`
12	1 = Operator can enable the setpoint's gradient limitation function `SP_RateOn`
13	1 = Operator can change the setpoint's high limit for the ramp `SP_UpRaLim`
14	1 = Operator can change the setpoint's low limit for the ramp `SP_DnRaLim`

Bit	Function
15	1 = Operator can switch between the time value or the value for the ramp `SP_RmpModTime`
16	1 = Operator can change the ramp time `SP_RmpTime`
17	1 = Operator can change the target setpoint `SP_RmpTarget` for the setpoint ramp
18	1 = Operator can enable the setpoint ramp function `SP_RmpOn`
19	Not used
20	1 = Operator can enable the track setpoint in manual mode function `SP_TrkPV`
21	1 = Operator can enable the bumpless switchover from external to internal `SP_TrkExt`
22	1 = Operator can change the gain parameter `Gain`
23	1 = Operator can change the integral time parameter `TI`
24	1 = Operator can change the derivative time parameter `TD`
25	1 = Operator can change the derivative gain parameter `DiffGain`
26	1 = Operator can change the dead band parameter `DeadBand`
27	Not used
28	1 = Operator can change the integral time parameter `MotorTime`
29	1 = Operator can change the integral time parameter `PulseTime`
30	1 = Operator can change the integral time parameter `BreakTime`
31	Not used

The block has the following permissions for the `OS1Perm` parameter:

Bit	Function
0	1 = Operator can change the limit (process value) `PV_AH_Lim` for the high alarm
1	1 = Operator can change the limit (process value) `PV_WH_Lim` for the high warning
2	1 = Operator can change the limit (process value) `PV_TH_Lim` for the high tolerance
3	1 = Operator can change the hysteresis (process value) `PV_Hyst`
4	1 = Operator can change the limit (process value) `PV_TL_Lim` for the low tolerance
5	1 = Operator can change the limit (process value) `PV_WL_Lim` for the low warning
6	1 = Operator can change the limit (process value) `PV_AL_Lim` for the low alarm
7	1 = Operator can change the limit (error signal) `ER_AH_Lim` for the high alarm
8	1 = Operator can change the hysteresis (error signal) `ER_Hyst`
9	1 = Operator can change the limit (error signal) `ER_AL_Lim` for the low alarm
10	1 = Operator can change the limit (position feedback) `RbkWH_Lim` for the high warning
11	1 = Operator can change the hysteresis (position feedback) `RbkHyst`
12	1 = Operator can change the limit (position feedback) `RbkWL_Lim` for the low warning
13	1 = Operator can open the valve
14	1 = Operator can close the valve
15	1 = Operator can stop the valve
16	1 = Operator can activate the Simulation function `SimOn`
17	1 = Operator can activate the Release for maintenance function `MS_RelOp`
18	1 = Operator can activate / deactivate messages via `PV_AH_MsgEn`
19	1 = Operator can activate / deactivate messages via `PV_WH_MsgEn`
20	1 = Operator can activate / deactivate messages via `PV_TH_MsgEn`
21	1 = Operator can activate / deactivate messages via `PV_TL_MsgEn`

Bit	Function
22	1 = Operator can activate / deactivate messages via PV_WL_MsgEn
23	1 = Operator can activate / deactivate messages via PV_AL_MsgEn
24	1 = Operator can activate / deactivate messages via ER_AH_MsgEn
25	1 = Operator can activate / deactivate messages via ER_AL_MsgEn
26	1 = Operator can activate / deactivate messages via RbkWH_MsgEn
27	1 = Operator can activate / deactivate messages via RbkWL_MsgEn
28	1 = Operator can change the simulation value SimPV
29	1 = Operator can change the simulation value SimRbk
30 - 31	Not used

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

Release for maintenance

The block provides the standard function Release for maintenance (Page 64).

Generating instance-specific messages

The block provides the standard function Generating instance-specific messages (Page 200) without the time stamp function in the I/O.

Suppressing messages using the MsgLock parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Specifying the display area for process and setpoint values as well as operations

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- OpenOp
- StopOp
- CloseOp

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter EventTsOut of the block EventTs or Event16Ts is connected to the input parameter EventTsIn of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

See also

FmCont messaging (Page 604)

FmCont I/Os (Page 606)

FmCont block diagram (Page 622)

FmCont modes (Page 591)

FmCont error handling (Page 602)

EventTs functions (Page 1634)

5.3.4 FmCont error handling

Error handling of FmCont

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Control system fault (CSF)

Overview of error numbers

The ErrorNum I/O can be used to output various error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
30	The value of PV can no longer be displayed in the REAL number field.
31	The value of SP_Ext can no longer be displayed in the REAL number field.

Error number	Meaning of the error number
33	The value of MV_Trk can no longer be displayed in the REAL number field.
35	The value of Rbk can no longer be displayed in the REAL number field.
36	The value of MV can no longer be displayed in the REAL number field.
50	The controller cannot be switched to program mode, because program mode with default setpoint (AdvCoModSP = 0) is not possible for step controllers without position feedback (WithRbk = 0).
51	AutModLi = 1 and ManModLi = 1 SP_LiOp = 1 and SP_IntLi = 1 and SP_ExtLi = 1 OpenLi = 1 and StopLi = 1 CloseLi = 1 and StopLi = 1 OpenLi = 1 and CloseLi = 1
59	= 1, "Gain is negative"
60	|TI| < SampleTime / 2
61	|TD| < SampleTime
62	DiffGain < 1 or DiffGain > 10
63	TD / DiffGain < SampleTime / 2
64	PropFacSP < 0 or PropFacSP > 1
66	NormPV_High = NormPV_Low
67	MotorTime < SampleTime
68	PulseTime < SampleTime
69	BreakTime < SampleTime
70	Channel < 1 or Channel > 4
71	(D_InSel < 0 or D_InSel > 4) and D_InSel ≠ 17

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

FmCont block diagram (Page 622)

FmCont I/Os (Page 606)

Description of FmCont (Page 587)

FmCont modes (Page 591)

FmCont functions (Page 592)

FmCont messaging (Page 604)

Setting switch or button mode (Page 166)

5.3.5 FmCont messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
`MsgEvId2`	SIG 6	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a process control fault is triggered (`MsgEvId2`, SIG 6).

Process messages

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 1	Alarm - high	$$BlockComment$$ `PV` - high alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ `PV` - high warning limit violated
	SIG 3	Tolerance - high	$$BlockComment$$ `PV` - high tolerance limit violated
	SIG 4	Tolerance - low	$$BlockComment$$ `PV` - low tolerance limit violated
	SIG 5	Warning - low	$$BlockComment$$ `PV` - low warning limit violated
	SIG 6	Alarm - low	$$BlockComment$$ `PV` - low alarm limit violated
	SIG 7	Alarm - high	$$BlockComment$$ `ER` - high alarm limit violated
	SIG 8	Alarm - low	$$BlockComment$$ `ER` - low alarm limit violated

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 7	Warning - high	$$BlockComment$$ Rbk - high warning limit violated
	SIG 8	Warning - low	$$BlockComment$$ Rbk - low warning limit violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You can use up to four instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 1	AS process control message - fault	$$BlockComment$$ External message 1 Status 16#@5%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ External message 2 Status 16#@6%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ External message 3 Status 16#@7%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 4 Status 16#@8%x@

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	Process value PV_Out
5	Control deviation ER
6	ExtVa106
7	ExtVa107
8	Not allocated
9	Not allocated
10	Not allocated

The associated values 6 ... 7 are allocated to the parameters ExtVa106 ... ExtVa107 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

Associated values for message instance MsgEvId2

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	Position feedback Rbk
5	Signal status ExtMsg1
6	Signal status ExtMsg2
7	Signal status ExtMsg3
8	Signal status ExtMsg4
9	ExtVa209
10	ExtVa210

The associated values 9 ... 10 are allocated to the parameters ExtVa209 ... ExtVa210 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

FmCont block diagram (Page 622)

FmCont modes (Page 591)

FmCont error handling (Page 602)

5.3.6 FmCont I/Os

I/Os of FmCont

Input parameters

Parameter	Description	Type	Default
AccMode*	1 = Transfer of operating parameters SubN1_ID, SubN2_ID, RackNo, SlotNo and Channel to internal processing	BOOL	1
AdvCoEn	1 = Enable "program mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
AdvCoModSP	Type of "program mode": 1 = Setpoint specification 0 = Manipulated variable specification	BOOL	1
AdvCoMstrOn	Activate (0-1) or deactivate (1-0) "program mode" via edge transition	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoMV	Specified value from the external program	REAL	0.0
AdvCoOn*	1 = Enable "program mode" via faceplate	BOOL	0
AutModLi*	1 = "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 0)	BOOL	0
BatchEn	1 = Enable allocation for batch control	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BreakTime*	Minimum break duration [s]	REAL	1.0
Channel	Controller channel number (1..4)	INT	1
CloseLi*	1 = Close via interconnection or CFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseOp*	1 = Close via operator	BOOL	0
CoordNo	Coordination number	INT	0
CPI_In	Input for control performance index, which is calculated by the assigned ConPerMon block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
D_InSel*	Input for differentiator: 0 = Error signal 1..4 = Channel 1..4 17 = Actual value to feedback	INT	0
DeadBand*	Width of dead band	REAL	0.0
DiffGain*	Gain of differentiator [1...10] DiffGain = TD/(delay time of D component)	STRUCT • Value: REAL • ST: BYTE	- • 5.0 • 16#80
EN	1 = Called block will be processed	BOOL	1
ER_A_DC*	Delay for incoming alarms during error signal monitoring	REAL	0.0

Parameter	Description	Type	Default
ER_A_DG*	Delay for outgoing alarms during error signal monitoring	REAL	0.0
ER_AH_En	1 = Activate alarm (high) for error signal monitoring	BOOL	1
ER_AH_Lim	Alarm limit (high) for error signal monitoring	REAL	100.0
ER_AH_MsgEn	1 = Activate messages for alarm (high) for error signal monitoring	BOOL	1
ER_AL_En	1 = Activate alarm (low) for error signal monitoring	BOOL	1
ER_AL_Lim	Alarm limit (low) for error signal monitoring	REAL	-100.0
ER_AL_MsgEn	1 = Activate messages for alarm (low) for error signal monitoring	BOOL	1
ER_Hyst	Alarm hysteresis for error signal	REAL	1.0
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg4	Binary input for freely selectable message 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa209	Associated value 9 for messages (MsgEvID2)	ANY	
ExtVa210	Associated value 10 for messages (MsgEvID2)	ANY	

Parameter	Description	Type	Default
FbkClosed	Low limit stop signal of position feedback	STRUCT • Value: BOOL • ST: BYTE	- 0 16#80
FbkOpened	High limit stop signal of position feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Feature	I/O for additional functions (Page 592)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
FM355	Module type: 0: FM 355 S; 1: FM 355 C	BOOL	0
FuzOptOn*	Fuzzy optimization	BOOL	0
Gain*	Proportional gain Gain.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#FF
LogAddr	Logical address FM 355	INT	0
Man*	Manual specification for the manipulated variable	REAL	0.0
ManHiLim*	Limit (high) for manual parameter Man	REAL	100.0
ManLoLim*	Limit (low) for manual parameter Man	REAL	0.0
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
Mode	Operating mode	DWORD	16#00000000
ModLiOp	Operating mode switchover between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MotorTime*	Motor actuating time [s]	REAL	30.0
MS	Maintenance status	DWORD	0
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ms_Ext	External maintenance status	DWORD	0
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MsgEvID1	Message number (assigned automatically)	DWORD	16#00000000
MsgEvID2	Message number (assigned automatically)	DWORD	16#00000000

Parameter	Description	Type	Default
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_FmTrkOn	1 = Manipulated variable tracking in the FM	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_HiLim*	Limit (high) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
MV_LoLim*	Limit (low) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_OpScale	OS display range for manipulated variable MV	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
MV_Safe*	Neutral position manipulated variable	REAL	0.0
MV_SafeOn	1 = Neutral position manipulated variable MV_Safe at output MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_Trk*	Tracking value for the manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_TrkOn	1 = Tracking of manipulated variable MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_Unit	Unit of measure for manipulated variable	INT	1342
NegGain*	0 = Effective proportional gain GainEff is positive 1 = Effective proportional gain GainEff is negative	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NormMV	Manipulated variable range (MV) for standardizing the proportional gain (GAIN)	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
NormPV	Process value range (PV) for standardizing the proportional gain (GAIN)	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
Occupied	Occupied by batch control	BOOL	0
OosLi	Edge transition (0-1) = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0

Parameter	Description	Type	Default
OP_Sel*	Operation via OP 0 = "Off" (P bus) 1 = "On" (K bus)	BOOL	0
OpenLi*	1 = Open via interconnection or CFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenOp*	1 = Open via operator	BOOL	0
OptimEn	1 = Enable optimization of PID parameters by PID tuner	BOOL	0
OptimOcc	1 = Optimization running	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 592)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OS1Perm	I/O for operator permissions (Page 592)	STRUCT • Bit 0: BOOL • Bit 18: BOOL • Bit 19: BOOL • Bit 31: BOOL	- • 1 • 1 • 1 • 1
P_FbkSel*	1 = P component in feedback	BOOL	0
PropSel*	1 = Activate P component	BOOL	1
PulseTime*	Minimum pulse duration [s]	REAL	1.0
PV_A_DC*	Delay time for incoming PV alarms [s]	REAL	0.0
PV_A_DG*	Delay time for outgoing PV alarms [s]	REAL	0.0
PV_AH_En	1 = Enable PV alarm limit (high)	BOOL	1
PV_AH_Lim	Limit PV alarm (high)	REAL	95.0
PV_AH_MsgEn	1 = Enable PV alarm (high) message	BOOL	1
PV_AL_En	1 = Enable PV alarm limit (low)	BOOL	1
PV_AL_Lim	PV alarm limit (low)	REAL	5.0
PV_AL_MsgEn	1 = Enable PV alarm (low) message	BOOL	1
PV_Hyst	Hysteresis for PV alarm, warning and tolerance limits	REAL	1.0
PV_OpScale	Limit for scale in PV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
PV_T_DC*	Delay time for incoming PV tolerance messages [s]	REAL	0.0
PV_T_DG*	Delay time for outgoing PV tolerance messages [s]	REAL	0.0

Parameter	Description	Type	Default
PV_TH_En	1 = Enable PV tolerance limit (high)	BOOL	0
PV_TH_Lim	Limit PV tolerance message (high)	REAL	85.0
PV_TH_MsgEn	1 = Enable message for PV tolerance message (high)	BOOL	1
PV_TL_En	1 = Enable PV tolerance limit (low)	BOOL	0
PV_TL_Lim	Limit PV tolerance message (low)	REAL	15.0
PV_TL_MsgEn	1 = Activate message for PV tolerance message (low)	BOOL	1
PV_Unit	Unit of measure for process value	INT	1001
PV_W_DC*	Delay time for incoming PV warnings [s]	REAL	0.0
PV_W_DG*	Delay time for outgoing PV warnings [s]	REAL	0.0
PV_WH_En	1 = Enable PV warning limit (high)	BOOL	1
PV_WH_Lim	Limit PV warning (high)	REAL	90.0
PV_WH_MsgEn	1 = Enable PV warning (high) message	BOOL	1
PV_WL_En	1 = Enable PV warning limit (low)	BOOL	1
PV_WL_Lim	Limit PV warning (low)	REAL	10.0
PV_WL_MsgEn	1 = Enable PV warning (low) message	BOOL	1
RackNo	Rack number	BYTE	16#FF
Rbk*	Position feedback for display on OS	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
RbkHyst	Alarm hysteresis for position feedback	REAL	1.0
RbkWH_En	1 = Enable warning (high) for position feedback	BOOL	1
RbkWH_Lim	Limit for position feedback of warning (high)	REAL	100.0
RbkWH_MsgEn	1 = Enable messages for warning (high) for position feedback	BOOL	1
RbkWL_En	1 = Enable warning (low) for position feedback	BOOL	1
RbkWL_Lim	Limit for position feedback of warning (low)	REAL	0.0
RbkWL_MsgEn	1 = Enable messages for warning (low) for position feedback	BOOL	1
RefStdDevIn	Reference value of PV standard deviation (sigma) in defined "good" state of control loop	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
S_RbkOnPIDTun	Simulation of position feedback on; For PCS 7 PID tuner only	BOOL	0
S_RbkPIDTun	Simulated position feedback	REAL	50.0
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1

Parameter	Description	Type	Default
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimPV*	Process value used for SimOn = 1	REAL	0.0
SimPV_Li	Process value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimRbk*	Position feedback used for SimOn = 1	REAL	0.0
SimRbkLi	Position feedback used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SlotNo	Slot number	BYTE	16#FF
SP_DnRaLim	Limit (low) for the gradient of the setpoint [SP_Unit/s]	REAL	100.0
SP_ExHiLim*	Limit (high) for external setpoint	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP_ExLoLim*	Limit (low) for external setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_Ext*	External setpoint of - (to interconnection)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtLi*	1 = Select external setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOp*	1 = Select external setpoint (via operator)	BOOL	0
SP_InHiLim*	Limit (high) of internal setpoint	REAL	100.0
SP_InLoLim*	Limit (low) of internal setpoint	REAL	0.0
SP_Int*	Internal setpoint for operation	REAL	0.0

Parameter	Description	Type	Default
SP_IntLi*	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_IntOp*	1 = Select internal setpoint (via operator)	BOOL	0
SP_LiOp	Select setpoint source (internal/external): 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_RateOn*	1 = Activate limitation of setpoint gradients	BOOL	0
SP_RmpModTime	1 = Use time (SP_RmpTime) for setpoint ramp, 0 = Use gradient	BOOL	0
SP_RmpOn*	1 = Activate setpoint ramp to target setpoint SP_RmpTarget	BOOL	0
SP_RmpTarget*	Target setpoint for setpoint ramp	REAL	0.0
SP_RmpTime*	Time for setpoint ramp [s] from current SP up to SP_RmpTarget	REAL	0.0
SP_TrkExt	1 = Bumpless switchover from external to internal setpoint active	BOOL	1
SP_TrkPV	1 = Setpoint follows PV in "manual mode" and with tracking	BOOL	0
SP_UpRaLim	Gradient limit (high) for the setpoint [SP_Unit/s]	REAL	100.0
StepCon	Controller type in the FM 355 S: 0 = Pulse controller 1 = Step controller	BOOL	0
StepNo	Batch step number	DWORD	16#00000000
StopLi*	1 = Stop via interconnection or CFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopOp*	1 = Stop via operator	BOOL	0
SubN1_ID	ID of the primary DP master system	BYTE	16#FF
SubN2_ID	ID of the redundant DP master system	BYTE	16#FF
TD*	Derivative time [s] TD.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
TextRef	Text reference	WORD	0
TI*	Integral time [s] TI.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#FF

Parameter	Description	Type	Default
TimeFactor	Time unit: 0 = Seconds 1 = Minutes 2 = Hours	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
EnCoord	Current coordination number	STRUCT • CO_ACT : INT	- • 0
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
AdvCoAct	1 = "Program mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoRdy	1 = "Program mode" available	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutAct	1 = "Automatic mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ChFM_Err	1 = Channel error on the module	BOOL	0
Close	Control output: 1 = Closed is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EnCoNum	Coordination number	BYTE	16#00
ENO	1 = Block algorithm completed without errors	BOOL	0
ER	Error signal	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
ER_AH_Act	1 = Alarm limit (high) for control deviation violated. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ER_AL_Act	1 = Alarm limit (low) for control deviation violated. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum	Output of pending error number; for error numbers that can be output by this block, see FmCont error handling (Page 602)	INT	-1
FbkClsOut	1 = Low limit stop of the position feedback reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpnOut	1 = High limit stop of the position feedback reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FFwdOut	Disturbance variable generated in the FM	STRUCT Value: REAL ST: BYTE	- 0.0 16#80
FuzCon	Controller type: 0 = PID controller 1 = Fuzzy controller	BOOL	0
FuzOptAct	1 = Optimization of fuzzy controller active	BOOL	0
FuzSP_PV_Act	Fuzzy controller display: Setpoint < actual value	BOOL	0
GainEff	Effective proportional gain depends on NegGain, Gain, NormPV, and NormMV	REAL	1.0
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LoopClosed	1 = Control loop closed 0 = Control loop open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80

Parameter	Description	Type	Default
ManARW_Act	1 = Tracking mode or anti-reset windup by secondary controller	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManHiOut	Limit (high) for "manual mode", corresponds to input parameter ManHiLim	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
ManLoOut	Limit (low) for "manual mode", corresponds to input parameter ManLoLim	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ModErr	1 = Module error	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgAckn2	Message acknowledgement status 2 (output ACK_STATE of second ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgErr2	Alarm error 2 (output ERROR of second ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
MsgStat2	Message status 2 (output STATUS of second ALARM_8P)	WORD	16#0000
MV	Manipulated variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_FmTrkAct	1 = Track manipulated variable in the FM enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_HiAct	1 = Limit (high) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MV_LoAct	1 = Limit (low) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_SafeAct	1 = Neutral position manipulated variable of the FM enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_SpliA	Manipulated variable A of split-range function	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_SpliB	Manipulated variable B of split-range function	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_UnitOut	Unit of measure for manipulated variable, for interconnecting to the MV_Unit input parameter of the ConPerMon block	INT	0
MV_Visible	1 = MV display visible Evaluated by block icon	BOOL	0
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Open	Control output: 1 = Open is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS1PermOut	Display of OS_Perm1	DWORD	16#FFFFFFFF
OS1PermLog	Display of OS1Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
ParFM_Err	1 = Direct parameter-assignment error of the FM or input Channel configured incorrectly	BOOL	0
PerAccErr	1 = I/O access error	BOOL	0
PV	Process value of the module	STRUCT • Value: REAL • ST:BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
PV_AH_Act	1 = PV alarm (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AL_Act	1 = PV alarm (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	Output for process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_TH_Act	1 = PV tolerance message (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_TL_Act	1 = PV tolerance message (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_ToleHi	Limit (high) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_ToleLo	Limit (low) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_UnitOut	Unit of measure for process value, for interconnecting to the PV_Unit input parameter of the ConPerMon block	INT	0
PV_WH_Act	1 = PV warning (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_WL_Act	1 = PV warning (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
RbkOut	Output for position feedback	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RbkVisible	1 = Rbk display visible Evaluated by block icon	BOOL	0
RbkWH_Act	1 = Warning (high) for position feedback active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkWL_Act	1 = Warning (low) for position feedback active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RetValue	Return value of WRREC / RDREC	WORD	16#0000
SP	Setpoint used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_DnRaAct	1 = Negative gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExHiAct	1 = Limit (high) for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExLoAct	1 = Limit (low) for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtAct	1 = External setpoint is active 0 = Internal setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOut	External setpoint, corresponds to input parameter SP_Ext	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_InHiOut	Limit (high) for SP_Int corresponds to input parameter SP_InHiLim	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80

Parameter	Description	Type	Default
SP_InLoOut	Limit (low) for SP_Int corresponds to input parameter SP_InLoLim	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_RateTarget	Target setpoint for the gradient limitation	REAL	0.0
SP_UpRaAct	Positive gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
SplitRange	1 = Split-range function has been activated	BOOL	0
Status1	Status word 1 (Page 587)	DWORD	16#00000000
Status2	Status word 2 (Page 587)	DWORD	16#00000000
Status3	Status word 2 (Page 587)	DWORD	16#00000000
Stop	Control output: 1 =Stopped is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SumMsgAct	1 = Active hardware interrupt	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
WithRbk	Controller type: 0 = Step controller without position feedback 1 = Step controller with position feedback	BOOL	0

See also

FmCont messaging (Page 604)

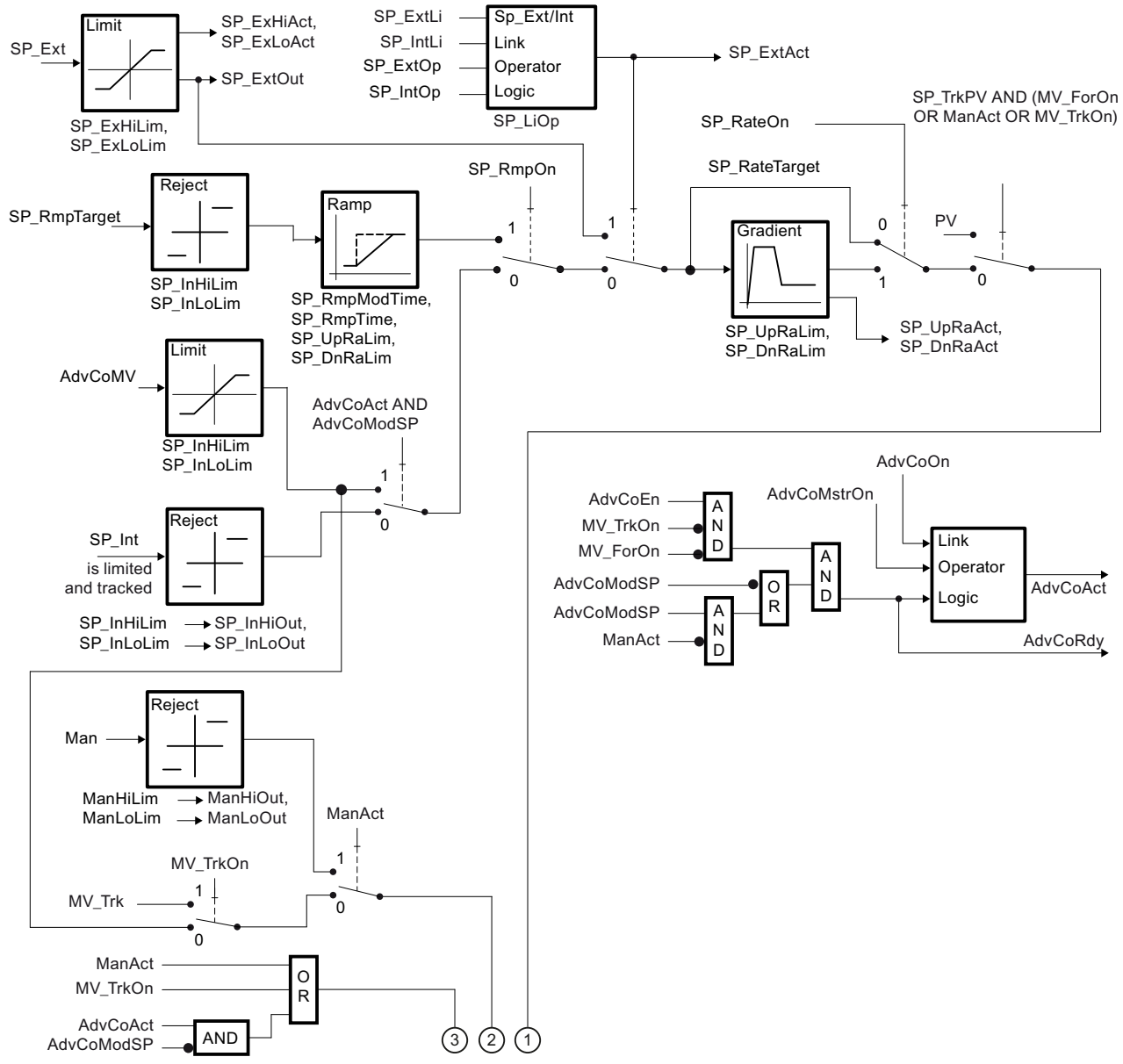
FmCont block diagram (Page 622)

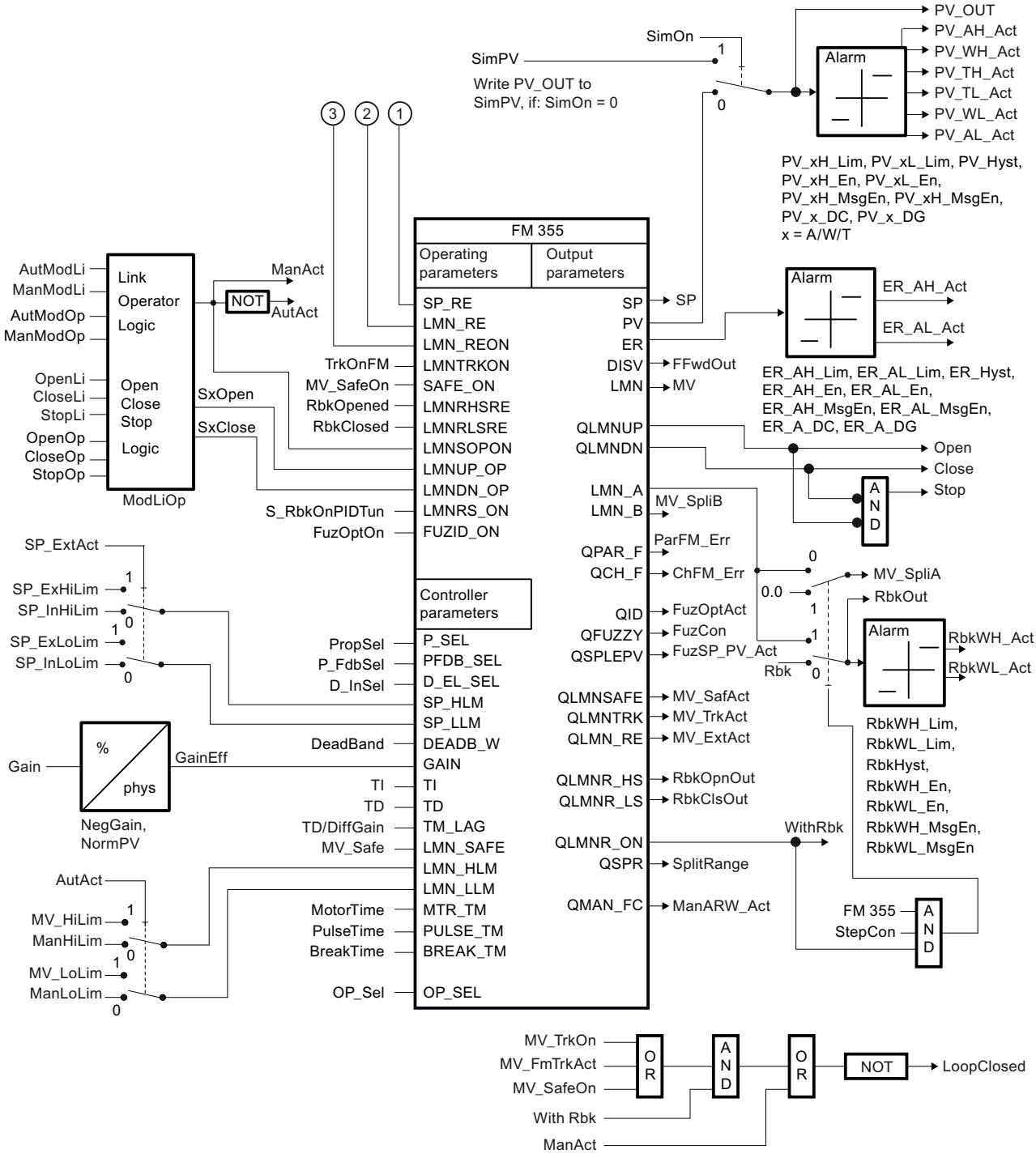
FmCont modes (Page 591)

Neutral position for motors, valves and controllers (Page 48)

5.3.7 FmCont block diagram

FmCont block diagram





See also

FmCont I/Os (Page 606)

FmCont messaging (Page 604)

FmCont error handling (Page 602)

FmCont functions (Page 592)

FmCont modes (Page 591)

Description of FmCont (Page 587)

5.3.8 Operator control and monitoring

5.3.8.1 FmCont views

Views of the FmCont block

The block FmCont provides the following views:

- FM controllers standard view (analog) (Page 255)
- FM controllers standard view (pulse controller) (Page 259)
- FM controllers standard view (step controller with position feedback) (Page 263)
- FM controllers standard view (step controller without position feedback) (Page 267)
- Alarm view (Page 296)
- Limit view of FM controllers (Page 282)
- Trend view (Page 299)
- Ramp view (Page 294)
- Parameter view of FM controllers (Page 278)
- Preview of FM controllers (Page 291)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icons for PID and FM controller (Page 235)

Refer to the Structure of the faceplate (Page 242) and Block icon structure (Page 226) sections for general information about the faceplate and block icon.

5.4 FmTemp - Interface to temperature controller modules FM 355-2

5.4.1 Description of FmTemp

Object name (type + number) and family

Type + number: FB 1819

Family: Control

Area of application for FmTemp

The block is used for the following applications:

- Fixed setpoint control
- Cascade control
- Ratio control
- Split-range control

How it works

Block FmTemp is used to interface the FM 355-2 temperature controller modules.

FmTemp can be used for the C (continuous controllers) and S (step and pulse controllers) module types. It contains the algorithms of the setpoint ramp, the setpoint rise limitation, and the limit monitoring of the process value, the control deviation, and the position feedback. Limit monitoring is not used on the module.

The control function itself (e.g. PID algorithm) is processed on the module. You can use the FmTemp block to monitor all relevant process values and to change all relevant controller parameters.

Application examples of the FM 355-2 and detailed descriptions of the associated input and output parameters can be found in the manual of the FM 355-2. temperature controller.

It is primarily used for controlling temperature processes, but can also control level and flow processes which are not excessively fast, for example.

Module FM 355-2 features online optimization of the PID parameters. You can set the corresponding parameters for performing online optimization in the CFC chart at block FmTemp .

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100). Set the input `LogAddr` to the module address from HW Config and the input `Channel` to the desired controller channel (0 ... 3)..

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The parameter CoordNo is set
- The in/out parameter Mode is interconnected to the corresponding OMODE_xx output parameter of the MOD block.
- The parameter FM 355_2 is set in accordance with the module type C/S
- The in/out parameter EnCoord is interconnected to the output EN_CO_x of the FM_CO block of the Basis Library (x = number of the rack)
- The output EnCoNum is interconnected to the input ENCOx_yy of the FM_CO block (x = number of the rack, yy = coordination number).

For the control loop monitoring to work as planned in the trend view of the controller faceplates, the

S7_xarchive:='Value, shortterm;'

attributes in the process tag types for control loops at the controller function block must be set for the following tags:

- Input parameters:
  - CPI_In
- Output parameters
  - MV
  - MV_HiAct
  - MV_LoAct
  - LoopClosed
  - SP
  - PV_Out
  - PV_ToleHi
  - PV_ToleLo

Startup characteristics

Use the Feature Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at RunUpCyc.

Status word allocation for Status1 parameter

You can find a description for each parameter in section FmTemp I/Os (Page 646).

Status bit	Parameter
0	Occupied
1	BatchEn
2	SimOn

Status bit	Parameter
3	OosAct.Value
4	OosLi.Value
5	AutAct.Value
6	Not used
7	ManAct.Value
8	SP_ExtAct.Value
9	MV_SafeOn.Value
10	MV_TrkOn.Value
11	MV.Value > ManLoLim for continuous or pulse controller NOT FbkClosed.Value for step controller with/without position feedback
12	Open.Value
13	Close.Value
14	Stop.Value
15	FbkOpened.Value
16	FbkClosed.Value
17	SimOn AND ManAct
18	SimOn AND ManAct
19	AdvCoAct
20	1 = Input parameter Rbk is not interconnected (RbkOut.ST = 16#FF)
21	NegGain
22 - 27	Not used
28	1 = Analog controller (FM 355_2 = 1)
29	1 = Pulse controller (FM 355_2 = 0 AND StepCon = 0)
30	1 = Step controller with position feedback (FM 355_2 = 0 AND StepCon = 1 AND WithRbk = 1)
31	1 = Step controller without position feedback (FM 355_2 = 0 AND StepCon = 1 AND WithRbk = 0)

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock
1	PV_AH_Act.Value
2	PV_WH_Act.Value
3	PV_TH_Act.Value
4	PV_TL_Act.Value
5	PV_WL_Act.Value
6	PV_AL_Act.Value
7	PV_AH_En
8	PV_WH_En
9	PV_TH_En
10	PV_TL_En

Status bit	Parameter
11	PV_WL_En
12	PV_AL_En
13	PV_AH_MsgEn
14	PV_WH_MsgEn
15	PV_TH_MsgEn
16	PV_TL_MsgEn
17	PV_WL_MsgEn
18	PV_AL_MsgEn
19	ER_AH_Act.Value
20	ER_AL_Act.Value
21	ER_AH_En
22	ER_AL_En
23	ER_AH_MsgEn
24	ER_AL_MsgEn
25	RbkWH_Act.Value
26	RbkWL_Act.Value
27	RbkWH_En
28	RbkWL_En
29	RbkWH_MsgEn
30	RbkWL_MsgEn
31	MS_RelOp

Status word allocation for `Status3` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via EventTsIn
1	Effective signal 2 of the message block connected via EventTsIn
2	Effective signal 3 of the message block connected via EventTsIn
3	Effective signal 4 of the message block connected via EventTsIn
4	Effective signal 5 of the message block connected via EventTsIn
5	Effective signal 6 of the message block connected via EventTsIn
6	Effective signal 7 of the message block connected via EventTsIn
7	Effective signal 8 of the message block connected via EventTsIn
8	Delay of the PV_AH_Lim message
9	Delay of the PV_WH_Lim message
10	Delay of the PV_TH_Lim message
11	Delay of the PV_TL_Lim message
12	Delay of the PV_WL_Lim message
13	Delay of the PV_AL_Lim message
14	Delay of the ER_AH_Lim message
15	Delay of the ER_AL_Lim message
16	Collection of message delays

Status bit	Parameter
17 - 26	Not used
27	SP_UpRaAct, SP_DnRaAct limits enabled for gradient mode (SP_RateOn = 1)
28	GrpErr.Value
29	RdyToStart.Value
30	SimLiOp.Value
31	Not used

Status word allocation for Status4 parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via EventTsIn
8 - 31	Not used

See also

FmTemp functions (Page 630)
FmTemp messaging (Page 643)
FmTemp modes (Page 629)
FmTemp error handling (Page 642)
FmTemp block diagram (Page 662)

5.4.2 FmTemp modes

FmTemp operating modes

The block can be operated using the following modes:

- Automatic mode (Page 72)
- Manual mode (Page 72)
- Program mode for controllers (Page 78)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and bumpless switchover in the Manual and automatic mode for control blocks (Page 72) section.

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the Manual and automatic mode for control blocks (Page 72) section.

"Program mode for controllers"

General information on "Program mode for controllers" is available in the section Program mode for controllers (Page 78).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of FmTemp (Page 625)

FmTemp functions (Page 630)

FmTemp error handling (Page 642)

FmTemp messaging (Page 643)

FmTemp I/Os (Page 646)

FmTemp block diagram (Page 662)

5.4.3 FmTemp functions

Functions of FmTemp

The functions for this block are listed below.

Module types

FmTemp can be used for the C (continuous controllers) and S (step controllers with and without position feedback and pulse controllers) module types. You can use the following parameters to identify which module type and controller type has been set:

FM 355	StepCon	WithRbk	Module type, controller type
1 or C	-	-	FM 355-2 C: Continuous controller
0 or S	1	1	FM 355-2 S: Step controller with position feedback
0 or S	1	0	FM 355-2 S: Step controller without position feedback
0 or S	0	-	FM 355-2 S: Pulse controller

Generating manipulated variables for continuous controllers, step controllers with position feedback, or pulse controllers

The manipulated variable MV and the actuating signals Open, Close and Stop are generated as follows:

MV_Safe-On	MV_FMTrk On	ManAct	MV_TrkOn	AdvCoAct AND NOT AdvCo-ModSP	MV =	Manipu-lated varia-ble limit	State	Open, Close, Stop
1	-	-	-	-	MV_Safe	MV_HiLim MV_LoLim	Tracking to safety value	**Cont. con-trol-ler:** Open, Close, Stop = 0 **Step con-troller with position feedback:** Depending on Rbk and MV, the out-put signals Open, Close and Stop are generated using the al-gorithm of a positioner. **Pulse con-troller:** De-pending on MV, the out-put signals Open and Close are generated using the al-gorithm of a pulse con-troller (Stop = 0).
0	1	-	-	-	Prepared FM analog input	MV_HiLim MV_LoLim	Tracking to an FM ana-log input	
0	0	1	-	-	Man	ManHiLim ManLoLim	Manual mode, set by the oper-ator	
0	0	0	1	-	MV_Trk	MV_HiLim MV_LoLim	Tracking to block input MV_Trk	
0	0	0	0	1	AdvCoMV	MV_HiLim MV_LoLim	Higher-lev-el program mode	
0	0	0	0	0	P_Part + I_Part + D_Part + FFwd	MV_HiLim MV_LoLim	Automatic mode (PID algorithm)	

Generating actuating signals for step controllers without position feedback (WithRbk = 0)

The manipulated variable signals `Open`, `Close` and `Stop` can be generated as follows:

ManAct	Open, Close, Stop	State	ManAct
1	The output signals are generated using input signals `OpenOp`/`Li`, CloseOp/`Li` or `StopOp`/`Li`	Manual mode, set by the operator	1
0	The output signals are generated using PID output variables `P_Part`, `I_Part`, `D_Part` and `FFwd`	Automatic mode (PID algorithm)	0

Tracking and limiting a manipulated variable (cont. controller, step controller with position feedback and pulse controller)

The block provides the standard function Tracking and limiting a manipulated variable (Page 192).

Neutral position

The controller modules have their own mechanism for feedforwarding a safety value (see manual for Temperature Controller FM 355-2).

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `ModErr`
- `ParFM_Err`
- `PerAccErr`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

"Actuator active" information

For continuous and pulse controllers: If the manipulated variable `MV` is greater than the minimum manual limit `ManLoLim`, this is recognized as actuator active.

For step controllers: If the parameter `FbkClosed` = 0 , this is recognized as "Actuator active".

This status can be used to indicate a customized symbol in the process image, for example, and is saved in the status word (see Status word section in Description of FmTemp (Page 625)).

Limit monitoring of position feedback (cont. controller, step controller with position feedback and pulse controller)

The block provides the standard function Limit monitoring of the feedback (Page 94).

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

External/internal setpoint specification

The block provides the standard function Setpoint specification - internal/external (Page 127).

Setpoint limiting for external setpoints

The block provides the standard function Setpoint limiting for external setpoints (Page 192).

Limitation of rate of change of setpoint

The block provides the standard function Gradient limit of the setpoint (Page 124).

Using setpoint ramp

The block provides the standard function Using setpoint ramp (Page 123).

Tracking setpoint in manual mode

The block provides the standard function Tracking setpoint in manual mode (Page 192).

Simulating signals

The block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Process value (`SimPV`, `SimPV_Li`)
- Position feedback (`SimRbk`, `SimRbkLi`)

Note

The simulated process value `SimPV` only affects alarm processing and not the PID algorithm in the control module.

Limit monitoring of the process value

The block provides the standard function Limit monitoring of the process value (Page 86).

Error signal generation and dead band

The block provides the standard function Error signal generation and dead band (Page 188).

Limit monitoring of error signal

The block provides the standard function Limit monitoring of setpoint, manipulated variable and control deviation (Page 95).

Inverting control direction

The block provides the standard function Inverting control direction (Page 188).

Physical standardization of setpoint, manipulated variable and process value

Controller gain `Gain` is entered either using a physical variable or as standardized value.

- `Gain` as a physical variable:
  The standardized variables retain their default values:
  - `NormPV.High` = 100 and `NormPV.Low` = 0
  - `NormMV.High` = 100 and `NormMV.Low` = 0

  For step controllers with/without position feedback and pulse controllers, the values of `NormMV.High` and `NormMV.Low` are not taken into account. The algorithm uses default values 0 and 100 for internal calculations.
  The effective gain is: `GainEff` = `Gain`
- Entering a standardized `Gain` (dimensionless):
  Change the standardized variables to the actual range of the process values and manipulated variables.
  - Internal and external setpoints; the process value and corresponding parameters are entered according to the physical measuring range of the process value.

  Continuous controller, pulse controller:
  - The manual value, the tracking value of the manipulated variable, disturbance variable feedforward and the corresponding parameters are set according to the physical measuring range of the manipulated variable.

  Step controller with position feedback:
  - The manual parameter, the tracking value of the manipulated variable, disturbance variable feedforward and the corresponding parameters are entered as a percentage 0 ... 100.

  Step controller without position feedback:
  - No physical measuring range available.

  The effective gain is:
  - Step controller with/without position feedback:
    `GainEff` = 100.0 / (`NormPV.High` - `NormPV.Low`) · `Gain`
  - Continuous controller, pulse controller:
    `GainEff` = (`NormMV.High` - `NormMV.Low`) / (`NormPV.High` - `NormPV.Low`) · `Gain`

Selecting a unit of measure

The block provides the standard function Selecting a unit of measure (Page 207).

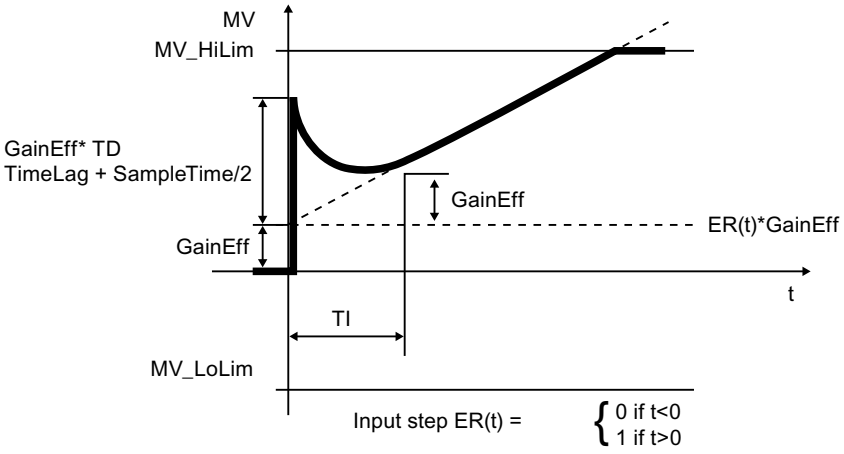
PID algorithm

The manipulated variable is generated in automatic mode according to the following algorithm:

MV = GainEff · (1 + 1 / (TI · s) + (TD · s) / (1 + TD / DiffGain · s)) · ER

Where: s = Complex number

The following step response occurs:



Note

This formula describes a standard application where P, I and D components are activated and the P and D components are not in the feedback circuit (PropSel = 1, TI ≠ 0, D_InSel = 0 and PropFacSP = 1).

The D component delay is derived from TD / DiffGain.

- The P component can be shut down by PropSel = 0 .
- The I component can be shut down by TI = 0 .
- The D component can be shut down by TD = 0 .

Structure segmentation at controllers

The PID controller algorithm of FM 355 features structure segmentation. It is activated via the PropFacSel and D_InSel parameters. The precise functionality is described in the FM 355 manual.

Online optimization of the PID controller parameters

- Optimization sequence
  The optimization sequence is as follows:
  - Create a steady state
  - Set `PID_On` = 1 (if PID parameters are required)
  - Assign parameters of `TunD_MV` / `TunC_MVLMN`
  - Set `TunOn` = 1 (phase 1, preparing for optimization)
  - Start the optimization using a step change in the setpoint or by setting `TunStart`

  If you have not made any configuration errors, the controller optimization is now in phase 2 and `StatusH` is 0.
  - When the point of inflection has been reached (PHASE ≥ 3) evaluate the diagnostics display at the `StatusH` parameter. Phase 0 is reached in a few cycles for process type I and the optimization is completed in full. For process types II and III, the optimization goes to phase 7 (checking the process type). If `StatusH` > 20000 , an estimation error has occurred or the point of inflection has not been reached. In this case, repeat the procedure.
- Result
  - Once optimization is complete, the parameters `PropFacSP`, `GAIN`, `TI`, `TD`, `DiffGain`, `ConZone` are updated (both for the module and at FmTemp). Furthermore, the PI or PID parameter sets are saved on the FM 355-2.
  - The precise procedure is described in the FM 355-2 manual of the temperature controller module.
- Permanent backup of optimized controller parameters
  - Save, compile and download the hardware configuration; the optimized controller parameters are now in the system data block (SDB).
  - Transfer the modified parameters to the offline data management of the CFC via Chart > Readback.

Anti-windup

The PID control algorithm of FM 355 has an anti-windup function. The I component is frozen or tracked after the manipulated variable has reached its limits (`MV_HiLim` or `MV_LoLim`).

Feedforwarding and limiting disturbance variables

The block provides a function for activating the disturbance variable feedforward. The precise functionality is described in the FM 355-2 manual.

Control zone

The block provides the standard function Using control zones (Page 190).

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

- Signal status for the setpoint value SP:
  The signal status of the SP output parameter is always equivalent to the signal status of input parameter SP_Ext or SP_Int, depending on how the setpoint is specified. If the internal setpoint SP_Int is used, the signal status is always output as 16#80.
- Signal status for PV_Out, RbkOut, Open, Close, Stop:
  When the manipulated value (for example, substitute value, simulation, last valid value) is activated, the signal status is always 16#60.
  When the ModErr.Value, ChFM_Err, ParFM_Err module error occurs, the signal status of PV_Out is always 16#0. RbkOut is always 16#0 for step controllers with position feedback
  Otherwise, the following applies:

```
PV_Out.ST: 16#80
Step controller: RbkOut.ST: = 16#80
Continuous controller or pulse controller: RbkOut.ST: = Rbk.ST
Open.ST := 16#80;
Close.ST := 16#80;
Stop.ST := 16#80;
```

- Signal status of the error signal ER:
  The signal status of output parameter ER is obtained from the worst signal status of the two output parameters PV_Out and SP and is output. The signal status 16#60 (external manipulated value - for example, substitute value, simulation, last valid value) is suppressed because the block acts as a sink with external simulation.
  Signal status for FbkOpnOut, FbkClsOut:

```
FbkOpnOut.ST := FbkOpened.ST;
FbkClsOut.ST := FbkClosed.ST;
```

- Signal status for the manipulated variable `MV`:
  The status signal from the output parameter `MV` is always set to `16#80` in "manual mode" and for step controllers without position feedback.
  In "automatic mode", the signal status for continuous controllers or pulse controllers is formed from the following parameters:
  `RbkOut.STFFwdOut.STER.ST`With step controllers, the `FbkOpnOut.STFbkClsOut.ST`parameters are also included. The signal status `16#60` (external manipulated value - for example, substitute value, simulation, last valid value) is suppressed because the block acts as a sink with an external manipulated value (for example, substitute value, simulation, last valid value).
- Worst signal status:
  The worst signal status `ST_Worst` for the block is formed from:
  - `PV_Out.ST`;
  - `SP:ST`;
  - `FFwdOut.ST`;
  - `RbkOut.ST`;

  With step controllers (FM355-2 = 0, StepCon = 1), the following are also included:
  - `FbkOpnOut.ST`;
  - `FbkClsOut.ST`;

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
4	Setting switch or button mode (Page 166)
6	Ramp rate calculation (Page 178)
10	Condition monitoring information at the channel blocks (Page 144)
22	Update acknowledgment and error status of the message call (Page 159)
23	SP following PV in open loop has no priority over SP_Ext and SP limits (Page 178)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)

Operator permissions

The block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode" `AutModOp`
1	1 = Operator can switch to "manual mode" `ManModOp`
2	1 = Operator can switch to "Out of service" mode `OosOp`
3	1 = Operator can switch to "program mode" `AdvCoEn`
4	1 = Operator can switch the setpoint to "external" `SP_ExtOp`
5	1 = Operator can switch the setpoint to "internal" `SP_IntOp`
6	1 = Operator can change the internal setpoint `SP_Int`
7	Continuous controllers, pulse controllers or step controllers with position feedback: 1 = Operator can change the manual parameter `Man` Step controller without position feedback: 1 = Operator can change the manual operation signals `OpenOp`, `StopOp`, `CloseOp`
8	1 = Operator can change operation high limit of the setpoint `SP_InHiLim`
9	1 = Operator can change operation low limit of the setpoint `SP_InLoLim`
10	1 = Operator can change the operation high limit of the manipulated variable `ManHiLim`
11	1 = Operator can change the operation low limit of the manipulated variable `ManLoLim`
12	1 = Operator can enable the setpoint's gradient limitation function `SP_RateOn`
13	1 = Operator can change the setpoint's high limit for the ramp `SP_UpRaLim`
14	1 = Operator can change the setpoint's low limit for the ramp `SP_DnRaLim`
15	1 = Operator can switch between the time value or the value for the ramp `SP_RmpModTime`
16	1 = Operator can change the ramp time `SP_RmpTime`
17	1 = Operator can change the target setpoint `SP_RmpTarget` for the setpoint ramp
18	1 = Operator can enable the setpoint ramp function `SP_RmpOn`
19	Not used
20	1 = Operator can enable the track setpoint in manual mode function `SP_TrkPV`
21	1 = Operator can enable the bumpless switchover from external to internal `SP_TrkExt`
22	1 = Operator can change the gain parameter `Gain`
23	1 = Operator can change the integral time parameter `TI`
24	1 = Operator can change the derivative time parameter `TD`
25	1 = Operator can change the derivative gain parameter `DiffGain`
26	1 = Operator can change the dead band parameter `DeadBand`
27	1 = Operator can change the control zone parameter `ConZone`
28	1 = Operator can change the integral time parameter `MotorTime`
29	1 = Operator can change the integral time parameter `PulseTime`
30	1 = Operator can change the integral time parameter `BreakTime`
31	Not used

The block has the following permissions for the `OS1Perm` parameter:

Bit	Function
0	1 = Operator can change the limit (process value) `PV_AH_Lim` for the high alarm
1	1 = Operator can change the limit (process value) `PV_WH_Lim` for the high warning
2	1 = Operator can change the limit (process value) `PV_TH_Lim` for the high tolerance

Bit	Function
3	1 = Operator can change the hysteresis (process value) `PV_Hyst`
4	1 = Operator can change the limit (process value) `PV_TL_Lim` for the low tolerance
5	1 = Operator can change the limit (process value) `PV_WL_Lim` for the low warning
6	1 = Operator can change the limit (process value) `PV_AL_Lim` for the low alarm
7	1 = Operator can change the limit (error signal) `ER_AH_Lim` for the high alarm
8	1 = Operator can change the hysteresis (error signal) `ER_Hyst`
9	1 = Operator can change the limit (error signal) `ER_AL_Lim` for the low alarm
10	1 = Operator can change the limit (position feedback) `RbkWH_Lim` for the high warning
11	1 = Operator can change the hysteresis (position feedback) `RbkHyst`
12	1 = Operator can change the limit (position feedback) `RbkWL_Lim` for the low warning
13	1 = Operator can open the valve
14	1 = Operator can close the valve
15	1 = Operator can stop the valve
16	1 = Operator can activate the Simulation function `SimOn`
17	1 = Operator can activate the Release for maintenance function `MS_RelOp`
18	1 = Operator can activate / deactivate messages via `PV_AH_MsgEn`
19	1 = Operator can activate / deactivate messages via `PV_WH_MsgEn`
20	1 = Operator can activate / deactivate messages via `PV_TH_MsgEn`
21	1 = Operator can activate / deactivate messages via `PV_TL_MsgEn`
22	1 = Operator can activate / deactivate messages via `PV_WL_MsgEn`
23	1 = Operator can activate / deactivate messages via `PV_AL_MsgEn`
24	1 = Operator can activate / deactivate messages via `ER_AH_MsgEn`
25	1 = Operator can activate / deactivate messages via `ER_AL_MsgEn`
26	1 = Operator can activate / deactivate messages via `RbkWH_MsgEn`
27	1 = Operator can activate / deactivate messages via `RbkWL_MsgEn`
28	1 = Operator can change the simulation value `SimPV`
29	1 = Operator can change the simulation value `SimRbk`
30 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Release for maintenance

The block provides the standard function Release for maintenance (Page 64).

Generating instance-specific messages

The block provides the standard function Generating instance-specific messages (Page 200) without the time stamp function in the I/O.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Specifying the display area for process and setpoint values as well as operations

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- `OpenOp`
- `StopOp`
- `CloseOp`

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

See also

FmTemp messaging (Page 643)

FmTemp I/Os (Page 646)

FmTemp modes (Page 629)

FmTemp block diagram (Page 662)

FmTemp error handling (Page 642)

EventTs functions (Page 1634)

5.4.4 FmTemp error handling

Error handling of FmTemp

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` output parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
30	The value of `PV` can no longer be displayed in the REAL number field.
31	The value of `SP_Ext` can no longer be displayed in the REAL number field.
33	The value of `MV_Trk` can no longer be displayed in the REAL number field.
35	The value of `Rbk` can no longer be displayed in the REAL number field.
36	The value of `MV` can no longer be displayed in the REAL number field.
50	The controller cannot be switched to program mode, because program mode with default setpoint (`AdvCoModSP` = 0) is not possible for step controllers without position feedback (`WithRbk` = 0).
51	`AutModLi` = 1 and `ManModLi` = 1 `SP_LiOp` = 1 and `SP_IntLi` = 1 and `SP_ExtLi` = 1 `OpenLi` = 1 and `StopLi` = 1 `CloseLi` = 1 and `StopLi` = 1 `OpenLi` = 1 and `CloseLi` = 1
59	= 1, "Gain is negative"
60	|`TI`| < `SampleTime` / 2
61	|`TD`| < `SampleTime`
62	`DiffGain` < 1 or `DiffGain` > 10
63	`TD` / `DiffGain` < `SampleTime` / 2
64	`PropFacSP` < 0 or `PropFacSP` > 1
66	`NormPV_High` = `NormPV_Low`
67	`MotorTime` < `SampleTime`
68	`PulseTime` < `SampleTime`
69	`BreakTime` < `SampleTime`
70	`Channel` < 0 or `Channel` > 3
71	(`D_InSel` < 0 or `D_InSel` > 4) and `D_InSel` ≠ 17
74	`ConZone < 0.0` FM 355-2 sets the output `ParFM_Err = 1`

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

Description of FmTemp (Page 625)

FmTemp modes (Page 629)

FmTemp functions (Page 630)

FmTemp messaging (Page 643)

FmTemp I/Os (Page 646)

FmTemp block diagram (Page 662)

Setting switch or button mode (Page 166)

5.4.5 FmTemp messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 6	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter CSF. If it changes to CSF = 1, a process control fault is triggered (MsgEvId2, SIG 6).

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	Alarm - high	$$BlockComment$$ PV - high alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ PV - high warning limit violated
	SIG 3	Tolerance - high	$$BlockComment$$ PV - high tolerance limit violated
	SIG 4	Tolerance - low	$$BlockComment$$ PV - low tolerance limit violated
	SIG 5	Warning - low	$$BlockComment$$ PV - low warning limit violated
	SIG 6	Alarm - low	$$BlockComment$$ PV - low alarm limit violated
	SIG 7	Alarm - high	$$BlockComment$$ ER - high alarm limit violated
	SIG 8	Alarm - low	$$BlockComment$$ ER - low alarm limit violated
MsgEvId2	SIG 7	Warning - high	$$BlockComment$$ Rbk - high warning limit violated
	SIG 8	Warning - low	$$BlockComment$$ Rbk - low warning limit violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You can use up to four instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 1	AS process control message - fault	$$BlockComment$$ External message 1 Status 16#@5%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ External message 2 Status 16#@6%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ External message 3 Status 16#@7%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 4 Status 16#@8%x@

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	Process value `PV_Out`
5	Control deviation `ER`
6	`ExtVa106`
7	`ExtVa107`
8	Not allocated
9	Not allocated
10	Not allocated

The associated values 6 ... 7 are allocated to the parameters `ExtVa106` ... `ExtVa107` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

Associated values for message instance `MsgEvId2`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	Position feedback `Rbk`
5	Signal status `ExtMsg1`
6	Signal status `ExtMsg2`
7	Signal status `ExtMsg3`
8	Signal status `ExtMsg4`
9	`ExtVa209`
10	`ExtVa210`

The associated values 9 ... 10 are allocated to the parameters `ExtVa209` ... `ExtVa210` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of FmTemp (Page 625)

FmTemp functions (Page 630)

FmTemp I/Os (Page 646)

FmTemp modes (Page 629)

FmTemp error handling (Page 642)

FmTemp block diagram (Page 662)

5.4.6 FmTemp I/Os

I/Os of FmTemp

Input parameters

Parameter	Description	Type	Default
AccMode*	1 = Transfer of operating parameters SubN1_ID, SubN2_ID, RackNo, SlotNo and Channel to internal processing	BOOL	1
AdvCoEn	1 = Enable "program mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoModSP	Type of "program mode": 1 = Setpoint specification 0 = Manipulated variable specification	BOOL	1
AdvCoMstrOn	Activate (0-1) or deactivate (1-0) "program mode" via edge transition	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoMV	Specified value from the external program	REAL	0.0
AdvCoOn*	1 = Enable "program mode" via faceplate	BOOL	0
AutModLi*	1 = "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 0)	BOOL	0
BatchEn	1 = Enable allocation for batch control	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BreakTime*	Minimum break duration [s]	REAL	1.0
Channel	Controller channel number (0..3)	INT	0
CloseLi*	1 = Close via interconnection or CFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseOp*	1 = Close via operator	BOOL	0
ConZone*	Control zone	REAL	0.0

Parameter	Description	Type	Default
CoordNo	Coordination number	INT	0
CPI_In	Input for control performance index, which is calculated by the assigned ConPerMon block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
D_InSel*	Input for differentiator: 0 = Error signal 1..4 = Channel 0..3 17 = Actual value to feedback	INT	0
DeadBand*	Width of dead band	REAL	0.0
DiffGain*	Gain of differentiator [1..10] DiffGain = TD / (delay time of D component)	STRUCT • Value: REAL • ST: BYTE	- • 5.0 • 16#80
EN	1 = Called block will be processed	BOOL	1
ER_A_DC*	Delay for incoming alarms during error signal monitoring	REAL	0.0
ER_A_DG*	Delay for outgoing alarms during error signal monitoring	REAL	0.0
ER_AH_En	1 = Activate alarm (high) for error signal monitoring	BOOL	1
ER_AH_Lim	Alarm limit (high) for error signal monitoring	REAL	100.0
ER_AH_MsgEn	1 = Activate messages for alarm (high) for error signal monitoring	BOOL	1
ER_AL_En	1 = Activate alarm (low) for error signal monitoring	BOOL	1
ER_AL_Lim	Alarm limit (low) for error signal monitoring	REAL	-100.0
ER_AL_MsgEn	1 = Activate messages for alarm (low) for error signal monitoring	BOOL	1
ER_Hyst	Alarm hysteresis for error signal	REAL	1.0
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	

Parameter	Description	Type	Default
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg4	Binary input for freely selectable message 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa209	Associated value 9 for messages (MsgEvID2)	ANY	
ExtVa210	Associated value 10 for messages (MsgEvID2)	ANY	
FbkClosed	Low limit stop signal of position feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpened	High limit stop signal of position feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Feature	I/O for additional functions (Page 630)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
FM355_2	Module type: 0: FM 355-2 S; 1: FM 355-2 C	BOOL	0
Gain*	Proportional gain Gain.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#FF
LoadPID*	Load optimized PI/PID parameters	BOOL	0
LogAddr	Logical address FM 355	INT	0
Man*	Manual specification for the manipulated variable	REAL	0.0
ManHiLim*	Limit (high) for manual parameter Man	REAL	100.0
ManLoLim*	Limit (low) for manual parameter Man	REAL	0.0

Parameter	Description	Type	Default
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
Mode	Operating mode	DWORD	16#00000000
ModLiOp	Operating mode switchover between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MotorTime*	Motor actuating time [s]	REAL	30.0
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MS	Maintenance status	DWORD	0
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ms_Ext	External maintenance status	DWORD	0
MsgEvID1	Message number (assigned automatically)	DWORD	16#00000000
MsgEvID2	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_FmTrkOn	1 = Manipulated variable tracking in the FM	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_HiLim*	Limit (high) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
MV_LoLim*	Limit (low) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_OpScale	OS display range for manipulated variable MV	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
MV_Safe*	Neutral position manipulated variable	REAL	0.0
MV_SafeOn	1 = Neutral position manipulated variable MV_Safe at output MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MV_Trk*	Tracking value for the manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_TrkOn	1 = Tracking of manipulated variable MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_Unit	Unit of measure for manipulated variable	INT	1342
NegGain*	0 = Effective proportional gain GainEff is positive 1 = Effective proportional gain GainEff is negative	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NormMV	Manipulated variable range (MV) for standardizing the proportional gain (GAIN)	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
NormPV	Process value range (PV) for standardizing the proportional gain (GAIN)	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
Occupied	Occupied by batch control	BOOL	0
OosLi	Edge transition (0-1) = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpenLi*	1 = Open via interconnection or CFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenOp*	1 = Open via operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 630)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OS1Perm	I/O for operator permissions (Page 630)	STRUCT • Bit 0: BOOL • Bit 18: BOOL • Bit 19: BOOL • Bit 31: BOOL	- • 1 • 1 • 1 • 1
PID_On*	1 = PID mode on	BOOL	0
PropFacSP*	Applying the P component to the feedback [0..1]. 0 = P component fully in feedback	REAL	1.0

Parameter	Description	Type	Default
PropSel*	1 = Activate P component	BOOL	1
PulseTime*	Minimum pulse duration [s]	REAL	1.0
PV_A_DC*	Delay time for incoming PV alarms [s]	REAL	0.0
PV_A_DG*	Delay time for outgoing PV alarms [s]	REAL	0.0
PV_AH_En	1 = Enable PV alarm limit (high)	BOOL	1
PV_AH_Lim	Limit PV alarm (high)	REAL	95.0
PV_AH_MsgEn	1 = Enable PV alarm (high) message	BOOL	1
PV_AL_En	1 = Enable PV alarm limit (low)	BOOL	1
PV_AL_Lim	PV alarm limit (low)	REAL	5.0
PV_AL_MsgEn	1 = Enable PV alarm (low) message	BOOL	1
PV_Hyst	Hysteresis for PV alarm, warning and tolerance limits	REAL	1.0
PV_OpScale	Limit for scale in PV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
PV_T_DC*	Delay time for incoming PV tolerance messages [s]	REAL	0.0
PV_T_DG*	Delay time for outgoing PV tolerance messages [s]	REAL	0.0
PV_TH_En	1 = Enable PV tolerance limit (high)	BOOL	0
PV_TH_Lim	Limit PV tolerance message (high)	REAL	85.0
PV_TH_MsgEn	1 = Enable message for PV tolerance message (high)	BOOL	1
PV_TL_En	1 = Enable PV tolerance limit (low)	BOOL	0
PV_TL_Lim	Limit PV tolerance message (low)	REAL	15.0
PV_TL_MsgEn	1 = Activate message for PV tolerance message (low)	BOOL	1
PV_Unit	Unit of measure for process value	INT	1001
PV_W_DC*	Delay time for incoming PV warnings [s]	REAL	0.0
PV_W_DG*	Delay time for outgoing PV warnings [s]	REAL	0.0
PV_WH_En	1 = Enable PV warning limit (high)	BOOL	1
PV_WH_Lim	Limit PV warning (high)	REAL	90.0
PV_WH_MsgEn	1 = Enable PV warning (high) message	BOOL	1
PV_WL_En	1 = Enable PV warning limit (low)	BOOL	1
PV_WL_Lim	Limit PV warning (low)	REAL	10.0
PV_WL_MsgEn	1 = Enable PV warning (low) message	BOOL	1
RackNo	Rack number	BYTE	16#FF
RatioFac*	Ratio factor	REAL	0.0
Rbk*	Position feedback for display on OS	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
RbkHyst	Alarm hysteresis for position feedback	REAL	1.0

Parameter	Description	Type	Default
RbkWH_En	1 = Enable warning (high) for position feedback	BOOL	1
RbkWH_Lim	Limit for position feedback of warning (high)	REAL	100.0
RbkWH_MsgEn	1 = Enable messages for warning (high) for position feedback	BOOL	1
RbkWL_En	1 = Enable warning (low) for position feedback	BOOL	1
RbkWL_Lim	Limit for position feedback of warning (low)	REAL	0.0
RbkWL_MsgEn	1 = Enable messages for warning (low) for position feedback	BOOL	1
RefStdDevIn	Reference value of PV standard deviation (sigma) in defined "good" state of control loop	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SavePar*	1 = Save PID controller parameters	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimPV*	Process value used for SimOn = 1	REAL	0.0
SimPV_Li	Process value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimRbk*	Position feedback used for SimOn = 1	REAL	0.0
SimRbkLi	Position feedback used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SlotNo	Slot number	BYTE	16#FF
SP_DnRaLim	Limit (low) for the gradient of the setpoint [SP_Unit/s]	REAL	100.0

Parameter	Description	Type	Default
SP_ExHiLim*	Limit (high) for external setpoint	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP_ExLoLim*	Limit (low) for external setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_Ext*	External setpoint of - (to interconnection)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtLi*	1 = Select external setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOp*	1 = Select external setpoint (via operator)	BOOL	0
SP_InHiLim*	Limit (high) of internal setpoint	REAL	100.0
SP_InLoLim*	Limit (low) of internal setpoint	REAL	0.0
SP_Int*	Internal setpoint for operation	REAL	0.0
SP_IntLi*	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_IntOp*	1 = Select internal setpoint (via operator)	BOOL	0
SP_LiOp	Select setpoint source (internal/external): 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_RateOn*	1 = Activate limitation of setpoint gradients	BOOL	0
SP_RmpModTime	1 = Use time (SP_RmpTime) for setpoint ramp, 0 = Use gradient	BOOL	0
SP_RmpOn*	1 = Activate setpoint ramp to target setpoint SP_RmpTarget	BOOL	0
SP_RmpTarget*	Target setpoint for setpoint ramp	REAL	0.0
SP_RmpTime*	Time for setpoint ramp [s] from current SP up to SP_RmpTarget	REAL	0.0
SP_TrkExt	1 = Bumpless switchover from external to internal setpoint active	BOOL	1
SP_TrkPV	1 = Setpoint follows PV in "manual mode" and with tracking	BOOL	0
SP_UpRaLim	Gradient limit (high) for the setpoint [SP_Unit/s]	REAL	100.0
StepNo	Batch step number	DWORD	16#00000000

Parameter	Description	Type	Default
StopLi*	1 = Stop via interconnection or CFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopOp*	1 = Stop via operator	BOOL	0
SubN1_ID	ID of the primary DP master system	BYTE	16#FF
SubN2_ID	ID of the redundant DP master system	BYTE	16#FF
TD*	Derivative time [s] TD.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
TextRef	Text reference	WORD	0
TI*	Integral time [s] TI.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#FF
TimeFactor	Time unit: 0 = Seconds 1 = Minutes 2 = Hours	INT	0
TunC_MV*	Delta manipulated variable for cooling optimization	REAL	-20.0
TunC_Start*	Start cooling optimization	BOOL	0
TunD_MV*	Delta manipulated variable for process trigger	REAL	20.0
TunOn*	Enable controller optimization	BOOL	0
TunStart*	Start controller optimization	BOOL	0
UndoPar*	Undo controller parameter changes	BOOL	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
EnCoord	Current coordination number	STRUCT • CO_ACT : INT	- • 0
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
AdvCoAct	1 = "Program mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoRdy	1 = "Program mode" available	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutAct	1 = "Automatic mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ChFM_Err	1 = Channel error on the module	BOOL	0
Close	Control output: 1 = Closed is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EnCoNum	Coordination number	BYTE	16#00
ENO	1 = Block algorithm completed without errors	BOOL	0
ER	Error signal	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ER_AH_Act	1 = Alarm limit (high) for control deviation violated. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ER_AL_Act	1 = Alarm limit (low) for control deviation violated. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum	Output of pending error number, for error numbers that can be output by this block, see FmTemp error handling (Page 642)	INT	-1

Parameter	Description	Type	Default
FFwdOut	Disturbance variable generated in the FM	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
FbkClsOut	1 = Low limit stop of the position feedback reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpnOut	1 = High limit stop of the position feedback reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GainEff	Effective proportional gain depends on NegGain, Gain, NormPV, and NormMV	REAL	1.0
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LoopClosed	1 = Control loop closed 0 = Control loop open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
ManARW_Act	1 = Tracking mode or anti-reset windup by secondary controller	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManHiOut	Limit (high) for "manual mode", corresponds to input parameter ManHiLim	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
ManLoOut	Limit (low) for "manual mode", corresponds to input parameter ManLoLim	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ModErr	1 = Module error	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Xchg	Exchange of the maintenance status	DWORD	0

Parameter	Description	Type	Default
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgAckn2	Message acknowledgement status 2 (output ACK_STATE of second ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgErr2	Alarm error 2 (output ERROR of second ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
MsgStat2	Message status 2 (output STATUS of second ALARM_8P)	WORD	16#0000
MV	Manipulated variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_FmTrkAct	1 = Track manipulated variable in the FM enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_HiAct	1 = Limit (high) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_LoAct	1 = Limit (low) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_SafeAct	1 = Neutral position manipulated variable of the FM enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_SpliA	Manipulated variable A of split-range function	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_SpliB	Manipulated variable B of split-range function	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_UnitOut	Unit of measure for manipulated variable, for interconnecting to the MV_Unit input parameter of the ConPerMon block	INT	0
MV_Visible	1 = MV display visible Is evaluated by the block icon	BOOL	0

Parameter	Description	Type	Default
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Open	Control output 1 = Open is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS1PermLog	Display of OS1Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS1PermOut	Display of OS_Perm1	DWORD	16#FFFFFFFF
ParFM_Err	1 = Direct parameter-assignment error of the FM or input Channel configured incorrectly	BOOL	0
PerAccErr	1 = I/O access error	BOOL	0
Phase	Phase of auto-tuning [0..7]	INT	0
PV	Process value of the module	STRUCT • Value: REAL • ST: BYTE	- 0.0 16#80
PV_AH_Act	1 = PV alarm (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- 0 16#80
PV_AL_Act	1 = PV alarm (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	Output for process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
PV_TH_Act	1 = PV tolerance message (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_TL_Act	1 = PV tolerance message (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_ToleHi	Limit (high) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_ToleLo	Limit (low) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_UnitOut	Unit of measure for process value, for interconnecting to the PV_Unit input parameter of the ConPerMon block	INT	0
PV_WH_Act	1 = PV warning (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_WL_Act	1 = PV warning (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkOut	Output for position feedback	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
RbkWH_Act	1 = Warning (high) for position feedback active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkWL_Act	1 = Warning (low) for position feedback active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RetValue	Return value of WRREC / RDREC	WORD	16#0000
RbkVisible	1 = Rbk display visible Is evaluated by the block icon	BOOL	0
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP	Setpoint used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_DnRaAct	1 = Negative gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExHiAct	1 = Limit (high) for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExLoAct	1 = Limit (low) for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtAct	1 = External setpoint is active 0 = Internal setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOut	External setpoint, corresponds to input parameter SP_Ext	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

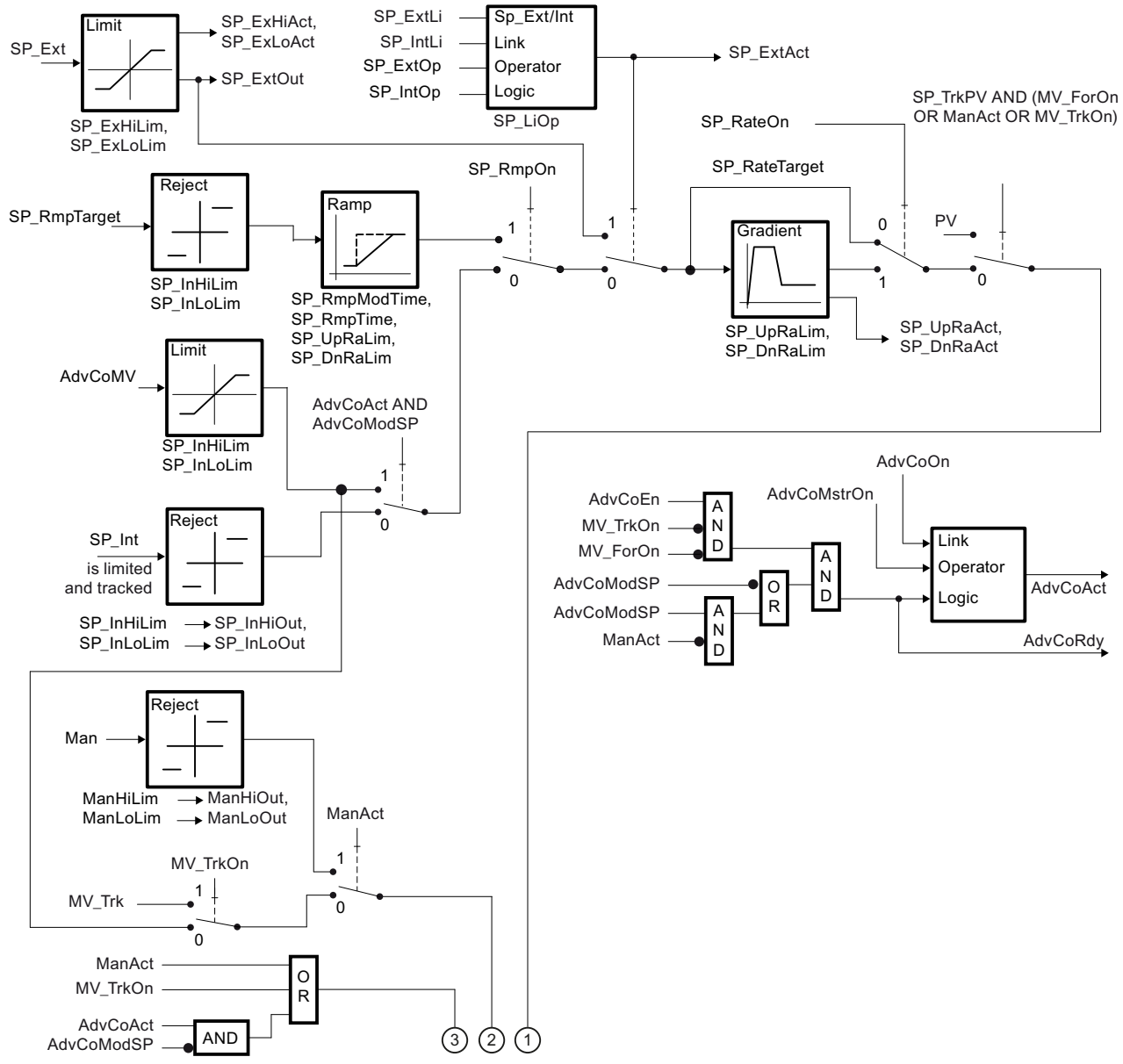
Parameter	Description	Type	Default
SP_InHiOut	Limit (high) for SP_Int corresponds to input parameter SP_InHiLim	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP_InLoOut	Limit (low) for SP_Int corresponds to input parameter SP_InLoLim	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_RateTarget	Target setpoint for the gradient limitation	REAL	0.0
SP_UpRaAct	Positive gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SplitRange	1 = Split-range function has been activated	BOOL	0
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 625)	DWORD	16#00000000
Status2	Status word 2 (Page 625)	DWORD	16#00000000
Status3	Status word 2 (Page 625)	DWORD	16#00000000
StatusC	Status of cooling optimization	INT	0
StatusD	Status of controller design	INT	0
StatusH	Status of heating optimization	INT	0
StepCon	1 = Step controller	BOOL	0
Stop	Control output: 1 =Stopped is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SumMsgAct	1 = Active hardware interrupt	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
TunAct	1 = Optimization running	BOOL	0
WithRbk	Controller type: 0 = Step controller without position feedback 1 = Step controller with position feedback	BOOL	0
ZoneTun	Controller channels grouped in one zone for parallel optimization	WORD	16#0000

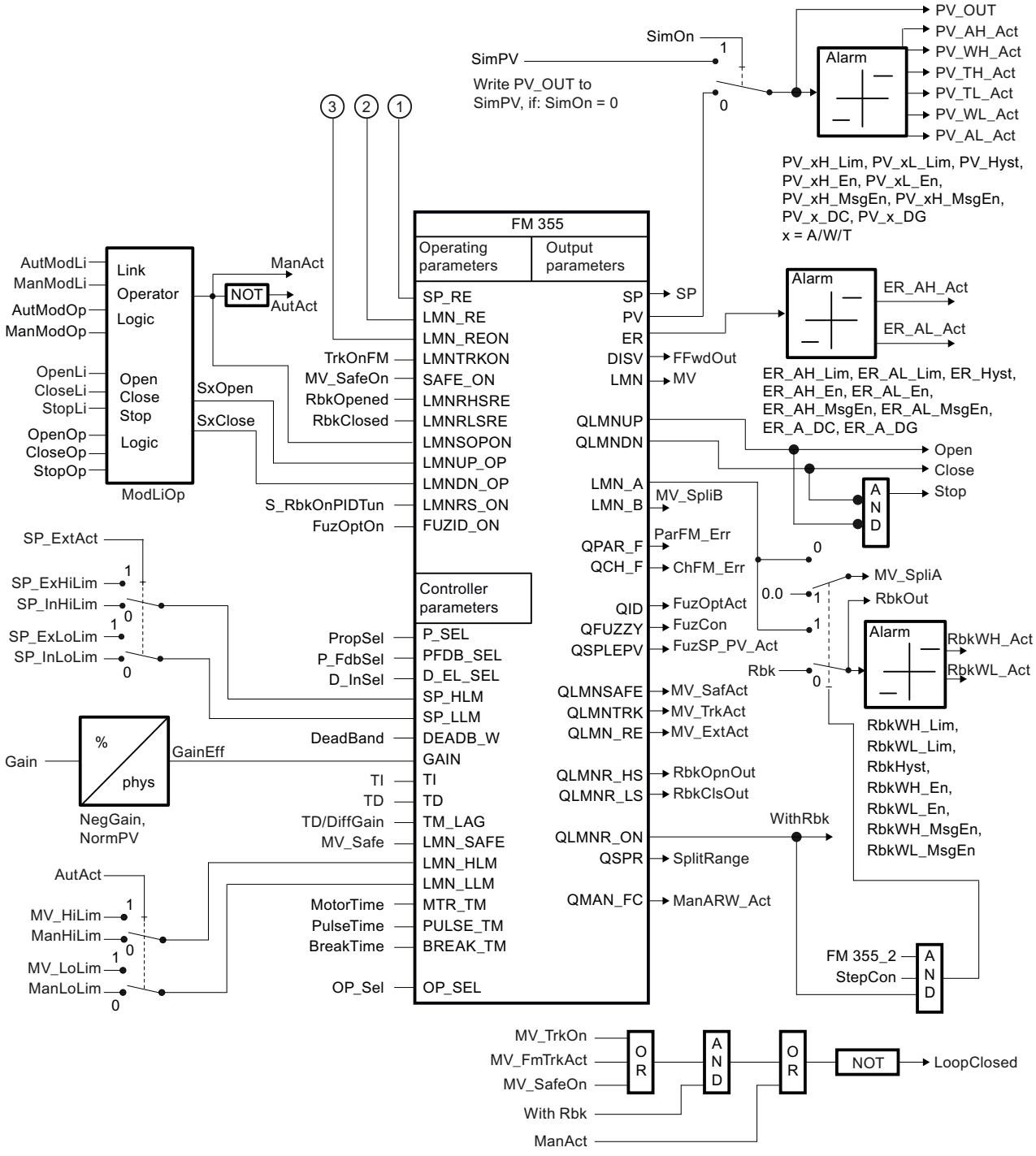
See also

FmTemp messaging (Page 643)
FmTemp modes (Page 629)
FmTemp block diagram (Page 662)

5.4.7 FmTemp block diagram

FmTemp block diagram





See also

Description of FmTemp (Page 625)

FmTemp modes (Page 629)

FmTemp functions (Page 630)

FmTemp error handling (Page 642)

FmTemp messaging (Page 643)

FmTemp I/Os (Page 646)

5.4.8 Operator control and monitoring

5.4.8.1 FmTemp views

Views of the FmTemp block

The block FmTemp provides the following views:

- FM controllers standard view (analog) (Page 255)
- FM controllers standard view (pulse controller) (Page 259)
- FM controllers standard view (step controller with position feedback) (Page 263)
- FM controllers standard view (step controller without position feedback) (Page 267)
- Alarm view (Page 296)
- Limit view of FM controllers (Page 282)
- Trend view (Page 299)
- Parameter view of FM controllers (Page 278)
- Preview of FM controllers (Page 291)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icons for PID and FM controller (Page 235)

Refer to the Structure of the faceplate (Page 242) and Block icon structure (Page 226) sections for general information about the faceplate and block icon.

5.5 GainSched - Adapting parameter values for a PID controller

5.5.1 Description of GainSched

Object name (type + number) and family

Type + number: FB 1820

Family: Control

Area of application for GainSched

The block is used for the following applications:

- Continuous adaptation of the parameter values of a PID controller to the current operating point of a non-linear process
- Controller gain
- Integral action time
- Derivative action time

How it works

If your process requires different PID controller parameters due to its non-linear response at different operating points, you can store optimum parameter sets for up to three different operating points in theGainSched block in the form of a table ("timetable"). The current operating point is represented by a continuously measurable variable X, typically by the actual value of the controller itself. The block ensures that the suitable optimum parameters `Gain`(j), `TI`(j) and `TD`(j) are made available to the controller for each operating point X(j).

If the process is between two operating points, the parameters are calculated by linear interpolation between the optimum values of the two nearest operating points. This allows a bumpless, continuous adaptation of the controller parameters while the process moves from one operating point to another.

The block should be considered a supplementary function for a PID controller to improve the control performance of the PID controller in non-linear processes. The GainSched faceplate is called from the parameter view of the corresponding PID controller using the "Gain scheduler" button.

In contrast to all other function blocks, the GainSched block is implemented as a CFC chart and is generated with the "Compile chart as block type" function. The source chart "`FbGainSchedLim`" is supplied with the library so that you have more options open to you:

- You can use the precompiled function block GainSched from the library if the standard functionality is adequate for your needs.
- If you require special additional functions for gain scheduling in your application (for example more than three operating points, additional logic functions for selecting the parameters), you will need to modify the CFC source chart and compile it as a block type with a different FB number.

If the current value of input parameter X is below the lowest value X1 in the table or above the highest value X3, precisely the controller parameters which are specified at the relevant boundary point X1 or X3 in the table are output.

Configuration

The GainSched block is placed in the same CFC chart as the assigned controller and interconnected with it as shown in the corresponding template: The output parameters Link2Gain, Link2TI and Link2TD are connected to the inputs Gain, TI and TD of the PID controller. The input X of GainSched is supplied with the measured value for the operating point, typically with the same value as PV of the controller.

You can open the standard view for the GainSched block from the parameter view of a controller (for example PIDConL). Additional information on this topic is available in the section Opening additional faceplates (Page 203).

To specify the parameters for gain scheduling, run separate controller optimizations at each of the intended operating points, for example with a tool such as the PID tuner. Use amplitudes as small as possible to excite the process to capture the approximately linear response in the area of the operating point under investigation. The optimum parameter values calculated by the PID tuner are entered in the relevant row belonging to the operating point in the table of the GainSched block. The table is clearly displayed in the standard view of the faceplate. Make sure that the numeric values are also permanently stored in the data management of the engineering system by reading back the numeric values of the parameters from AS to the ES or entering them manually at the inputs of the CFC block.

For the GainSched block, the Advanced Process Library contains a template for process tag types as an example with an application scenario for this block.

Example of process tag types:

- PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301)

Gain scheduling for batch processes

A typical area of application for gain scheduling is in batch processes that, in contrast to continuous processes, cannot be linearized around a fixed operating point because they need to be moved backwards and forwards between different operating points during the course of the batch. Here, there are three application scenarios:

- The controller parameters depend on a single continuously measurable variable that is representative of the operating point, for example, the reactor temperature. This is the normal use case for the GainSched block: The management of the controller parameters is handled in the block and is independent of batch recipes.
- The controller parameters depend on a continuously measurable variable that is representative of the operating point, but there is also a dependency on the materials used in the reaction. Suitable parameter sets for gain scheduling can then be anchored in the recipe and transferred by SIMATIC BATCH to the GainSched block.

- The controller parameters depend only the current phase of the batch. They can then be written directly from the Batch package to the PID controller and no gain scheduling block is necessary. The disadvantage of this is that there is bump in the controller parameters at the transfer from one phase to the next. The controller should be put into manual mode temporarily at the time of the transfer to avoid a bump in the manipulated variable.
- The recipe only specifies which of the controller parameter sets 1 ... 3 is currently required from the GainSched block. The numerical values of the parameter, however, are not anchored in the recipe. In this case input variable `X` of the GainSched block can then be used as the number of the required data record and assigned by the recipe instead of being linked with a measurable process variable. In this case, there are only three discrete values for `X` and the precautions against a change of controller parameters with bump outlined above must be taken because the interpolation abilities of the GainSched block are not used.

In general, it is not necessary to manage the batch parameters (batch number, batch name, etc.) in the GainSched block because the block does not generate any separate messages and there is always a 1-to-1 relationship with a controller block that knows the batch parameters.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

GainSched functions (Page 668)
GainSched messaging (Page 669)
GainSched I/Os (Page 670)
GainSched block diagram (Page 672)
GainSched error handling (Page 669)
GainSched modes (Page 667)

5.5.2 GainSched modes

GainSched modes

The block can be operated using the following modes:

- Automatic mode (Page 72)
- Manual mode (Page 72)

"Automatic mode"

In "automatic mode" (`ManParOn` = 0) the controller parameters are determined through a polygon in accordance with the settings in the "automatic" area of the parameter view.

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for control blocks (Page 72) section.

"Manual mode"

In "manual mode" (`ManParOn` = 1) the controller parameters correspond to the settings in the "manual" area.

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for control blocks (Page 72).

See also

GainSched block diagram (Page 672)

GainSched I/Os (Page 670)

GainSched messaging (Page 669)

GainSched error handling (Page 669)

GainSched functions (Page 668)

Description of GainSched (Page 665)

5.5.3 GainSched functions

Functions of GainSched

The functions for this block are listed below.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) section. The following functionality is available for this block at the relevant bits:

Bit	Function
24	Enabling local operator authorization (Page 157)

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

See also

Description of GainSched (Page 665)
GainSched messaging (Page 669)
GainSched I/Os (Page 670)
GainSched block diagram (Page 672)
GainSched error handling (Page 669)
GainSched modes (Page 667)
GainSched standard view (Page 673)

5.5.4 GainSched error handling

GainSched error handling

The block does not report any errors.

See also

GainSched block diagram (Page 672)
GainSched I/Os (Page 670)
GainSched messaging (Page 669)
GainSched functions (Page 668)
GainSched modes (Page 667)
Description of GainSched (Page 665)

5.5.5 GainSched messaging

Messaging

This block does not offer messaging.

See also

Description of GainSched (Page 665)
GainSched functions (Page 668)
GainSched I/Os (Page 670)
GainSched block diagram (Page 672)

GainSched error handling (Page 669)

GainSched modes (Page 667)

5.5.6 GainSched I/Os

GainSched I/Os

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`Gain1`	PID gain for operating point 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Gain2`	PID gain for operating point 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Gain3`	PID gain for operating point 3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`GainOp`	PID gain: Input for "manual mode"	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Feature`	I/O for additional functions (Page 668)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`ManParOn`	1 = input of PID parameters in "manual mode" 0 = use the planned controller parameters from the table	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`OpSt_In`	Input parameter for local operating permission, connected with the `Out` output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
`TD_Op`	PID derivative action time [s]: Manual input for the operator	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`TD1`	PID derivative action time [s] for operating point 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
TD2	PID derivative action time [s] for operating point 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
TD3	PID derivative action time [s] for operating point 3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
TI_Op	PID integral action time [s]: Manual input for the operator	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
TI1	PID integral action time [s] for operating point 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
TI2	PID integral action time [s] for operating point 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
TI3	PID integral action time [s] for operating point 3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
X	Process value that defines the operating point	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
X1	Operating point 1 (support point) for X	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
X2	Operating point 2 (support point) for X	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
X3	Operating point 3 (support point) for X	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
X_Unit	Unit of measure for the operating point	INT	1001

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Link2Gain	Calculated controller gain	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
`Link2TD`	Calculated integral action time	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Link2TI`	Calculated derivative action time	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`OpSt_Out`	Value of the `OpSt_In` input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by `Feature` bit 24	DWORD	16#00000000
`OS_PermLog`	Parameter used to hide operator controls in the faceplate	DWORD	16#00000000

See also

Description of GainSched (Page 665)
GainSched messaging (Page 669)
GainSched block diagram (Page 672)
GainSched modes (Page 667)

5.5.7 GainSched block diagram

GainSched block diagram

A block diagram is not provided for this block.

See also

GainSched I/Os (Page 670)
GainSched messaging (Page 669)
GainSched error handling (Page 669)
GainSched functions (Page 668)
GainSched modes (Page 667)
Description of GainSched (Page 665)

5.5.8 Operator control and monitoring

5.5.8.1 GainSched views

Views of the GainSched block

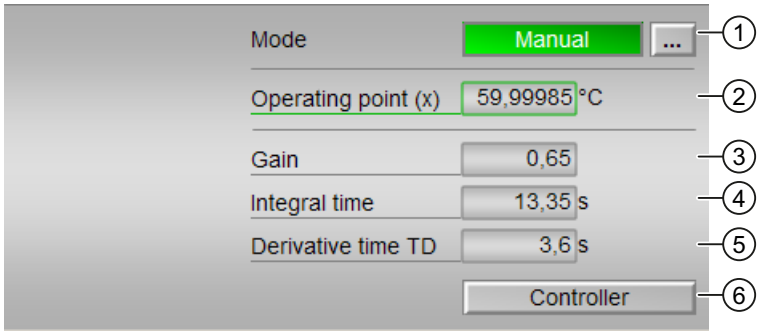
The GainSched block provides the following views:

- GainSched standard view (Page 673)
- GainSched parameter view (Page 674)
- GainSched preview (Page 675)
- Memo view (Page 298)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

5.5.8.2 GainSched standard view

GainSched standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 72)
- Automatic mode (Page 72)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

In "manual mode", you can specify the values on the OS in the parameter view of this block; they are then output directly via the corresponding output parameters.

In "automatic mode", an interpolation is performed through the interpolation points, which can also be specified in the parameter view.

(2) Displaying the operating point (X)

Currently used operating point.

(3) Displaying the gain

The controller gain currently output at the `Link2Gain` output parameter.

(4) Displaying the integration time `TI`

Integration time currently output at the `Link2TI` output parameter.

(5) Displaying and changing the derivative time `TD`

Derivative time currently output at the `Link2TD` output parameter.

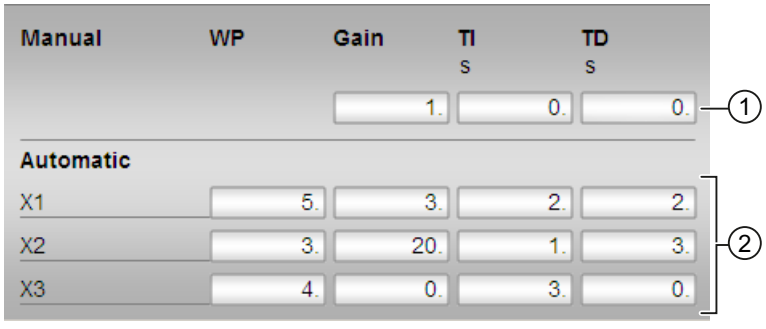
(6) Navigation button to GainSched block

Use this navigation button to reach the standard view of a controller block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

Additional information is available in the section Opening additional faceplates (Page 203).

5.5.8.3 GainSched parameter view

Parameter view of GainSched



(1) Displaying and changing the values for the controller parameters in "manual mode"

This is where you enter the values for the parameters to be used in "manual mode" at the corresponding output parameters of the block:

- "Gain": Input parameter `GainOp`
- "TI": Integration time, input parameter `TI_Op`
- "TD": Derivative time, input parameter `TD_Op`

Refer to the section titled Changing values (Page 253) for information on changing the values.

(2) Displaying and changing the values for the controller parameters in "automatic mode"

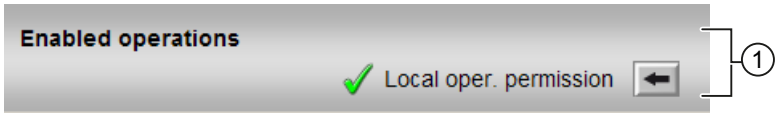
This is where you enter the values for the parameters to be used in "automatic mode" for the interpolation (max. 3 values):

- "X1": Operating point 1, input parameter `X1`
- "X2": Operating point 2, input parameter `X2`
- "X3": Operating point 3, input parameter `X3`
- "Gain": Input parameter `Gain1` ... `Gain3`
- "TI": Integration time, input parameter `TI1` ... `TI3`
- "TD": Derivative time, input parameter `TD1` ... `TD3`

Refer to the section titled Changing values (Page 253) for information on changing the values.

5.5.8.4 GainSched preview

Preview of GainSched



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

The following enabled operations are shown here:

- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

5.6 ModPreCon - Model predictive controller

5.6.1 Description of ModPreCon

Object name (type + number) and family

Type + number: FB 1843

Family: Control

Area of application for ModPreCon

The block is used in much the same way as a PIDConL block for the following applications:

- Fixed setpoint control
- Cascade control
- Ratio control
- Split-range control

In contrast to the PID controllers, this is a multivariable controller.

Compared to Advanced Process Library V7.1.4, the new version provides the following functional enhancements:

- Integrated static operating point optimization,
- Prediction without control response and display of the prediction of the free motion,
- Automatic process trigger for model identification,
- Verification of the sampling time.

Method of operation and area of application

The block is used for multivariable control (Page 2364) of dynamic processes. It can handle up to four dependent manipulated and controlled variables as well as a measurable disturbance.

In special situations, the ModPreCon block can also be used for particularly difficult dynamic, single-variable controls. It is better than a PID controller, for example, in systems with non-minimum phase (Page 2365) or a strongly oscillating response.

The ModPreCon algorithm only works for stable processes with a step response that settles to a fixed value in a finite time. If the process is unstable in one of the main transfer functions or includes an integrator (level control, for example), the corresponding partial transfer function must be stabilized with a secondary controller.

A simple P controller (proportional component only) is sufficient as a subordinate controller for integrating processes. It is inserted between the manipulated variable output from ModPreCon and the input of the unstable process section, and receives the output of the integrating process section as a manipulated variable. (Unstable data links are stabilized with this approach.)

You can find an explanation of "multivariable controls" and "non-phase-minimum response" in the Help on the Advanced Process Library > Definitions.

Note on the area of application of the controller: Longer run times

Due to the principle on which they work, the runtime for multivariable controllers is significantly longer than that for PID controllers, because the matrix calculations in the algorithm are much more complex. The runtime is also determined by the number of the process and manipulated variables in the control algorithm. This is why the multivariable controller is unsuited for rapid control and is usually used for slow, complex control tasks.

The computation time required on the CPU is compensated for by the fact that very slow sample times of > 20s are used for the typical ModPreCon applications (see Advanced control templates). The ModPreCon is then typically in OB30 and can be interrupted by faster OBs.

Optimization is called within the ModPreCon block in the program section in which OB1 is executed. In cyclic operation (OB3x), this avoids the additional computing time required by the optimizer, the acyclic time, i.e. which is only relevant for changes to the optimizer inputs. The calculation time load caused by the ModPreCon block is hardy more with optimization than it is for ModPreCon without optimization.

Operating principle

The ModPreCon block is a model-based predictive multivariable controller. It uses a mathematical model of the process dynamics including all interactions as part of the controller. This model allows the process response to be predicted over a defined period in the future, also known as the prediction horizon.

Based on this prediction, a performance index

$$J = (\vec{w} - \vec{y})^T \cdot R \cdot (\vec{w}-\vec{y}) + \Delta\vec{u}^T \cdot Q \, \Delta\vec{u}$$

is optimized (minimized) where the following applies:

- w contains the time series of the future setpoints,
- y contains the vector of the controlled variables in the future,
- Δu contains the future changes to the manipulated variable.

If you increase the weighting in the Q diagonal matrix, the controller moves its manipulated variables more cautiously resulting in a slower but more robust control response. Using the weighting factors in the R diagonal matrix, you specify the relative significance of the individual controlled variables. A higher weighting (priority) for a controlled variable means that this CV moves more quickly towards the setpoint and remains more accurately at the setpoint in steady state if it is not possible to achieve all setpoints precisely.

The algorithm is a variant of the DMC procedure (**D**ynamic **M**atrix **C**ontrol) in which the optimization problem is solved in the design phase ignoring the constraints. The function block itself contains the analytical solution of the optimization problem. Manipulated variable limitations, both absolute and relating to the gradients, are treated in the algorithm of the function block as hard constraints that must not be violated. This means that precise setpoints or target zones for the controlled variables are taken into account during optimization as well as possible. The target zones for the controlled variables are therefore soft limits, which are

maintained as well as possible although they cannot be guaranteed. Using a reference variable filter for future setpoint settings, the control response of the controller can be fine-tuned during operation.

You can achieve significant improvements in control performance when individual disturbances can be measured, for example variations in throughput. In this case, it is a good idea to take into account a model of the influence of this disturbance on the controlled variables when predicting the controlled variables so that the controller can react preemptively to such disturbances.

Operating point optimization

The integrated static operating point optimization can be used when at least one controlled variable provides certain degrees of freedom. No exact setpoint is specified for such controlled variables. Instead there is a tolerance band, e.g.

SP2OptHiLim…SP2OptLoLim

within which the process value, CV2, must remain. These areas can be defined for any subset of the relevant controlled variables. From an economic perspective, different values within the tolerance range can be more or less favorable. With the help of the optimization function, the optimal economic point can be found within the tolerance range. This is done by defining a target function (performance criterion), which depends on the manipulated variable and controlled variable of the predictive controller. This can be, for example, the economic yield of plant operation per time unit or it may involve specific costs or energy consumption.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

After installation in CFC, follow the steps outlined below:

1. Excite the process with the controller in manual mode by applying a series of manipulated variable step changes.
2. Record the measured data with the CFC trend display and export it to an archive file.
3. Select the ModPreCon instance in the CFC. Start the MPC Configurator with the command Edit > MPC Configurator. The working steps process identification, controller design and simulation of the closed control loop are executed in the MPC Configurator. You can find a detailed description of this procedure in the MPC Configurator help. You can find the help with a button in the MPC Configurator or directly under …\Program Files (x86)\SIEMENS \STEP7\S7JMPC\s7jmpctb.chm. (the last letter is the language code).
4. Using the Configurator, create an SCL source code for the user data block (DB). It contains the models and matrices required for a ModPreCon instance.

5. Compile the SCL source code in the engineering system and download it to the AS.
6. Enter the number of the data block at the DB_No input parameter of the ModPreCon block. The values are adopted in the controller by restarting the block using the Restart input parameter.

Note

During controller design in the MPC Configurator, a controller cycle time and an OB sampling time are calculated, displayed, and stored in the user data block. You yourself are responsible for the ModPreCon block being called in the cyclic interrupt level suitable for the OB sampling time. This is checked in the current ModPreCon version during initialization. If the SampleTime parameter of the function block does not match the OB_SampleTime parameter of the user data block, a parameter assignment error (ErrorNum=3) is displayed. For controller cycle times greater than 5 s, specify the ModPreCon block in the OB30 and specify the appropriate cycle time for the OB30 in the hardware configuration of the Simatic CPU. Controller cycle times slower than 20 s cannot be set in the hardware configuration. The block would then be called every 20s and the slower sampling time automatically realized by an internal cycle reduction ratio in the block.

For the ModPreCon block, the Advanced Process Library contains a template for a process tag type as examples and there is an example project (where APL_Example_xx, xx designates the language variant) containing various application cases for this block. Several application cases are simulated in the example project and serve to explain how the block works.

Example of a process tag type:

- Model-based predictive control (you can find additional information on this in the Help on Advanced Process Library > PCS 7 Advanced Process Control Templates > Process tag types > Model-based predictive control (ModPreCon))

Application cases in example project:

- Predictive control of a 2x2 multi-variable controlled system (you can find additional information on this in the Help on Advanced Process Library > PCS 7 Advanced Process Control Templates > Example project APL_Example_xx > Predictive control of a 2x2 multi-variable controlled system)
- Predictive control of a non-linear process (you can find additional information on this in the Help on Advanced Process Library > PCS 7 Advanced Process Control Templates > Example project APL_Example_xx > Predictive control of a non-linear process)

Startup characteristics

When the CPU starts up, the block always starts in manual mode. It is only possible to change to automatic mode when a user data block is loaded and the internal measured value memory in ModPreCon is filled with data.

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

You can find additional information on the feature bit "Set startup characteristics" in the Help on Advanced Process Library > Basics of APL > Selectable block response> Set startup characteristics.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section ModPreCon I/Os (Page 696)

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutAct.Value`
6	Not used
7	`ManAct.Value`
8 - 9	Not used
10	`MV1TrkOn.Value` AND NOT (`ManAct.Value` OR `OosAct.Value`)
11	`MV2TrkOn.Value` AND NOT (`ManAct.Value` OR `OosAct.Value`)
12	`MV3TrkOn.Value` AND NOT (`ManAct.Value` OR `OosAct.Value`)
13	`MV4TrkOn.Value` AND NOT (`ManAct.Value` OR `OosAct.Value`)
14	Not used
15	`DB_Loaded`
16	`DV_Model Available`
17	OptimAct
18	`NOT(OptimAct)`
19	`SimOn AND ManAct`
20	J_Mini
21	NOT(J_Mini)
22	ExciteOn AND ManAct.Value
23-30	Not allocated
31	Feature.Bit31: Display of the predictions in the faceplate

Status word allocation for `Status2` parameter

Status bit	Parameter
0 - 30	Not used
31	`MS_RelOp`

See also

ModPreCon functions (Page 682)

ModPreCon messaging (Page 696)

ModPreCon block diagram (Page 707)

ModPreCon error handling (Page 695)

ModPreCon modes (Page 681)

Model-based predictive control (ModPreCon) (Page 2325)

Predictive control of a 2x2 multi-variable controlled system (ModPreConSim) (Page 2355)

Predictive control of a non-linear process (ModPreConNonLinSim) (Page 2355)

5.6.2 ModPreCon modes

Operating modes of ModPreCon

The block can be operated using the following modes:

- Automatic mode (Page 72)
- Manual mode (Page 72)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

The aforementioned operating modes are valid for the block with all control channels (`MV1` ... `MV4`). Moreover, individual control channels can be tracked; see chapter ModPreCon functions (Page 682) for more information.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and bumpless switchover in the Manual and automatic mode for control blocks (Page 72) section.

Note

In contrast to PID controllers, it is permitted to run the ModPreCon block in "automatic mode" without its actuating signals affecting the process because there is no risk of integrator windup.

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the Manual and automatic mode for control blocks (Page 72) section.

"Out of service"

You can find general information about the "Out of service" mode in the Out of service (Page 71) section.

See also

ModPreCon block diagram (Page 707)

ModPreCon I/Os (Page 696)

ModPreCon messaging (Page 696)
ModPreCon error handling (Page 695)
Description of ModPreCon (Page 676)

5.6.3 ModPreCon functions

Functions of ModPreCon

The functions for this block are listed below.

Generating and limiting the manipulated variable

The manipulated variable `MV1` ... `MV4` (hereinafter referred to as `MVx`, x = 1 ... 4) can be generated as follows:

ManAct	MVxTrkOn	MVx	Limit monitoring	State
1	-	`Manx`	`ManxHiLim` `ManxLoLim`	"Manual mode", set by the operator
0	1	`MVxTrk`	`MVxHiLim` `MVxLoLim`	Tracking with limitation
0	0	Automatic manipulated variable	`MVxHiLim` `MVxLoLim`	"Automatic mode": Predictive controller algorithm

Remark

If the controller is in "Out of service" mode, the output parameters `MV1` ... `MV4`, depending on the `Feature` Bit (Neutral position manipulated variable takes effect at startup (Page 165)), are set to the last valid value in manual mode or the corresponding neutral position manipulated variable (`SafePos1` ... `SafePos4`). Refer to the Out of service (Page 71) section for more on this.

The limited operating range (between `MVxHiLim` and `MVxLoLim`) is typically smaller in automatic mode than in manual mode. With regard to the limited range of validity of a linear process model for approximating a non-linear process response, this allows the stability of the closed control loop to be guaranteed within the control range in automatic mode.

The gradients of the manipulated variable (changes per second) are limited to `MV1RaLim` to `MV4RaLim` in "automatic mode". Gradient limitation applies both to the positive and negative directions.

Tracking and limiting a manipulated variable

The block provides the standard function Tracking and limiting a manipulated variable (Page 192).

In contrast to PID controllers, tracking the manipulated variables (`MV1` ... `MV4`) is enabled for specific channels via one of the input parameters `MV1TrkOn` ... `MV4TrkOn`. The corresponding manipulated variable is then tracked by the interconnectable input parameters `MV1Trk` ... `MV4Trk`.

Setting the setpoint internally

With this block, the setpoint must always be set internally at the I/Os SP1 ... SP4. These are normally operated in the faceplate. In special situations, you can interconnect the setpoints but they can then no longer be changed using the faceplate.

Setpoint tracking in manual mode

In this situation (SP_TrackCV = 1), the internal setpoints SP1 ... SP4 are tracked to the assigned process values CV1 ... CV4 in "manual mode". This function allows a bumpless transfer to "automatic mode". After the transfer, the setpoints can be changed by the operator again.

Setpoint filters

The setpoint filter is the only way of changing the action of the predictive controller without having to create a new user data block with the MPC Configurator and reinitialize the controller. The specified time constant PreFilt1 ... PreFilt4 of the setpoint filter can be interpreted as the required settling time of this CV channel following a setpoint step change. As the time constant setting increases, the controller works more slowly and less aggressively. In particular, this reduces the influence of a setpoint step change in one control channel on neighboring control channels.

Internally, the ModPreCon block works with sets of future setpoints that are compared with the predicted movements of the controlled variables. Without the setpoint filter, it is assumed that the current setpoint will continue to remain valid in the future within the prediction horizon. If there is a setpoint step change, this means that the full value of the new setpoint will be required in the near future although the process cannot achieve this (according to the prediction). With the setpoint filter, an asymptotic setpoint trajectory (first order) is calculated from the current process value to the required setpoint so that the required setpoint is reached in the specified time.

Note

The setpoint filter also comes into effect without a setpoint step if the process value deviates significantly from the setpoint due to disturbances. This means that the filter not only slows down the control response but indirectly also the response to disturbances.

The control response can only be slowed down by the setpoint filter and not accelerated; when the value is 0, the prefilter is deactivated. It is therefore advisable to set the basic controller action in the MPC Configurator with the "Manipulated variable change penalty" parameter and then to optimize this in the software using the function for simulation of the closed control loop.

The setpoint filter should then only be used for fine tuning of the action in the operational system.

Note

Do not select an excessively long filter time constant. Values that are greater than the control horizon of the predictive controller (s. ModPreCon user DB, header comment of the SCL source: "control horizon" times "controller sample time" in seconds) can have undesirable effects when the dead band is used at the same time. If you wish to dramatically slow down the reaction of the controller, it is better to use the penalty of the manipulated variable changes in the MPC Configurator.

Simulating signals

The block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Controlled variable (`SimCVx` , `SimCVxLi`)

Selecting a unit of measure

The block provides the standard function Selecting a unit of measure (Page 207).

Error signal generation and dead band

The block provides the standard function Error signal generation and dead band (Page 188).

In the predictive controller, the error signal is generated over the entire prediction horizon for each control channel as the deviation between the predicted movement of the process (starting at the current process value `CV1` ... `CV4`) and future setpoint settings (ending at `SP1` ... `SP4`) and used to calculate the manipulated variable.

In principle, the effect of the dead bands `SP1DeadBand` to `SP4Deadband` is the same as in a PID controller, but extends over the entire future prediction horizon. In other words, if, for example, the predicted controlled variable `CV1` in the entire prediction horizon is within the band `SP1` ± `SP1DeadBand`, the controller sees no reason whatsoever to change any manipulated variable. These are therefore also known as `CV` bands. In contrast to the manipulated variable limits, these are not hard constraints that need to be adhered to at all costs.

In multivariable controllers, it is advisable to make use of the fact that from the perspective of the application only some of the controlled variables need to move to a specified setpoint exactly while others only need to remain within a defined range.

A typical example would be quality characteristics for which a tolerance band is specified. While a dead band in a PID controller tends to put stability at risk, `CV` bands in individual controller channels generally relieve the multivariable controller overall.

Using `CV` bands, the action of a soft override control can be achieved.

Use case for error signal generation with dead band

As long as the pressure in a reactor remains within the set safety limits, the controller is interested only in product quality. However, as soon as the pressure threatens to leave the permitted range (in other words, in the prediction it moves towards an illegal value in the future), the pressure control cuts in. By weighting the controlled variables in the performance index (see MPC Configurator), the user can specify that predicted violations of the pressure limits are given a particularly high weighting.

Predictive controller algorithm

The ModPreCon block is derived from the familiar DMC algorithm (**D**ynamic **M**atrix **C**ontrol) . Future changes to the manipulated variable within the control horizon are calculated according to the formula:

$$\Delta\underline{\vec{u}} = \underline{C} \cdot (\underline{\vec{w}} - \underline{\vec{f}})$$

Where:

- w contains the time series of the future setpoints
- f contains the predicted free movement of the controlled variables (with constant manipulated variables) in the future
- C is the constant controller matrix calculated by the MPC Configurator. C includes both the process model and the weighting of the manipulated variable changes and the controlled variables from the performance index of the optimization.

Based on the principle of the receding horizon, only the first value is taken from the vector of the optimum manipulated variable changes over the entire control horizon and applied to the process. In the next step, the newly arrived process values are taken into account and the calculation repeated over the entire prediction horizon.

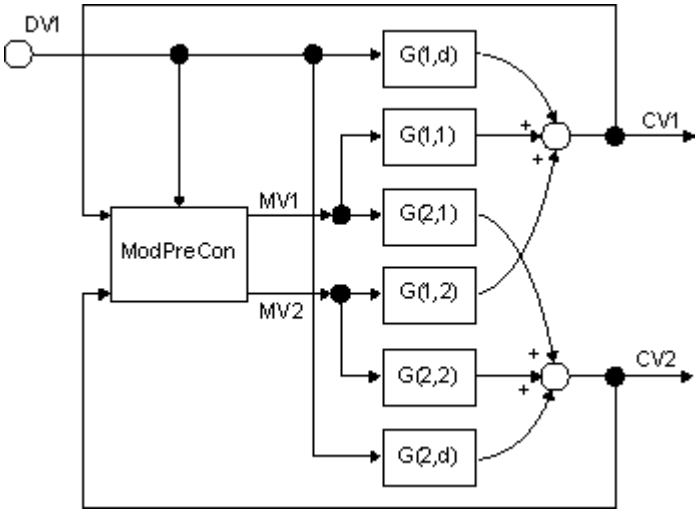
With predictive controllers, the manipulated variable changes are based on the control deviations predicted in the future, while with a PID controller, they are based on error signal of the past (possibly also integrated). This can be interpreted as a "looking ahead" strategy.

Anti-windup

When manipulated variable limits are active, anti-windup measures are taken automatically within the controller. The prediction equations use the real limited values of the manipulated variable instead of theoretically calculated values.

Model-based disturbance compensation

Model-based disturbance compensation can and should be used when a known disturbance has a strong influence on the process and its cause can be measured.



The effects of a measurable disturbance (`DV1` I/O) on all controlled variables `CV1` ... `CV4` can be estimated when the controller is taken into manual mode. This means that no movements of the controlled variables whatsoever result from changes to the manipulated variable and all movements result from the disturbance. If the disturbance can be measured but cannot be actively adjusted, it may be necessary to search through a data archive to find the time segments in which the disturbance changed.

The identification of the transfer functions from the disturbance variable `DV1` to all controlled variables `CV1` ... `CV4` (disturbance model, in the graphic above G(1.d) and G(2.d)) is performed with the MPC Configurator and is analogous to the identification of the main transfer functions (G(1.1) to G(2.2)). The measured disturbance variable is then switched to the `DV1` input of the ModPreCon block and disturbance compensation is activated with `DV_On` = 1. As a result, the effect of the measurable disturbance is taken into account in the prediction and the controller can start counter measures in advance before the disturbance can have a massive influence on the controlled variables.

Such disturbance compensation is especially effective when the disturbance is constant in sections and changes from time to time. If a disturbance changes constantly or oscillates, however, the feedforward control is not enabled during operation of the controller to avoid constant oscillation of the manipulated variables, although it should be taken into account in the MPC Configurator when creating the process model.

If there is no disturbance model in the user data block, the `DV1` input is ignored.

Typical examples of measurable disturbances are inlet volumes in distillation columns or throughput of continuous reactors.

Predictive controller with more than one measurable disturbance

If you want to plan for more than one measurable disturbance in an application, but do not need all four disturbances from the ModPreCon block, you can dedicate the first of the previously unused control channels for disturbance variable feedforwarding.

Example: You only have two control actions available, so only use MV1 and MV2. Then connect the additional measurable disturbance variable DV2 to the MV3Trk input parameter and set MV3TrkOn = 1. For recording training data for the predictive controller, declare MV3Trk as the third disturbance and also use the CFC trend recorder to record the effect of changes to DV2 on all controlled variables.

Use the MPC Configurator to then determine a process model that describes the effects of DV2. However, if the DV2 in "automatic mode" of the controller changes due to external influences, the effect of such change is taken into consideration for predicting future process reactions through MV3Trk, and predictive methods can be used to compensate. The performance of the disturbance compensation is exactly the same as that for regular feedforward control via the input parameters DV and DV_On = 1.

If you want to disable this disturbance compensation in runtime with the unused MV3 manipulated variable, you need to insert a MV3Trk a selector block in front of the SelA02In input. This allows you to set a constant zero for MV3Trk instead of the measured value DV2, which stops the effect of MV3 on prediction. (Due to this reassignment, MV3TrkOn must always remain 1 to prevent the controller from changing the value of MV3.)

This way, up to four measurable disturbances can be selected. However, the sum of the manipulated variables and disturbances may not exceed a total number of five.

Control of square and non-square systems

In multivariable controllers, the number of manipulated variables should ideally be the same as the number of controlled variables. This is known as a "square system". As long as constraints do not influence operation, the controller can, in principle, track all controlled variables exactly to the selected setpoints.

If there are less manipulated variables than controlled variables, or individual manipulated variables have reached their limits, there is no freedom in the control problem. This means that it is not possible for all setpoints to be reached exactly.

The ModPreCon algorithm then finds a compromise that can be influenced by the selection of controlled variable weights (priorities) in the MPC Configurator: Controlled variables with higher priority will have lower control deviations.

Note

Since the ModPreCon block is a lean predictive controller algorithm without online optimization, there can be no general guarantee that the compromise found is optimum in a mathematical sense; in other words, it is the minimum of the fit function taking into account the manipulated variable limits. In most practical situations, however, the controller finds sensible compromises.

The static operating point optimization it not a dynamic online optimization in this sense, i.e. it does not change anything in the above-mentioned restriction.

If there are more manipulated variables than controlled variables or if some of the controlled variables are already within their setpoint bands, there is surplus freedom in the control problem. A lean predictive controller algorithm, however, cannot recognize this situation explicitly and use the free manipulated variables for optimization. The ModPreCon block therefore moves all manipulated variables to values that meet the aims in terms of controlled variables and then leaves them there. In some situations, however, it can be useful to provide

the controller with more manipulated variables than controlled variables, for example when the effect of individual manipulated variables is too restricted.

Another approach is to define the excess manipulated variables as pseudo controlled variables at the same time. You do this by assigning a setpoint with low priority to the pseudo controlled variables. The controller then attempts to achieve the important control aims as first priority and, at the same time, attempts to reach certain ideal values for the individual manipulated variables.

Control of linear and non-linear systems

The ModPreCon algorithm is based on a linear, time invariant process model. As a result, in much the same way as a PID controller, it is suitable above all for controlling non-linear systems around a fixed operating point.

Again analogous to the PID controller, there are, however, several possibilities with which the area of application can be extended with non-linear systems:

Compensation functions between controller and controlled system:

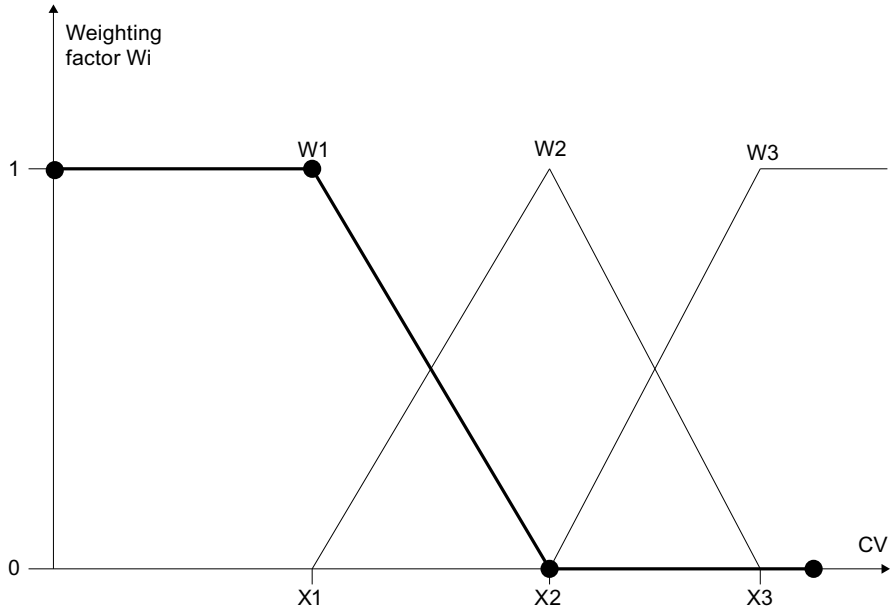
It is, for example, possible to compensate the effect of a non-linear valve characteristic curve using a polygon block between the `MV` output and the control input of the valve block. Care must be taken when implementing the manipulated variable limits. In the same way, the effect of a non-linearity at the output of the controlled system (for example a sensor characteristic curve) can be compensated by a polygon block before the `CV` input of the controller. Remember that the corresponding SP must also be transformed accordingly. In both cases, the compensation functions become part of the controlled system from the perspective of the controller. The aim is always to keep the overall response of the controlled system consisting of process and compensation elements as linear as possible.

Multimodel control:

This approach is related to the basic idea of operating-point-based parameter control with PID controllers. Since the model parameter of the ModPreCon block cannot be modified in runtime, however, the control strategy for selecting the suitable parameter set becomes a control strategy for selecting the suitable model.

Several ModPreCon instances with different models for different operating points run at the same time. The local optimal models are determined by starting the process at the various operating points with small amplitudes, so that only the reaction of the non-linear process in the ambience of this operating point is registered.

The final manipulated variable for each manipulated variable is formed as a weighted mean value of the manipulated variables proposed by the controller instances. (It is recommended that experiments for starting the process for the MPC Configurator are only performed after implementing the functions for adding the manipulated variables, in order to ensure that the same conditions applicable when the model is in operation actually take effect.)



The weighting factors 0 ... 1 are formed in the same way as the membership functions known from fuzzy logic so that the sum of all weights is always one and each controller has the highest weighting at its own operating point. A polygon with 4 or 5 interpolation points is used to calculate each individual weighting factor. The weighting factors are calculated based on a specific measurable `PV` variable of the process, which is representative for the operating point of the process. This can be one of the `CVx` controlled variables, although it does not have to be. The abscissa of the interpolation points of all polygons is selected in such a way that they cover the entire value range of `PV` in order to avoid extrapolation errors.

One should note in this regard that the only non-linear effects in the full multi-variable control loop that can be modeled are those that correlate exactly to a representative `PV` variable. This approach is therefore not suitable for cases in which individual partial transfer functions demonstrate non-linear effects that depend on various, totally independent variables.

To ensure the stability of the overall control loop, all subcontrollers must be at least stable at all operating points. In contrast to PID controllers however, an MPC is not affected by windup problems if it temporarily runs in "automatic mode" but cannot intervene in the real process (weighting factor zero).

One of the controller instances is defined as the main controller and shown in the operator faceplate on the OS. All others are connected in such a way that they adopt the operating mode ("manual"/"automatic mode") and the setpoints from the main controller. The manual manipulated variables are passed to the secondary controller via the tracking inputs. This means that no operator intervention is required on secondary controllers.

Note

The manipulated variables of the main controller can be used when switching from automatic to manual mode. If the process was in the operating range of a secondary controller beforehand, the current manipulated variables in the process can deviate significantly. In this case, you should apply the actual manipulated values in effect by manual input in the faceplate of the primary controller.

You can find an example for multi-model controlling in the example project of Advanced Process Library under Predictive control of a non-linear process (ModPreConNonLinSim) (Page 2355).

Trajectory control:

This approach neatly combines the advantages of an open loop controller (Feedforward Control) with those of a closed loop controller with process value feedback (Closed Loop Control). The controller follows a previously optimized trajectory of setpoints and manipulated variables; in other words, it only needs to compensate small deviations between the stored trajectory and the current plant state. A trajectory is an optimum series of manipulated variables over time and the process values that match them. The required manipulated variables are read via the inputs `MV1Traj` ... `MV4Traj` into the ModPreCon block and added to the values of the manipulated variable calculated by the algorithm (in automatic mode only). Among other things, the advantage of this is that the effective manipulated variable acting on the process can be configured and is limited to the sum of the trajectory and controller action. The process values from the trajectory are switched to the corresponding setpoint specifications `SP1` ... `SP4` of the controller. As long as the process reacts exactly as planned in the trajectory, it will respond to the series of manipulated variables from the trajectory with the corresponding series of process values and the control deviation is zero. It is generally known that a non-linear dynamic process can linearized around a fixed operating point or a steady state of the system. It is also possible to linearize it around a trajectory.

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `DV1.ST`
- `CV1.ST`
- `CV2.ST`
- `CV3.ST`
- `CV4.ST`

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
4	Setting switch or button mode (Page 166)
15	Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165)
16	Neutral position manipulated variable takes effect at startup (Page 165)
24	Enabling local operator authorization (Page 157)

Operator permissions

The block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	Not used
2	1 = Operator can switch to "Out of service" mode
3	Not used
4	Not used
5	1 = Operator can change the setpoint 1
6	1 = Operator can change the manipulated variables of all channels
7	1 = Operator can change operating high limits of the setpoints for all channels
8	1 = Operator can change operating low limits of the setpoints for all channels
9	1 = Operator can change the setpoint 2
10	1 = Operator can change the setpoint 3
11	1 = Operator can change the setpoint 4
12	1 = Operator can change the setpoint filter of all channels
13 - 16	Not used
17	1 = Operator can enable the track setpoint in "manual mode" function
18	1 = Operator can activate the model-based disturbance compensation function
19	1 = Operator can enable the function "Prediction without control response"
20 - 22	Not used
23	1 = Operator can change the dead band parameter of all channels
24	1 = Operator can change the tuning parameters
25	1 = Operator can change the simulation value SimPV1..4
26	1 = Operator can activate the Simulation function
27	1 = Operator can activate the Release for maintenance function
28	1 = Operator can change the manipulated variable limits of all channels
29	1 = Operator can change the gradient limits of manipulated variables of all channels
30	0 = Operator can set the integrated static operating point optimization
31	1 = Operator can set the integrated static operating point optimization

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Release for maintenance

The block provides the standard function Release for maintenance (Page 64).

Specifying the display area for process and setpoint values as well as operations

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

In contrast to PID controllers, there are no separate parameters for bar limits. The setpoints limits are used for all setpoint and actual value bars; manual limits are used as bar limits for all manipulated variable bars.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Integrated static operating point optimization

The integrated static operating point optimization can be used if no exact SPi setpoint is specified for at least one controlled variable (index i =1...4), but rather the CVi process value must remain within a tolerance range SPiOptHiLim…SPiOptLoLim. The tolerance range must, of course, be within the valid setpoint limits SPiHiLim…SPiLoLim for this control channel. The tolerance range SPiOptHiLim…SPiOptLoLim is not coupled to the operator-controlled setpoint SPi. If the setpoint is changed, the tolerance range is not automatically shifted. If you want to do this nevertheless, interconnect the SPiOpOut output parameters via two adders to the width of the tolerance range at the inputs SPiOptHiLim and SPiOptLoLim.

From an economic perspective, different values within the tolerance range can be more or less favorable. With the help of the optimization function, the optimal economic point can be found within the tolerance range.

This is done by defining a target function (performance criterion), which depends on the manipulated variable and controlled variable of the predictive controller. This can be, for example, the economic yield of plant operation per time unit or it may involve specific costs or energy consumption.

J= GradMV1*MV1 +GradMV2*MV2 +GradMV3*MV3 +GradMV4*MV4

+GradCV1*CV1 +GradCV2*CV2 +GradCV3*CV3 +GradCV4*CV4

+J0

You specify the individual GradXVi coefficients of the gradient vector as input variables at the ModPreCon function block in the CFC or in the parameter view of the faceplate. If individual coefficients vary with time, e.g. they are dependent on current market prices, you can also interconnect these input variables. If individual manipulated or controlled variables have no influence on the performance criterion, leave the corresponding coefficients at the default value, zero.

You can use the binary J_Mini input parameter to specify whether the target function is to be maximized or minimized, based on whether this involves yields or costs (J_Mini = 1 : minimization).

The term J0 combines all contributions to the target function that do not depend on manipulated variables and controlled variables. These contributions have no effect on the

optimum values in the decision variables, but are applied for calculating the current value of employed performance criterion similar to the above-mentioned formula.

Within the controller, the terms of the target function that depend on manipulated variables are converted to make their dependence on the controlled variables visible. To do this, the inverse stationary process model from the MPC Configurator is used. This requires that the number of manipulated variables matches the number of controlled variables. If the number of manipulated variables does not match the number of controlled variables, the largest possible square submodel in the matrix of the transfer functions is truncated from the top left. If there are more manipulated variables than controlled variables, for example, only the first manipulated variables are used, in accordance with the number of controlled variables.

Constraints for the controlled variables take the form of the above-mentioned tolerance ranges for setpoints. The controller takes care of adhering to the manipulated variable limits, in any case; they do not have to separately specified as constraints for the optimization.

Enable the optimization using the binary input variable OptimizeOn in the controller faceplate. The optimizer then returns setpoints within the tolerance ranges that are optimal for the performance criterion. These setpoints are then sent to the control algorithm, which handles them in the same way as conventionally specified setpoints (with or without dead band). The operable SP1...SP4 setpoints are not tracked to the optimized setpoints; when optimization is disabled, the old setpoints from the faceplate take effect once again. When selecting variables for archiving and graphic plotter, ensure you use the SP1Out...SP4Out setpoint actually in effect and not the SP1...SP4. input variables.

The current value of the performance criterion is displayed at the J_Actual output variables.

You can find additional information about the topic of static operating point optimization in the online help for the MPC configuration editor.

Display of the prediction of free movement

The prediction of free motion is a forecast for the future behavior of the process within the overall prediction horizon, under the assumption that all manipulated variables are frozen at their current values. The time length of the prediction horizon is indicated in the output parameter PrediHorizon in the [s] unit.

The prediction of free motion is recalculated in each sampling step within the control algorithm. If the manipulated variables is to a constant value in manual mode, the prediction of the free movement is actually a realistic prediction for the future process response. It can therefore be represented graphically in the faceplate at least in terms of quality. For this purpose, five equidistant interpolation values are copied from the prediction horizon, and displayed in the standard view of the faceplate as a vertical bar next to the current process value.

Example: The prediction horizon is 1800s=30min and the current time is labeled with the index k. The prediction for k+6min, and next to it k+12min, up to k+30min then appears on the right next to the bar of the current process value. If the upper border of the bar is conceptually connected with a line (red in the picture red), you can imaging the curve created by the future course of the process value over the next half hour.

In automatic mode, the value of the manipulated variables changes with each sampling step. The prediction of free movement is then only a fictional mathematical formulation within the algorithm, and not a realistic prediction for the future process response. This is why the prediction is only displayed in manual mode. The display can be generally suppressed using Feature.Bit31.

Prediction without control response

In this special "operating mode" (comparable to the block-internal simulation), the controller only monitors the process and indicates what it would like to do in the next sampling step without actively intervening in the process. This allows you as the user to build trust before "switching active" the controller the first time, i.e. actively intervening in the process.

"Prediction Mode" is activated via the binary input variable PredictMode or in "Parameters" in the faceplate view. Setpoints and process values are read as in normal automatic mode. The prediction of the free movement and manipulated variable change for the next sampling step are calculated as in normal automatic mode. The starting point for the prediction of the manipulated variable for the next sampling step, however, is the current process value of the follow-up control loop at the MV1Trk…MV4Trk tracking inputs. The predicted manipulated variables are not output at the normal outputs MV1...4, but rather at the MV1Pred...MV4Pred outputs, which were especially introduced for this purpose, and are displayed in the standard view of the faceplate on the left next to MV1...4, as long as "Prediction Mode" is active.

- When the controller is in automatic mode, in "Prediction Mode" all MVi (i=1..4) are set to match the assigned MViTrk input parameters, similar to tracking mode.
- When the controller in manual mode, all MVi are set to the desired manual values regardless of "Prediction Mode".
- When "Prediction Mode" is disabled, all MViPred always equal the assigned MVi.

Automatic process trigger for model identification

In order to determine the process model for the model predictive controller, the process must be artificially triggered in order to observe its dynamic response and record it in the form of training data. This trigger can be specified manually in manual mode of the controller.

Alternatively, a suitable trigger signal can be generated automatically in the form a defined, symmetrical sequence of manipulated variable jumps. The trigger signals are calculated by an auxiliary function block, "AutoExcitation", which is built into the process tag type and interconnected with ModPreCon .

Additional MV1Excite…MV4Excite input variables are required for this on the controller. Process triggering performed in manual mode of the controller, because automatic mode cannot be activated before modeling. The new process trigger "operating mode" can only be controlled via the ExciteOn input bit on the Engineering System, and not on the operator station, since the CFC is needed for data recording in any case. However, process triggering must be displayed on the OS in standard view at the lower left.

Manual intervention per faceplate remains possible even during the triggering. The values of the MV1Excite...MV4Excite input parameters are therefore only written to the MV1Man...MV4Man manual values event-based, but only if they change.

You can find additional details on automatic process triggering in the online help for the MPC configuration editor.

See also

Description of ModPreCon (Page 676)

ModPreCon messaging (Page 696)

ModPreCon I/Os (Page 696)

ModPreCon block diagram (Page 707)
ModPreCon error handling (Page 695)
ModPreCon modes (Page 681)

5.6.4 ModPreCon error handling

Error handling of ModPreCon

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` output parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Default value when the block is installed; this message is irrelevant.
0	There is no error.
3	Configuration error: Block is inserted in the wrong OB, cycle time of the OB is not suited for controller configuration.
32	The value of `CV1` can no longer be displayed in the real number field or is not a number.
33	The value of `CV2` can no longer be displayed in the real number field or is not a number.
34	The value of `CV3` can no longer be displayed in the real number field or is not a number.
35	The value of `CV4` can no longer be displayed in the real number field or is not a number.
36	The `MV_Trk1` value can no longer be displayed in the real number field or is not a number.
37	The `MV_Trk2` value can no longer be displayed in the real number field or is not a number.
38	The `MV_Trk3` value can no longer be displayed in the real number field or is not a number.
39	The `MV_Trk4` value can no longer be displayed in the real number field or is not a number.
51	`AutModLi` = 1 and `ManModLi` = 1
90	The controller matrix could not be loaded from the user data block.

The ErrorOpt output parameter is used to output the status of the lower-level LPOptim block. See ModPreCon I/Os (Page 696) for more.

See also

ModPreCon block diagram (Page 707)
ModPreCon messaging (Page 696)
ModPreCon functions (Page 682)
ModPreCon modes (Page 681)
Description of ModPreCon (Page 676)
Description of LPOptim (Page 917)

5.6.5 ModPreCon messaging

Messaging

This block does not offer messaging.

See also

Description of ModPreCon (Page 676)
ModPreCon functions (Page 682)
ModPreCon I/Os (Page 696)
ModPreCon block diagram (Page 707)
ModPreCon error handling (Page 695)
ModPreCon modes (Page 681)

5.6.6 ModPreCon I/Os

I/Os of ModPreCon

Input parameters

Parameter	Description	Type	Default
`AutModLi*`	1 = Automatic mode via interconnection or SFC (controlled by `ModLiOp = 1`)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`AutModOp*`	1 = Automatic mode via operator (controlled by `ModLiOp = 0`)	BOOL	0
`BatchEn`	1 = Enabled for allocation by batch control	BOOL	0
`BatchID`	Batch ID	DWORD	16#00000000
`BatchName`	Batch name	STRING[32]	''

Parameter	Description	Type	Default
CV1	Control variable 1 (process value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
CV1_Unit	Unit of measure for control variable 1 (process value)	INT	1001
CV2	Manipulated variable 2 (process value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
CV2_Unit	Unit of measure for manipulated variable 2 (process value)	INT	1001
CV3	Manipulated variable 3 (process value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
CV3_Unit	Unit of measure for manipulated variable 3 (process value)	INT	1001
CV4	manipulated variable 4 (process value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
CV4_Unit	Unit of measure for manipulated variable 4 (process value)	INT	1001
DB_No	Number of the data block in which the controller data is saved.	INT	0
DV_On	1 = Activate the feedforward control from DV1	BOOL	1
DV1	Disturbance variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
EN	1 = Called block will be processed	BOOL	1
ExciteOn	1 = Automatic process trigger; MViExcite input parameters are written to the MVi outputs	BOOL	0
Feature	I/O for additional ModPreCon functions (Page 682)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
GradCV1	Gradient vector for performance criterion, element (factor) for CV1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
GradCV2	Gradient vector for performance criterion, element (factor) for CV2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
GradCV3	Gradient vector for performance criterion, element (factor) for CV3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
GradCV3	Gradient vector for performance criterion, element (factor) for CV3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
GradCV4	Gradient vector for performance criterion, element (factor) for CV4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
GradMV1	Gradient vector for performance criterion, element (factor) for MV1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
GradMV2	Gradient vector for performance criterion, element (factor) for MV2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
GradMV3	Gradient vector for performance criterion, element (factor) for MV3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
GradMV4	Gradient vector for performance criterion, element (factor) for MV4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
J_Actual_Unit	Physical unit of performance J_Actual	INT	0
J_Mini	1 = Minimize, 0 = maximize	BOOL	0
J0	Value of the performance criterion in the operating point	REAL	0
ManModLi*	1 = Manual mode via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = Manual mode via OS operator (controlled by ModLiOp = 0)	BOOL	0
ModLiOp	Operating mode switchover by: • 0 = Operator • 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MV1_Unit	Unit of measure for manipulated variable 1	INT	1342
MV1Excite	MV1 for automatic process trigger	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV1HiLim	High limit of manipulated variable MV1	REAL	100.0
MV1LoLim	Low limit of manipulated variable MV1	REAL	• 0.0

Parameter	Description	Type	Default
MV1Man*	Manual value: Operator input for setting the manipulated variable MV1 in manual mode	REAL	0.0
MV1ManHiLim	High limit of manipulated variable MV1 in manual mode	REAL	100.0
MV1ManLoLim	Low limit of manipulated variable MV1 in manual mode	REAL	0.0
MV1RaLim	Gradient limit of the manipulated variable MV1 per sampling step	REAL	100.0
MV1Traj	Trajectory value that is added to the manipulated variable MV1	REAL	0.0
MV1Trk	Tracking value for the manipulated variable MV1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV1TrkOn	1 = Tracking of manipulated variable MV1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV2_Unit	Unit of measure for manipulated variable 2	INT	1342
MV2Excite	MV2 for automatic process trigger	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV2HiLim	High limit of manipulated variable MV2	REAL	100.0
MV2LoLim	Low limit of manipulated variable MV2	REAL	0.0
MV2Man*	Manual value: Operator input for setting the manipulated variable MV2 in manual mode	REAL	0.0
MV2ManHiLim	High limit of manipulated variable MV2 in manual mode	REAL	100.0
MV2ManLoLim	Low limit of manipulated variable MV2 in manual mode	REAL	0.0
MV2RaLim	Gradient limit of the manipulated variable MV2 per sampling step	REAL	100.0
MV2Traj	Trajectory value that is added to the manipulated variable MV2	REAL	0.0
MV2Trk	Tracking value for the manipulated variable MV2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV2TrkOn	1 = Tracking of manipulated variable MV2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV3_Unit	Unit of measure for manipulated variable 3	INT	1342
MV3Excite	MV3 for automatic process trigger	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV3HiLim	High limit of manipulated variable MV3	REAL	100.0
MV3LoLim	Low limit of manipulated variable MV3	REAL	0.0

Parameter	Description	Type	Default
MV3Man*	Manual value: Operator input for setting the manipulated variable MV3 in manual mode	REAL	0.0
MV3ManHiLim	High limit of manipulated variable MV3 in manual mode	REAL	100.0
MV3ManLoLim	Low limit of manipulated variable MV3 in manual mode	REAL	0.0
MV3RaLim	Gradient limit of the manipulated variable MV3 per sampling step	REAL	100.0
MV3Traj	Trajectory value that is added to the manipulated variable MV3	REAL	0.0
MV3Trk	Tracking value for the manipulated variable MV3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV3TrkOn	1 = Tracking of manipulated variable MV3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV4_Unit	Unit of measure for manipulated variable 4	INT	1342
MV4Excite	MV4 for automatic process trigger	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV4HiLim	High limit of manipulated variable MV4	REAL	100.0
MV4LoLim	Low limit of manipulated variable MV4	REAL	0.0
MV4Man*	Manual value: Operator input for setting the manipulated variable MV4 in manual mode	REAL	0.0
MV4ManHiLim	High limit of manipulated variable MV4 in manual mode	REAL	100.0
MV4ManLoLim	Low limit of manipulated variable MV4 in manual mode	REAL	0.0
MV4RaLim	Gradient limit of the manipulated variable MV4 per sampling step	REAL	100.0
MV4Traj	Trajectory value that is added to the manipulated variable MV4	REAL	0.0
MV4Trk	Tracking value for the manipulated variable MV4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV4TrkOn	1 = Tracking of manipulated variable MV4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = Allocated by SIMATIC BATCH	BOOL	0
OosLi	1 = Out of service, via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = Out of service, via OS operator	BOOL	0

Parameter	Description	Type	Default
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations	DWORD	16#00000000
OptimOffOp	1 = Disable optimization, normal setpoints SP1...SP4 in effect	BOOL	0
OptimOnOp	1 = Enable optimization, optimized setpoints SP1Out ... SP4Out in effect	BOOL	0
OS_Perm	I/O for operator permissions	STRUCT • Bit 0: BOOL • Bit 25: BOOL • Bit 31: BOOL	- • 1 • 1 • 1
PredictMode	1 = "Prediction mode" enabled, prediction only, no intervention in the process	BOOL	0
PreFilt1	Settling time [s] of the setpoint filter for setpoint SP1	REAL	0.0
PreFilt2	Time constant [s] of the setpoint filter for setpoint SP2	REAL	0.0
PreFilt3	Time constant [s] of the setpoint filter for setpoint SP3	REAL	0.0
PreFilt4	Time constant [s] of the setpoint filter for setpoint SP4	REAL	0.0
Restart*	1 = Restart of the block and adoption of the data from the user block that is entered at the input parameter DB_No	BOOL	1
SafePos1	Neutral position for MV1	BOOL	0
SafePos2	Neutral position for MV2	BOOL	0
SafePos3	Neutral position for MV3	BOOL	0
SafePos4	Neutral position for MV4	BOOL	0
SampleTime	Sampling time [s] (assigned automatically)	REAL	1.0
SelFp1	1 = Call a block saved in this parameter as an additional faceplate in the standard view	ANY	-
SelFp2	1 = Call a block saved in this parameter as an additional faceplate in the preview	ANY	-
SimCV1*	Controlled variable CV1 (process value) which is used with SimOn = 1	REAL	0.0
SimCV2*	Controlled variable CV2 (process value) which is used with SimOn = 1	REAL	0.0
SimCV3*	Controlled variable CV3 (process value) which is used with SimOn = 1	REAL	0.0
SimCV4	Controlled variable CV4 (process value) which is used with SimOn = 1	REAL	0.0
SimCV1Li	Controlled variable CV1 (process value) that is used for SimOnLi.Value = 1 (SimLiOp.Value =1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF

Parameter	Description	Type	Default
SimCV2Li	Controlled variable CV2 (process value) that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
SimCV3Li	Controlled variable CV3 (process value) that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
SimCV4Li	Controlled variable CV4 (process value) that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SP_TrkCV	1 = Setpoints follow the CVs in manual mode and in tracking	BOOL	0
SP1*	Setpoint 1 SP1.ST=FF: Operable in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
SP1DeadBand	Width of the dead band control of CV1	REAL	0.0
SP1HiLim	High limit for setpoint 1	REAL	100.0
SP1LoLim	Low limit for setpoint 1	REAL	0.0
SP1OptHiLim	High limit for optimization of setpoint 1	REAL	100.0
SP1OptLoLim	Low limit for optimization of setpoint 1	REAL	0.0
SP2*	Setpoint 2 SP2.ST=FF: Operable in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
SP2DeadBand	Width of the dead band control of CV2	REAL	0.0
SP2HiLim	High limit for setpoint 2	REAL	100.0
SP2LoLim	Low limit for setpoint 2	REAL	0.0
SP2OptHiLim	High limit for optimization of setpoint 2	REAL	100.0
SP2OptLoLim	Low limit for optimization of setpoint 2	REAL	0.0
SP3*	Setpoint 3 SP3.ST=FF: Operable in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
SP3DeadBand	Width of the dead band control of CV3	REAL	0.0
SP3HiLim	High limit for setpoint 3	REAL	100.0
SP3LoLim	Low limit for setpoint 3	REAL	0.0
SP3OptHiLim	High limit for optimization of setpoint 3	REAL	100.0

Parameter	Description	Type	Default
SP3OptLoLim	Low limit for optimization of setpoint 3	REAL	0.0
SP4*	Setpoint 4 SP4.ST=FF: Operable in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
SP4DeadBand	Width of the dead band control of CV4	REAL	0.0
SP4HiLim	High limit for setpoint 4	REAL	100.0
SP4LoLim	Low limit for setpoint 4	REAL	0.0
SP4OptHiLim	High limit for optimization of setpoint 4	REAL	100.0
SP4OptLoLim	Low limit for optimization of setpoint 4	REAL	0.0
StepNo	Batch step number	DWORD	16#00000000
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = Automatic mode is active 0 = Manual mode is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CV1Out	Output of manipulated variable 1 (process value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
CV2Out	Output of manipulated variable 2 (process value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
CV3Out	Output of manipulated variable 3 (process value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
CV4Out	Output of manipulated variable 4 (process value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. You can find information via the error numbers that are output by this block at ModPreCon error handling (Page 695)	INT	-1
ErrorOpt	Error number of the integrated optimization function, see the description of function block LPOptim	INT	0
Fut1_y1... Fut1_y5	Prediction of free movement of CV1 for five future points in time within the prediction horizon	REAL	0

Parameter	Description	Type	Default
`Fut2_y1... Fut2_y5`	Prediction of free movement of CV2 for five future points in time within the prediction horizon	REAL	0
`Fut3_y1... Fut3_y5`	Prediction of free movement of CV3 for five future points in time within the prediction horizon	REAL	0
`Fut4_y1... Fut4_y5`	Prediction of free movement of CV4 for five future points in time within the prediction horizon	REAL	0
`J_Actual`	Current value of the performance criterion	REAL	0
`Loop1Closed`	1 = Control loop for CV1 closed 0 = Control loop for CV1 open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`Loop2Closed`	1 = Control loop for CV2 closed 0 = Control loop for CV2 open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`Loop3Closed`	1 = Control loop for CV3 closed 0 = Control loop for CV3 open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`Loop4Closed`	1 = Control loop for CV4 closed 0 = Control loop for CV4 open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ManAct`	1 = Manual mode active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`MS_Release`	Release for maintenance: 1 = Release by OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`MV1`	Control variable 1 (control signal)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`MV1HiAct`	1 = High limit of manipulated variable 1 reached or exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`MV1LoAct`	1 = Low limit of manipulated variable 1 reached or exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`MV1Pred`	One-step prediction for MV1 in the "Prediction without control response" mode	REAL	0
`MV2`	Manipulated variable 2 (control signal)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
MV2HiAct	1 = High limit of manipulated variable 2 reached or exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV2LoAct	1 = Low limit of manipulated variable 2 reached or exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV2Pred	One-step prediction for MV2 in the "Prediction without control response" mode	REAL	0
MV3	Manipulated variable 3 (control signal)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV3HiAct	1 = High limit of manipulated variable 3 reached or exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV3LoAct	1 = Low limit of manipulated variable 3 reached or exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV3Pred	One-step prediction for MV3 in the "Prediction without control response" mode	REAL	0
MV4	Manipulated variable 4 (control signal)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV4HiAct	1 = High limit of manipulated variable 4 reached or exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV4LoAct	1 = Low limit of manipulated variable 4 reached or exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV4Pred	One-step prediction for MV4 in the "Prediction without control response" mode	REAL	0
NumberCVs	Number of controlled variables (process values) used	INT	0
NumberDVs	Number of disturbances used	INT	0
NumberMVs	Number of manipulated variables used	INT	4
OosAct	1 = Block is "out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OptimAct	1 = Optimization is active	BOOL	0

Parameter	Description	Type	Default
OptimAvailable	1 = Optimization available, 0 = Optimization not available, because old user data block is loaded	BOOL	0
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
PrediHorizon	Prediction horizon [s]	REAL	0
SP1OpOut	Copy of the operable setpoint 1 for step-enabling	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP2OpOut	Copy of the operable setpoint 2 for step-enabling	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP3OpOut	Copy of the operable setpoint 3 for step-enabling	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP4OpOut	Copy of the operable setpoint 4 for step-enabling	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP1Out	Setpoint 1 used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP2Out	Setpoint 2 used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP3Out	Setpoint 3 used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP4Out	Setpoint 4 used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1	DWORD	16#00
Status2	Status word 2	DWORD	16#00

See also

Description of ModPreCon (Page 676)
ModPreCon messaging (Page 696)
ModPreCon block diagram (Page 707)
ModPreCon modes (Page 681)

Neutral position for motors, valves and controllers (Page 48)

Opening additional faceplates (Page 203)

Description of OpStations (Page 399)

5.6.7 ModPreCon block diagram

ModPreCon block diagram

A block diagram is not provided for this block.

See also

ModPreCon I/Os (Page 696)

ModPreCon messaging (Page 696)

ModPreCon error handling (Page 695)

ModPreCon functions (Page 682)

ModPreCon modes (Page 681)

Description of ModPreCon (Page 676)

5.6.8 Operator control and monitoring

5.6.8.1 ModPreCon views

Views of the ModPreCon block

The block ModPreCon provides the following views:

- ModPreCon standard view (Page 708)
- ModPreCon trend view (Page 717)
- ModPreCon parameter view (Page 712)
- ModPreCon parameter view channel 1 to 4 (Page 714)
- ModPreCon preview (Page 715)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for ModPreCon (Page 718)

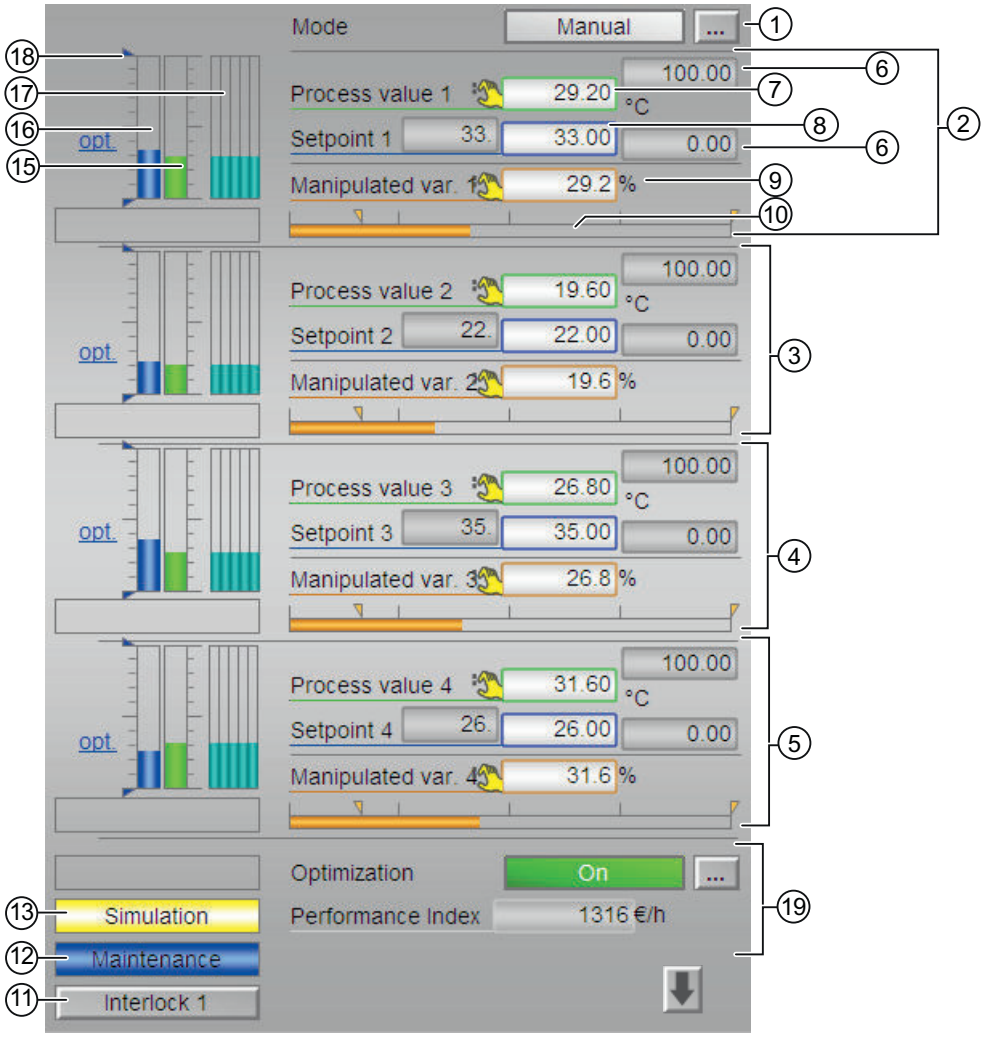
Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

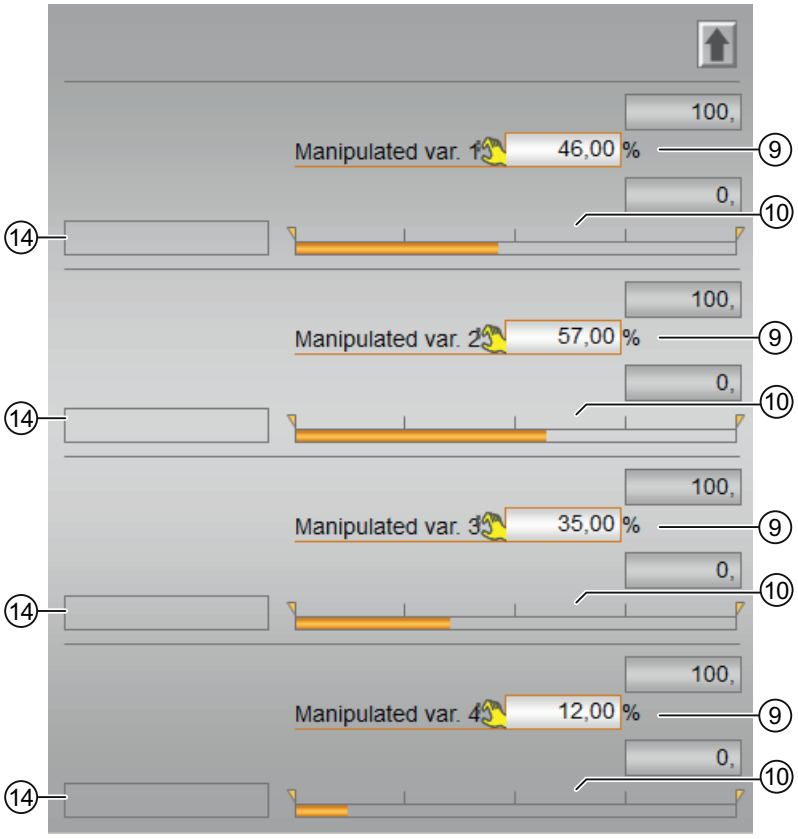
See also

Trend view (Page 299)

5.6.8.2 ModPreCon standard view

ModPreCon standard view





The standard view has an upper half and a lower half. You can change between the two halves with the arrow keys. The upper half shows all available controlled variable channels with their setpoints, while the lower half shows all available manipulated variable channels.

Upper screen half (controlled variables)

(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 72)
- Automatic mode (Page 72)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2), (3), (4) and (5) Displaying and switching for values for channels 1 to 4

This area always has the same layout for channels 1 to 4:

(6) High and low scale range for the process value

These values provide information on the display range for the bar graph of the process value. The scale range is defined in the engineering system.

(7) Displaying and changing the process value including signal status

This area shows the current process value with the corresponding signal status.

(8) Displaying and changing the setpoint including signal status

This area shows the current setpoint with the corresponding signal status. Refer to the Changing values (Page 253) section for information on changing the setpoint.

(11) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(12) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

- "Process excitation"
  The automatic process trigger is fed forward using the upstream block AutoExcitation for recording learning data for the MPC configurator. The manipulated variable step changes are added to the manipulated values 1 to 4 according to schedule. Avoid external disturbances to the process while the process trigger is running. The manipulated variables can be changed manually while the process trigger is running.

(13) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in the section Simulating signals (Page 58).

(15) Bar graph for the process value 1

There is a bar graph for the process value for every channel 1 to 4.

This area shows the current process value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(16) Bar graph for the setpoint 1

There is a bar graph for the setpoint for every channel 1 to 4.

This area shows the current setpoint in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(17) Prediction of free movement

This area shows you the prediction of free movement in the form of a bar graph. For each channel from 1 to 4, there is a bar graph for the prediction of free movement, that is, for the future behavior of the process within the overall prediction horizon, under the assumption that all manipulated variables are frozen at their current values.

This is why the prediction of free movement is only displayed in manual mode.

The value range of the bar graph matches the value range of the assigned setpoint and current value bar.

ModPreCon functions (Page 682)

(18) Displaying the limits

These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the engineering system (ES).

(19) Static operating point optimization

Activate the optimization using the button at bottom right. Activation means that the optimized setpoints SP1Out...SP4Out are actually used instead of the SP1...SP4 setpoints specified in the faceplate for the closed-loop control. (The actual calculation of the optimum setpoints depends on this, and is only performed if one of the input variables for the optimization has changed.) The current value the economic performance criterion J appears in the display field below.

When optimization is enabled, the optimum setpoints are displayed on the setpoint bar as small, horizontal lines and highlighted with the abbreviation "opt.". The numerical values of the optimum setpoints are then displayed left of the input fields for the setpoints.

Lower screen half (manipulated variables)

(9) Displaying and changing the manipulated variable including signal status

This area shows the current manipulated variable with the corresponding signal status. Refer to the Changing values (Page 253) section for information on changing the manipulated variable. You can only make a change in manual mode.

(10) Bar graph for the manipulated variable with limit display

This area shows the current manipulated variable in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES):

- Limits: `MVxHiLim` and `MVxLoLim`
- Display area: `MVxManHiLim` and `MVxManLoLim`

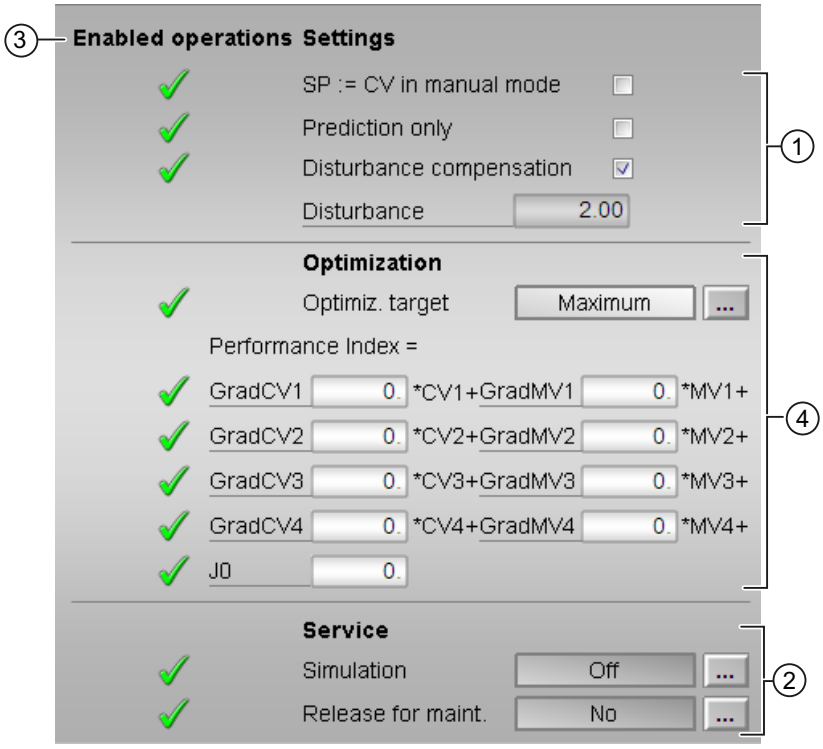
(14) Display for block states

There is a display for the states of the block for every channel 1 to 4:

- "Tracking"

5.6.8.3 ModPreCon parameter view

Parameter view of ModPreCon



(1) Settings

You can activate the following functions for the controller in this area:

- "SP := CV in manual mode": ☑ Bumpless switchover from "manual mode" to "automatic mode"
- "Prediction only" activate this special "operating mode" by selecting the check box. The controller then only listens in on the process and indicates what it would like to do in the next sampling step without actively intervening in the process
- "Disturbance compensation": ☑ Select disturbance feedforward
- "Disturbance variable"

You cannot change the disturbance variable, it can only be displayed.

(2) Service

You can select the following functions in this area:

- "Simulation"
- "Release for maintenance"

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Simulating signals (Page 58)
- Release for maintenance (Page 64)

(3) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

(4) Optimization

Direction of the optimization (minimize or maximize)

By default, the optimizer seeks to maximize the performance function, in the assumption that it is dealing with economic yield. If you want to search a minimum, however, because you are dealing with costs or consumption values, click this button.

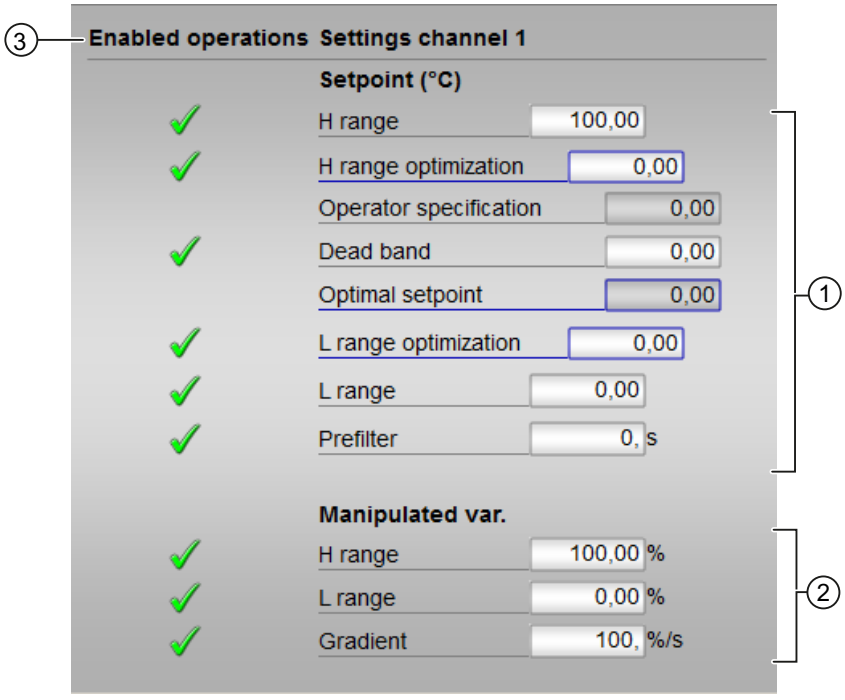
Specification of performance criterion for the operating point optimization

The performance criterion consists of a weighted sum of all manipulated and controlled variables. For each manipulated variable and controlled variable, enter the appropriate weighting factor, i.e. the coefficient of the gradient vector. Zero means that the value of the corresponding manipulated variable or controlled variable no direct influence on the economic yield. If the controller has less than four manipulated variables or controlled variables, the irrelevant variables are hidden automatically.

5.6.8.4 ModPreCon parameter view channel 1 to 4

Parameter view channel 1 to 4 for ModPreCon

The layout of the parameter view for channels 1 to 4 is always identical:



(1) Displaying and changing the limit parameters for the setpoint

You can change the following parameters for the setpoint in this area:

- "H range": High limit for setpoint operation
- "H range optimization": High limit for optimizing the setpoint
- "Operator input": Display of the setpoint entered in the standard view, cannot be operated here.
- "Dead band": Dead band (Page 61), Error signal generation and dead band section
- "Optimal setpoint": Calculated by the optimization, cannot be operated
- "L range optimization": Low limit for optimizing the setpoint
- "L range": Low limit for setpoint operation
- "Prefilter": ModPreCon functions (Page 682), Setpoint filter section

You can find additional information on this in the section Changing values (Page 253) .

(2) Displaying and changing the limit parameters for the manipulated variable

You can change the following parameters for the manipulated variable in this area:

- "H range": Upper limit of the manipulated variable for automatic mode
- "L range": Low limit of manipulated variable for automatic mode
- "Gradient limit": Maximum (absolute) change in the manipulated variable per sampling step

(3) Enabled operation

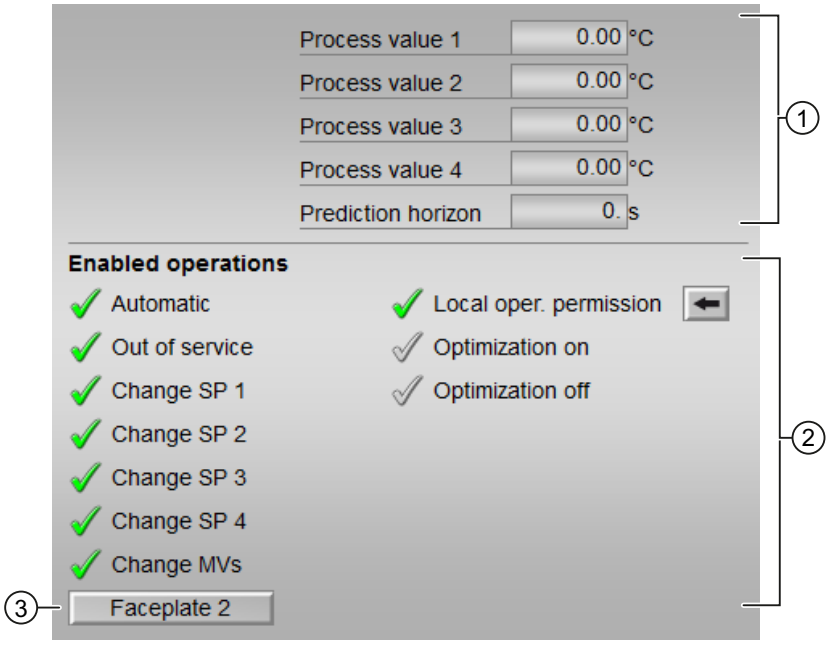
This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

5.6.8.5 ModPreCon preview

Preview for ModPreCon



(1) Process value

This area displays the real process values (`PVx`) and the prediction horizon.

Prediction horizon

The prediction horizon specifies how far the controller looks into the furutre of its calculation.

(2) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

The following enabled operations for parameters are shown here:

- "Automatic": You can switch to "automatic mode".
- "Out of service": You can switch to "Out of service" operating mode.
- " "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248) .
- "Change SP1": You can change the setpoint 1
- "Change SP2": You can change the setpoint 2
- "Change SP3": You can change the setpoint 3
- "Change SP4": You can change the setpoint 4
- "Change MVs": You can change the manipulated variables

Note

The OS operator must always be able to switch to "manual mode". That is why the switch to "manual mode" is not shown here in the faceplate.

(3) Navigation button for switching to the standard view of any faceplate

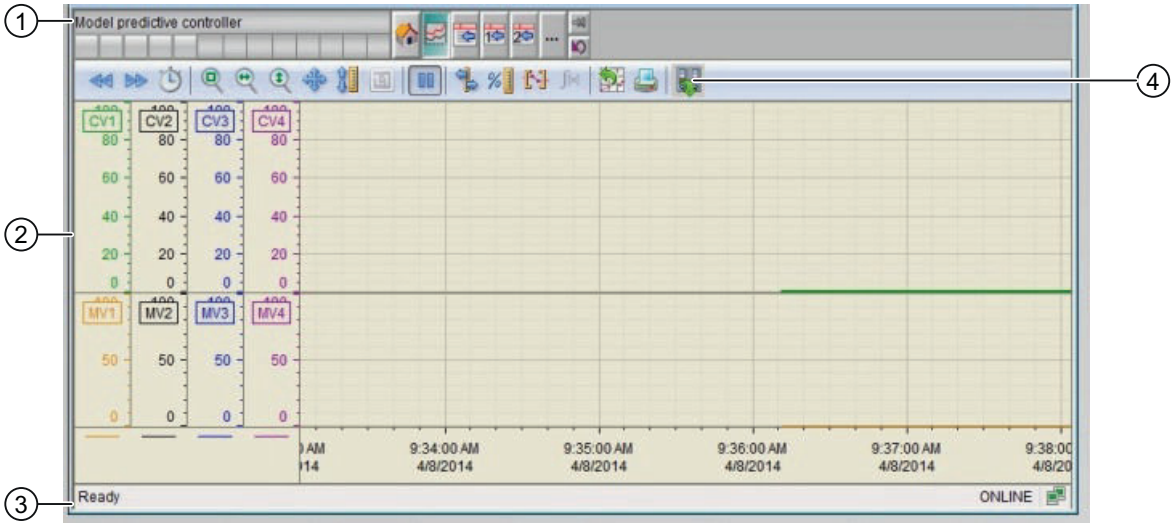
Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

5.6.8.6 ModPreCon trend view

ModPreCon trend view

There is a block-specific trend view for ModPreCon which is supplied as file @PG_APL_TrendMPC.pdl and which you can modify if necessary.



(1) Toolbar

(2) Display area for trends

(3) Status bar

(4) Button for switching between archive tags and online tags. The status bar shows if the trend view is working with online data or archive data.

The Export button is only visible and operable with the "Higher-level process control" operating permission.

For additional information about the trend view, refer to the *WinCC Information System* Online Help.

The trend view is divided into two screen halves.

The upper screen half shows all controlled variables with their associated setpoints. The setpoint is shown in the same color as the associated process value to allow the assignment to be identified straight away. Setpoints are dashed lines, process values are bold lines. If a controlled variable is exactly on the setpoint, it hides the setpoint.

The lower screen half shows all manipulated variables.

Both screen halves use the same color sequence for the individual channels. The sequence starts at channel 1 with green (standard color for the process value with the PID controller) and then goes through the color spectrum from top to bottom as far as gray and black. Each channel has its own y-axis in the corresponding color.

See also

ModPreCon views (Page 707)

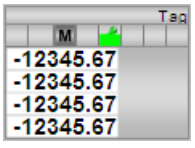
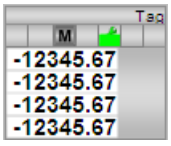
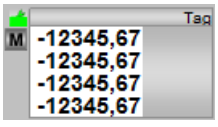
5.6.8.7 Block icon for ModPreCon

Block icons for ModPreCon

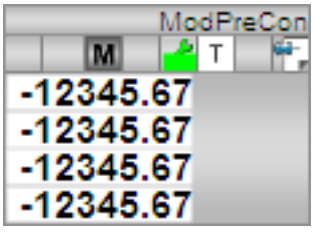
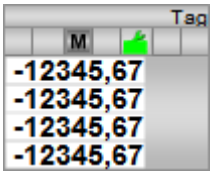
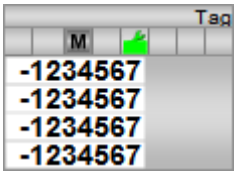
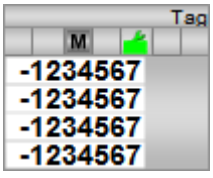
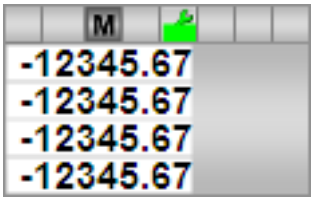
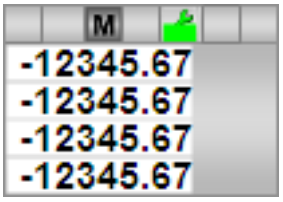
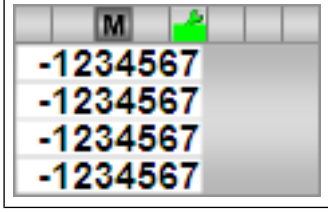
A variety of block icons are available with the following functions:

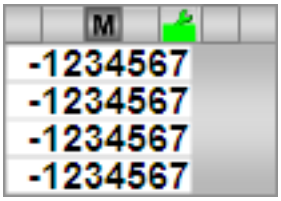
- Process tag type
- Operating modes
- Signal status, release for maintenance
- Tracking
- Memo display
- Process value (black, with and without decimal places)

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	5	
	6	
	7	

Icons	Selection of the block icon in CFC	Special features
	8	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

5.7 PIDConL - Continuous PID controller (Large)

5.7.1 Description of PIDConL

Object name (type + number) and family

Type and number: FB 1874

Family: Control

Area of application for PIDConL

The block is used for the following applications:

- Fixed setpoint control
- Cascade control
- Ratio control
- Split-range control
- Smith predictor closed-loop control
- Override control (override)

How it works

The block is a PID controller with continuous output signal (manipulated variable). It is used to activate a final controlling element with continuous action input.

The block functions following the PID algorithm with a delayed D component and an integrator with double precision.

The block is suitable for controlling sluggish control loops, for example, for temperatures and filling levels, and high-speed control loops, for example, for flow rates and speed. For a given CPU, a compromise has to be made between the number of controllers and the frequency with which the individual controllers have to be processed. The faster the modulated control loops are, i.e. the more frequently the manipulated variables have to be calculated per time unit, the lower the number of controllers that can be installed.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the control loop monitoring to work as planned in the trend view of the controller faceplates, the

```
S7_xarchive:='Value, shortterm;'
```

attributes in the process tag types for control loops at the controller function block must be set for the following tags:

- Input parameters:
  - `CPI_In`
- Output parameters
  - `MV`
  - `MV_HiAct`
  - `MV_LoAct`
  - `LoopClosed`
  - `SP`
  - `PV_Out`
  - `PV_ToleHi`
  - `PV_ToleLo`

For the PIDConL block, the Advanced Process Library contains templates for process tag types as examples and there is a example project (APL_Example_xx, xx designates the language variant) containing different application cases for this block. Several application cases are simulated in the example project and serve to explain how the block works.

Examples of process tag types:

- PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298)
- Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl) (Page 2309)
- Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312)
- Cascade control with control loop monitoring through ConPerMon (CascadeControl) (Page 2316)
- PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301)
- PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303)
- Override control (Page 2322)
- PID controller with Smith predictor (SmithPredictorControl) (Page 2306)
- PID controller for PA/FF devices (PIDControl_Lean_Fb) (Page 2297)

Application cases in example project:

- Process simulation including noise generator (ProcSimC; ProcSimS) (Page 2347)
- Cascade control of temperature by using the heat flow (CascadeSim) (Page 2349)
- Control loop monitoring for simulation with colored noise (ConPerMonSim) (Page 2351)
- Feedforward control to compensate a measurable disturbance variable (DisturbCompSim) (Page 2351)

- Operating point-oriented adaptation of parameters (gain scheduling) for non-linear processes (GainSchedSim) (Page 2352)
- Override control on a pipeline (OverrideSim) (Page 2353)
- Smith predictor for a dead time system (SmithPredictorSim) (Page 2353)
- Filtering of noisy measured values in a control loop (SigSmoothSim) (Page 2354)

Startup characteristics

Use the Feature bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

The following output parameters are written with the corresponding input parameters:

- `PV_HysOut`
- `PV_AH_Out`
- `PV_WH_Out`
- `PV_TH_Out`
- `PV_AL_Out`
- `PV_WL_Out`
- `PV_TL_Out`

Status word allocation for `Status1` parameter

You can find a description for each parameter in section PIDConL I/Os (Page 742).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutAct.Value`
6	Not used
7	`ManAct.Value`
8	`SP_ExtAct.Value`
9	`MV_ForOn.Value`
10	`MV_TrkOn.Value` AND NOT (`ManAct.Value` OR `OosAct.Value` OR `MV_ForOn.Value`)
11	`MV.Value` > `ManLoLim`
12	`SimLiOp.Value`
13	"Interlock" button is enabled OR `Intlock.ST` ≠ 16#FF
14	0 = Open padlock in the block icon 1 = Closed padlock in block icon

Status bit	Parameter
15 - 17	Not used
18	SimOn AND ManAct
19	AdvCoAct
20	1 = Input parameter Rbk is not interconnected (RbkOut.ST = 16#FF)
21	NegGain
22	1 = SP ramp active
23	OptimEn
24	OptimOcc
25	Not used
26	Display of BypassAct.Value in faceplate (display and operator controls) and block icon
27 - 28	Not used
29	BypProt
30	Dead band is temporarily disabled
31	Not used

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock
1	PV_AH_Act.Value
2	PV_WH_Act.Value
3	PV_TH_Act.Value
4	PV_TL_Act.Value
5	PV_WL_Act.Value
6	PV_AL_Act.Value
7	PV_AH_En
8	PV_WH_En
9	PV_TH_En
10	PV_TL_En
11	PV_WL_En
12	PV_AL_En
13	PV_AH_MsgEn
14	PV_WH_MsgEn
15	PV_TH_MsgEn
16	PV_TL_MsgEn
17	PV_WL_MsgEn
18	PV_AL_MsgEn
19	ER_AH_Act.Value
20	ER_AL_Act.Value
21	ER_AH_En
22	ER_AL_En

Status bit	Parameter
23	ER_AH_MsgEn
24	ER_AL_MsgEn
25	RbkWH_Act.Value
26	RbkWL_Act.Value
27	RbkWH_En
28	RbkWL_En
29	RbkWH_MsgEn
30	RbkWL_MsgEn
31	MS_RelOp

Status word allocation for Status3 parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via EventTsIn
1	Effective signal 2 of the message block connected via EventTsIn
2	Effective signal 3 of the message block connected via EventTsIn
3	Effective signal 4 of the message block connected via EventTsIn
4	Effective signal 5 of the message block connected via EventTsIn
5	Effective signal 6 of the message block connected via EventTsIn
6	Effective signal 7 of the message block connected via EventTsIn
7	Effective signal 8 of the message block connected via EventTsIn
8	Delay of the PV_AH_Lim message
9	Delay of the PV_WH_Lim message
10	Delay of the PV_TH_Lim message
11	Delay of the PV_TL_Lim message
12	Delay of the PV_WL_Lim message
13	Delay of the PV_AL_Lim message
14	Delay of the ER_AH_Lim message
15	Delay of the ER_AL_Lim message
16	Collection of message delays
17	BypassAct.Value
18 - 23	Not used
24	Hidden bypass signal in Intlock
25	Feature2 bit 2: Separate bypass signal
26	LockAct.Value
27	SP_UpRaAct, SP_DnRaAct limits enabled for gradient mode (SP_RateOn = 1)
28	GrpErr.Value
29	RdyToStart.Value
30	1 = Input parameter MV_ChnST is interconnected
31	Not used

Status word allocation for `Status4` parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via `EventTsIn`
8 - 31	Not used

See also

PIDConL functions (Page 727)

PIDConL messaging (Page 739)

PIDConL block diagram (Page 758)

PIDConL error handling (Page 738)

PIDConL modes (Page 726)

5.7.2 PIDConL modes

PIDConL operating modes

The block can be operated using the following modes:

- Automatic mode (Page 72)
- Manual mode (Page 72)
- Program mode for controllers (Page 78)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and bumpless switchover in the Manual and automatic mode for control blocks (Page 72) section.

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the Manual and automatic mode for control blocks (Page 72) section.

"Program mode for controllers"

General information on "Program mode for controllers" is available in the section Program mode for controllers (Page 78).

"Out of service"

You can find general information about the "Out of service" mode in the Out of service (Page 71) section.

See also

PIDConL block diagram (Page 758)

PIDConL I/Os (Page 742)

PIDConL messaging (Page 739)

PIDConL error handling (Page 738)

PIDConL functions (Page 727)

Description of PIDConL (Page 721)

5.7.3 PIDConL functions

Functions of PIDConL

The functions for this block are listed below.

Generation of manipulated variables

The manipulated variable `MV` can be generated as follows:

MV_For-On	Intlock	ManAct	MV_Trk On	AdvCoAct AND NOT AdvCoModSP	MV =	Limit monitoring	State
1	-	-	-	-	`MV_Forced`	none	Forced tracking through constraint without limitation
0	1	-	-	-	Neutral position: depends on `SafePos`, `SafePos2`	none	Interlock state
0	0	1	-	-	`Man`	`ManHiLim` `ManLoLim`	Manual mode, set by the operator
0	0	0	1	-	`MV_Trk`	`MV_HiLim` `MV_LoLim`	Tracking with limitation
0	0	0	0	1	`AdvCoMV`	`MV_HiLim` `MV_LoLim`	Higher-level program mode
0	0	0	0	0	`P_Part` + `I_Part` + `D_Part` + `FFwd`	`MV_HiLim` `MV_LoLim`	Automatic mode (PID algorithm)

If the controller is in "out of service" mode, the output parameter MV is set to the last valid value in manual mode or the neutral position manipulated variable depending on the Feature Bit (Neutral position manipulated variable takes effect at startup (Page 165)). Refer to the Out of service (Page 71) section for more on this.

Tracking and limiting a manipulated variable

The block provides the standard function Tracking and limiting a manipulated variable (Page 192).

Neutral position

The block provides the standard function Neutral position for motors, valves and controllers (Page 48).

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- CSF

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

"Actuator active" information

If the manipulated variable MV is greater than the minimum manual limit ManLoLim, this is recognized as actuator active. This status can be used to indicate a customized symbol in the process image, for example, and is saved in the status word (see Status word section in Description of PIDConL (Page 721)).

Limit monitoring of position feedback

The block provides the standard function Limit monitoring of the feedback (Page 94).

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

External/internal setpoint specification

The block provides the standard function Setpoint specification - internal/external (Page 127).

Setpoint limiting for external setpoints

The block provides the standard function Setpoint limiting for external setpoints (Page 192).

Gradient limit of the setpoint

The block provides the standard function Gradient limit of the setpoint (Page 124).

Using setpoint ramp

The block provides the standard function Using setpoint ramp (Page 123).

Tracking setpoint in manual mode

The block provides the standard function Tracking setpoint in manual mode (Page 192).

Simulating signals

The block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Process value (`SimPV, SimPV_Li`)
- Position feedback (`SimRbk, SimRbkLi`)

Bypass function

This block provides the standard function Bypassing signals (Page 107).

Limit monitoring of the process value

The block provides the standard function Limit monitoring of the process value (Page 86) with the alarm delay type Two time values per limit pair (Page 197). With the `Feature` Bit Separate delay times for each alarm (Page 169), the alarm delay type Two time values for each individual limit (Page 198) can be activated.

Providing PV limit at the output

For further connections to the other blocks, the following input parameters are also displayed with the corresponding output parameters:

- `PV_HysOut` := `PV_Hyst`
- `PV_AH_Out` := `PV_AH_Lim`
- `PV_WH_Out` := `PV_WH_Lim`
- `PV_TH_Out` := `PV_TH_Lim`
- `PV_TL_Out` := `PV_TL_Lim`

- `PV_WL_Out` := `PV_WL_Lim`
- `PV_AL_Out` := `PV_AL_Lim`

Error signal generation and dead band

The block provides the standard function Error signal generation and dead band (Page 188).

The `Feature` bit 14 can be used to feedforward an external error signal `ER_Ext`. When the external error signal is activated, `ER_Ext` affects both the dead band and the error signal alarm generation.

Delay alarm for control deviation at setpoint step changes

The block provides the standard function Delay alarm for control deviation at setpoint step changes (Page 186).

Limit monitoring of error signal

The block provides the standard function Limit monitoring of setpoint, manipulated variable and control deviation (Page 95). The monitoring of error signal works with the alarm delay type Two time values per limit pair (Page 197). With the `Feature` Bit Separate delay times for each alarm (Page 169), the alarm delay type Two time values for each individual limit (Page 198) can be activated.

Inverting control direction

The block provides the standard function Inverting control direction (Page 188).

Physical standardization of setpoint, manipulated variable and process value

Controller gain `Gain` is entered either using a physical variable or as standardized value.

`Gain` as a physical variable:

The standardized variables retain their default values:

- `NormPV.High` = 100 and `NormPV.Low` = 0
- `NormMV.High` = 100 and `NormMV.Low` = 0

The effective gain is:

`GainEff` = `Gain`

Entering a standardized `Gain` (dimensionless):

Change the standardized variables to the actual range of the process values and manipulated variables.

- Internal and external setpoints; the process value and corresponding parameters are entered according to the physical measuring range of the process value.
- The manual value, the tracking value of the manipulated variable, feedforward control and the corresponding parameters are set according to the physical measuring range of the manipulated variable.

The effective gain is:

`GainEff` = (`NormMV.High` - `NormMV.Low`) / (`NormPV.High` - `NormPV.Low`) · `Gain`

Selecting a unit of measure

The block provides the standard function Selecting a unit of measure (Page 207).

PID algorithm

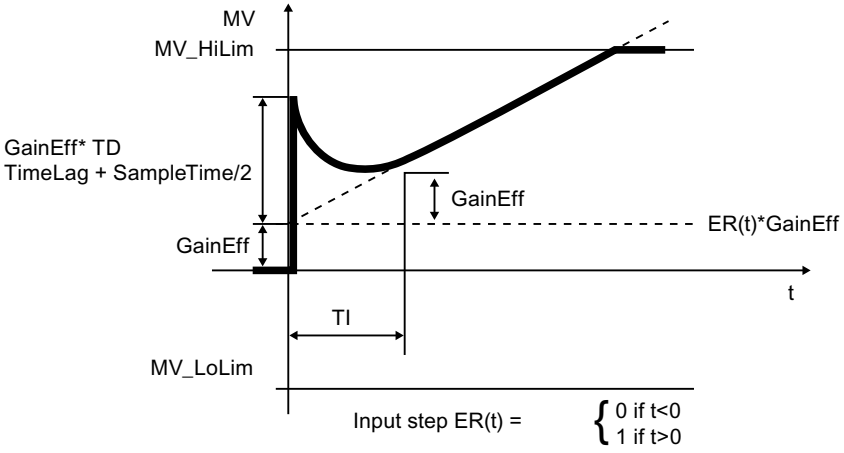
The manipulated variable is generated in automatic mode according to the following algorithm:

`MV` = `GainEff` · (1 + 1 / (`TI` · s) + (`TD` · s) / (1 + `TD` / `DiffGain` · s)) · `ER`

Where:

s = Complex number

The following step response occurs:



Note

The formula describes a standard application where P, I and D components are activated and the P and D components are not in the feedback circuit (`PropSel` = `1`, `TI` <> `0` and `IntSel` = `1`, `TD` <> `0` and `DIffSel` = `1`, `DiffToFbk` = `0` and `PropFacSP` = 1).

The D component delay is derived from `TD` / `DiffGain`.

- The P component is displayed at the `P_Part` I/O and can be deactivated using `PropSel` = 0 .
- The I component is displayed at the I_Part I/O and can be deactivated using `TI = 0` or IntSel = 0. In deactivated state, I_Part is specified by MV_Offset and added to the manipulated variable. Make a selection for this value so that the remaining control deviation equals zero at the control loop's typical operating point, at least. `IntSel` is used for temporary deactivation of the I component. The I component is not reactivated until `TI <> 0` and `IntSel = 1`. After the I component is activated, the integrator continues working starting from `MV_Offset`.
- The D component is displayed at the `D_Part` I/O and can be deactivated using `TD` = 0 or `DiffSel` = `0`.

Structure segmentation at controllers

The block provides the standard function Structure segmentation at controllers (Page 194).

Anti-windup

The controller has an anti-windup function. The I component is frozen after the manipulated variable has reached limits (`MV_HiLim` or `MV_LoLim`).

Feedforwarding and limiting disturbance variables

The block provides the standard function Feedforwarding and limiting disturbance variables (Page 193).

Control zone

The block provides the standard function Using control zones (Page 190).

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

- Signal status for the process value `PV_Out`:
  The signal status of the output parameter `PV_Out` always corresponds to the signal status of input parameter `PV` or, if the block is in simulation mode, 16#60.
- Signal status for the setpoint value `SP`:
  The signal status of the `SP` output parameter is always equivalent to the signal status of input parameter `SP_Ext` or `SP_Int`, depending on how the setpoint is specified. If the internal setpoint `SP_Int` is used, the signal status is always output as 16#80.

- Signal status of the error signal ER:
  The signal status of output parameter ER is obtained from the worst signal status of the two output parameters PV_Out and SP and is output.
  The signal status 16#60 (external simulation) is suppressed because the block acts as a sink with external simulation.
  If the external error signal is activated (Feature bit14 = 1), the signal status of ER_Ext.ST is applied.
- Signal status for the manipulated variable MV:
  The signal status of output parameter MV is obtained in "automatic mode" or in "program mode" with default setpoint from the worst signal status of the two parameters FFwd and ER and is output. In "manual mode", the signal status is output as good. The signal status 16#60 (external simulation) is suppressed because the block acts as a sink with external simulation. In "manual mode", the signal status is output as good.
- Signal status for position feedback RbkOut:
  The signal status of RbkOut always corresponds to the signal status of input parameter Rbk or, if the block is in simulation mode, 16#60.
- Worst signal status:
  The worst signal status ST_Worst for the block is formed from the following parameters:
  Feature bit 14 = 1 (external error signal)
  - ER_Ext.ST;
  - FFwd.ST;
  - RbkOut.ST;
  - MV_ChnST.ST;

  Feature bit 14 = 0 (external error signal)
  - SP.ST;
  - PV_Out.ST;
  - FFwd.ST;
  - RbkOut.ST;
  - MV_ChnST.ST;

Configurable reactions using the Feature parameter

You can find an overview of all reactions provided by the Feature parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
4	Setting switch or button mode (Page 166)
6	Ramp rate calculation (Page 178)
8	Separate delay times for each alarm (Page 169)
9	Substitution value is active if the block is in bypass (Page 184)

Bit	Function
11	Gradient limitation with time duration (Page 181)
12	Control zone with specified I component (Page 159)
13	Control zone with frozen I component (Page 159)
14	External control deviation (Page 150)
15	Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165)
16	Neutral position manipulated variable takes effect at startup (Page 165)
18	Disabling bumpless switchover to automatic mode for controllers (Page 172)
22	Update acknowledgment and error status of the message call (Page 159)
23	SP following PV in open loop has no priority over SP_Ext and SP limits (Page 178)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Dead band is temporarily disabled (Page 140)

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)

Operator permissions

The block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode" `AutModOp`
1	1 = Operator can switch to "manual mode" `ManModOp`
2	1 = Operator can switch to "Out of service" mode `OosOp`
3	1 = Operator can switch to "program mode" `AdvCoEn`
4	1 = Operator can switch the setpoint to "external" `SP_ExtOp`
5	1 = Operator can switch the setpoint to "internal" `SP_IntOp`
6	1 = Operator can change the internal setpoint `SP_Int`
7	1 = Operator can change the manual parameter `Man`
8	1 = Operator can change operation high limit of the setpoint `SP_InHiLim`
9	1 = Operator can change operation low limit of the setpoint `SP_InLoLim`
10	1 = Operator can change the operation high limit of the manipulated variable `ManHiLim`
11	1 = Operator can change the operation low limit of the manipulated variable `ManLoLim`
12	1 = Operator can enable the setpoint's gradient limitation function `SP_RateOn`

Bit	Function
13	1 = Operator can change the setpoint's high limit for the ramp `SP_UpRaLim`
14	1 = Operator can change the setpoint's low limit for the ramp `SP_DnRaLim`
15	1 = Operator can switch between the time value or the value for the ramp `SP_RmpModTime`
16	1 = Operator can change the ramp time `SP_RmpTime`
17	1 = Operator can change the target setpoint `SP_RmpTarget` for the setpoint ramp
18	1 = Operator can enable the setpoint ramp function `SP_RmpOn`
19	1 = Operator can permit the PID optimization function `OptimEn`
20	1 = Operator can enable the track setpoint in "manual mode" function `SP_TrkPV`
21	1 = Operator can enable the bumpless switchover from external to internal `SP_TrkExt`
22	1 = Operator can change the gain parameter `Gain`
23	1 = Operator can change the integral time parameter `TI`
24	1 = Operator can change the derivative time parameter `TD`
25	1 = Operator can change the derivative gain parameter `DiffGain`
26	1 = Operator can change the dead band parameter `DeadBand`
27	1 = Operator can change the control zone parameter `ConZone`
28	1 = Operator can change the derivative gain parameter `ER_AH_DFac`
29	1 = Operator can change the derivative gain parameter `ER_AL_DFac`
30	Not used
31	Not used

The block has the following permissions for the `OS1Perm` parameter:

Bit	Function
0	1 = Operator can change the limit (process value) `PV_AH_Lim` for the high alarm
1	1 = Operator can change the limit (process value) `PV_WH_Lim` for the high warning
2	1 = Operator can change the limit (process value) `PV_TH_Lim` for the high tolerance
3	1 = Operator can change the hysteresis (process value) `PV_Hyst`
4	1 = Operator can change the limit (process value) `PV_TL_Lim` for the low tolerance
5	1 = Operator can change the limit (process value) `PV_WL_Lim` for the low warning
6	1 = Operator can change the limit (process value) `PV_AL_Lim` for the low alarm
7	1 = Operator can change the limit (error signal) `ER_AH_Lim` for the high alarm
8	1 = Operator can change the hysteresis (error signal) `ER_Hyst`
9	1 = Operator can change the limit (error signal) `ER_AL_Lim` for the low alarm
10	1 = Operator can change the limit (position feedback) `RbkWH_Lim` for the high warning
11	1 = Operator can change the hysteresis (position feedback) `RbkHyst`
12	1 = Operator can change the limit (position feedback) `RbkWL_Lim` for the low warning
13	"Interlock" button is enabled
14	1 = Operator can activate bypass functionality
15	1 = Operator can deactivate bypass functionality
16	1 = Operator can activate the Simulation function `SimOn`
17	1 = Operator can activate the Release for maintenance function `MS_RelOp`
18	1 = Operator can activate / deactivate messages via `PV_AH_MsgEn`
19	1 = Operator can activate / deactivate messages via `PV_WH_MsgEn`

Bit	Function
20	1 = Operator can activate / deactivate messages via `PV_TH_MsgEn`
21	1 = Operator can activate / deactivate messages via `PV_TL_MsgEn`
22	1 = Operator can activate / deactivate messages via `PV_WL_MsgEn`
23	1 = Operator can activate / deactivate messages via `PV_AL_MsgEn`
24	1 = Operator can activate / deactivate messages via `ER_AH_MsgEn`
25	1 = Operator can activate / deactivate messages via `ER_AL_MsgEn`
26	1 = Operator can activate / deactivate messages via `RbkWH_MsgEn`
27	1 = Operator can activate / deactivate messages via `RbkWL_MsgEn`
28	1 = Operator can change the simulation value `SimPV`
29	1 = Operator can change the simulation value `SimRbk`
30	1 = Operator can activate the derivative action to the feedback path `DiffToFbk`
31	1 = Operator can change the proportional action to the feedback path `PropFacSP`

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Interlocks

This block provides the following interlocks:

- Interlock without reset ("Interlock")

If interlock is active without forcing (`MV_ForOn` =0) the manipulated value `MV.Value` is set to the neutral position value (Neutral position for motors, valves and controllers (Page 48)).

You can find additional information on this in the section Interlocks (Page 99).

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Release for maintenance

The block provides the standard function Release for maintenance (Page 64)

Generating instance-specific messages

The block provides the standard function Generating instance-specific messages (Page 200) without the time stamp function in the I/O.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205).

Instance-specific text can be configured for the following parameters:

- AutModOp
- ManModOp
- AdvCoOn
- OosOp
- SP_ExtOp
- SP_IntOp

Specifying the display area for process and setpoint values as well as operations

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

See also

PIDConL I/Os (Page 742)

PIDConL block diagram (Page 758)

PIDConL error handling (Page 738)

PIDConL modes (Page 726)

EventTs functions (Page 1634)

5.7.4 PIDConL error handling

Error handling of PIDConL

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` output parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
30	The value of `PV` can no longer be displayed in the REAL number field.
31	The value of `SP_Ext` can no longer be displayed in the REAL number field.
32	The value of `FFwd` can no longer be displayed in the REAL number field.
33	The value of `MV_Trk` can no longer be displayed in the REAL number field.
34	The value of `MV_Forced` can no longer be displayed in the REAL number field.
35	The value of `Rbk` can no longer be displayed in the REAL number field.
36	The value of `MV` can no longer be displayed in the REAL number field.
51	`AutModLi` = 1 and `ManModLi` = 1 `SP_LiOp` = 1 and `SP_IntLi` = 1 and `SP_ExtLi` = 1
59	= 1, "Gain is negative"
60	|`TI`| < `SampleTime` / 2
61	|`TD`| < `SampleTime`
62	`DiffGain` < 1 or `DiffGain` > 10
63	`TD` / `DiffGain` < `SampleTime` / 2
64	`PropFacSP` < 0 or `PropFacSP` > 1
66	`NormPV_High` = `NormPV_Low`
74	`ConZone = < 0.0`

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

PIDConL block diagram (Page 758)

PIDConL I/Os (Page 742)

PIDConL messaging (Page 739)
PIDConL functions (Page 727)
PIDConL modes (Page 726)
Description of PIDConL (Page 721)
Setting switch or button mode (Page 166)

5.7.5 PIDConL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
`MsgEvId2`	SIG 6	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a process control fault is triggered (`MsgEvId2`, SIG 6).

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	Alarm - high	$$BlockComment$$ PV - high alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ PV - high warning limit violated
	SIG 3	Tolerance - high	$$BlockComment$$ PV - high tolerance limit violated
	SIG 4	Tolerance - low	$$BlockComment$$ PV - low tolerance limit violated
	SIG 5	Warning - low	$$BlockComment$$ PV - low warning limit violated
	SIG 6	Alarm - low	$$BlockComment$$ PV - low alarm limit violated
	SIG 7	Alarm - high	$$BlockComment$$ ER - high alarm limit violated
	SIG 8	Alarm - low	$$BlockComment$$ ER - low alarm limit violated
MsgEvId2	SIG 7	Warning - high	$$BlockComment$$ Rbk - high warning limit violated
	SIG 8	Warning - low	$$BlockComment$$ Rbk - low warning limit violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You can use up to four instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 1	AS process control message - fault	$$BlockComment$$ External message 1 Status 16#@5%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ External message 2 Status 16#@6%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ External message 3 Status 16#@7%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 4 Status 16#@8%x@

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	Process value `PV_Out`
5	Control deviation `ER`
6	`ExtVa106`
7	`ExtVa107`
8	Not allocated
9	Not allocated
10	Not allocated

The associated values 6 ... 7 are allocated to the parameters `ExtVa106` ... `ExtVa107` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

Associated values for message instance `MsgEvId2`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	Position feedback `Rbk`
5	Signal status `ExtMsg1`
6	Signal status `ExtMsg2`
7	Signal status `ExtMsg3`
8	Signal status `ExtMsg4`
9	`ExtVa209`
10	`ExtVa210`

The associated values 9 ... 10 are allocated to the parameters `ExtVa209` ... `ExtVa210` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of PIDConL (Page 721)

PIDConL functions (Page 727)

PIDConL I/Os (Page 742)

PIDConL block diagram (Page 758)

PIDConL error handling (Page 738)

PIDConL modes (Page 726)

5.7.6 PIDConL I/Os

I/Os of PIDConL

Input parameters

Parameter	Description	Type	Default
AdvCoEn	1 = Enable "program mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoOn*	1 = Enable "program mode" via faceplate	BOOL	0
AdvCoModSP	Type of "program mode": 1 = Setpoint specification 0 = Manipulated variable specification	BOOL	1
AdvCoMstrOn	Activate (0-1) or deactivate (1-0) "program mode" via edge transition	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoMV	Specified value from the external program	REAL	0.0
AutModLi*	1 = "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 0)	BOOL	0
BatchEn	1 = Enable allocation for batch control	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Interlock bypass during simulation	BOOL	0
BypLiOp	1 = Bypass commands via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypLock	1 = Bypass activation or deactivation is locked for operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypPV	Substitution value if block is in bypass	REAL	0.0
BypPVLi	1 = Select bypass PV (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
BypPVOp	1 = Select bypass PV (via operator)	BOOL	0
ConZone	Width of control zone	REAL	0.0
CPI_In	Input for control performance index, which is calculated by the assigned ConPerMon block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DeadBand	Width of dead band	REAL	0.0
DiffGain	Gain of differentiator [1..10] DiffGain = TD / (delay time of D component)	STRUCT • Value: REAL • ST: BYTE	- • 5.0 • 16#80
DiffSel	1 = D component activated	BOOL	1
DiffToFbk	1 = D component is placed in the feedback	BOOL	0
EN	1 = Called block will be processed	BOOL	1
ER_A_DC*	Delay time for incoming ER high/low or only low alarms [s]	REAL	0.0
ER_AH_DC*	Delay time for incoming ER high alarms [s]	REAL	0.0
ER_A_DG*	Delay time for outgoing ER high/low or only low alarms [s]	REAL	0.0
ER_AH_DG*	Delay time for outgoing ER high alarms [s]	REAL	0.0
ER_AH_En	1 = Activate alarm (high) for error signal monitoring	BOOL	1
ER_AH_DFac*	Delay factor at positive setpoint step changes for incoming alarms at the error signal monitoring ER_AH_Lim	REAL	0.0
ER_AH_Lim	Alarm limit (high) for error signal monitoring	REAL	100.0
ER_AH_MsgEn	1 = Activate messages for alarm (high) for error signal monitoring	BOOL	1
ER_AL_DFac*	Delay factor at negative setpoint step changes for incoming alarms at the error signal monitoring ER_AL_Lim	REAL	0.0
ER_AL_En	1 = Activate alarm (low) for error signal monitoring	BOOL	1
ER_AL_Lim	Alarm limit (low) for error signal monitoring	REAL	-100.0
ER_AL_MsgEn	1 = Activate messages for alarm (low) for error signal monitoring	BOOL	1
ER_Ext	External error signal	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ER_Hyst	Alarm hysteresis for error signal	REAL	1.0

Parameter	Description	Type	Default
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	1 = Binary input for freely selectable message 1 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	1 = Binary input for freely selectable message 2 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	1 = Binary input for freely selectable message 3 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg4	1 = Binary input for freely selectable message 4 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa209	Associated value 9 for messages (MsgEvID2)	ANY	
ExtVa210	Associated value 10 for messages (MsgEvID2)	ANY	
Feature	I/O for additional functions (Page 727)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Feature2	I/O for additional functions (Page 727)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
FFwd*	Input for additive disturbance variable activation	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
FFwdHiLim	Limit (high) for additive disturbance variable activation	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
FFwdLoLim	Limit (low) for additive disturbance variable activation	STRUCT • Value: REAL • ST: BYTE	- • -100.0 • 16#80
Gain	Proportional gain Gain.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#FF
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is in effect	BOOL	1
IntHoldNeg	1 = Integrator cannot run in negative direction	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
IntHoldPos	1 = Integrator cannot run in positive direction	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Intlock	0 = Interlock without reset is in effect Once the interlock condition has cleared, the block can be operated without reset 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
IntSel	1 = I component activated	BOOL	1
Man*	Manual specification for the manipulated variable	REAL	0.0
ManHiLim	Limit (high) for manual parameter Man	REAL	100.0
ManLoLim	Limit (low) for manual parameter Man	REAL	0.0
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
ModLiOp	Operating mode switchover between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MsgEvID1	Message number (assigned automatically)	DWORD	16#00000000
MsgEvID2	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Hotspot-Text (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MV_ChnST	Signal status of output channel for MV Should be connected to an output channel block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
MV_Forced*	Forced manipulated variable that is not limited and assumes top priority	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_ForOn	1 = Output forced manipulated variable MV_Forced unlimited at output MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_HiLim	Limit (high) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
MV_LoLim	Limit (low) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_Mean	Mean value of the MV in the time window	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_Offset	Manipulated variable for ER=0, operating point for controller with deactivated I component	REAL	0.0
MV_OpScale	OS display range for manipulated variable MV	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
MV_SafePos	Manipulated variable neutral position	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_Trk*	Tracking value for the manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_TrkOn	1 = Tracking of manipulated variable MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_Unit	Unit of measure for manipulated variable	INT	1342
NegGain	0 = Effective proportional gain GainEff is positive 1 = Effective proportional gain GainEff is negative	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NormMV	Manipulated variable range (MV) for standardizing the proportional gain (GAIN)	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

Parameter	Description	Type	Default
NormPV	Process value range (PV) for standardizing the proportional gain (GAIN)	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
Occupied	1 = Occupied by batch control	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OptimEn*	1 = Enable optimization of PID parameters by PID tuner	BOOL	0
OptimOcc*	1 = Optimization running	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 727)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OS1Perm	I/O for operator permissions (Page 727)	STRUCT • Bit 0: BOOL • Bit 18: BOOL • Bit 19: BOOL • Bit 31: BOOL	- • 1 • 1 • 1 • 1
PropFacSP	Applying the P component to the feedback [0..1]. 0 = P component fully in feedback	REAL	1.0
PropSel	1 = Activate P component	BOOL	1
PV*	Process value (controlled variable)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_A_DC*	Delay time for incoming PV high/low or only low alarms [s]	REAL	0.0
PV_AH_DC*	Delay time for incoming PV high alarms [s]	REAL	0.0
PV_A_DG*	Delay time for outgoing PV high/low or only low alarms [s]	REAL	0.0
PV_AH_DG*	Delay time for outgoing PV high alarms [s]	REAL	0.0
PV_AH_En	1 = Enable PV alarm limit (high)	BOOL	1
PV_AH_Lim	Limit PV alarm (high)	REAL	95.0
PV_AH_MsgEn	1 = Enable PV alarm (high) message	BOOL	1
PV_AL_En	1 = Enable PV alarm limit (low)	BOOL	1
PV_AL_Lim	PV alarm limit (low)	REAL	5.0
PV_AL_MsgEn	1 = Enable PV alarm (low) message	BOOL	1

Parameter	Description	Type	Default
PV_Hyst	Hysteresis for PV alarm, warning and tolerance limits	REAL	1.0
PV_OpScale	Limit for scale in PV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
PV_T_DC*	Delay time for incoming PV high/low or only low tolerance messages [s]	REAL	0.0
PV_TH_DC*	Delay time for incoming PV high tolerance messages [s]	REAL	0.0
PV_T_DG*	Delay time for outgoing PV high/low or only low tolerance messages [s]	REAL	0.0
PV_TH_DG*	Delay time for outgoing PV high tolerance messages [s]	REAL	0.0
PV_TH_En	1 = Enable PV tolerance limit (high)	BOOL	0
PV_TH_Lim	Limit PV tolerance message (high)	REAL	85.0
PV_TH_MsgEn	1 = Enable message for PV tolerance message (high)	BOOL	1
PV_TL_En	1 = Enable PV tolerance limit (low)	BOOL	0
PV_TL_Lim	Limit PV tolerance message (low)	REAL	15.0
PV_TL_MsgEn	1 = Enable message for tolerance message (low)	BOOL	1
PV_Unit	Unit of measure for process value	INT	1001
PV_W_DC*	Delay time for incoming PV high/low or only low warnings [s]	REAL	0.0
PV_WH_DC*	Delay time for incoming PV high warnings [s]	REAL	0.0
PV_W_DG*	Delay time for outgoing PV high/low or only low warnings [s]	REAL	0.0
PV_WH_DG*	Delay time for outgoing PV high warnings [s]	REAL	0.0
PV_WH_En	1 = Enable PV warning limit (high)	BOOL	1
PV_WH_Lim	Limit PV warning (high)	REAL	90.0
PV_WH_MsgEn	1 = Enable PV warning (high) message	BOOL	1
PV_WL_En	1 = Enable PV warning limit (low)	BOOL	1
PV_WL_Lim	Limit PV warning (low)	REAL	10.0
PV_WL_MsgEn	1 = Enable PV warning (low) message	BOOL	1
Rbk*	Position feedback for display on OS	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
RbkHyst	Alarm hysteresis for position feedback	REAL	1.0
RbkWH_En	1 = Enable warning (high) for position feedback	BOOL	1
RbkWH_Lim	Limit for position feedback of warning (high)	REAL	100.0
RbkWH_MsgEn	1 = Enable messages for warning (high) for position feedback	BOOL	1

Parameter	Description	Type	Default
RbkWL_En	1 = Enable warning (low) for position feedback	BOOL	1
RbkWL_Lim	Limit for position feedback of warning (low)	REAL	0.0
RbkWL_MsgEn	1 = Enable messages for warning (low) for position feedback	BOOL	1
RefStdDevIn	Reference value of PV standard deviation (sigma) in defined "good" state of control loop	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
RstBypLi	1 = Reset bypass PV (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstBypOp	1 = Reset bypass PV (via operator)	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SafePos	1 = Neutral position (Page 48) for controller manipulated variable is ManHiLim 0 = Neutral position for controller manipulated variable is ManLoLim	BOOL	0
SafePos2	Neutral position for controller manipulated variable: 0 = SafePos is valid 1 = Neutral position is MV_SafePos 2 = Neutral position is last manipulated variable (stop)	INT	0
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
SettliTime	Settling time [s] of the control loop determined by the ConPerMon block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#78
SettliFactor	Factor to increase the settling time to adjust the dead band	REAL	2.0
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Vale: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimPV*	Process value used for SimOn = 1	REAL	0.0

Parameter	Description	Type	Default
SimPV_Li	Process value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimRbk*	Position feedback used for SimOn = 1	REAL	0.0
SimRbkLi	Position feedback used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_DnRaLim	Limit (low) for the gradient of the setpoint [SP_Unit/s]	REAL	100.0
SP_ExHiLim	Limit (high) for external setpoint	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP_ExLoLim	Limit (low) for external setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_Ext*	external setpoint - (to interconnection)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtLi*	1 = Select external setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOp*	1 = Select external setpoint (via operator)	BOOL	0
SP_InHiLim	Limit (high) of internal setpoint	REAL	100.0
SP_InLoLim	Limit (low) of internal setpoint	REAL	0.0
SP_Int*	Internal setpoint for operation	REAL	0.0
SP_IntLi*	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_IntOp*	1 = Select internal setpoint (via operator)	BOOL	0
SP_LiOp	Select setpoint source (internal/external): 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_RateOn*	1 = Activate limitation of setpoint gradients	BOOL	0
SP_RmpModTime	1 = Use time (SP_RmpTime) for setpoint ramp 0 = Use gradient	BOOL	0
SP_RmpOn*	1 = Activate setpoint ramp to target setpoint SP_RmpTarget	BOOL	0
SP_RmpTarget*	Target setpoint for setpoint ramp	REAL	0.0
SP_RmpTime*	Time for setpoint ramp [s] from current SP up to SP_RmpTarget	REAL	0.0
SP_TrkExt	1 = Bumpless switchover from external to internal setpoint active	BOOL	1

Parameter	Description	Type	Default
SP_TrkPV	1 = Setpoint follows PV in "manual mode" and with tracking	BOOL	0
SP_UpRaLim	Gradient limit (high) for the setpoint [SP_Unit/s]	REAL	100.0
StepNo	Batch step number	DWORD	16#00000000
TD	Derivative time [s] TD.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
TI	Integral time [s] TI.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#FF
TimeFactor	Time unit: 0 = Seconds 1 = Minutes 2 = Hours	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AdvCoAct	1 = "Program mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoRdy	1 = "Program mode" available	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutAct	1 = "Automatic mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypassAct	1 = Bypass is activated in this block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CascaCut	Cascade connection: 1 = Control chain from primary to secondary controller is interrupted	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DynDeadBand	Dynamic dead band	REAL	0.0
D_Part	D component of PID algorithm	REAL	0.0
ENO	1 = Block algorithm completed without errors	BOOL	0

Parameter	Description	Type	Default
ER	Error signal	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ER_A_DCOut	Effective delay time [s] for incoming alarms at the error signal monitoring	REAL	0.0
ER_AH_Act	1 = Alarm limit (high) for control deviation violated. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ER_AL_Act	1 = Alarm limit (low) for control deviation violated. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see PIDConL error handling (Page 738)	INT	-1
FFwdHiAct	1 = Limit (high) for additive disturbance variable activation violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FFwdLoAct	1 = Limit (low) for additive disturbance variable activation violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GainEff	Effective proportional gain depends on NegGain, Gain, NormPV, and NormMV	REAL	1.0
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
I_Part	I component of PID algorithm	REAL	0.0
LockAct	1 = Interlock is in effect	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LoopClosed	1 = Control loop closed 0 = Control loop open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80

Parameter	Description	Type	Default
ManHiOut	Limit (high) for "manual mode", corresponds to input parameter ManHiLim	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
ManLoOut	Limit (low) for "manual mode", corresponds to input parameter ManLoLim	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgAckn2	Message acknowledgement status 2 (output ACK_STATE of second ALARM_8P)	WORD	16#0000
MsgErr1	1 = Alarm error 1 (output ERROR of the first ALARM_8P)	BOOL	0
MsgErr2	1 = Alarm error 2 (output ERROR of the second ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
MsgStat2	Message status 2 (output STATUS of second ALARM_8P)	WORD	16#0000
MV	Manipulated variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_HiAct	1 = Limit (high) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_LoAct	1 = Limit (low) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_UnitOut	Unit of measure for manipulated variable, for interconnecting to the MV_Unit input parameter of the ConPerMon block	INT	0
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF

Parameter	Description	Type	Default
OS1PermOut	Display of OS1Perm	DWORD	16#FFFFFFFF
OS1PermLog	Display of OS1Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
PhaseDeadBand	Phase for the dynamic adjustment of the dead band 0: Dead band enabled 1: Dead band disabled 2: Settling time	INT	0
P_Part	P component of PID algorithm	REAL	0.0
PV_AH_Act	1 = PV alarm (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AH_Out	PV - High alarm limit output	REAL	0.0
PV_AL_Act	1 = PV alarm (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AL_Out	PV - Low alarm limit output	REAL	0.0
PV_HysOut	PV - Alarm hysteresis output	REAL	0.0
PV_Out	Output for process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_TH_Act	1 = PV tolerance message (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_TH_Out	PV - High tolerance limit output	REAL	0.0
PV_TL_Act	1 = PV tolerance message (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_TL_Out	PV - Low tolerance limit output	REAL	0.0
PV_ToleHi	Limit (high) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
PV_ToleLo	Limit (low) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_WH_Act	1 = PV warning (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_WH_Out	PV - High warning limit output	REAL	0.0
PV_WL_Act	1 = PV warning (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_WL_Out	PV - Low warning limit output	REAL	0.0
PV_UnitOut	Unit of measure for process value, for interconnecting to the PV_Unit input parameter of the ConPerMon block	INT	0
RbkOut	Output for position feedback	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RbkWH_Act	1 = Warning (high) for position feedback active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkWL_Act	1 = Warning (low) for position feedback active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SettlingTimer	Settling time for a closed control loop	REAL	0.0
SP	Setpoint used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
SP_DnRaAct	1 = Negative gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExHiAct	1 = Limit (high) for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExLoAct	1 = Limit (low) for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtAct	1 = External setpoint is active 0 = Internal setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOut	External setpoint, corresponds to input parameter SP_Ext	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_InHiOut	Limit (high) for SP_Int corresponds to input parameter SP_InHiLim	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP_InLoOut	Limit (low) for SP_Int corresponds to input parameter SP_InLoLim	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_RateTarget	Target setpoint for the gradient limitation	REAL	0.0
SP_UpRaAct	1 = Positive gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_RemRT	Remaining ramp time of the setpoint	REAL	0.0
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 721)	DWORD	16#00000000
Status2	Status word 2 (Page 721)	DWORD	16#00000000
Status3	Status word 2 (Page 721)	DWORD	16#00000000
SumMsgAct	1 = Active hardware interrupt	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Calculation of the output parameter ER_A_DCOut

ER_A_DC is assigned by default to the output before a setpoint change.

ER_A_DCOut = ER_A_DC

In the case of a setpoint change in the positive direction during automatic mode, the output is calculated as follows:

ER_A_DCOut = Maximum (ER_A_DC, ER_AH_DFac * Setpoint difference)

In the case of a setpoint change in the negative direction during automatic mode, the output is calculated as follows:

ER_A_DCOut = Maximum (ER_A_DC, -1*ER_AH_DFac * Setpoint difference)

When the control circuit has stabilized again, meaning

(ER_AL_Lim + ER_Hyst) <= ER <= (ER_AH_Lim – ER_Hyst)

and the delay time for outgoing alarms ER_A_DG has expired, the output is reset again to ER_A_DC: ER_A_DCOut = ER_A_DC

Activating and deactivating the function:

The function is deactivated (default) when the following applies: ER_AH_DFac = 0.0 and ER_AL_DFac = 0.0

See also

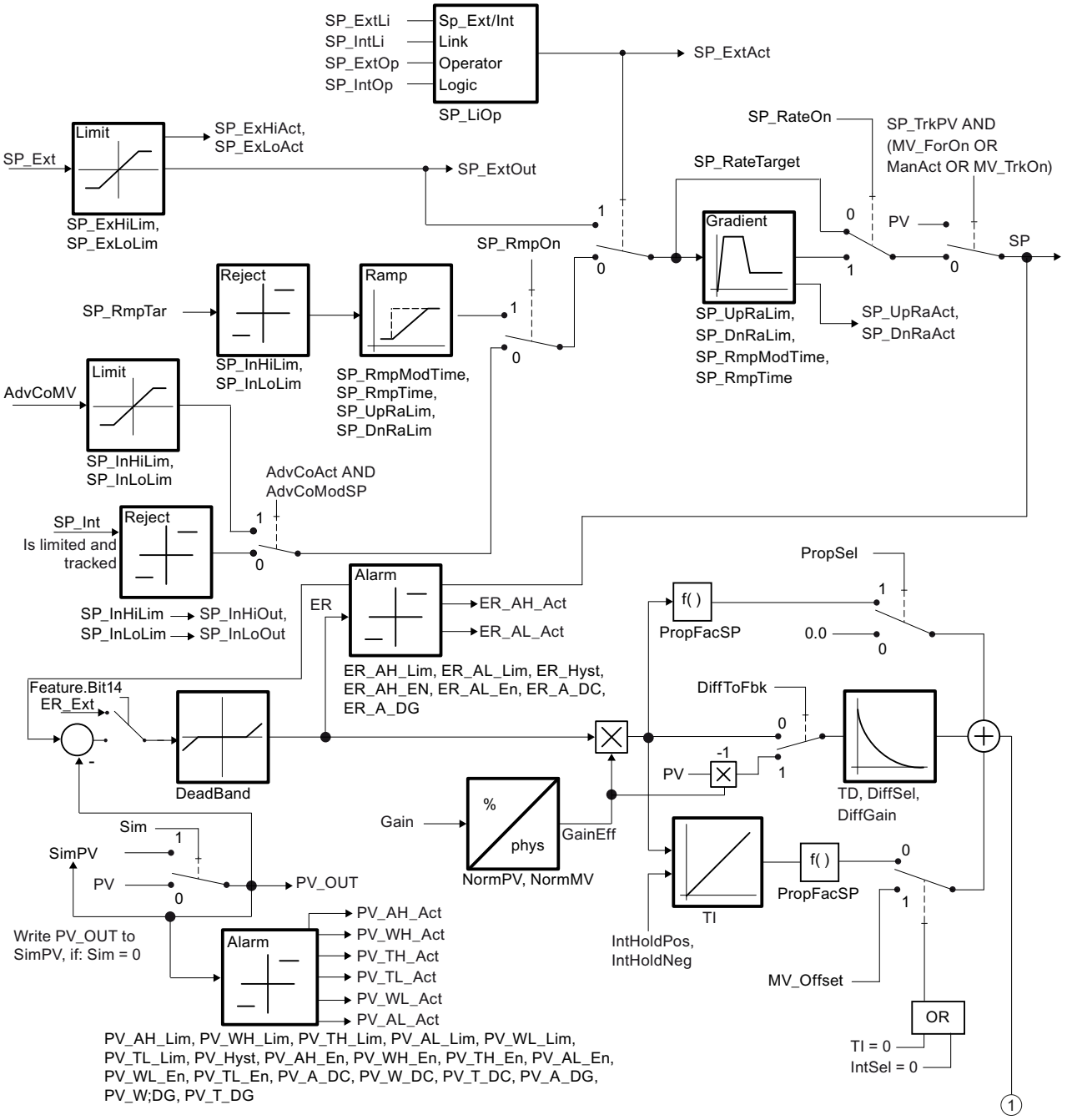
PIDConL messaging (Page 739)

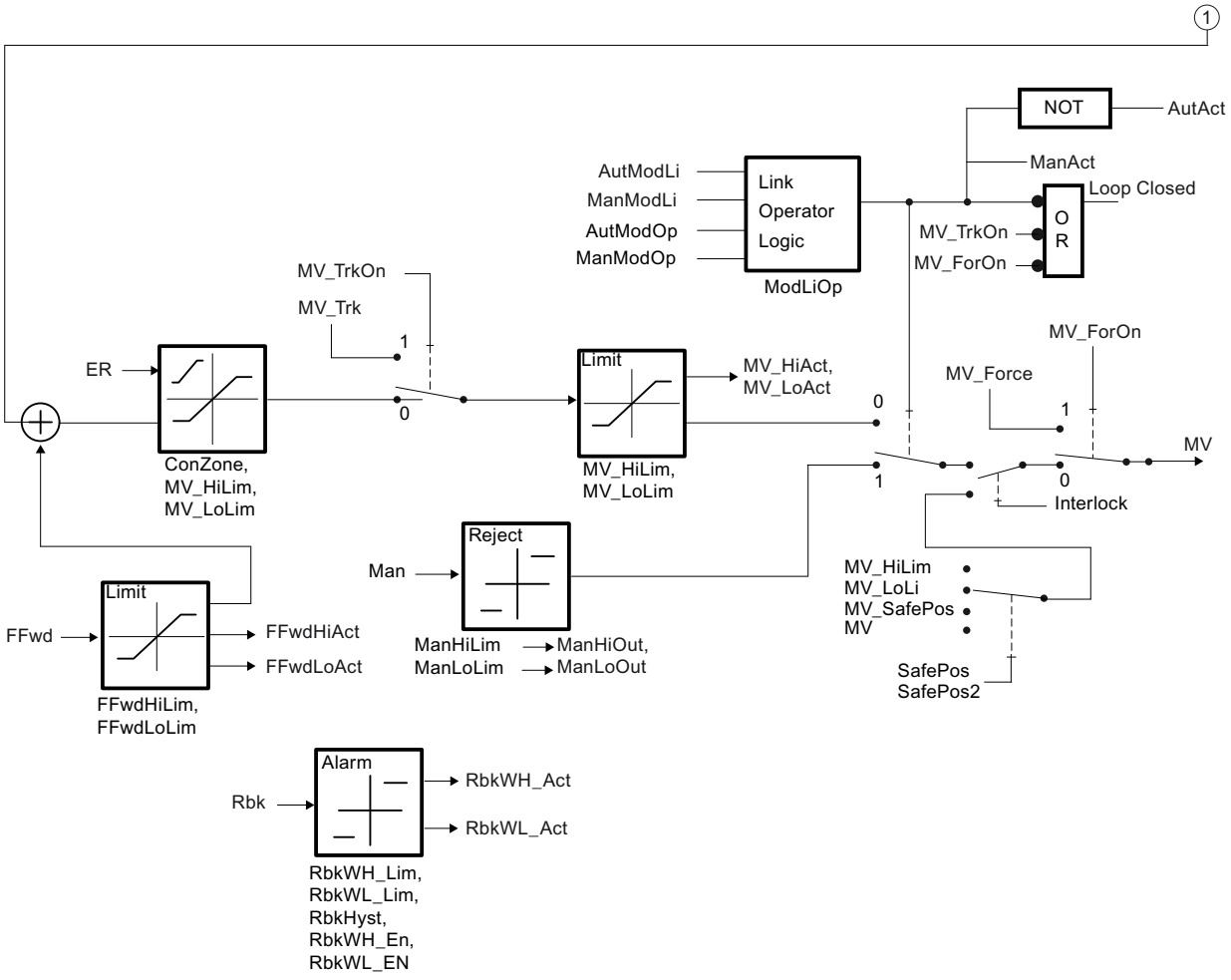
PIDConL block diagram (Page 758)

PIDConL modes (Page 726)

5.7.7 PIDConL block diagram

PIDConL block diagram





See also

PIDConL I/Os (Page 742)

PIDConL messaging (Page 739)

PIDConL error handling (Page 738)

PIDConL functions (Page 727)

PIDConL modes (Page 726)

Description of PIDConL (Page 721)

5.7.8 Operator control and monitoring

5.7.8.1 PIDConL views

Views of the PIDConL block

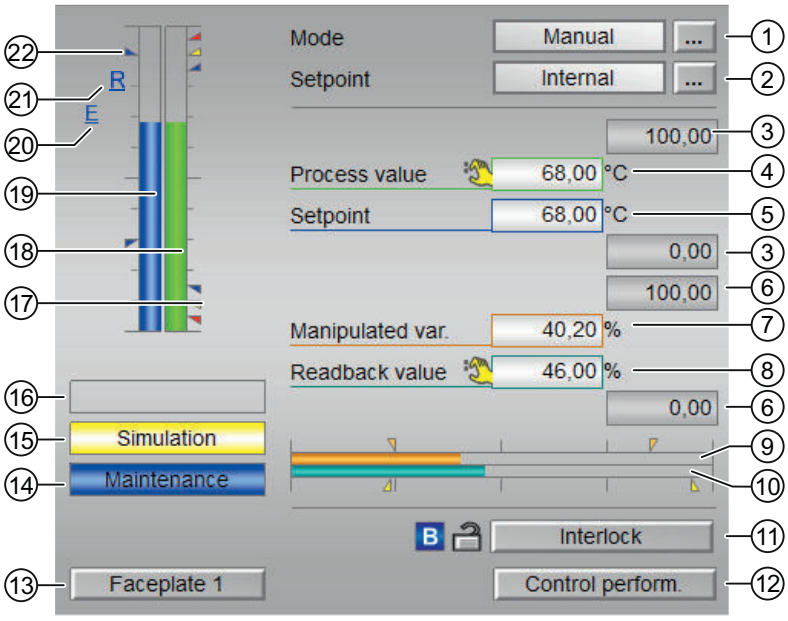
The block PIDConL provides the following views:

- PIDConL, PIDConS and PIDConR standard views (Page 761)
- Alarm view (Page 296)
- Limit view of PID controllers (Page 285)
- Trend view (Page 299)
- Ramp view (Page 294)
- Parameter view of PID controllers (Page 275)
- PIDConL, PIDConS and PIDConR previews (Page 766)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icons for PID and FM controller (Page 235)

Refer to the Structure of the faceplate (Page 242) and Block icon structure (Page 226) sections for general information about the faceplate and block icon.

5.7.8.2 PIDConL, PIDConS and PIDConR standard views

PIDConL standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 72)
- Automatic mode (Page 72)
- Program mode for controllers (Page 78) (not with PIDConS)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Displaying and switching the setpoint

This area shows how to specify the setpoint. The setpoint can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the setpoint specification.

You can find additional information on this in the section Setpoint specification - internal/ external (Page 127).

Note

With the PIDConR block, this area is only visible if you have set the `Feature` Bit Switching operator controls for external setpoint to visible (Page 143) to 1.

(3) High and low scale range for the process value

These values provide information on the display range for the bar graph of the process value. The scale range is defined in the engineering system.

(4) Display of the process value including signal status

This area shows the current process value with the corresponding signal status.

(5) Displaying and changing the setpoint including signal status

This area shows the current setpoint with the corresponding signal status.

Refer to the Changing values (Page 253) section for information on changing the setpoint. The setpoint specification also needs to be set to "Internal" for this block.

(6) High and low scale range for the setpoint

This area is already set and cannot be changed.

(7) Displaying and changing the manipulated variable including signal status

This area shows the current manipulated variable with the corresponding signal status.

Refer to the Changing values (Page 253) section for information on changing the manipulated variable. You can only make a change in manual mode.

(8) Display of the position feedback including signal status

This display is only visible when the corresponding block input is connected.

This area shows the current feedback of the manipulated variable with the corresponding signal status.

(9) Bar graph for the manipulated variable

This area shows the current manipulated variable in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(10) Bar graph for position feedback

This display is only visible when the corresponding block input is connected.

This area shows the current position feedback in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(10)Bar graph for position feedback

This display is only visible when the corresponding block input is connected.

This area shows the current position feedback in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(11)Operator control and display area for interlock functions of the block (not with PIDConS)

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the sectionInterlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (seeForming the group status for interlock information (Page 104)), e.g.:

- Signal status (seeForming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (seeForming the group status for interlock information (Page 104)):

(12)Navigation button for the standard view of theConPerMonblock

Use this navigation button to open the standard view of theConPerMonblock. The visibility of this navigation button depends on the configuration in the engineering system (ES).

See also the sectionOpening additional faceplates (Page 203)for more on this.

(13)Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

See also the sectionOpening additional faceplates (Page 203)for more on this.

(14)Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in sectionRelease for maintenance (Page 64)Display area for block states.

(15) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in the section Simulating signals (Page 58).

(16) Display area for block states

This area provides additional information on the operating state of the block (from high to low according to priority):

- "Optimizing"
- "Tracking"
- "Forced tracking"
- "Load SP"
- "SP ramp active"

(17) Limit display

These colored triangles indicate the configured limits in the respective bar graph:

- Red: Alarm
- Yellow: Warning
- Blue: Tolerance

(18) Bar graph for the process value

This area shows the current process value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(19) Bar graph for the setpoint

This area shows the current setpoint in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(20) Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

(21) Display for the target setpoint of the setpoint ramp

This display [R] shows the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

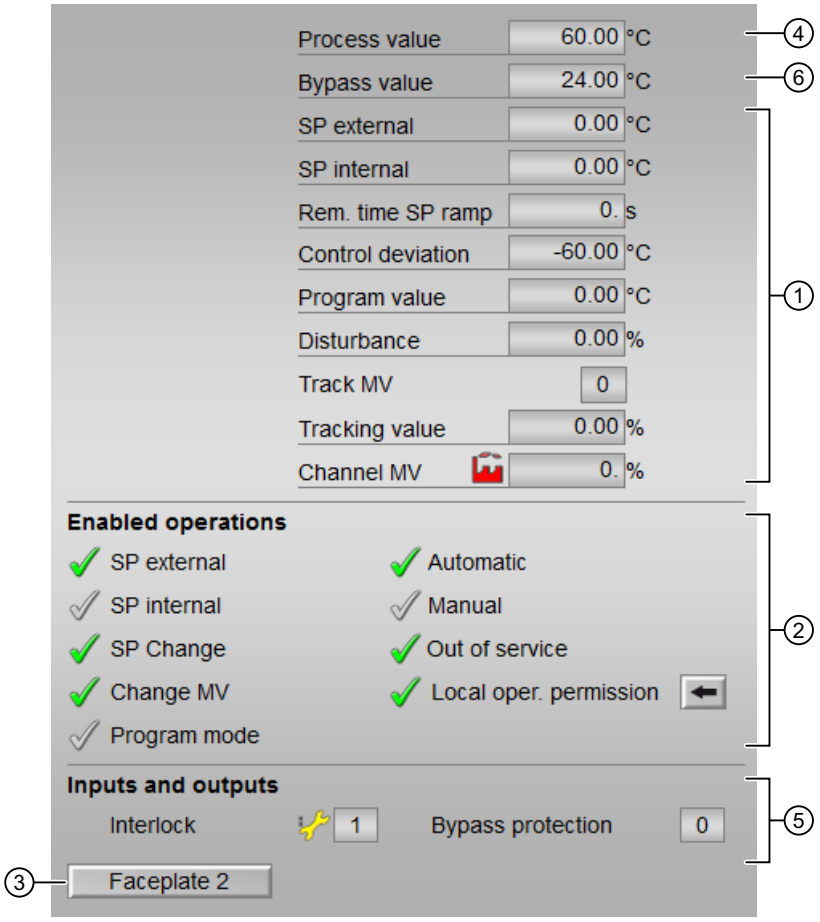
(22) Limit display for the setpoint

These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the engineering system (ES).

5.7.8.3 PIDConL, PIDConS and PIDConR previews

Preview of PIDConL

The preview shows you the parameters that you, as an OS operator, can control. You cannot control anything in this view, however.



(1) Preview area

This area shows you a preview for the following values:

- "SP external": currently applicable external setpoint
  - With the PIDConR block, this area is only visible if you have set the `Feature` Bit Switching operator controls for external setpoint to visible (Page 143) to 1
- "SP internal": currently applicable internal setpoint
- "Rem. time SP ramp" : Remaining time to reach the ramp target value (not with PIDConS).
- "Error signal": Current control deviation
- "Program value": Default value for program mode (not with PIDConS)
- "Disturbance variable": Additive value for feedforward control (not with PIDConS)

- "Track MV": Track manipulated variable (value is 1)
- "Tracking value": effective manipulated variable for "Track manipulated variable"
- "Channel MV": Display of the manipulated variable by the output channel block

(2) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: The OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

The following enabled operations are shown here:

- "SP external": You can feedforward the external setpoint.
- "SP internal": You can feedforward the internal setpoint.
- "Change SP": You can change the setpoint.
- "Change MV": You can change the manipulated variable.
- "Program mode": You can switch to "program mode". (not with PIDConS)
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248).

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

See also the section Opening additional faceplates (Page 203) for more on this.

(4) Process value

This area displays the real process value (`PV`).

(5) Inputs and outputs

This area shows the following parameters:

- "Interlock" (not with PIDConS):
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Interlock deact." (not with PIDConS):
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"

(6) Bypass value

This area displays the bypass value (`BypPV`).

5.8 PIDConS - Continuous PID controller (Small)

5.8.1 Description of PIDConS

Object name (type + number) and family

Type and number: FB 1830

Family: Control

Area of application for PIDConS

The block is used for the following applications:

- Fixed setpoint control

How it works

The block is a PID controller with continuous output signal (manipulated variable). It is used to activate a final controlling element with continuous action input.

The block functions following the PID algorithm with a delayed D component and an integrator with double precision.

The block is suitable for controlling sluggish control loops, for example, for temperatures and filling levels, and high-speed control loops, for example, for flow rates and speed. For a given CPU, a compromise has to be made between the number of controllers and the frequency with which the individual controllers have to be processed. The faster the modulated control loops are, i.e. the more frequently the manipulated variables have to be calculated per time unit, the lower the number of controllers that can be installed.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the control loop monitoring to work as planned in the trend view of the controller faceplates, the

`S7_xarchive:='Value, shortterm;'`

attributes in the process tag types for control loops at the controller function block must be set for the following tags:

- Input parameters:
  - `CPI_In`
- Output parameters
  - `MV`
  - `MV_HiAct`
  - `MV_LoAct`
  - `LoopClosed`
  - `SP`
  - `PV_Out`
  - `PV_ToleHi`
  - `PV_ToleLo`

There are no templates for the PIDConS block for process tag types or simulated use cases in the example project (APL_Example_xx, xx represents the language variant). However, you can replace PIDConL with PIDConS in the following process tag types and use cases.

Examples of process tag types:

- PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298)

Application cases in example project:

- Process simulation including noise generator (ProcSimC; ProcSimS) (Page 2347)
- Control loop monitoring for simulation with colored noise (ConPerMonSim) (Page 2351)
- Operating point-oriented adaptation of parameters (gain scheduling) for non-linear processes (GainSchedSim) (Page 2352)
- Filtering of noisy measured values in a control loop (SigSmoothSim) (Page 2354)

Startup characteristics

Use the Feature bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section PIDConS I/Os (Page 782).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`

Status bit	Parameter
3	OosAct.Value
4	OosLi.Value
5	AutAct.Value
6	Not used
7	ManAct.Value
8	SP_ExtAct.Value
9	1 = Input parameter MV_ChnST is interconnected
10	MV_TrkOn.Value AND NOT (ManAct.Value OR OosAct.Value)
11	MV.Value > MV_LoLim.Value
12 - 17	Not used
18	SimOn AND ManAct
19	Not used
20	1 = Input parameter Rbk is not interconnected (Rbk.ST = 16#FF)
21-22	Not used
23	OptimEn
24	OptimOcc
25 - 31	Not used

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock
1	PV_AH_Act.Value
2	PV_WH_Act.Value
3-4	Not used
5	PV_WL_Act.Value
6	PV_AL_Act.Value
7	PV_AH_En
8	PV_WH_En
9-10	Not used
11	PV_WL_En
12	PV_AL_En
13	PV_AH_MsgEn
14	PV_WH_MsgEn
15-16	Not used
17	PV_WL_MsgEn
18	PV_AL_MsgEn
19	Delay of the PV_AH_Lim message
20	Delay of the PV_WH_Lim message
21	Delay of the PV_WL_Lim message
22	Delay of the PV_AL_Lim message
23	Collection of message delays

Status bit	Parameter
24-28	Not used
29	`GrpErr.Value`
30	`RdyToStart.Value`
31	`MS_RelOp`

See also

PIDConS modes (Page 772)

PIDConS functions (Page 773)

PIDConS error handling (Page 779)

PIDConS messaging (Page 780)

PIDConS block diagram (Page 789)

5.8.2 PIDConS modes

PIDConS operating modes

The block can be operated using the following modes:

- Automatic mode (Page 72)
- Manual mode (Page 72)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and bumpless switchover in the Manual and automatic mode for control blocks (Page 72) chapter.

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the Manual and automatic mode for control blocks (Page 72) chapter.

"Out of service"

You can find general information about the "Out of service" mode in the Out of service (Page 71) chapter.

See also

Program mode for controllers (Page 78)
PIDConS block diagram (Page 789)
Description of PIDConS (Page 769)
PIDConS functions (Page 773)
PIDConS error handling (Page 779)
PIDConS messaging (Page 780)
PIDConS I/Os (Page 782)

5.8.3 PIDConS functions

Functions of PIDConS

The functions for this block are listed below.

Generation of manipulated variables

The manipulated variable `MV` can be generated as follows:

ManAct	MV_TrkOn	AdvCoAct AND NOT AdvCo-ModSP	MV =	Limit monitoring	State
1	-	-	`Man`	`MV_HiLim` `MV_LoLim`	Manual mode, set by the operator
0	1	-	`MV_Trk`	`MV_HiLim` `MV_LoLim`	Tracking with limitation
0	0	0	`P_Part` + `I_Part` + `D_Part`	`MV_HiLim` `MV_LoLim`	Automatic mode (PID algorithm)

If the controller is in "out of service" mode, the output parameter `MV` is set to the last valid value in manual mode or the neutral position manipulated variable depending on the `Feature` Bit (Neutral position manipulated variable takes effect at startup (Page 165)). Refer to the Out of service (Page 71) section for more on this.

Tracking and limiting a manipulated variable

The block provides the standard function Tracking and limiting a manipulated variable (Page 192).

Neutral position

The block provides the standard function Neutral position for motors, valves and controllers (Page 48).

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

"Actuator active" information

If the manipulated variable `MV` is greater than the minimum manual limit `MV_LoLim.Value`, this is recognized as actuator active. This status can be used to indicate a customized symbol in the process image, for example, and is saved in the status word (see Status word section in Description of PIDConS (Page 769)).

External/internal setpoint specification

The block provides the standard function Setpoint specification - internal/external (Page 127).

Setpoint limitation

With this function, you can limit the setpoint to a range by means of the parameters `SP_HiLim` (high limit) and `SP_LoLim` (low limit). If the setpoint lies outside the range defined by you, it is limited to the valid range.

If the setpoint is at or above the limit `SP_HiLim`, this is displayed at the output `SP_HiAct = 1`.

If the setpoint is at or below the limit `SP_LoLim`, this is displayed at the output `SP_LoAct = 1`

Tracking setpoint in manual mode

The block provides the standard function Tracking setpoint in manual mode (Page 192).

Simulating signals

The block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Process value (`SimPV`)

Limit monitoring of the process value

The block provides the standard function Limit monitoring of the process value (Page 86).

Error signal generation and dead band

The block provides the standard function Error signal generation and dead band (Page 188).

Selecting a unit of measure

The block provides the standard function Selecting a unit of measure (Page 207).

PID algorithm

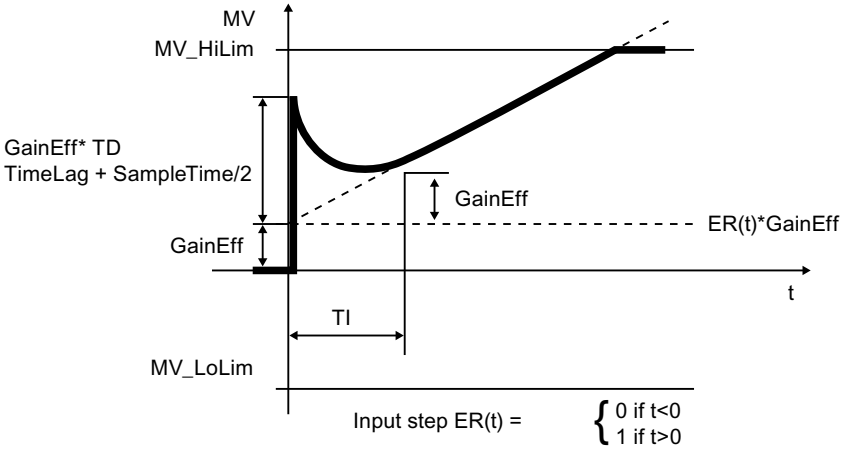
The manipulated variable is generated in automatic mode according to the following algorithm:

`MV` = `GainEff` · (1 + (1 / (`TI` · s) + (`TD` · s) / (1 + `TD` / `DiffGain` · s)) · `ER`

Where:

s = Complex number

The following step response occurs:



Note

The formula describes a standard application where P, I and D components are activated and the P and D components are not in the feedback circuit (`PropSel = 1, TI <> 0, DiffToFbk = 0` and `PropFacSP = 1`).

The D component delay is derived from `TD` / `DiffGain`.

- The P component is displayed at the `P_Part` I/O and can be deactivated using `PropSel` = 0 .
- The I component is displayed at the `I_Part` I/O and can be deactivated using `TI` = 0 .
- The D component is displayed at the `D_Part` I/O and can be deactivated using `TD` = 0 .

Anti-windup

The controller has an anti-windup function. The I component is frozen after the manipulated variable has reached limits (`MV_HiLim` or `MV_LoLim`).

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

- Signal status for the process value `PV_Out`:
  The signal status of the output parameter `PV_Out` always corresponds to the signal status of input parameter `PV` or, if the block is in simulation mode, 16#60.
- Signal status for the setpoint value `SP`:
  The signal status of the `SP` output parameter is always equivalent to the signal status of the input parameter `SP_Ext` or `SP_Int`, depending on how the setpoint is specified. If the internal setpoint `SP_Int` is used, the signal status is always outputted as `16#80`.
- Signal status of the error signal `ER`:
  The signal status of output parameter `ER` is obtained from the worst signal status of the two output parameters `PV_Out` and `SP`, and is outputted.
  The signal status `16#60` (external simulation) is suppressed because the block acts as a sink with external simulation.
- Signal status for the manipulated variable `MV`:
  The signal status of output parameter `MV` is formed in "Automatic mode" from the error signal `ER` and output. In "manual mode", the signal status is output as good. The signal status `16#60` (external simulation) is suppressed because the block acts as a sink with external simulation. In "manual mode", the signal status is output as good.
- Worst signal status:
  The worst signal status `ST_Worst` for the block is formed from `SP`, `PV_Out`, `MV_ChnST` and `Rbk` .

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
4 - 14	Not used.
15	Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165)
16	Neutral position manipulated variable takes effect at startup (Page 165)
17 - 22	Not used.
23	SP following PV in open loop has no priority over SP_Ext and SP limits (Page 178)
24	Enabling local operator authorization (Page 157)

Bit	Function
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
27 - 31	Not used.

Operator permissions

The block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode" `AutModOp`
1	1 = Operator can switch to "manual mode" `ManModOp`
2	1 = Operator can switch to "Out of service" mode `OosOp`
3	1 = Operator can switch the setpoint to "external" `SP_ExtOp`
4	1 = Operator can switch the setpoint to "internal" `SP_IntOp`
5	1 = Operator can change the internal setpoint `SP_Int`
6	1 = Operator can change the manual parameter `Man`
7	1 = Operator can change the high operation limit of the setpoint `SP_HiLim`
8	1 = Operator can change the low operation limit of the setpoint `SP_LoLim`
9	1 = Operator can change the operation high limit of the manipulated variable `ManHiLim`
10	1 = Operator can change the operation low limit of the manipulated variable `ManLoLim`
11	1 = Operator can permit the PID optimization function `OptimEn`
12	1 = Operator can enable the track setpoint in "manual mode" function `SP_TrkPV`
13	1 = Operator can enable the bumpless switchover from external to internal `SP_TrkExt`
14	1 = Operator can change the gain parameter `Gain`
15	1 = Operator can change the integral time parameter `TI`
16	1 = Operator can change the derivative time parameter `TD`
17	1 = Operator can change the derivative gain parameter `DiffGain`
18	1 = Operator can change the dead band parameter `DeadBand`
19	1 = Operator can change the limit (process value) `PV_AH_Lim` for the high alarm
20	1 = Operator can change the limit (process value) `PV_WH_Lim` for the high warning
21	1 = Operator can change the hysteresis (process value) `PV_Hyst`
22	1 = Operator can change the limit (process value) `PV_WL_Lim` for the low warning
23	1 = Operator can change the limit (process value) `PV_AL_Lim` for the low alarm
24	1 = Operator can activate the Simulation function `SimOn`
25	1 = Operator can activate the Release for maintenance function `MS_RelOp`
26	1 = Operator can change the simulation value `SimPV`
27	1 = Operator can activate / deactivate messages via `PV_AH_MsgEn`
28	1 = Operator can activate / deactivate messages via `PV_WH_MsgEn`
29	1 = Operator can activate / deactivate messages via `PV_WL_MsgEn`
30	1 = Operator can activate / deactivate messages via `PV_AL_MsgEn`
31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Release for maintenance

The block provides the standard function Release for maintenance (Page 64)

Generating instance-specific messages

The block provides the standard function Generating instance-specific messages (Page 200) without the time stamp function in the I/O.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Specifying the display area for process and setpoint values as well as operations

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Labeling of buttons and text (Page 205)

EventTs functions (Page 1634)

PIDConS modes (Page 772)

PIDConS error handling (Page 779)

PIDConS messaging (Page 780)

PIDConS I/Os (Page 782)

PIDConS block diagram (Page 789)

5.8.4 PIDConS error handling

Error handling of PIDConS

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` output parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
30	The value of `PV` can no longer be displayed in the REAL number field.
31	The value of `SP_Ext` can no longer be displayed in the REAL number field.
33	The value of `MV_Trk` can no longer be displayed in the REAL number field.
35	The value of `Rbk` can no longer be displayed in the REAL number field.
36	The value of `MV` can no longer be displayed in the REAL number field.
51	`AutModLi` = 1 and `ManModLi` = 1 `SP_LiOp` = 1 and `SP_IntLi` = 1 and `SP_ExtLi` = 1
60	|`TI`| < `SampleTime` / 2
61	|`TD`| < `SampleTime`
62	`DiffGain` < 1 or `DiffGain` > 10
63	`TD` / `DiffGain` < `SampleTime` / 2

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the section Error handling (Page 119).

See also

Description of PIDConS (Page 769)

PIDConS modes (Page 772)

PIDConS functions (Page 773)

PIDConS messaging (Page 780)

PIDConS I/Os (Page 782)

PIDConS block diagram (Page 789)

5.8.5 PIDConS messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

The following control system fault messages can be output:

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 5	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter CSF. If it changes to CSF = 1, a process control fault is triggered (MsgEvId1, SIG 5).

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	Alarm - high	$$BlockComment$$ PV - high alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ PV - high warning limit violated
	SIG 3	Warning - low	$$BlockComment$$ PV - low warning limit violated
	SIG 4	Alarm - low	$$BlockComment$$ PV - low alarm limit violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You have the option of using one or two instance-specific messages for this block.

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 6	AS process control message - fault	$$BlockComment$$ External message 1 Status 16#@8%x@
	SIG 7	AS process control message - fault	$$BlockComment$$ External message 2 Status 16#@9%x@

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance **MsgEvId1**

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	Process value PV_Out
5	Control deviation ER
6	Signal status ExtMsg1
7	Signal status ExtMsg2
8	ExtVa108
9	ExtVa109
10	Not allocated

The associated values 8 ... 9 are allocated to the parameters ExtVa108 ... ExtVa109 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of PIDConS (Page 769)

PIDConS modes (Page 772)

PIDConS functions (Page 773)

PIDConS error handling (Page 779)

PIDConS I/Os (Page 782)

PIDConS block diagram (Page 789)

5.8.6 PIDConS I/Os

I/Os of PIDConS

Input parameters

Parameter	Description	Type	Default
AutModLi*	1 = "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 0)	BOOL	0
BatchEn	1 = Enable allocation for batch control	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
CPI_In	Input for control performance index, which is calculated by the assigned ConPerMon block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DeadBand	Width of dead band	REAL	0.0
DiffGain	Gain of differentiator [1..10] DiffGain = TD / (delay time of D component)	STRUCT • Value: REAL • ST: BYTE	- • 5.0 • 16#80
EN	1 = Called block will be processed	BOOL	1
ExtMsg1	1 = Binary input for freely selectable message 1 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	1 = Binary input for freely selectable message 2 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
ExtVa109	Associated value 9 for messages (MsgEvID1)	ANY	
Feature	I/O for additional PIDConS functions (Page 773)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Gain	Proportional gain Gain.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#FF

Parameter	Description	Type	Default
IntHoldNeg	1 = Integrator cannot run in negative direction	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
IntHoldPos	1 = Integrator cannot run in positive direction	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Man*	Manual specification for the manipulated variable	REAL	0.0
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
ModLiOp	Operating mode switchover between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MsgEvID1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_ChnST	Signal status of output channel for MV Should be connected to an output channel block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
MV_HiLim	Limit (high) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
MV_LoLim	Limit (low) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_Offset	Manipulated variable for ER=0, operating point for controller with deactivated I component	REAL	0.0
MV_OpScale	OS display range for manipulated variable MV	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
MV_SafePos	Manipulated variable neutral position	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_Trk*	Tracking value for the manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
MV_TrkOn	1 = Tracking of manipulated variable MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_Unit	Unit of measure for manipulated variable	INT	1342
Occupied	1 = Occupied by batch control	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OptimEn*	1 = Enable optimization of PID parameters by PID tuner	BOOL	0
OptimOcc*	1 = Optimization running	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permission (Page 773)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
PropSel	1 = Activate P component	BOOL	1
PV*	Process value (controlled variable)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_A_DC*	Delay time for incoming PV alarms [s]	REAL	0.0
PV_AH_En	1 = Enable PV alarm limit (high)	BOOL	1
PV_AH_Lim	Limit PV alarm (high)	REAL	95.0
PV_AH_MsgEn	1 = Enable PV alarm (high) message	BOOL	1
PV_AL_En	1 = Enable PV alarm limit (low)	BOOL	1
PV_AL_Lim	PV alarm limit (low)	REAL	5.0
PV_AL_MsgEn	1 = Enable PV alarm (low) message	BOOL	1
PV_Hyst	Hysteresis for PV alarm, warning and tolerance limits	REAL	1.0
PV_OpScale	Limit for scale in PV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
PV_Unit	Unit of measure for process value	INT	1001
PV_W_DC*	Delay time for incoming PV warnings [s]	REAL	0.0
PV_WH_En	1 = Enable PV warning limit (high)	BOOL	1
PV_WH_Lim	Limit PV warning (high)	REAL	90.0
PV_WH_MsgEn	1 = Enable PV warning (high) message	BOOL	1
PV_WL_En	1 = Enable PV warning limit (low)	BOOL	1

Parameter	Description	Type	Default
PV_WL_Lim	Limit PV warning (low)	REAL	10.0
PV_WL_MsgEn	1 = Enable PV warning (low) message	BOOL	1
Rbk*	Position feedback for display on OS	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SafePos	1 = Neutral position (Page 48) for controller manipulated variable is MV_HiLim 0 = Neutral position for controller manipulated variable is MV_LoLim	BOOL	0
SafePos2	Neutral position for control valve: 0 = SafePos is valid 1 = Neutral position is MV_SafePos 2 = Neutral position is last manipulated variable (stop)	INT	0
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SimOn*	1 = Simulation on	BOOL	0
SimPV*	Process value used for SimOn = 1	REAL	0.0
SP_Ext*	external setpoint - (to interconnection)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtLi*	1 = Select external setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOp*	1 = Select external setpoint (via operator)	BOOL	0
SP_HiLim	Limit (high) of internal SP	STRUCT • Value: BOOL • ST: BYTE	 • 100.0 • 16#80
SP_Int*	Internal setpoint for operation	REAL	0.0
SP_IntLi*	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_IntOp*	1 = Select internal setpoint (via operator)	BOOL	0
SP_LiOp	Select setpoint source (internal/external): 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_LoLim	Limit (low) of internal SP	STRUCT • Value: REAL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
SP_TrkPV	1 = Setpoint follows PV in "manual mode" and with tracking	BOOL	0
StepNo	Batch step number	DWORD	16#00000000
TD	Derivative time [s] TD.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
TI	Integral time [s] TI.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#FF
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
D_Part	D component of PID algorithm	REAL	0.0
ENO	1 = Block algorithm completed without errors	BOOL	0
ER	Error signal	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see PIDConS error handling (Page 779)	INT	-1
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
I_Part	I component of PID algorithm	REAL	0.0
LoopClosed	1 = Control loop closed 0 = Control loop open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MV	Manipulated variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_HiAct	1 = Limit (high) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_LoAct	1 = Limit (low) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_UnitOut	Unit of measure for manipulated variable, for interconnecting to the MV_Unit input parameter of the ConPerMon block	INT	0
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
P_Part	P component of PID algorithm	REAL	0.0
PV_AH_Act	1 = PV alarm (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AL_Act	1 = PV alarm (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	Output for process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_ToleHi	Limit (high) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_ToleLo	Limit (low) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

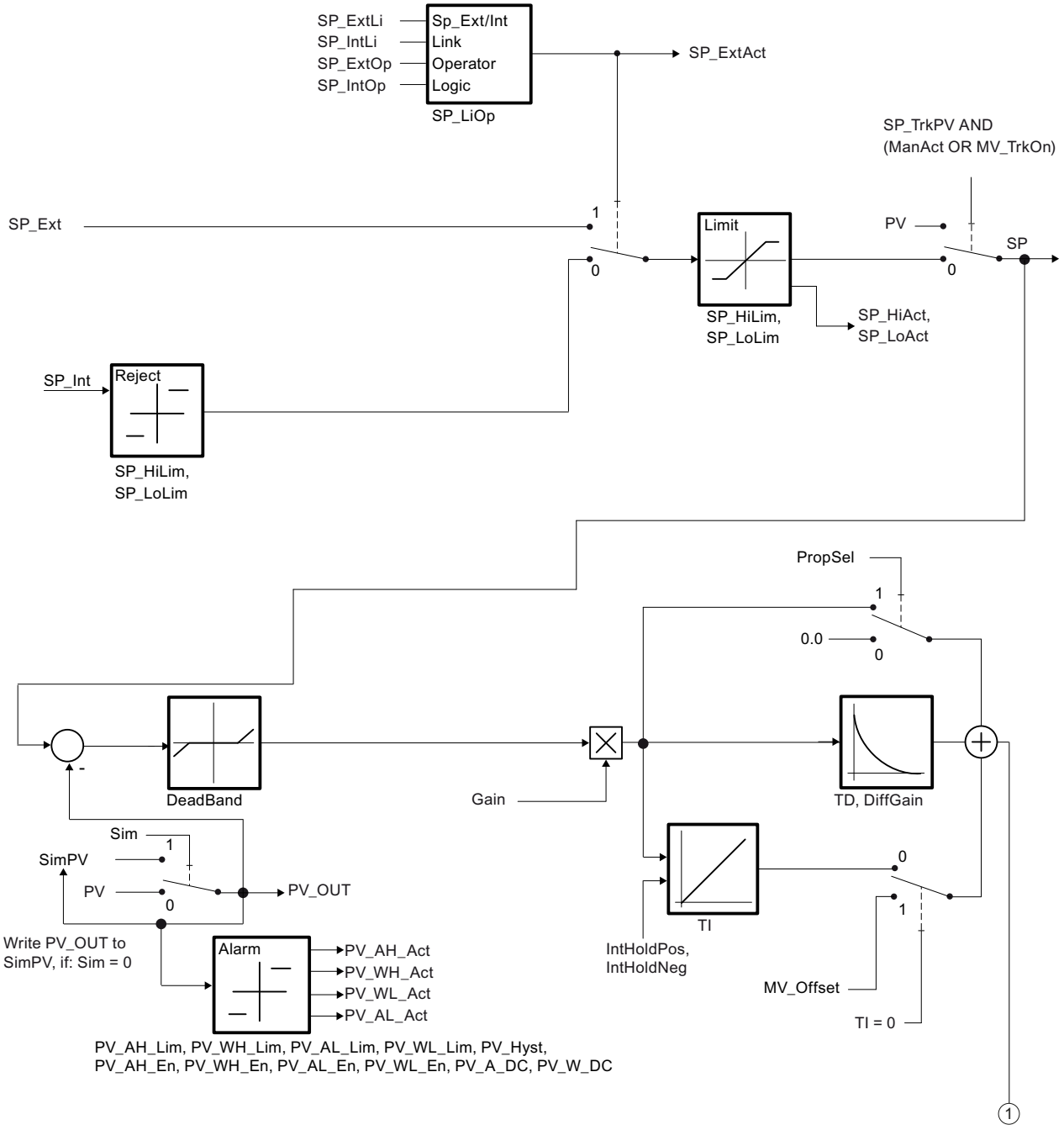
Parameter	Description	Type	Default
PV_WH_Act	1 = PV warning (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_WL_Act	1 = PV warning (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_UnitOut	Unit of measure for process value, for interconnecting to the PV_Unit input parameter of the ConPerMon block	INT	0
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP	Setpoint used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtAct	1 = External setpoint is active 0 = Internal setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_HiOut	Limit (high) for SP corresponds to input parameter SP_HiLim	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP_LoOut	Limit (low) for SP corresponds to input parameter SP_LoLim	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 769)	DWORD	16#00000000
Status2	Status word 2 (Page 769)	DWORD	16#00000000
SumMsgAct	1 = Active hardware interrupt	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

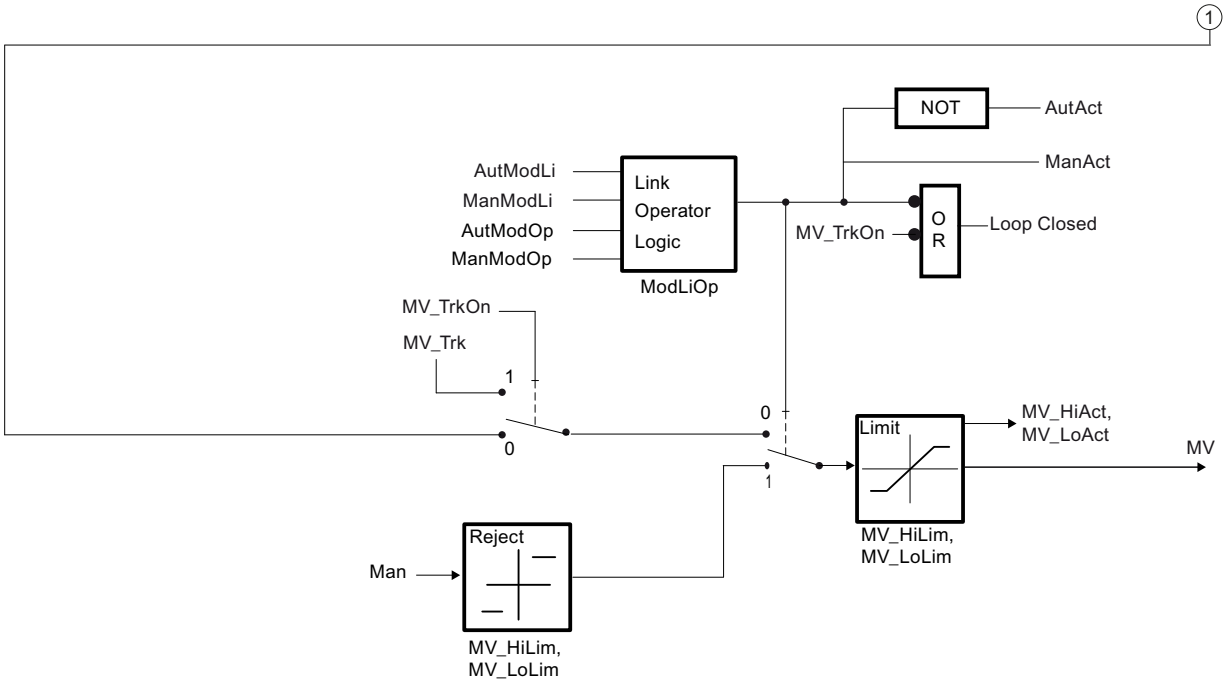
See also

PIDConS modes (Page 772)
PIDConS messaging (Page 780)
PIDConS block diagram (Page 789)

5.8.7 PIDConS block diagram

PIDConS block diagram





See also

PIDConS I/Os (Page 782)

PIDConS messaging (Page 780)

PIDConS error handling (Page 779)

PIDConS functions (Page 773)

PIDConS modes (Page 772)

Description of PIDConS (Page 769)

5.8.8 Operator control and monitoring

5.8.8.1 PIDConS views

Views of the PIDConS block

The block PIDConS provides the following views:

- PIDConL, PIDConS and PIDConR standard views (Page 761)
- Alarm view (Page 296)
- Limit view of PID controllers (Page 285)
- Trend view (Page 299)
- Parameter view of PID controllers (Page 275)
- PIDConL, PIDConS and PIDConR previews (Page 766)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icons for PID and FM controller (Page 235)

Refer to the Structure of the faceplate (Page 242) and Block icon structure (Page 226) chapters for general information about the faceplate and block icon.

See also

Ramp view (Page 294)

5.9 PIDConR - Continuous PID controller with external reset

5.9.1 Description of PIDConR

Object name (type + number) and family

Type + number: FB 1875

Family: Control

Area of application for PIDConR

The block is used for the following applications:

- Fixed setpoint control
- Cascade control
- Ratio control
- Split-range control
- Smith predictor closed-loop control
- Override control (override)

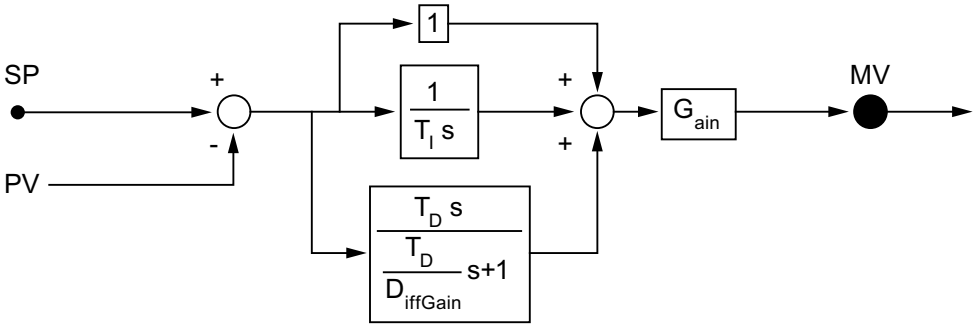
Unlike PIDConL, the PIDConR permits an external reset and satisfies the special requirements of the US market.

How it works

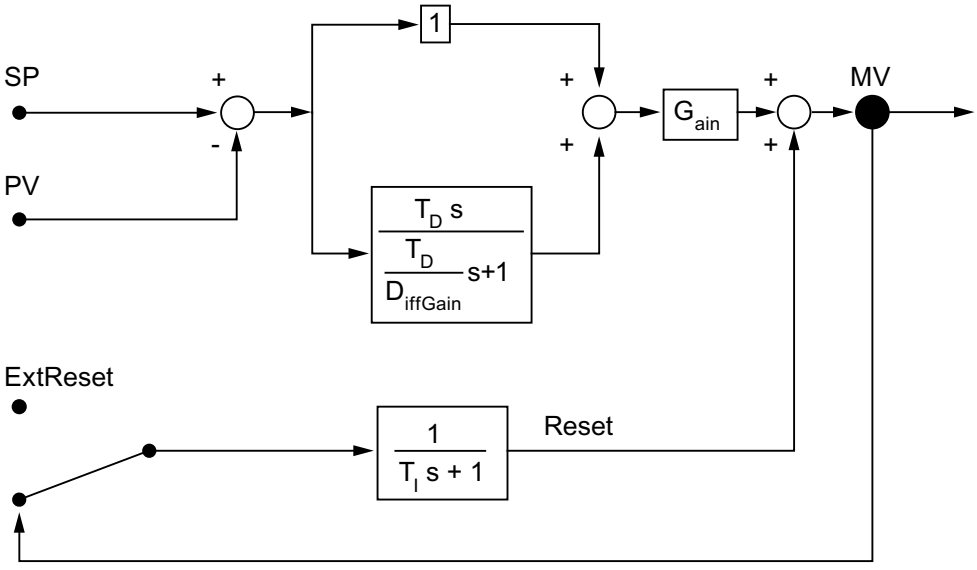
The block is a PID controller with continuous output signal (manipulated variable). It is used to activate a final controlling element with continuous action input.

The block functions following the PID algorithm with a delayed D component and an integrator with double precision. Its particular feature is that it is an incremental control algorithm with a serial-interactive structure. Incremental means that the current manipulated variable is calculated from the old manipulated variable of the last sampling step. In place of the old manipulated variable, an output point for the manipulated variable calculation (external reset) can be specified externally by interconnection. The difference between the parallel controller structure of the PIDConL and the serial-interactive structure of the PIDConR is shown in the next two diagrams.

PIDConL



PIDConR



The block is suitable for controlling sluggish control loops, for example, for temperatures and filling levels, and high-speed control loops, for example, for flow rates and speed. For a given CPU, a compromise has to be made between the number of controllers and the frequency with which the individual controllers have to be processed. The faster the modulated control loops are, i.e. the more frequently the manipulated variables have to be calculated per time unit, the lower the number of controllers that can be installed.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the control loop monitoring to work as planned in the trend view of the controller faceplates, the

`S7_xarchive:='Value, shortterm;'`

attributes in the process tag types for control loops at the controller function block must be set for the following tags:

- Input parameters:
  - `CPI_In`
- Output parameters
  - `MV`
  - `MV_HiAct`
  - `MV_LoAct`
  - `LoopClosed`
  - `SP`
  - `PV_Out`
  - `PV_ToleHi`
  - `PV_ToleLo`

For the PIDConR block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Cascade control with PIDConR (CascadeR) (Page 2319)
- Override control with PIDConR (OverrideR) (Page 2324)
- PIDConR with safety logic and control loop monitoring (PIDConR_ConPerMon) (Page 2299)
- Ratio control with PIDConR (RatioR) (Page 2315)

Note

The meaning of the controller parameters of both structures is different for all controller parameter settings with a D component. If you want to transfer parameter values from one structure to another, they have to be converted according to the formula in the PIDConR function .

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

The following output parameters are written with the corresponding input parameters:

- `PV_HysOut`
- `PV_AH_Out`
- `PV_WH_Out`
- `PV_TH_Out`
- `PV_AL_Out`

- PV_WL_Out
- PV_TL_Out

Status word allocation for Status1 parameter

You can find a description for each parameter in section PIDConR I/Os (Page 818).

Status bit	Parameter
0	Occupied
1	BatchEn
2	SimOn
3	OosAct.Value
4	OosLi.Value
5	AutAct.Value
6	Not used
7	ManAct.Value
8	SP_ExtAct.Value
9	MV_ForOn.Value
10	MV_TrkOn.Value AND NOT (ManAct.Value OR OosAct.Value OR MV_ForOn.Value)
11	MV.Value > ManLoLim
12	SimLiOp.Value
13	"Interlock" button is enabled OR Intlock.ST ≠ 16#FF
14	0 = Open padlock in the block icon 1 = Closed padlock in block icon
15	SP_LoadOn.Value
16	SP_LoadOn.Value AND NOT (MV_TrkOn.Value OR OosAct.Value OR MV_ForOn.Value)
17	Feature Bit 19
18	Feature Bit 21
19	AdvCoAct
20	1 = Input parameter Rbk is not interconnected (RbkOut.ST = 16#FF)
21	NegGain
22	1 = SP ramp active
23	OptimEn
24	OptimOcc
25	Not used
26	Display of BypassAct.Value in faceplate (display and operator controls) and block icon
27	Not used
28	SimOn AND ManAct
29	BypProt
30	Dead band is temporarily disabled
31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	`PV_AH_Act.Value`
2	`PV_WH_Act.Value`
3	`PV_TH_Act.Value`
4	`PV_TL_Act.Value`
5	`PV_WL_Act.Value`
6	`PV_AL_Act.Value`
7	`PV_AH_En`
8	`PV_WH_En`
9	`PV_TH_En`
10	`PV_TL_En`
11	`PV_WL_En`
12	`PV_AL_En`
13	`PV_AH_MsgEn`
14	`PV_WH_MsgEn`
15	`PV_TH_MsgEn`
16	`PV_TL_MsgEn`
17	`PV_WL_MsgEn`
18	`PV_AL_MsgEn`
19	`ER_AH_Act.Value`
20	`ER_AL_Act.Value`
21	`ER_AH_En`
22	`ER_AL_En`
23	`ER_AH_MsgEn`
24	`ER_AL_MsgEn`
25	`RbkWH_Act.Value`
26	`RbkWL_Act.Value`
27	`RbkWH_En`
28	`RbkWL_En`
29	`RbkWH_MsgEn`
30	`RbkWL_MsgEn`
31	`MS_RelOp`

Status word allocation for `Status3` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via `EventTsIn`
1	Effective signal 2 of the message block connected via `EventTsIn`
2	Effective signal 3 of the message block connected via `EventTsIn`
3	Effective signal 4 of the message block connected via `EventTsIn`
4	Effective signal 5 of the message block connected via `EventTsIn`

Status bit	Parameter
5	Effective signal 6 of the message block connected via `EventTsIn`
6	Effective signal 7 of the message block connected via `EventTsIn`
7	Effective signal 8 of the message block connected via `EventTsIn`
8	Delay of the `PV_AH_Lim` message
9	Delay of the `PV_WH_Lim` message
10	Delay of the `PV_TH_Lim` message
11	Delay of the `PV_TL_Lim` message
12	Delay of the `PV_WL_Lim` message
13	Delay of the `PV_AL_Lim` message
14	Delay of the `ER_AH_Lim` message
15	Delay of the `ER_AL_Lim` message
16	Collection of message delays
17	`BypassAct.Value`
18 - 23	Not used
24	Hidden bypass signal in Intlock
25	`Feature`2 bit 2: Separate bypass signal
26	`LockAct.Value`
27	`SP_UpRaAct`, `SP_DnRaAct` limits enabled for gradient mode (`SP_RateOn = 1`)
28	`GrpErr.Value`
29	`RdyToStart.Value`
30	1 = Input parameter `MV_ChnST` is interconnected
31	Not used

Status word allocation for `Status4` parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via `EventTsIn`
8 - 31	Not used

See also

PIDConR block diagram (Page 834)

PIDConR messaging (Page 815)

PIDConR error handling (Page 814)

PIDConR functions (Page 800)

PIDConR modes (Page 798)

5.9.2 PIDConR modes

Operating modes of PIDConR

The block can be operated using the following modes:

- Automatic mode (Page 72)
- Manual mode (Page 72)
- Program mode for controllers (Page 78)
- Out of service (Page 71)

In "manual mode", the control settings for the device are made manually by the operator. The operator decides how to change the block's manipulated variable (output signal).

In "automatic mode", the controller's manipulated variable is calculated automatically by the block algorithm.

Changing between operating modes

The switchover between manual and automatic modes takes place as shown in the following schematic:

Switchover by using faceplates: The switchover between operating modes is carried out in the standard view of the faceplate. In the function block, the parameters `ManModOp` for "manual mode" and `AutModOp` for "automatic mode" are used.

Switchover via interconnection (CFC or SFC instance): The switchover between the operating modes is carried out by means of interconnection on the function block.

Note

You can access the variable parameters `AutModOp` and `ManModOp` from a normal SFC (in contrast to the instance of an SFC type). The SFC can thus change the operating mode without revoking the access rights of the operator.

Points to note for this block

With PIDConR the switchover between operating modes via interconnectable input parameters follows a different logic than that used for other controller blocks. This logic is oriented towards the special requirements of the US market. The basic approach is to issue the controller with certain commands by interconnection using an individual input parameter. The commands issued by interconnection take priority over the entries in the faceplate, i.e. if such a command input is made, the associated operator controls are locked in the faceplate:

- If the interconnectable `AutExtSet` = 1 input parameter is set, the controller goes into "automatic mode" with an external setpoint. This is also known as the "cascade" operating mode.
- If the interconnectable `AutIntSet` = 1 input parameter is set, the controller goes into "automatic" mode with internal setpoint. `AutIntSet` has higher priority than`AutExtSet`. For more information about the internal and external setpoint, refer to section PIDConR functions (Page 800).

- If the interconnectable input parameter ManSet = 1 is set, the controller goes into "manual" mode. This command has higher priority than AutIntSet and AutExtSet.
- If one of the interconnectable input parameters MV_Close or MV_Open is set, the controller also goes into "manual" operating mode and performs the corresponding command.

The ModLiOp, ManModLi and AutModLi input parameters are not therefore present in PIDConR.

Switchover from automatic mode to manual mode

When changing over from "automatic mode" to "manual mode", the last valid control settings (**M**anipiulated **V**alue MV) for the controller set in "automatic mode" remain valid until you change the control settings manually.

Switchover from manual mode to automatic mode

The switchover from manual to automatic mode can take place with or without the internal setpoint tracking the process value. You specify this behavior on the SP_TrkPV I/O, which can also be operated from the faceplate in the parameter view (Option "SP : = PV in Manual").

Switchover with tracked internal setpoint

(SP_TrkPV = 1) means that in "manual" mode the setpoint (SP) tracks the process variable (PV). This results in bumpless switchover, which means the manipulated variable remains constant after switching back to "automatic mode" until the setpoint (SP) is changed or the process value (PV) changes.

Switchover without tracked internal setpoint

(SP_TrkPV = 0) means that the setpoint does not track the actual value in manual mode and that the block immediately recalculates the manipulated variable based on the setpoint and process value (PV) when the mode is changed. PIDConR only offers **switchover without P step**: During switchover, the I action (reset) of the controller is set in such a way that the switchover is carried out without a P step (virtually bumpless referring to the manipulated variable). In case of a large control deviation, when switching from the manual mode to the automatic mode, it may happen that the integral action is set far outside the manipulated variable limits. See Feature bit With accelerated return of the integral action from the manipulated variable limit (Page 179).

Reaction of signals when operating mode is changed

Using the Feature Bit Resetting the commands for changing the mode (Page 160), you can specify whether the block automatically resets the signal for changing the operating mode.

"Program mode for controllers"

You can find general information about the "Program mode for controller" in the section Program mode for controllers (Page 78).

In case of a large control deviation, when switching from the program mode with manipulated variable specification to the automatic mode, it may happen that the integral action is set far

outside the manipulated variable limits. See `Feature` bit With accelerated return of the integral action from the manipulated variable limit (Page 179).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

PIDConR block diagram (Page 834)

PIDConR I/Os (Page 818)

PIDConR messaging (Page 815)

PIDConR error handling (Page 814)

Description of PIDConR (Page 792)

Disabling bumpless switchover to automatic mode for controllers (Page 172)

5.9.3 PIDConR functions

Functions of PIDConR

The functions for this block are listed below.

Generation of manipulated variables

The manipulated variable `MV` can be generated as follows:

MV_ForOn	MV_BumpOn	MV_Close	MV_Open	Man-Act	MV_TrkOn	Adv-CoAct AND NOT AdvCo-ModSP	MV =	Limit monitoring	State
1	-	-	-	-	-	-	`MV_Forced`	none	Forced tracking through constraint without limitation

0	1	-	-	-	-	-	`MV_Bump`	Manual mode:
								`ManHiLim`
								`ManLoLim`
								Automatic mode:
								`MV_HiLim`
								`MV_LoLim`
									Sudden manipulation of the manipulated variable

MV_ForOn	MV_BumpOn	MV_Close	MV_Open	Man-Act	MV_TrkOn	Adv-CoAct AND NOT AdvCo-ModSP	MV =	Limit monitor-ing	State
0	0	1	-	-	-	-	ManLoLim	ManHiLim ManLoLim	Close by interconnection
0	0	0	1	-	-	-	ManHiLim	ManHiLim ManLoLim	Open by interconnection
0	0	0	0	1	-	-	Man	ManHiLim ManLoLim	Manual mode, set by the OS operator or via the ManSet = 1 command
0	0	0	0	0	1	-	MV_Trk	MV_HiLim MV_LoLim	Tracking with limitation
0	0	0	0	0	0	1	AdvCoMV	MV_HiLim MV_LoLim	Higher-level program mode
0	0	0	0	0	0	0	PID.OUT + FFwd	MV_HiLim MV_LoLim	Automatic mode (PID algorithm)

If the controller is in "out of service" mode, the output parameter MV is set to the last valid value in manual mode or the neutral position manipulated variable depending on the Feature Bit Neutral position manipulated variable takes effect at startup (Page 165). Refer to the Out of service (Page 71) section for more on this.

The PIDConR offers the following special ways of influencing the generation of manipulated variables via interconnectable input parameters. The commands issued by interconnection take priority over the entries in the faceplate, i.e. if such a command input is made, the associated operator controls are locked in the faceplate.

MV_BumpOn is used for sudden manipulation of the manipulated variable and has a similar effect to MV_ForOn, the difference being that the appropriate limits are applied. If MV_BumpOn = 1 is set in "automatic mode", the MV_Bump manipulated variable is written in a limited manner between MV_HiLim and MV_LoLim to output MV. If MV_BumpOn = 1 is set in "manual mode", the MV_Bump manipulated variable is written in a limited manner between ManHiLim and ManLoLim to output MV. If MV_BumpOn is reset to 0, the controller returns to its previous mode.

MV_Close is used to close the adjustment valve. MV_Close = 1 switches the controller to "manual mode" with manipulated variable MV = ManLoLim. If MV_Close is reset to 0, the controller remains in "manual mode".

MV_Open is used to open the adjustment valve. MV_Open = 1 switches the controller to "manual mode" with manipulated variable MV = ManHiLim. If MV_Open is reset to 0, the controller remains in "manual mode".

These commands have higher priority than "automatic mode", but lower priority than forced tracking by MV_ForOn.

For meshed controller structures, such as a cascade control and closed-loop control with the PIDConR block, the ExtReset input parameter is used with ExtRstOn = 1 rather than MV_Trk with MV_TrkOn = 1 (also refer to the process tag types Cascade control with PIDConR (CascadeR) (Page 2319) and Override control with PIDConR (OverrideR) (Page 2324)).

Displaying additional information relating to the manipulated variable on the output

The manual manipulated variable limitations `ManLoLim` and `ManHiLim` are copied to the `ManLoOut` and `ManHiOut` output parameters so that they can be further interconnected to the secondary controller as setpoint limits `SP_ExtLoLim` and `SP_ExtHiLim`. If you want to use the same limit pairs for "manual" and "automatic" mode, you can interconnect output parameters `ManLoOut` and `ManHiOut` to input parameters `MV_LoLim` and `MV_HiLim` of the same block and thereby control the limits for manipulated variable limitation in the faceplate. In most cases, a valve's complete control range can be passed through in "manual" mode. You can then further interconnect the `ManLoOut` and `ManHiOut` output parameters to the `LoScale` and `HiScale` input parameters of the assigned analog output channel block. Caution: If the valves have an open neutral position (`SafePos` = 1), this interconnection must be crossed over: `LoScale` = `ManHiOut` and `HiScale` = `ManLoOut`. On the controller, 0% is always interpreted as the valve being closed and 100% as the valve being open, but the channel block then issues a control signal of 0% for a 100% controller manipulated variable.

Tracking and limiting a manipulated variable

The block provides the standard function Tracking and limiting a manipulated variable (Page 192).

In case of a large control deviation, when disabling the tracking or forced tracking function and switching to the automatic mode, it may happen that the integral action is set far outside the manipulated variable limits. See `Feature` bit With accelerated return of the integral action from the manipulated variable limit (Page 179).

Neutral position

The block provides the standard function Neutral position for motors, valves and controllers (Page 48).

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

"Actuator active" information

If the manipulated variable `MV` is greater than the minimum manual limit `ManLoLim`, this is recognized as actuator active. This status can be used to display a custom symbol in the process image, for example, and is stored in the status word (you can find additional information under "Status word" in the Description of PIDConR (Page 792) section).

Limit monitoring of position feedback

The block provides the standard function Limit monitoring of the feedback (Page 94).

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

External/internal setpoint specification

This input of setpoints is carried out either using a CFC/SFC program or using the faceplate (operator). The operator can specify an internal setpoint value (`SP_Int`) or a higher-level open-loop control will specify an external setpoint value (`SP_Ext`).

Setpoint specification internally and externally using faceplate

With PIDConR, the setpoint signal source is selected via the faceplate at the SP_IntOp = 1 parameter for the internal setpoint specification SP_ExtOp = 1 for external setpoint specification, just as it is with other controllers.

Note

In contrast to the other controllers, with PIDConR you can only switch to external setpoint specification via SP_ExtOp = 1 in automatic mode or in the program SP mode.

Setpoint specification internally and externally using interconnection

With PIDConR the switchover between internal and external setpoint via interconnectable input parameters follows a different logic than that used for other controller blocks. This logic is oriented towards the special requirements of the US market.

The setpoint signal source (internal/external) can be selected by interconnection along with selection of the operating mode using the `AutIntSet` input parameter for automatic with an internal setpoint and `AutExtSet` for automatic with an external setpoint (additional information is available in the PIDConR modes (Page 798) section). The commands issued by interconnection take priority over the entries in the faceplate, i.e. if such a command input is made, the associated operator controls are locked in the faceplate. Input parameters `SP_LiOp`, `SP_ExtLi` and `SP_IntLi` on PIDConR are not therefore needed.

The PIDConR also offers a special way of influencing the setpoint specification via interconnectable input parameters. Loading setpoints. If the `SP_LoadOn` = 1 input parameter is set, the controller goes into "automatic" mode. The value of the `SP_Load` input parameter is limited according to an internal setpoint and used for control purposes. If the parameter `SP_LoadOn` changes back from 1 to 0, the controller remains in "automatic" mode and sets the default setpoint back to the internal setpoint.

Loading a setpoint via `SP_LoadOn` takes priority over all other forms of setpoint specification.

Bumpless switchover from external to internal setpoint

The parameter `SP_TrkExt` = 1 is used so that the internal setpoint tracks the external setpoint to achieve a bumpless switchover from the external to the internal setpoint. This allows unwanted jumps at the output parameter to be avoided.

Setpoint limiting for external setpoints

The block provides the standard function Setpoint limiting for external setpoints (Page 192).

Gradient limit of the setpoint

The block provides the standard function Gradient limit of the setpoint (Page 124).

Using setpoint ramp

The block provides the standard function Using setpoint ramp (Page 123).

Tracking setpoint in manual mode

The block provides the standard function Tracking setpoint in manual mode (Page 192).

Simulating signals

The block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Process value (`SimPV`, `SimPV_Li`)
- Position feedback (`SimRbk`, `SimRbkLi`)

Bypass function

This block provides the standard function Bypassing signals (Page 107).

Limit monitoring of the process value

The block provides the standard function Limit monitoring of the process value (Page 86) with the alarm delay type Two time values for each individual limit (Page 198).

The PIDConR is the only block to have separate input parameters for the alarm delay at the high and low limits. Delay alarm for control deviation at setpoint step changes (Page 186)

Providing PV limit at the output

For further connections to the other blocks, the following input parameters are also displayed with the corresponding output parameters:

- `PV_HysOut` := `PV_Hyst`
- `PV_AH_Out` := `PV_AH_Lim`

- PV_WH_Out := PV_WH_Lim
- PV_TH_Out := PV_TH_Lim
- PV_TL_Out := PV_TL_Lim
- PV_WL_Out := PV_WL_Lim
- PV_AL_Out := PV_AL_Lim

Error signal generation and dead band

The block provides the standard function Error signal generation and dead band (Page 188).

Delay alarm for control deviation at setpoint step changes

The block provides the standard function Delay alarm for control deviation at setpoint step changes (Page 186)

Limit monitoring of error signal

The block provides the standard function Limit monitoring of setpoint, manipulated variable and control deviation (Page 95). The monitoring of error signal works with the alarm delay type Two time values per limit pair (Page 197). With the Feature Bit Separate delay times for each alarm (Page 169), the alarm delay type Two time values for each individual limit (Page 198) can be activated.

Inverting control direction

The block provides the standard function Inverting control direction (Page 188).

Physical standardization of setpoint, manipulated variable and process value

Controller gain Gain is entered either using a physical variable or as standardized value.

Gain as a physical variable [MV_Unit / PV_Unit]:

The standardized variables retain their default values:

- NormPV.High = 100 and NormPV.Low = 0
- NormMV.High = 100 and NormMV.Low = 0

The effective gain is:

GainEff = Gain

Entering a standardized Gain (dimensionless):

Change the standardized variables to the actual range of the process values and manipulated variables.

- Internal and external setpoints; the process value and corresponding parameters are entered according to the physical measuring range of the process value.
- The manual value, the tracking value of the manipulated variable, disturbance variable feedforward and the corresponding parameters are set according to the physical measuring range of the manipulated variable.

The effective gain is:

`GainEff` = (`NormMV.High` - `NormMV.Low`) / (`NormPV.High` - `NormPV.Low`) · `Gain`

Selecting a unit of measure

The block provides the standard function Selecting a unit of measure (Page 207).

PID algorithm

The manipulated variable is generated in automatic mode according to the following algorithm:

$$MV = \quad = G_{ainEff}\left[1+\frac{T_D s}{\frac{T_D}{D_{iffGain}}s+1}\right](PV-SP)+R_{eset}\left(\frac{1}{T_I s+1}\right)$$

Where:

s = complex number of the Laplace transformation

This formula describes a standard application where P, I and D component is activated

Unlike with PIDConL, the `Gain` gain factor is not applied to the I component.

The D component delay is derived from `TD` / `DiffGain`.

- The P component can be deactivated by `PropSel` = 0 .
- The I component can be deactivated by `IntSel` = 0 . The `MV_Offset` input parameter can then be used to add a constant value to the manipulated variable. Select this value such that the remaining control deviation equals zero at least at the control loop's typical operating point.
- The D component can be deactivated by `TD` = 0 or `DiffSel` = `1`.

Note

The formula describes a standard application where P, I and D components are activated and the D component is in the feedback circuit, while the P component is formed by the error signal (`PropSel` = `1`, `TI` <> `0` and `IntSel` = `1`, `TD` <> `0` and `DiffSel` = `1`, `DiffToFbk` = `1`, and `PropFacSP` = `1`).

If you set the `Feature` Bit Enabling bumpless change to the proportional gain, derivative time and amplification of the differentiator (Page 171) to 1, changes in the proportional gain `Gain` are carried out in a bumpless manner in "automatic" mode by converting the internal reset.

If a GainSched block is linked to the controller, you have to make the parameter settings for the `Feature` Bit Enabling bumpless change to the proportional gain, derivative time and amplification of the differentiator (Page 171) with 0.

The PID core algorithm is implemented in the PIDKernR function block which in turn calls a series of auxiliary functions for the 64-bit arithmetic. Every edge transition on the `InitPid` input parameter forces initialization equations to be carried out by PIDKernR. In so doing, the internal reset is calculated such that the `MV` output does not suffer a P step.

During initialization in the automatic mode with a large control deviation, it may happen that the integral action is set far outside the manipulated variable limits. See `Feature` bit With accelerated return of the integral action from the manipulated variable limit (Page 179).

Note

The meaning of the controller parameters with PIDConR is different to that with the other PID controllers for all controller parameter settings with a D component. If you want to transfer parameter values from one kind of controller to another, they have to be converted using the following formulas. The calculation is based on an ideal transfer function without a delay in the D component, and all three control channels are applied to the `ER` = `PV` - `SP` control deviation.

Parallel controller structure (for example, PIDConL):

$$MV = G_{ainEff}\left(1+\frac{1}{T_I s}+T_D s\right)ER$$

The parameters of the serial-interactive controller structure (PIDConR):

$$MV = G'_{ainEff}\left(1+T'_D s\right)\left(1+\frac{1}{T'_I s}\right)ER$$

These parameters are marked using a speech mark. Both controllers calculate the same manipulated variable if the parameter values of the serial-interactive structure are determined by the following substitution:

$$G'_{ainEff} = \alpha G_{ainEff}\ ,\quad T'_I = \alpha T_I\ ,\quad T'_D = \frac{1}{\alpha}T_D$$

where α is the conversion factor:

$$\alpha = \frac{1}{2}\pm\sqrt{\frac{1}{4}-\frac{T_D}{T_I}}$$

The argument of the square root is negative and the conversion is impossible if:

$$\frac{T_D}{T_I} > \frac{1}{4}$$

In such cases, the output of PIDConL cannot be exactly replicated by PIDConR. In all other cases, there are two solutions for α and therefore two parameter sets that will generate the same control action of the ideal controller structures. The conversion in the other direction, from serial-interactive to parallel controller structure is always possible by:

$$\alpha = \frac{T_I'}{T_I' + T_D'}$$

You can see that the conversion factor only equals one if $T_D = 0$.

In PID-Tuner, the conversion for ideal controller structures is performed automatically before downloading of parameters to the PIDConR function block. The first of the two possible solutions (plus square root) is used. If the argument of the square root is negative, it will be set to zero, which implies that PIDConR control performance may deteriorate slightly.

Structure segmentation at controllers

The block provides the standard function Structure segmentation at controllers (Page 194).

Use output point for the manipulated variable calculation (external reset)

For the `ExtResOn` = 0 input parameter (default setting), the starting point for the manipulated variable calculation within the block is taken from manipulated variable `MV`. In other words, this is the manipulated variable of the last sampling step. If `ExtResOn` = 1, the `ExtReset` start parameter is used. This is used in particular for networked control structures, such as cascade or transfer control.

Anti-windup

A controller with incremental algorithm (external reset) inherently has an anti-windup reaction because the starting point for the manipulated variable calculation (external reset value) is limited provided it is taken internally from manipulated variable `MV` or is taken from another signal source with limitation. If the starting point for the manipulated variable calculation (external reset value) is at the limit (`MV_HiLim` or `MV_LoLim`), the I component is automatically frozen.

Feedforwarding and limiting disturbance variables

The block provides the standard function Feedforwarding and limiting disturbance variables (Page 193).

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

- Signal status for the process value `PV_Out`:
  The signal status of the output parameter `PV_Out` always corresponds to the signal status of input parameter `PV` or, if the block is in simulation mode, 16#60.
- Signal status for the setpoint value `SP`:
  The signal status of the SP output parameter is always equivalent to the signal status of input parameter `SP_Ext` or `SP_Int`, depending on how the setpoint is specified. If the internal setpoint `SP_Int` is used, the signal status is always output as 16#80.
- Signal status of the error signal `ER`:
  The signal status of output parameter `ER` is obtained from the worst signal status of the two output parameters `PV_Out` and `SP` and is output. The signal status `16#60` (external simulation) is suppressed because the block acts as a sink with external simulation.
- Signal status for the manipulated variable `MV`:
  The signal status of output parameter `MV` is obtained in "automatic mode" or in "program mode" with default setpoint from the worst signal status of the two parameters `FFwd` and `ER` and is output. In "manual mode", the signal status is output as good. The signal status `16#60` (external simulation) is suppressed because the block acts as a sink with external simulation. In "manual mode", the signal status is output as good.
- Signal status for position feedback `RbkOut`:
  The signal status of `RbkOut` always corresponds to the signal status of input parameter `Rbk` or, if the block is in simulation mode, `16#60`.
- Worst signal status:
  The worst signal status `ST_Worst` for the block is formed from the following parameters:
  - `SP.ST`;
  - `PV_Out.ST`;
  - `FFwd.ST`;
  - `RbkOut.ST`;
  - `MV_ChnST.ST`;

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
6	Ramp rate calculation (Page 178)
8	Separate delay times for each alarm (Page 169)
9	Substitution value is active if the block is in bypass (Page 184)

Bit	Function
11	Gradient limitation with time duration (Page 181)
15	Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165)
16	Neutral position manipulated variable takes effect at startup (Page 165)
17	With accelerated return of the integral action from the manipulated variable limit (Page 179)
19	Enabling program mode (Page 158)
20	Enabling bumpless change to the proportional gain, derivative time and amplification of the differentiator (Page 171)
21	Switching operator controls for external setpoint to visible (Page 143)
22	Update acknowledgment and error status of the message call (Page 159)
23	SP following PV in open loop has no priority over SP_Ext and SP limits (Page 178)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Dead band is temporarily disabled (Page 140)

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)

Operator permissions

The block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode" `AutModOp`
1	1 = Operator can switch to "manual mode" `ManModOp`
2	1 = Operator can switch to "Out of service" mode `OosOp`
3	1 = Operator can switch to "program mode" `AdvCoOn`
4	1 = Operator can switch the setpoint to "external" `SP_ExtOp`
5	1 = Operator can switch the setpoint to "internal" `SP_IntOp`
6	1 = Operator can change the internal setpoint `SP_Int`
7	1 = Operator can change the manual parameter `Man`
8	1 = Operator can change operation high limit of the setpoint `SP_InHiLim`
9	1 = Operator can change operation low limit of the setpoint `SP_InLoLim`
10	1 = Operator can change the operation high limit of the manipulated variable `ManHiLim`
11	1 = Operator can change the operation low limit of the manipulated variable `ManLoLim`

Bit	Function
12	1 = Operator can enable the setpoint's gradient limitation function SP_RateOn
13	1 = Operator can raise the gradient limit SP_UpRaLim of the setpoint
14	1 = Operator can lower the gradient limit SP_DnRaLim of the setpoint
15	1 = Operator can switch between the time value or the gradient value for specifying the ramp SP_RmpModTime
16	1 = Operator can change the ramp time SP_RmpTime
17	1 = Operator can change the target setpoint SP_RmpTarget for the setpoint ramp
18	1 = Operator can enable the setpoint ramp function SP_RmpOn
19	1 = Operator can permit the PID optimization function OptimEn
20	1 = Operator can enable the track setpoint in manual mode function SP_TrkPV
21	1 = Operator can enable the bumpless switchover from external to internal SP_TrkExt
22	1 = Operator can change the gain parameter Gain
23	1 = Operator can change the integral time parameter TI
24	1 = Operator can change the derivative time parameter TD
25	1 = Operator can change the derivative gain parameter DiffGain
26	1 = Operator can change the dead band parameter DeadBand
27	Not used
28	1 = Operator can change the derivative gain parameter ER_AH_DF
29	1 = Operator can change the derivative gain parameter ER_AL_DF
30 - 31	Not used

The block has the following permissions for the OS1Perm parameter:

Bit	Function
0	1 = Operator can change the limit (process value) PV_AH_Lim for the high alarm
1	1 = Operator can change the limit (process value) PV_WH_Lim for the high warning
2	1 = Operator can change the limit (process value) PV_TH_Lim for the high tolerance
3	1 = Operator can change the hysteresis (process value) PV_Hyst
4	1 = Operator can change the limit (process value) PV_TL_Lim for the low tolerance
5	1 = Operator can change the limit (process value) PV_WL_Lim for the low warning
6	1 = Operator can change the limit (process value) PV_AL_Lim for the low alarm
7	1 = Operator can change the limit (error signal) ER_AH_Lim for the high alarm
8	1 = Operator can change the hysteresis (error signal) ER_Hyst
9	1 = Operator can change the limit (error signal) ER_AL_Lim for the low alarm
10	1 = Operator can change the limit (position feedback) RbkWH_Lim for the high warning
11	1 = Operator can change the hysteresis (position feedback) RbkHyst
12	1 = Operator can change the limit (position feedback) RbkWL_Lim for the low warning
13	"Interlock" button is enabled
14	1 = Operator can activate bypass functionality
15	1 = Operator can deactivate bypass functionality
16	1 = Operator can activate the Simulation function SimOn
17	1 = Operator can activate the Release for maintenance function MS_RelOp
18	1 = Operator can activate / deactivate messages via PV_AH_MsgEn
19	1 = Operator can activate / deactivate messages via PV_WH_MsgEn

Bit	Function
20	1 = Operator can activate / deactivate messages via `PV_TH_MsgEn`
21	1 = Operator can activate / deactivate messages via `PV_TL_MsgEn`
22	1 = Operator can activate / deactivate messages via `PV_WL_MsgEn`
23	1 = Operator can activate / deactivate messages via `PV_AL_MsgEn`
24	1 = Operator can activate / deactivate messages via `ER_AH_MsgEn`
25	1 = Operator can activate / deactivate messages via `ER_AL_MsgEn`
26	1 = Operator can activate / deactivate messages via `RbkWH_MsgEn`
27	1 = Operator can activate / deactivate messages via `RbkWL_MsgEn`
28	1 = Operator can change the simulation value `SimPV`
29	1 = Operator can change the simulation value `SimRbk`
30	1 = Operator can activate the derivative action to the feedback path `DiffToFbk`
31	1 = Operator can change the proportional action to the feedback path `PropFacSP`

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Interlocks

This block provides the following interlocks:

- Interlock without reset ("Interlock")

You can find additional information on this in the section Interlocks (Page 99).

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Release for maintenance

The block provides the standard function Release for maintenance (Page 64).

Generating instance-specific messages

The block provides the standard function Generating instance-specific messages (Page 200) without the time stamp function in the I/O.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205).

Instance-specific text can be configured for the following parameters:

- AutModOp
- ManModOp
- AdvCoOn
- OosOp
- SP_ExtOp
- SP_IntOp

Specifying the display area for process and setpoint values as well as operations

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

See also

PIDConR I/Os (Page 818)

PIDConR messaging (Page 815)

PIDConR error handling (Page 814)

Program mode for controllers (Page 78)

EventTs functions (Page 1634)

5.9.4 PIDConR error handling

Error handling of PIDConR

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` output parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
30	The value of `PV` can no longer be displayed in the REAL number field.
31	The value of `SP_Ext` can no longer be displayed in the REAL number field.
32	The value of `FFwd` can no longer be displayed in the REAL number field.
33	The value of `MV_Trk` can no longer be displayed in the REAL number field.
34	The value of `MV_Forced` can no longer be displayed in the REAL number field.
35	The value of `Rbk` can no longer be displayed in the REAL number field.
36	The value of `MV` can no longer be displayed in the REAL number field.
59	= 1 "Gain is negative"
60	|`TI`| < `SampleTime` / 2
61	|`TD`| < `SampleTime`
62	`DiffGain` < 1 or `DiffGain` > 10
63	`TD` / `DiffGain` < `SampleTime` / 2
64	`PropFacSP < 0` or `PropFacSP > 1`
66	`NormPV_High` = `NormPV_Low`

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

PIDConR block diagram (Page 834)

PIDConR I/Os (Page 818)

PIDConR messaging (Page 815)

PIDConR functions (Page 800)

PIDConR modes (Page 798)
Description of PIDConR (Page 792)

5.9.5 PIDConR messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
`MsgEvId2`	SIG 6	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a process control fault is triggered (`MsgEvId2`, SIG 6).

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	Alarm - high	$$BlockComment$$ PV - high alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ PV - high warning limit violated
	SIG 3	Tolerance - high	$$BlockComment$$ PV - high tolerance limit violated
	SIG 4	Tolerance - low	$$BlockComment$$ PV - low tolerance limit violated
	SIG 5	Warning - low	$$BlockComment$$ PV - low warning limit violated
	SIG 6	Alarm - low	$$BlockComment$$ PV - low alarm limit violated
	SIG 7	Alarm - high	$$BlockComment$$ ER - high alarm limit violated
	SIG 8	Alarm - low	$$BlockComment$$ ER - low alarm limit violated
MsgEvId2	SIG 7	Warning - high	$$BlockComment$$ Rbk - high warning limit violated
	SIG 8	Warning - low	$$BlockComment$$ Rbk - low warning limit violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You can use up to four instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 1	AS process control message - fault	$$BlockComment$$ External message 1 Status 16#@5%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ External message 2 Status 16#@6%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ External message 3 Status 16#@7%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 4 Status 16#@8%x@

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance **MsgEvId1**

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	Process value PV_Out
5	Control deviation ER
6	ExtVa106
7	ExtVa107
8	Not allocated
9	Not allocated
10	Not allocated

The associated values 6 ... 7 are allocated to the parameters ExtVa106 ... ExtVa107 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

Associated values for message instance **MsgEvId2**

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	Position feedback Rbk
5	Signal status ExtMsg1
6	Signal status ExtMsg2
7	Signal status ExtMsg3
8	Signal status ExtMsg4
9	ExtVa209
10	ExtVa210

The associated values 9 ... 10 are allocated to the parameters ExtVa209 ... ExtVa210 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

PIDConR block diagram (Page 834)

PIDConR I/Os (Page 818)

PIDConR error handling (Page 814)

PIDConR functions (Page 800)

PIDConR modes (Page 798)

Description of PIDConR (Page 792)

5.9.6 PIDConR I/Os

I/Os of PIDConR

Input parameters

Parameter	Description	Type	Default
AdvCoEn	1 = Enable "program mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoOn*	1 = Enable "program mode" via faceplate	BOOL	0
AdvCoModSP	Type of "program mode": 1 = Setpoint specification 0 = Manipulated variable specification	BOOL	1
AdvCoMstrOn	1 = Enable (0-1) or disable (1-0) "program mode" via edge transition	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoMV	Specified value from the external program	REAL	0.0
AutExtSet*	1 = Activate "automatic" mode with external setpoint by interconnection (SP = SP_Ext)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutIntSet*	1 = Activate "automatic" mode with internal setpoint by interconnection (SP = SP_Op). AutIntSet has higher priority thanAutExtSet.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic" mode via operator	BOOL	0
BatchEn	1 = Enable allocation for batch control	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Interlock bypass during simulation	BOOL	0
BypLiOp	1 = Bypass commands via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypLock	1 = Bypass activation or deactivation is locked for operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
BypPV	Substitution value if block is in bypass	REAL	0.0
BypPVLi	1 = Select bypass PV (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypPVOp	1 = Select bypass PV (via operator)	BOOL	0
CPI_In	Input for control performance index, which is calculated by the assigned ConPerMon block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DeadBand	Width of dead band	REAL	0.0
DiffGain	Gain of differentiator [1..10] DiffGain = TD / (delay time of D component)	STRUCT • Value: REAL • ST: BYTE	- • 5.0 • 16#80
DiffSel	1 = D component activated	BOOL	1
DiffToFbk*	1 = D component is placed in the feedback	BOOL	1
EN	1 = Called block will be processed	BOOL	1
ER_A_DC*	Delay time for incoming ER high/low or only low alarms [s]	REAL	0.0
ER_AH_DC*	Delay time for incoming ER high alarms [s]	REAL	0.0
ER_AH_DFac*	Delay factor at positive setpoint step changes for incoming alarms at the error signal monitoring ER_AH_Lim	REAL	0.0
ER_A_DG*	Delay time for outgoing ER high/low or only low alarms [s]	REAL	0.0
ER_AH_DG*	Delay time for outgoing ER high alarms [s]	REAL	0.0
ER_AH_En	1 = Activate alarm (high) for error signal monitoring	BOOL	1
ER_AH_Lim	Alarm limit (high) for error signal monitoring	REAL	100.0
ER_AH_MsgEn	1 = Activate messages for alarm (high) for error signal monitoring	BOOL	1
ER_AL_DFac*	Delay factor at negative setpoint step changes for incoming alarms at the error signal monitoring ER_AL_Lim	REAL	0.0
ER_AL_En	1 = Activate alarm (low) for error signal monitoring	BOOL	1
ER_AL_Lim	Alarm limit (low) for error signal monitoring	REAL	-100.0
ER_AL_MsgEn	1 = Activate messages for alarm (low) for error signal monitoring	BOOL	1

Parameter	Description	Type	Default
ER_Hyst	Alarm hysteresis for error signal	REAL	1.0
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	1 = Binary input for freely selectable message 1 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	1 = Binary input for freely selectable message 2 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	1 = Binary input for freely selectable message 3 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg4	1 = Binary input for freely selectable message 4 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtReset	Value to which a reset is made if ExtRstOn = 1.	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ExtRstOn	1 = Reset externally	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa209	Associated value 9 for messages (MsgEvID2)	ANY	
ExtVa210	Associated value 10 for messages (MsgEvID2)	ANY	
Feature	I/O for additional functions (Page 800)	STRUCT • Bit 0: BOOL • ... • Bit 20: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0

Parameter	Description	Type	Default
Feature2	I/O for additional functions (Page 800)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
FFwd*	Input for additive disturbance variable activation	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
FFwdHiLim	Limit (high) for additive disturbance variable activation	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
FFwdLoLim	Limit (low) for additive disturbance variable activation	STRUCT • Value: REAL • ST: BYTE	- • -100.0 • 16#80
Gain	Proportional gain Gain.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#FF
InitPid*	InitPid edge transitions result in the PID algorithm's initialization equations being carried out. Is used, for example, for SP changes without MV jumps	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is in effect	BOOL	1
Intlock	0 = Interlock without reset is in effect Once the interlock condition has cleared, the block can be operated without reset 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
IntSel	1 = I component activated	BOOL	1
Man*	Manual specification for the manipulated variable	REAL	0.0
ManHiLim	Limit (high) for manual parameter Man	REAL	100.0
ManLoLim	Limit (low) for manual parameter Man	REAL	0.0
ManModOp*	1 = "Manual" mode via OS operator	BOOL	1
ManSet*	1 = Activate "manual" mode via interconnection. ManSet has higher priority thanAutIntSet.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MsgEvID1	Message number (assigned automatically)	DWORD	16#00000000
MsgEvID2	Message number (assigned automatically)	DWORD	16#00000000

Parameter	Description	Type	Default
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_Bump*	Default value for controller output MV if MV_BumpOn = 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_BumpOn	1 = Set controller output MV:= MV_Bump without (!) taking the controller into "manual" mode.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_ChnST	Signal status of output channel for MV Should be connected to an output channel block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
MV_Close	1 = Close adjustment valve by interconnection, i.e. MV:= MV_LoLim MV_Close has higher priority than MV_Open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_Forced*	Forced manipulated variable that is not limited and assumes top priority	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_ForOn	1 = Output forced manipulated variable MV_Forced unlimited at output MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_HiLim	Limit (high) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
MV_Mean	Mean value of the MV in the time window	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#00
MV_LoLim	Limit (low) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_Offset	Manipulated variable for ER=0, operating point for controller with deactivated I component	REAL	0.0
MV_Open	1 = Open adjustment valve by interconnection, i.e. MV:= MV_HiLim MV_Open has higher priority than MV_TrkOn	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_OpScale	OS display range for manipulated variable MV	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

Parameter	Description	Type	Default
MV_SafePos	Manipulated variable neutral position	STRUCT • Value: REAL • ST: BYTE	- • 0 • 16#80
MV_Trk*	Tracking value for the manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_TrkOn	1 = Tracking of manipulated variable MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_Unit	Unit of measure for manipulated variable	INT	1342
NegGain	0 = Effective proportional gain GainEff is positive 1 = Effective proportional gain GainEff is negative	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NormMV	Manipulated variable range (MV) for standardizing the proportional gain (GAIN)	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
NormPV*	Process value range (PV) for standardizing the proportional gain (GAIN)	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
Occupied	Occupied by batch control	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OptimEn*	1 = Enable optimization of PID parameters by PID tuner	BOOL	0
OptimOcc*	1 = Optimization running	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 800)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OS1Perm	I/O for operator permissions (Page 800)	STRUCT • Bit 0: BOOL • Bit 18: BOOL • Bit 19:BOOL • Bit 31: BOOL	- • 1 • 1 • 1 • 1

Parameter	Description	Type	Default
PropFacSP	Applying the P component to the feedback [0..1]. 0 = P component fully in feedback	REAL	1.0
PropSel*	1 = Activate P component	BOOL	1
PV*	Process value (controlled variable)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Unit	Unit of measure for process value	INT	1001
PV_AH_DC*	Delay time for incoming PV high alarms [s]	REAL	0.0
PV_AH_DG*	Delay time for outgoing PV high alarms [s]	REAL	0.0
PV_AH_En	1 = Enable PV alarm limit (high)	BOOL	1
PV_AH_Lim	Limit PV alarm (high)	REAL	95.0
PV_AH_MsgEn	1 = Enable PV alarm (high) message	BOOL	1
PV_AL_En	1 = Enable PV alarm limit (low)	BOOL	1
PV_AL_Lim	PV alarm limit (low)	REAL	5.0
PV_AL_MsgEn	1 = Enable PV alarm (low) message	BOOL	1
PV_AL_DC*	Delay time for incoming PVlow alarms [s]	REAL	0.0
PV_AL_DG*	Delay time for outgoing PV low alarms [s]	REAL	0.0
PV_Hyst	Hysteresis for PV alarm, warning and tolerance limits	REAL	1.0
PV_OpScale	Limit for scale in PV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
PV_TH_DC*	Delay time for incoming PV high tolerance messages [s]	REAL	0.0
PV_TH_DG*	Delay time for outgoing PV high tolerance messages [s]	REAL	0.0
PV_TH_En	1 = Enable PV tolerance limit (high)	BOOL	0
PV_TH_Lim	Limit PV tolerance message (high)	REAL	85.0
PV_TH_MsgEn	1 = Enable message for PV tolerance message (high)	BOOL	1
PV_TL_DC*	Delay time for incoming PV low tolerance messages [s]	REAL	0.0
PV_TL_DG*	Delay time for outgoing PV low tolerance messages [s]	REAL	0.0
PV_TL_En	1 = Enable PV tolerance limit (low)	BOOL	0
PV_TL_Lim	Limit PV tolerance message (low)	REAL	15.0
PV_TL_MsgEn	1 = Activate message for PV tolerance message (low)	BOOL	1
PV_WH_DC*	Delay time for incoming PV high warnings [s]	REAL	0.0

Parameter	Description	Type	Default
PV_WH_DG*	Delay time for outgoing PV high warnings [s]	REAL	0.0
PV_WH_En	1 = Enable PV warning limit (high)	BOOL	1
PV_WH_Lim	Limit PV warning (high)	REAL	90.0
PV_WH_MsgEn	1 = Enable PV warning (high) message	BOOL	1
PV_WL_DC*	Delay time for incoming PV low warnings [s]	REAL	0.0
PV_WL_DG*	Delay time for outgoing PV low warnings [s]	REAL	0.0
PV_WL_En	1 = Enable PV warning limit (low)	BOOL	1
PV_WL_Lim	Limit PV warning (low)	REAL	10.0
PV_WL_MsgEn	1 = Enable PV warning (low) message	BOOL	1
Rbk*	Position feedback for display on OS	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
RbkHyst	Alarm hysteresis for position feedback	REAL	1.0
RbkW_DC*	Delay time for incoming Rbk warnings [s]	REAL	0.0
RbkW_DG*	Delay time for outgoing Rbk warnings [s]	REAL	0.0
RbkWH_En	1 = Enable warning (high) for position feedback	BOOL	1
RbkWH_Lim	Limit for position feedback of warning (high)	REAL	100.0
RbkWH_MsgEn	1 = Enable messages for warning (high) for position feedback	BOOL	1
RbkWL_En	1 = Enable warning (low) for position feedback	BOOL	1
RbkWL_Lim	Limit for position feedback of warning (low)	REAL	0.0
RbkWL_MsgEn	1 = Enable messages for warning (low) for position feedback	BOOL	1
RefStdDevIn	Reference value of PV standard deviation (sigma) in defined "good" state of control loop	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
RstBypLi	1 = Reset bypass PV (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstBypOp	1 = Reset bypass PV (via operator)	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SafePos	1 = Neutral position (Page 48) for controller manipulated variable is ManHiLim 0 = Neutral position for controller manipulated variable is ManLoLim	BOOL	0

Parameter	Description	Type	Default
SafePos2	Neutral position for controller manipulated variable: 0 = SafePos is valid 1 = Neutral position is MV_SafePos 2 = Neutral position is last manipulated variable (stop)	INT	0
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
SettliFactor	Factor to increase the settling time to adjust the dead band	REAL	2.0
SettliTime	Settling time [s] of the control loop determined by the ConPerMon block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#78
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimPV*	Process value used for SimOn = 1	REAL	0.0
SimPV_Li	Process value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimRbk*	Position feedback used for SimOn = 1	REAL	0.0
SimRbkLi	Position feedback used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_DnRaLim	Limit (low) for the gradient of the setpoint [SP_Unit/s]	REAL	100.0
SP_ExHiLim	Limit (high) for external setpoint	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP_ExLoLim	Limit (low) for external setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
SP_Ext*	external setpoint - (to interconnection)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtOp*	1 = Select external setpoint (via operator)	BOOL	0
SP_InHiLim	Limit (high) of internal setpoint	REAL	100.0
SP_InLoLim	Limit (low) of internal setpoint	REAL	0.0
SP_Int*	Internal setpoint for operation	REAL	0.0
SP_IntOp*	1 = Select internal setpoint (via operator)	BOOL	0
SP_Load*	Defined setpoint, if SP_LoadOn = 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_LoadOn*	1 = Take controller into "automatic mode" with internal setpoint and set setpoint at SP:= SP_Load. This kind of setpoint specification has maximum priority.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_RateOn*	1 = Activate limitation of setpoint gradients	BOOL	0
SP_RmpModTime	1 = Use time (SP_RmpTime) for setpoint ramp 0 = Use gradient	BOOL	0
SP_RmpOn*	1 = Activate setpoint ramp to target setpoint SP_RmpTarget	BOOL	0
SP_RmpTarget*	Target setpoint for setpoint ramp	REAL	0.0
SP_RmpTime*	Time for setpoint ramp [s] from current SP up to SP_RmpTarget	REAL	0.0
SP_TrkExt	1 = Bumpless switchover from external to internal setpoint active	BOOL	1
SP_TrkPV	1 = Setpoint follows PV in "manual mode" and with tracking	BOOL	0
SP_UpRaLim	Gradient limit (high) for the setpoint [SP_Unit/s]	REAL	100.0
StepNo	Batch step number	DWORD	16#00000000
TD	Derivative time [s] TD.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
TI	Integral time [s] TI.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#FF

Parameter	Description	Type	Default
TimeFactor	Time unit: 0 = Seconds 1 = Minutes 2 = Hours	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AdvCoAct	1 = "Program mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoRdy	1 = "Program mode" available	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutAct	1 = "Automatic mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypassAct	1 = Bypass is activated in this block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CascaCut	Cascade connection: 1 = Control chain from primary to secondary controller is interrupted	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DynDeadBand	Dynamic dead band	REAL	0.0
PhaseDeadBand	Phase for the dynamic adjustment of the dead band 0: Dead band enabled 1: Dead band disabled 2: Settling time	INT	0
ENO	1 = Block algorithm completed without errors	BOOL	0
ER	Error signal	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ER_A_DCOut	Effective delay time [s] for incoming alarms at the error signal monitoring	REAL	0.0

Parameter	Description	Type	Default
ER_AH_Act	1 = Alarm limit (high) for control deviation violated. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ER_AL_Act	1 = Alarm limit (low) for control deviation violated. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum*	Output of pending error number. For error numbers that can be output by this block, see PIDConR error handling (Page 814)	INT	-1
FFwdHiAct	1 = Limit (high) for additive disturbance variable activation violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FFwdLoAct	1 = Limit (low) for additive disturbance variable activation violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GainEff	Effective proportional gain depends on NegGain, Gain, NormPV, and NormMV	REAL	1.0
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock is in effect	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LoopClosed	1 = Control loop closed 0 = Control loop open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
ManHiOut	Limit (high) for "manual mode", corresponds to input parameter ManHiLim	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
ManLoOut	Limit (low) for "manual mode", corresponds to input parameter ManLoLim	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgAckn2	Message acknowledgement status 2 (output ACK_STATE of second ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgErr2	Alarm error 2 (output ERROR of second ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
MsgStat2	Message status 2 (output STATUS of second ALARM_8P)	WORD	16#0000
MV	Manipulated variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_HiAct	1 = Limit (high) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_LoAct	1 = Limit (low) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_UnitOut	Unit of measure for manipulated variable, for interconnecting to the MV_Unit input parameter of the ConPerMon block	INT	0
OosAct	1 = Block is "out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS1PermOut	Display of OS1Perm	DWORD	16#FFFFFFFF
OS1PermLog	Display of OS1Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF

Parameter	Description	Type	Default
PV_AH_Act	1 = PV alarm (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AH_Out	PV - High alarm limit output	REAL	0.0
PV_AL_Act	1 = PV alarm (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AL_Out	PV - Low alarm limit output	REAL	0.0
PV_HysOut	PV - Alarm hysteresis output	REAL	0.0
PV_Out	Output for process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_TH_Act	1 = PV tolerance message (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_TH_Out	PV - High tolerance limit output	REAL	0.0
PV_TL_Act	1 = PV tolerance message (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_TL_Out	PV - Low tolerance limit output	REAL	0.0
PV_ToleHi	Limit (high) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_ToleLo	Limit (low) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_UnitOut	Unit of measure for process value, for interconnecting to the PV_Unit input parameter of the ConPerMon block	INT	0
PV_WH_Act	1 = PV warning (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
PV_WH_Out	PV - High warning limit output	REAL	0.0
PV_WL_Act	1 = PV warning (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_WL_Out	PV - Low warning limit output	REAL	0.0
RbkOut	Output for position feedback	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RbkWH_Act	1 = Warning (high) for position feedback active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkWL_Act	1 = Warning (low) for position feedback active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SettlingTimer	Settling time for a closed control loop	REAL	0.0
SP	Setpoint used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_DnRaAct	1 = Negative gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExHiAct	1 = Limit (high) for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExLoAct	1 = Limit (low) for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtAct	1 = External setpoint is active 0 = Internal setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
SP_ExtOut	External setpoint, corresponds to input parameter SP_Ext	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_InHiOut	Limit (high) for SP_Int corresponds to input parameter SP_InHiLim	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP_InLoOut	Limit (low) for SP_Int corresponds to input parameter SP_InLoLim	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_RateTarget	Target setpoint for the gradient limitation	REAL	0.0
SP_UpRaAct	Positive gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_RemRT	Remaining ramp time of the setpoint	REAL	0.0
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 792)	DWORD	16#00000000
Status2	Status word 2 (Page 792)	DWORD	16#00000000
Status3	Status word 2 (Page 792)	DWORD	16#00000000
SumMsgAct	1 = Active hardware interrupt	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

* Values can be written back to these inputs during processing of the block by the block algorithm.

Calculation of the output parameter ER_A_DCOut

ER_A_DC is assigned by default to the output before a setpoint change.

ER_A_DCOut = ER_A_DC

In the case of a setpoint change in the positive direction during automatic mode, the output is calculated as follows:

ER_A_DCOut = Maximum (ER_A_DC, ER_AH_DFac * Setpoint difference)

In the case of a setpoint change in the negative direction during automatic mode, the output is calculated as follows:

ER_A_DCOut = Maximum (ER_A_DC, -1*ER_AH_DFac * Setpoint difference)

When the control circuit has stabilized again, meaning

(ER_AL_Lim + ER_Hyst) <= ER <= (ER_AH_Lim – ER_Hyst)

and the delay time for outgoing alarms ER_A_DG has expired, the output is reset again to ER_A_DC: ER_A_DCOut = ER_A_DC

Activating and deactivating the function:

The function is deactivated (default) when the following applies: ER_AH_DFac = 0.0 and ER_AL_DFac = 0.0

See also

PIDConR block diagram (Page 834)

PIDConR messaging (Page 815)

PIDConR modes (Page 798)

5.9.7 PIDConR block diagram

PIDConRblock diagram

A block diagram is not provided for this block.

See also

PIDConR I/Os (Page 818)

PIDConR messaging (Page 815)

PIDConR error handling (Page 814)

PIDConR functions (Page 800)

PIDConR modes (Page 798)

Description of PIDConR (Page 792)

5.9.8 Operator control and monitoring

5.9.8.1 PIDConR views

Views of the PIDConR block

This block has the same views as the PIDConL block. Refer to the section Operator control and monitoring (Page 760) of the PIDConL block.

5.10 PIDStepL - Step controller

5.10.1 Description of PIDStepL

Object name (type + number) and family

Type + number: FB 1878

Family: Control

Area of application for PIDStepL

The block is used for the following applications:

- Fixed setpoint control
- Cascade control
- Ratio control
- Split-range control
- Smith predictor closed-loop control
- Override control (override)

Note on restricting the area of application: The block is not intended for pulse-duration modulation. Examples of applications: Temperature control with electric heater, which can switched on an off via a (semiconductor) relay. The heating power required by the controller, for example 80%, is output in the form of binary pulses, whereby the duration of the on pulses in this case is four times as long as the pause duration. Such control is built with a combination of a continuous controller (such as PIDConL) and the pulse generator module PULSEGEN from the CFC library ELEM_400.

How it works

The block is a PID step controller with binary output signals (manipulated variable signals). It is used to control integral action actuators (e.g. valves driven by motors).

The block functions following the PID algorithm with a delayed D component and an integrator with double precision.

The step controller can function both with and without a position feedback for the valve position.

The block is suitable for controlling sluggish control loops, for example, for temperatures and filling levels, and high-speed control loops, for example, for flow rates and speed. For a given CPU, a compromise has to be made between the number of controllers and the frequency with which the individual controllers have to be processed. The faster the modulated control loops are, i.e. the more frequently the manipulated variables have to be calculated per time unit, the lower the number of controllers that can be installed.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the control loop monitoring to work as planned in the trend view of the controller faceplates, the

`S7_xarchive:='Value, shortterm;'`

attributes in the process tag types for control loops at the controller function block must be set for the following tags:

- Input parameters:
  - `CPI_In`
- Output parameters
  - `MV`
  - `MV_HiAct`
  - `MV_LoAct`
  - `LoopClosed`
  - `SP`
  - `PV_Out`
  - `PV_ToleHi`
  - `PV_ToleLo`

For the PIDStepL block, the Advanced Process Library contains templates for process tag types as examples and there is a example project (APL_Example_xx, xx designates the language variant) containing different application cases for this block.

Several process tag types are simulated in the example project and serve to explain the block function.

Examples of process tag types:

- Step controller with direct access to the actuator and without position feedback (StepControlDirect) (Page 2307)
- Step controller with assigned actuator block and position feedback (StepControlActor) (Page 2308)

Startup characteristics

Use the Feature Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section PIDStepL I/Os (Page 855).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutAct.Value`
6	Not used
7	`ManAct.Value`
8	`SP_ExtAct.Value`
9	`MV_ForOn.Value`
10	`MV_TrkOn.Value`
11	NOT `RbkClosed.Value`
12	`Open.Value`
13	`Close.Value`
14	`Stop.Value`
15	`FbkOpened.Value`
16	`FbkClosed.Value`
17	`SimLiOp.Value`
18	`SimOn` AND `ManAct`
19	`AdvCoAct`
20	1 = Input parameter `Rbk` is not interconnected (`RbkOut.ST` = 16#FF)
21	`NegGain`
22	1 = SP ramp active
23	`OptimEn`
24	`OptimOcc`
25	Not used
26	Display of `BypassAct.Value` in faceplate (display and operator controls) and block icon
27	Not used
28	1 = Input parameter `OpenChnST` is interconnected
29	1 = Input parameter `CloseChnST` is interconnected
30	1 = Input parameter `StopChnST` is interconnected
31	`WithRbk` = 1 (Step controller with position feedback)

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	`PV_AH_Act.Value`
2	`PV_WH_Act.Value`

Status bit	Parameter
3	PV_TH_Act.Value
4	PV_TL_Act.Value
5	PV_WL_Act.Value
6	PV_AL_Act.Value
7	PV_AH_En
8	PV_WH_En
9	PV_TH_En
10	PV_TL_En
11	PV_WL_En
12	PV_AL_En
13	PV_AH_MsgEn
14	PV_WH_MsgEn
15	PV_TH_MsgEn
16	PV_TL_MsgEn
17	PV_WL_MsgEn
18	PV_AL_MsgEn
19	ER_AH_Act.Value
20	ER_AL_Act.Value
21	ER_AH_En
22	ER_AL_En
23	ER_AH_MsgEn
24	ER_AL_MsgEn
25	RbkWH_Act.Value
26	RbkWL_Act.Value
27	RbkWH_En
28	RbkWL_En
29	RbkWH_MsgEn
30	RbkWL_MsgEn
31	MS_RelOp

Status word allocation for `Status3` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via EventTsIn
1	Effective signal 2 of the message block connected via EventTsIn
2	Effective signal 3 of the message block connected via EventTsIn
3	Effective signal 4 of the message block connected via EventTsIn
4	Effective signal 5 of the message block connected via EventTsIn
5	Effective signal 6 of the message block connected via EventTsIn
6	Effective signal 7 of the message block connected via EventTsIn
7	Effective signal 8 of the message block connected via EventTsIn
8	Delay of the PV_AH_Lim message

Status bit	Parameter
9	Delay of the PV_WH_Lim message
10	Delay of the PV_TH_Lim message
11	Delay of the PV_TL_Lim message
12	Delay of the PV_WL_Lim message
13	Delay of the PV_AL_Lim message
14	Delay of the ER_AH_Lim message
15	Delay of the ER_AL_Lim message
16	Collection of message delays
17	BypassAct.Value
18 - 26	Not used
27	SP_UpRaAct, SP_DnRaAct limits enabled for gradient mode (SP_RateOn = 1)
28	GrpErr.Value
29	RdyToStart.Value
30 - 31	Not used

Status word allocation for Status4 parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via EventTsIn
8 - 31	Not used

See also

PIDStepL functions (Page 840)

PIDStepL messaging (Page 853)

PIDStepL modes (Page 839)

PIDStepL error handling (Page 851)

PIDStepL block diagram (Page 869)

5.10.2 PIDStepL modes

PIDStepL modes

The block can be operated using the following modes:

- Automatic mode (Page 72)
- Manual mode (Page 72)
- Program mode for controllers (Page 78)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and bumpless switchover in the section Manual and automatic mode for control blocks (Page 72).

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for control blocks (Page 72).

"Program mode for controllers"

You can find general information about the "Program mode for controller" in the section Program mode for controllers (Page 78).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of PIDStepL (Page 835)

PIDStepL functions (Page 840)

PIDStepL error handling (Page 851)

PIDStepL messaging (Page 853)

PIDStepL I/Os (Page 855)

PIDStepL block diagram (Page 869)

5.10.3 PIDStepL functions

Functions of PIDStepL

The functions for this block are listed below.

Generation of manipulated variables with position feedback (`WithRbk` = 1)

The manipulated variable `MV` and therefore the actuating signals `Open`, `Close` and `Stop` can be generated as follows:

MV_ForOn	Man-Act	MV_TrkOn	Adv-CoAct AND NOT AdvCo-ModSP	MV =	Limit monitoring	State	Open, Close, Stop
1	-	-	-	`MV_Forced`	none	Forced tracking through constraint without limitation	Depending on `Rbk` and `MV`, the output signals `Open`, `Close` and `Stop` are generated using the algorithm of a positioner.
0	1	-	-	`Man`	`ManHiLim` `ManLoLim`	Manual mode, set by the operator	
0	0	1	-	`MV_Trk`	`MV_HiLim` `MV_LoLim`	Tracking with limitation	
0	0	0	1	`AdvCoMV`	`MV_HiLim` `MV_LoLim`	Higher-level program mode	
0	0	0	0	`P_Part` + `I_Part` + `D_Part` + `FFwd`	`MV_HiLim` `MV_LoLim`	Automatic mode (PID algorithm)	

If the controller is in "out of service" mode, the output parameter `MV` is set to the last valid value in manual mode or the neutral position manipulated variable depending on the `Feature` Bit (Neutral position manipulated variable takes effect at startup (Page 165)). Refer to the Out of service (Page 71) section for more on this.

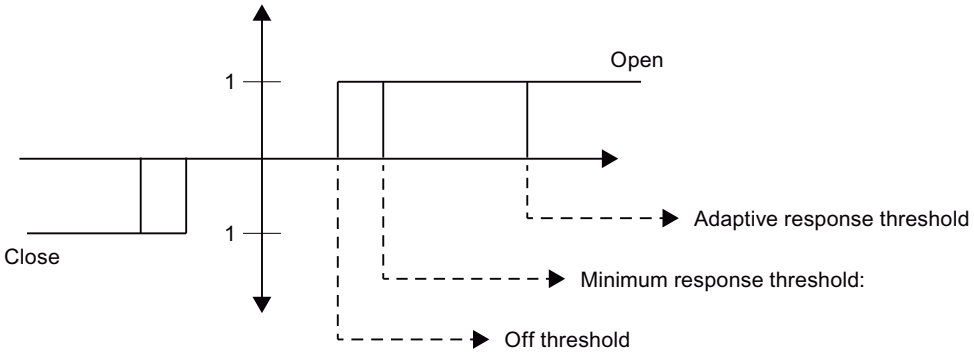
Generation of actuating signal without position feedback (`WithRbk` = 0)

The manipulated variable `MV` can be generated as follows:

`ManAct`	Open, Close, Stop	State
1	The output signals are generated using input signals `OpenOp/Li`, `CloseOp/Li` or `StopOp/Li`	Manual mode, set by the operator
0	The output signals are generated using PD output variables `P_Part`, `D_Part` and `FFwd`	Automatic mode (PID algorithm)

If the controller is in "out of service" mode, the output parameter `MV` is set to the last valid value in manual mode or the neutral position manipulated variable depending on the `Feature` Bit (Neutral position manipulated variable takes effect at startup (Page 165)). Refer to the Out of service (Page 71) section for more on this.

Three-position element with hysteresis and pulse generation



The difference (`MV - Rbk`) for a step controller with position feedback forms the input for the three-position element. The difference for step controllers without position feedback is formed from the output of the PD block and an internal feedback, which is weighted by the common integration of the position signal Open/Close `MotorTime` and the I-component weighted by `TI` (see block diagram).

You can calculate the response threshold of the three position element as follows:

- Minimum response threshold: `100.0 * Max(PulseTime, SampleTime) / MotorTime`
- Off threshold: `0.5 * 110.0 / MotorTime * SampleTime;`

Using "`ThrAdaOn`" you can turn on an adaption of the response threshold to reduce the switching frequency. This only affects the closed control loop (automatic operation without tracking) and is limited to:

- Down: Minimum response threshold
- Up: 10

The currently effective response threshold can be read at the output "`ThesOn`".

The pulse generation after the three position element ensures that `PulseTime` and `BreakTime` are maintained during pulse generation.

Tracking and limiting a manipulated variable

The block provides the standard function Tracking and limiting a manipulated variable (Page 192).

Note

This function is only available to you if position feedback is enabled for the controller (`WithRbk` = 1).

Neutral position

The block provides the standard function Neutral position for motors, valves and controllers (Page 48).

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming group errors:

- `CSF`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

"Actuator active" information

If the parameter `FbkClosed` = 0, this is recognized as "Actuator active". This status can be used to indicate a customized symbol in the process image, for example, and is saved in the status word (see Status word section in Description of PIDStepL (Page 835)).

Limit monitoring of position feedback

The block provides the standard function Limit monitoring of the feedback (Page 94).

Note

This function is only available to you if position feedback is enabled for the controller (`WithRbk` = 1).

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

External/internal setpoint specification

The block provides the standard function Setpoint specification - internal/external (Page 127).

Setpoint limiting for external setpoints

The block provides the standard function Setpoint limiting for external setpoints (Page 192).

Gradient limit of the setpoint

The block provides the standard function Gradient limit of the setpoint (Page 124).

Using setpoint ramp

The block provides the standard function Using setpoint ramp (Page 123).

Tracking setpoint in manual mode

The block provides the standard function Tracking setpoint in manual mode (Page 192).

Simulating signals

The block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Process value (`SimPV, SimPV_Li`)
- Position feedback (`SimRbk, SimRbkLi`)

Bypass function

This block provides the standard function Bypassing signals (Page 107).

Limit monitoring of the process value

The block provides the standard function Limit monitoring of the process value (Page 86).

Error signal generation and dead band

The block provides the standard function Error signal generation and dead band (Page 188).

Limit monitoring of error signal

The block provides the standard function Limit monitoring of setpoint, manipulated variable and control deviation (Page 95).

Inverting control direction

The block provides the standard function Inverting control direction (Page 188).

Physical standardization of setpoint, manipulated variable and process value

Controller gain `Gain` is entered either using a physical variable or as standardized value.

`Gain` as a physical variable:

The standardized variables retain their default values:

- `NormPV.High` = 100 and `NormPV.Low` = 0

The effective gain is:

`GainEff` = `Gain`

Entering a standardized `Gain` (dimensionless):

Change the standardized variables to the actual range of the process values and manipulated variables.

- Internal and external setpoints; the process value and corresponding parameters are entered according to the physical measuring range of the process value.
- The manual parameter, the tracking value of the manipulated variable, feedforward control and the corresponding parameters are entered as a percentage 0...100.

The effective gain is:

`GainEff` = 100.0 / (`NormPV.High` - `NormPV.Low`) · `Gain`

Selecting a unit of measure

The block provides the standard function Selecting a unit of measure (Page 207).

PID algorithm

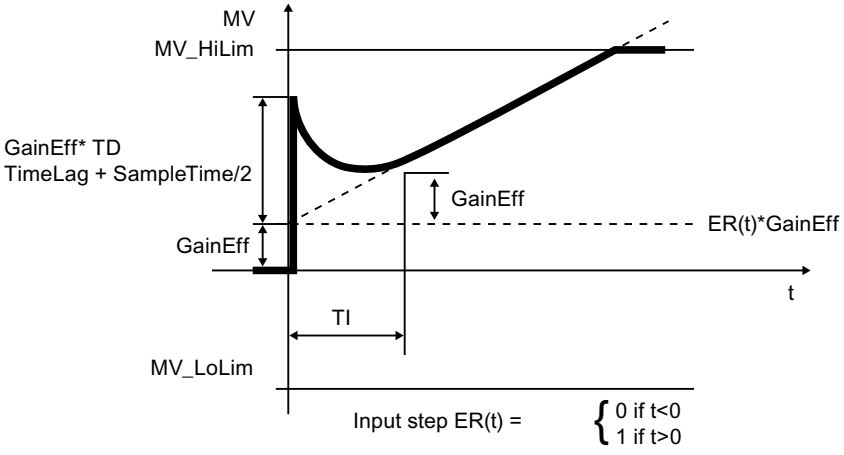
The manipulated variable is generated in automatic mode according to the following algorithm:

`MV` = `GainEff` · (1 + 1 / (`TI` · s) + (`TD` · s) / (1 + `TD` / `DiffGain` · s)) · `ER`

Where:

s = Complex number

The following step response occurs:



Note

The formula describes a standard application where P, I and D components are activated and the P and D components are not in the feedback circuit (`PropSel` = 1, `TI` <> 0, `DiffToFbk` = 0 `and IntSel` = 1, `DiffToFbk` = 0 and `PropFacSP` = 1).

The D component delay is derived from TD / DiffGain.

- The P component is at the P_Part I/O and can be deactivated using PropSel = 0 .
- The I component is displayed at the I_Part I/O and can be deactivated using TI = 0 or IntSel = 0. In deactivated state, I_Part is specified by MV_Offset and added to the manipulated variable. Make a selection for this value so that the remaining control deviation equals zero at the control loop's typical operating point, at least. IntSel is used for temporary deactivation of the I component. The I component is not reactivated until TI <> 0 and IntSel = 1. After the I component is activated, the integrator continues working starting from MV_Offset.
- The D component is displayed at the D_Part I/O and can be deactivated using TD = 0 or DiffSel = 0.

Structure segmentation at controllers

The block provides the standard function Structure segmentation at controllers (Page 194).

Anti-windup

The controller has an anti-windup function. The I component is frozen after the manipulated variable has reached limits (MV_HiLim or MV_LoLim).

Feedforwarding and limiting disturbance variables

The block provides the standard function Feedforwarding and limiting disturbance variables (Page 193).

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

- Signal status for the process value PV_Out:
  The signal status of the output parameter PV_Out always corresponds to the signal status of input parameter PV or, if the block is in simulation mode, 16#60.
- Signal status for the setpoint value SP:
  The signal status of the SP output parameter is always equivalent to the signal status of input parameter SP_Ext or SP_Int, depending on how the setpoint is specified. If the internal setpoint SP_Int is used, the signal status is always output as 16#80.
- Signal status of the error signal ER:
  The signal status of output parameter ER is obtained from the worst signal status of the two output parameters PV_Out and SP and is output.
  The signal status 16#60 (external simulation) is suppressed because the block acts as a sink with external simulation.
- Signal status for position feedback RbkOut:
  The signal status of RbkOut always corresponds to the signal status of input parameter Rbk or, if the block is in simulation mode, 16#60. RbkOut is always 16#80 for step controllers without position feedback.

- Signal status for the manipulated variable `MV`:
  The signal status of output parameter `MV` is obtained in "automatic mode" from the worst signal status of the following parameters and is output:
  `ER.STFFwd.STFbkOpened.STFbkClosed.STRbkOut.ST`The signal status `16#60` (external simulation) is suppressed because the block acts as a sink with external simulation. In "manual mode" and for step controllers without position feedback, the signal status is always output as good.
- Signal status for `Open`, `Close`, `Stop`:
  The signal status is `16#60` when simulation is activated; otherwise, it is always `16#80`.
- Worst signal status:
  The worst signal status `ST_Worst` for the block is formed from the following parameters:
  - `SP.ST`;
  - `PV_Out.ST`;
  - `FFwd.ST`;
  - `FbkOpened.ST`;
  - FbkClosed.ST;
  - `RbkOut.ST`;
  - `OpenChnST.ST`;
  - `CloseChnST.ST`;
  - `StopChnST.ST`;

Note

The `RbkOut.ST` parameter is always 16#80for a step controller without position feedback (`WithFbk` = 0) .

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
4	Setting switch or button mode (Page 166)
6	Ramp rate calculation (Page 178)
9	Substitution value is active if the block is in bypass (Page 184)
11	Gradient limitation with time duration (Page 181)
15	Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165)
16	Neutral position manipulated variable takes effect at startup (Page 165)
18	Disabling bumpless switchover to automatic mode for controllers (Page 172)

Bit	Function
22	Update acknowledgment and error status of the message call (Page 159)
23	SP following PV in open loop has no priority over SP_Ext and SP limits (Page 178)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)

Operator permissions

The block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode" `AutModOp`
1	1 = Operator can switch to "manual mode" `ManModOp`
2	1 = Operator can switch to "Out of service" mode `OosOp`
3	1 = Operator can switch to "program mode" `AdvCoEn`
4	1 = Operator can switch the setpoint to "external" `SP_ExtOp`
5	1 = Operator can switch the setpoint to "internal" `SP_IntOp`
6	1 = Operator can change the internal setpoint `SP_Int`
7	1 = Operator can change the manual parameter `Man` (`WithRbk` = 1)
8	1 = Operator can change operation high limit of the setpoint `SP_InHiLim`
9	1 = Operator can change operation low limit of the setpoint `SP_InLoLim`
10	1 = Operator can change the operation high limit of the manipulated variable `ManHiLim`
11	1 = Operator can change the operation low limit of the manipulated variable `ManLoLim`
12	1 = Operator can enable the setpoint's gradient limitation function `SP_RateOn`
13	1 = Operator can change the setpoint's high limit for the ramp `SP_UpRaLim`
14	1 = Operator can change the setpoint's low limit for the ramp `SP_DnRaLim`
15	1 = Operator can switch between the time value or the value for the ramp `SP_RmpModTime`
16	1 = Operator can change the ramp time `SP_RmpTime`
17	1 = Operator can change the target setpoint `SP_RmpTarget` for the setpoint ramp
18	1 = Operator can enable the setpoint ramp function `SP_RmpOn`
19	1 = Operator can permit the PID optimization function `OptimEn`
20	1 = Operator can enable the track setpoint in "manual mode" function `SP_TrkPV`
21	1 = Operator can enable the bumpless switchover from external to internal `SP_TrkExt`
22	1 = Operator can change the gain parameter `Gain`
23	1 = Operator can change the integral time parameter `TI`
24	1 = Operator can change the derivative time parameter `TD`
25	1 = Operator can change the derivative gain parameter `DiffGain`
26	1 = Operator can change the dead band parameter `DeadBand`
27	1 = Operator can activate bypass functionality
28	1 = Operator can change the integral time parameter `MotorTime`

Bit	Function
29	1 = Operator can change the integral time parameter `PulseTime`
30	1 = Operator can change the integral time parameter `BreakTime`
31	1 = Operator can deactivate bypass functionality

The block has the following permissions for the `OS1Perm` parameter:

Bit	Function
0	1 = Operator can change the limit (process value) `PV_AH_Lim` for the high alarm
1	1 = Operator can change the limit (process value) `PV_WH_Lim` for the high warning
2	1 = Operator can change the limit (process value) `PV_TH_Lim` for the high tolerance
3	1 = Operator can change the hysteresis (process value) `PV_Hyst`
4	1 = Operator can change the limit (process value) `PV_TL_Lim` for the low tolerance
5	1 = Operator can change the limit (process value) `PV_WL_Lim` for the low warning
6	1 = Operator can change the limit (process value) `PV_AL_Lim` for the low alarm
7	1 = Operator can change the limit (error signal) `ER_AH_Lim` for the high alarm
8	1 = Operator can change the hysteresis (error signal) `ER_Hyst`
9	1 = Operator can change the limit (error signal) `ER_AL_Lim` for the low alarm
10	1 = Operator can change the limit (position feedback) `RbkWH_Lim` for the high warning
11	1 = Operator can change the hysteresis (position feedback) `RbkHyst`
12	1 = Operator can change the limit (position feedback) `RbkWL_Lim` for the low warning
13	1 = Operator can open the valve: `OpenOp` (`WithRbk` = 0)
14	1 = Operator can close the valve: `CloseOp` (`WithRbk` = 0)
15	1 = Operator can stop the valve: `StopOp` (`WithRbk` = 0)
16	1 = Operator can activate the Simulation function `SimOn`
17	1 = Operator can activate the Release for maintenance function `MS_RelOp`
18	1 = Operator can activate / deactivate messages via `PV_AH_MsgEn`
19	1 = Operator can activate / deactivate messages via `PV_WH_MsgEn`
20	1 = Operator can activate / deactivate messages via `PV_TH_MsgEn`
21	1 = Operator can activate / deactivate messages via `PV_TL_MsgEn`
22	1 = Operator can activate / deactivate messages via `PV_WL_MsgEn`
23	1 = Operator can activate / deactivate messages via `PV_AL_MsgEn`
24	1 = Operator can activate / deactivate messages via `ER_AH_MsgEn`
25	1 = Operator can activate / deactivate messages via `ER_AL_MsgEn`
26	1 = Operator can activate / deactivate messages via `RbkWH_MsgEn`
27	1 = Operator can activate / deactivate messages via `RbkWL_MsgEn`
28	1 = Operator can change the simulation value `SimPV`
29	1 = Operator can change the simulation value `SimRbk`
18 - 29	Not used
30	1 = Operator can activate the derivative action to the feedback path `DiffToFbk`
31	1 = Operator can change the proportional action to the feedback path `PropFacSP`

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Release for maintenance

The block provides the standard function Release for maintenance (Page 64).

Generating instance-specific messages

The block provides the standard function Generating instance-specific messages (Page 200) without the time stamp function in the I/O.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Specifying the display area for process and setpoint values as well as operations

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- `OpenOp`
- `StopOp`
- `CloseOp`

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

See also

PIDStepL messaging (Page 853)

PIDStepL I/Os (Page 855)

PIDStepL error handling (Page 851)

PIDStepL modes (Page 839)

PIDStepL block diagram (Page 869)

Evaluation of the signal status of the interlock signals (Page 141)

5.10.4 PIDStepL error handling

Error handling of PIDStepL

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` output parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
30	The value of `PV` can no longer be displayed in the REAL number field.
31	The value of `SP_Ext` can no longer be displayed in the REAL number field.
32	The value of `FFwd` can no longer be displayed in the REAL number field.
33	The value of `MV_Trk` can no longer be displayed in the REAL number field.
34	The value of `MV_Forced` can no longer be displayed in the REAL number field.
35	The value of `Rbk` can no longer be displayed in the REAL number field.
36	The value of `MV` can no longer be displayed in the REAL number field.

Error number	Meaning of the error number
50	The controller cannot be switched to program mode, because program mode with default setpoint (`AdvCoModSP` = 0) is not possible for step controllers without position feedback (`WithRbk` = 0).
51	`AutModLi` = 1 and `ManModLi` = 1 `SP_LiOp` = 1 and `SP_IntLi` = 1 and `SP_ExtLi` = 1 `OpenLi` = 1 and `StopLi` = 1 `CloseLi` = 1 and `StopLi` = 1 `OpenLi` = 1 and `CloseLi` = 1
59	= 1 "Gain is negative"
60	|`TI`| < `SampleTime` / 2
61	|`TD`| < `SampleTime`
62	`DiffGain` < 1 or `DiffGain` > 10
63	`TD` / `DiffGain` < `SampleTime` / 2
64	`PropFacSP` < 0 or `PropFacSP` > 1
66	`NormPV_High` = `NormPV_Low`
67	`MotorTime` < `SampleTime`
68	`PulseTime` < `SampleTime`
69	`BreakTime` < `SampleTime`

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

PIDStepL functions (Page 840)

Description of PIDStepL (Page 835)

PIDStepL modes (Page 839)

PIDStepL messaging (Page 853)

PIDStepL I/Os (Page 855)

PIDStepL block diagram (Page 869)

Setting switch or button mode (Page 166)

5.10.5 PIDStepL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 6	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter CSF. If it changes to CSF = 1, a process control fault is triggered (MsgEvId2, SIG 6).

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	Alarm - high	$$BlockComment$$ PV - high alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ PV - high warning limit violated
	SIG 3	Tolerance - high	$$BlockComment$$ PV - high tolerance limit violated
	SIG 4	Tolerance - low	$$BlockComment$$ PV - low tolerance limit violated
	SIG 5	Warning - low	$$BlockComment$$ PV - low warning limit violated
	SIG 6	Alarm - low	$$BlockComment$$ PV - low alarm limit violated
	SIG 7	Alarm - high	$$BlockComment$$ ER - high alarm limit violated
	SIG 8	Alarm - low	$$BlockComment$$ ER - low alarm limit violated

Message instance	Message identifier	Message class	Event
`MsgEvId2`	SIG 7	Warning - high	$$BlockComment$$ `Rbk` - high warning limit violated
	SIG 8	Warning - low	$$BlockComment$$ `Rbk` - low warning limit violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You can use up to four instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
`MsgEvId2`	SIG 1	AS process control message - fault	$$BlockComment$$ External message 1 Status 16#@5%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ External message 2 Status 16#@6%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ External message 3 Status 16#@7%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 4 Status 16#@8%x@

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	Process value `PV_Out`
5	Control deviation `ER`
6	`ExtVa106`
7	`ExtVa107`
8	Not allocated
9	Not allocated
10	Not allocated

The associated values 6 ... 7 are allocated to the parameters ExtVa106 ... ExtVa107 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

Associated values for message instance MsgEvId2

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	Position feedback Rbk
5	Signal status ExtMsg1
6	Signal status ExtMsg2
7	Signal status ExtMsg3
8	Signal status ExtMsg4
9	ExtVa209
10	ExtVa210

The associated values 9 ... 10 are allocated to the parameters ExtVa209 ... ExtVa210 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of PIDStepL (Page 835)
PIDStepL functions (Page 840)
PIDStepL I/Os (Page 855)
PIDStepL modes (Page 839)
PIDStepL error handling (Page 851)
PIDStepL block diagram (Page 869)

5.10.6 PIDStepL I/Os

I/Os of PIDStepL

Input parameters

Parameter	Description	Type	Default
AdvCoEn	1 = Enable "program mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoOn*	1 = Enable "program mode" via faceplate	BOOL	0

Parameter	Description	Type	Default
AdvCoModSP	Type of "program mode": 1 = Setpoint specification 0 = Manipulated variable specification	BOOL	1
AdvCoMstrOn	Activate (0-1) or deactivate (1-0) "program mode" via edge transition	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoMV	Specified value from the external program	REAL	0.0
AutModLi*	1 = "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 0)	BOOL	0
BatchEn	1 = Enable allocation for batch control	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BreakTime	Minimum break duration [s]	REAL	1.0
BypLiOp	1 = Bypass commands via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypLock	1 = Bypass activation or deactivation is locked for operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypPV	Substitution value if block is in bypass	REAL	0.0
BypPVLi	1 = Select bypass PV (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypPVOp	1 = Select bypass PV (via operator)	BOOL	0
CloseChnST	Output channel state of Close Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
CloseLi*	1 = Close via interconnection or CFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseOp*	1 = Close via operator	BOOL	0
CPI_In	Input for control performance index, which is calculated by the assigned ConPerMon block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DeadBand	Width of dead band	REAL	0.0

Parameter	Description	Type	Default
DiffGain	Gain of differentiator [1..10] DiffGain = TD / (delay time of D component)	STRUCT • Value: REAL • ST: BYTE	- • 5.0 • 16#FF
DiffSel	1 = D component activated	BOOL	1
DiffToFbk	1 = D component is placed in the feedback	BOOL	0
EN	1 = Called block will be processed	BOOL	1
ER_A_DC*	Delay for incoming alarms during error signal monitoring	REAL	0.0
ER_A_DG*	Delay for outgoing alarms during error signal monitoring	REAL	0.0
ER_AH_En	1 = Activate alarm (high) for error signal monitoring	BOOL	1
ER_AH_Lim	Alarm limit (high) for error signal monitoring	REAL	100.0
ER_AH_MsgEn	1 = Activate messages for alarm (high) for error signal monitoring	BOOL	1
ER_AL_En	1 = Activate alarm (low) for error signal monitoring	BOOL	1
ER_AL_Lim	Alarm limit (low) for error signal monitoring	REAL	-100.0
ER_AL_MsgEn	1 = Activate messages for alarm (low) for error signal monitoring	BOOL	1
ER_Hyst	Alarm hysteresis for error signal	REAL	1.0
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	1 = Binary input for freely selectable message 1 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	1 = Binary input for freely selectable message 2 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	1 = Binary input for freely selectable message 3 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg4	1 = Binary input for freely selectable message 4 is used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa209	Associated value 9 for messages (MsgEvID2)	ANY	
ExtVa210	Associated value 10 for messages (MsgEvID2)	ANY	
FbkClosed	Low limit stop signal of position feedback	STRUCT • Value: BOOL • ST: BYTE	- 0 16#80
FbkOpened	High limit stop signal of position feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Feature	I/O for additional functions (Page 840)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
FFwd*	Input for additive disturbance variable activation	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
FFwdHiLim	Limit (high) for additive disturbance variable activation	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
FFwdLoLim	Limit (low) for additive disturbance variable activation	STRUCT • Value: REAL • ST: BYTE	- • -100.0 • 16#80
Gain	Proportional gain Gain.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#FF
IntHoldNeg	1 = Integrator cannot run in negative direction	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
IntHoldPos	1 = Integrator cannot run in positive direction	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
IntSel	1 = I component activated	BOOL	1
Man*	Manual specification for the manipulated variable	REAL	0.0
ManHiLim	Limit (high) for manual parameter Man	REAL	100.0
ManLoLim	Limit (low) for manual parameter Man	REAL	0.0

Parameter	Description	Type	Default
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
ModLiOp	Operating mode switchover between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MotorTime	Motor actuating time [s]	REAL	30.0
MS_RelOp*	1 = Release for maintenance by OS operator	BOOL	0
MsgEvID1	Message number (assigned automatically)	DWORD	16#00000000
MsgEvID2	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_Forced*	Forced manipulated variable that is not limited and assumes top priority	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_ForOn	1 = Output forced manipulated variable MV_Forced unlimited at output MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_HiLim	Limit (high) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
MV_LoLim	Limit (low) for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_Offset	Manipulated variable for ER = 0, operating point for controller with deactivated I component	REAL	0.0
MV_Trk*	Tracking value for the manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_TrkOn	1 = Tracking of manipulated variable MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NegGain	0 = Effective proportional gain GainEff is positive 1 = Effective proportional gain GainEff is negative	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
NormPV	Process value range (PV) for standardizing the proportional gain (GAIN)	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
Occupied	1 = Occupied by batch control	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpenChnST	Output channel state of Open Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
OpenLi*	1 = Open via interconnection or CFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenOp*	1 = Open via operator	BOOL	0
OptimEn*	1 = Enable optimization of PID parameters by PID tuner	BOOL	0
OptimOcc*	1 = Optimization running	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 840)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OS1Perm	I/O for operator permissions (Page 840)	STRUCT • Bit 0: BOOL • Bit 18: BOOL • Bit 19: BOOL • Bit 31: BOOL	- • 1 • 1 • 1 • 1
PropFacSP	Applying the P component to the feedback [0..1]. 0 = P component fully in feedback	REAL	1.0
PropSel	1 = Activate P component	BOOL	1
PulseTime	Minimum pulse duration [s]	REAL	1.0
PV*	Process value (controlled variable)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Unit	Unit of measure for process value	INT	1001
PV_A_DC*	Delay time for incoming PV alarms [s]	REAL	0.0
PV_A_DG*	Delay time for outgoing PV alarms [s]	REAL	0.0
PV_AH_En	1 = Enable PV alarm limit (high)	BOOL	1

Parameter	Description	Type	Default
PV_AH_Lim	Limit PV alarm (high)	REAL	95.0
PV_AH_MsgEn	1 = Enable PV alarm (high) message	BOOL	1
PV_AL_En	1 = Enable PV alarm limit (low)	BOOL	1
PV_AL_Lim	PV alarm limit (low)	REAL	5.0
PV_AL_MsgEn	1 = Enable PV alarm (low) message	BOOL	1
PV_Hyst	Hysteresis for PV alarm, warning and tolerance limits	REAL	1.0
PV_OpScale	Limit for scale in PV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
PV_T_DC*	Delay time for incoming PV tolerance messages [s]	REAL	0.0
PV_T_DG*	Delay time for outgoing PV tolerance messages [s]	REAL	0.0
PV_TH_En	1 = Enable PV tolerance limit (high)	BOOL	0
PV_TH_Lim	Limit PV tolerance message (high)	REAL	85.0
PV_TH_MsgEn	1 = Enable message for PV tolerance message (high)	BOOL	1
PV_TL_En	1 = Enable PV tolerance limit (low)	BOOL	0
PV_TL_Lim	Limit PV tolerance message (low)	REAL	15.0
PV_TL_MsgEn	1 = Activate message for PV tolerance message (low)	BOOL	1
PV_Unit	Unit of measure for process value	INT	1001; ^{O}C
PV_W_DC*	Delay time for incoming PV warnings [s]	REAL	0.0
PV_W_DG*	Delay time for outgoing PV warnings [s]	REAL	0.0
PV_WH_En	1 = Enable PV warning limit (high)	BOOL	1
PV_WH_Lim	Limit PV warning (high)	REAL	90.0
PV_WH_MsgEn	1 = Enable PV warning (high) message	BOOL	1
PV_WL_En	1 = Enable PV warning limit (low)	BOOL	1
PV_WL_Lim	Limit PV warning (low)	REAL	10.0
PV_WL_MsgEn	1 = Enable PV warning (low) message	BOOL	1
Rbk*	Position feedback	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
RbkHyst	Alarm hysteresis for position feedback	REAL	1.0
RbkWH_En	1 = Enable warning (high) for position feedback	BOOL	1
RbkWH_Lim	Limit for position feedback of warning (high)	REAL	100.0
RbkWH_MsgEn	1 = Enable messages for warning (high) for position feedback	BOOL	1
RbkWL_En	1 = Enable warning (low) for position feedback	BOOL	1
RbkWL_Lim	Limit for position feedback of warning (low)	REAL	0.0

Parameter	Description	Type	Default
RbkWL_MsgEn	1 = Enable messages for warning (low) for position feedback	BOOL	1
RefStdDevIn	Reference value of PV standard deviation (sigma) in defined "good" state of control loop	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#78
RstBypLi	1 = Reset bypass PV (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstBypOp	1 = Reset bypass PV (via operator)	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
S_RbkOnPIDTun	Simulation of position feedback on; For PCS 7 PID tuner only	BOOL	0
S_RbkPIDTun*	Simulated position feedback	REAL	50.0
SafePos	Neutral position (Page 48) for step controller actuating signals: 0 = Close 1 = Open 2 = Stop	INT	0
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the pre-view	ANY	-
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimPV*	Process value used for SimOn = 1	REAL	0.0
SimPV_Li	Process value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimRbk*	Position feedback used for SimOn = 1	REAL	0.0
SimRbkLi	Position feedback used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_DnRaLim	Limit (low) for the gradient of the setpoint [SP_Unit/s]	REAL	100.0

Parameter	Description	Type	Default
SP_ExHiLim	Limit (high) for external setpoint	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP_ExLoLim	Limit (low) for external setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_Ext*	External setpoint of - (to interconnection)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtLi*	1 = Select external setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOp*	1 = Select external setpoint (via operator)	BOOL	0
SP_InHiLim	Limit (high) of internal setpoint	REAL	100.0
SP_InLoLim	Limit (low) of internal setpoint	REAL	0.0
SP_Int*	Internal setpoint for operation	REAL	0.0
SP_IntLi*	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_IntOp*	1 = Select internal setpoint (via operator)	BOOL	0
SP_LiOp	Select setpoint source (internal/external): 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_RateOn*	1 = Activate limitation of setpoint gradients	BOOL	0
SP_RmpModTime	1 = Use time (SP_RmpTime) for setpoint ramp 0 = Use gradient	BOOL	0
SP_RmpOn*	1 = Activate setpoint ramp to target setpoint SP_RmpTarget	BOOL	0
SP_RmpTarget*	Target setpoint for setpoint ramp	REAL	0.0
SP_RmpTime*	Time for setpoint ramp [s] from current SP up to SP_RmpTarget	REAL	0.0
SP_TrkExt	1 = Bumpless switchover from external to internal setpoint active	BOOL	1
SP_TrkPV	1 = Setpoint follows PV in "manual mode" and with tracking	BOOL	0
SP_UpRaLim	Gradient limit (high) for the setpoint [SP_Unit/s]	REAL	100.0
StepNo	Batch step number	DWORD	16#00000000
StopChnST	Output channel state of Stop Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF

Parameter	Description	Type	Default
StopLi*	1 = Stop via interconnection or CFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopOp*	1 = Stop via operator	BOOL	0
TD	Derivative time in [s] TD.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
ThrAdaOn	Adaptation of threshold 0 = Hold constant	BOOL	1
TI	Integral time [s] TI.ST = 16#FF: Enabled in faceplate	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#FF
TimeFactor	Time unit: 0 = Seconds 1 = Minutes 2 = Hours	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
WithRbk	1 = Feedback value available for the manipulated variable	BOOL	0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AdvCoAct	1 = "Program mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AdvCoRdy	1 = "Program mode" available	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutAct	1 = "Automatic mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypassAct	1 = Bypass is activated in this block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CascaCut	Cascade connection: 1 = Control chain from primary to secondary controller is interrupted	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Close	Control output: 1 = Closed is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
D_Part	D component of PID algorithm	REAL	0.0
ENO	1 = Block algorithm completed without errors	BOOL	0
ER	Error signal	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ER_AH_Act	1 = Alarm limit (high) for control deviation violated. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ER_AL_Act	1 = Alarm limit (low) for control deviation violated. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see PIDStepL error handling (Page 851)	INT	-1
FFwdHiAct	1 = Limit (high) for additive disturbance variable activation violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FFwdLoAct	1 = Limit (low) for additive disturbance variable activation violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GainEff	Effective proportional gain depends on NegGain, Gain, NormPV, and NormMV	REAL	1.0
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
I_Part	I component of PID algorithm	REAL	0.0
LoopClosed	1 = Control loop closed 0 = Control loop open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
ManHiOut	Limit (high) for "manual mode", corresponds to input parameter ManHiLim	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80

Parameter	Description	Type	Default
ManLoOut	Limit (low) for "manual mode", corresponds to input parameter ManLoLim	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgAckn2	Message acknowledgement status 2 (output ACK_STATE of second ALARM_8P)	WORD	16#0000
MsgErr1	1 = Alarm error 1 (output ERROR of the first ALARM_8P)	BOOL	0
MsgErr2	1 = Alarm error 2 (output ERROR of the second ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
MsgStat2	Message status 2 (output STATUS of second ALARM_8P)	WORD	16#0000
MV	Manipulated variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_HiAct	1 = Limit (high) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_LoAct	1 = Limit (low) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Block is "out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Open	Control output: 1 = Open is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS1PermOut	Display of OS1Perm	DWORD	16#FFFFFFFF
OS1PermLog	Display of OS1Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
P_Part	P component of PID algorithm	REAL	0.0

Parameter	Description	Type	Default
PV_AH_Act	1 = PV alarm (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AL_Act	1 = PV alarm (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	Output for process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_TH_Act	1 = PV tolerance message (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_TL_Act	1 = PV tolerance message (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_ToleHi	Limit (high) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_ToleLo	Limit (low) of 3-sigma band around the setpoint is calculated when a ConPerMon block is connected	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_UnitOut	Unit of measure for process value, for interconnecting to the PV_Unit input parameter of the ConPerMon block	INT	0
PV_WH_Act	1 = PV warning (high) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_WL_Act	1 = PV warning (low) active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkOut	Output for position feedback	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
`RbkWH_Act`	1 = Warning (high) for position feedback active. You can change the reaction for this parameter with `Feature` bit 28 (Disabling operating points (Page 144)) and with `Feature` bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`RbkWL_Act`	1 = Warning (low) for position feedback active. You can change the reaction for this parameter with `Feature` bit 28 (Disabling operating points (Page 144)) and with `Feature` bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`RdyToStart`	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SP`	Setpoint used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`SP_DnRaAct`	1 = Negative gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SP_ExHiAct`	1 = Limit (high) for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SP_ExLoAct`	1 = Limit (low) for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SP_ExtAct`	1 = External setpoint is active 0 = Internal setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SP_ExtOut`	External setpoint, corresponds to input parameter `SP_Ext`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`SP_InHiOut`	Limit (high) for `SP_Int` corresponds to input parameter `SP_InHiLim`	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
`SP_InLoOut`	Limit (low) for `SP_Int` corresponds to input parameter `SP_InLoLim`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`SP_RateTarget`	Target setpoint for the gradient limitation	REAL	0.0
`SP_UpRaAct`	Positive gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
SP_RemRT	Remaining ramp time of the setpoint	REAL	0.0
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 835)	DWORD	16#00000000
Status2	Status word 2 (Page 835)	DWORD	16#00000000
Status3	Status word 2 (Page 835)	DWORD	16#00000000
Stop	Control output: 1 =Stopped is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SumMsgAct	1 = Active hardware interrupt	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Threson	Adaptive threshold [%]	REAL	0.0

See also

PIDStepL messaging (Page 853)

PIDStepL modes (Page 839)

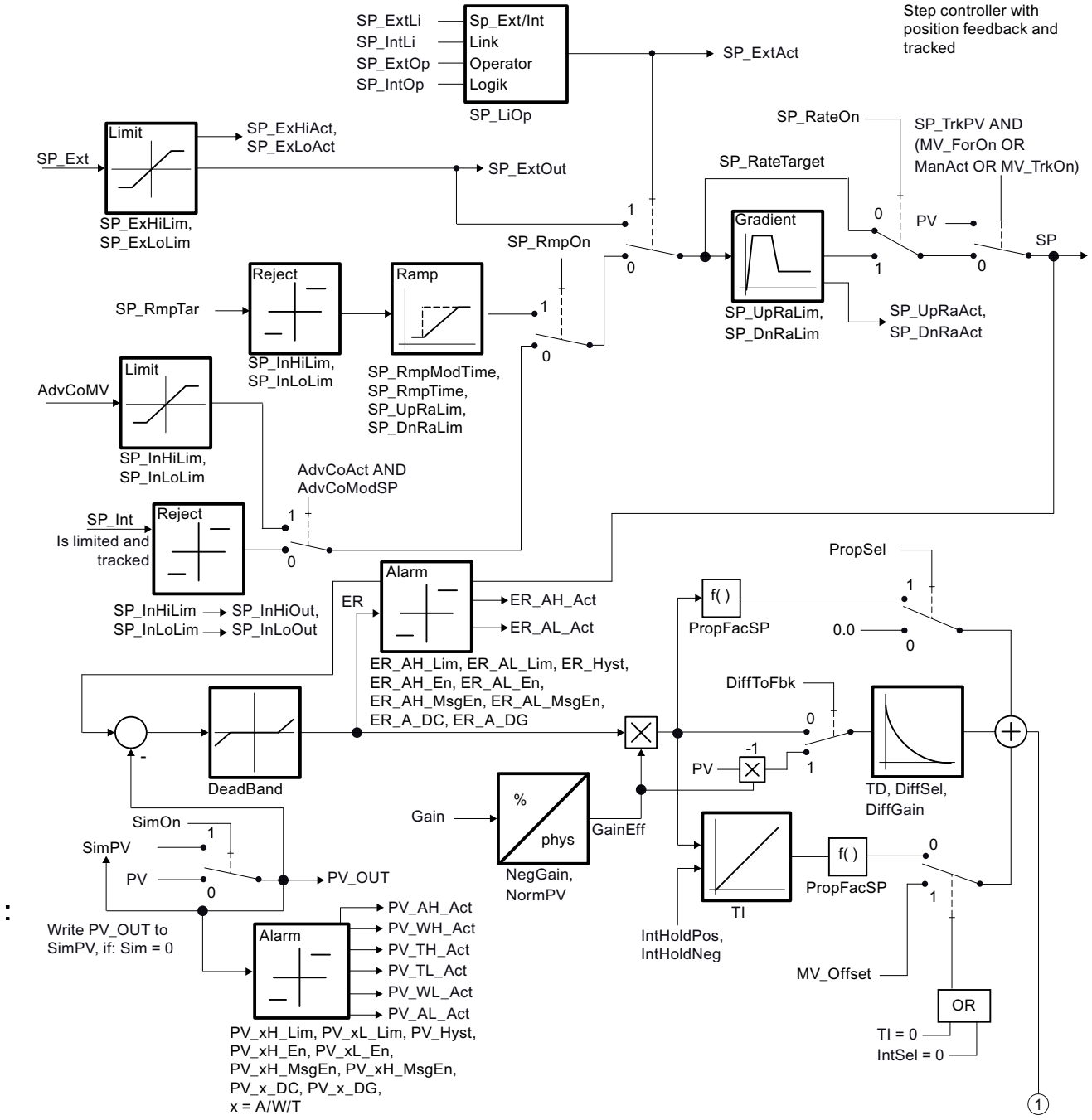
PIDStepL block diagram (Page 869)

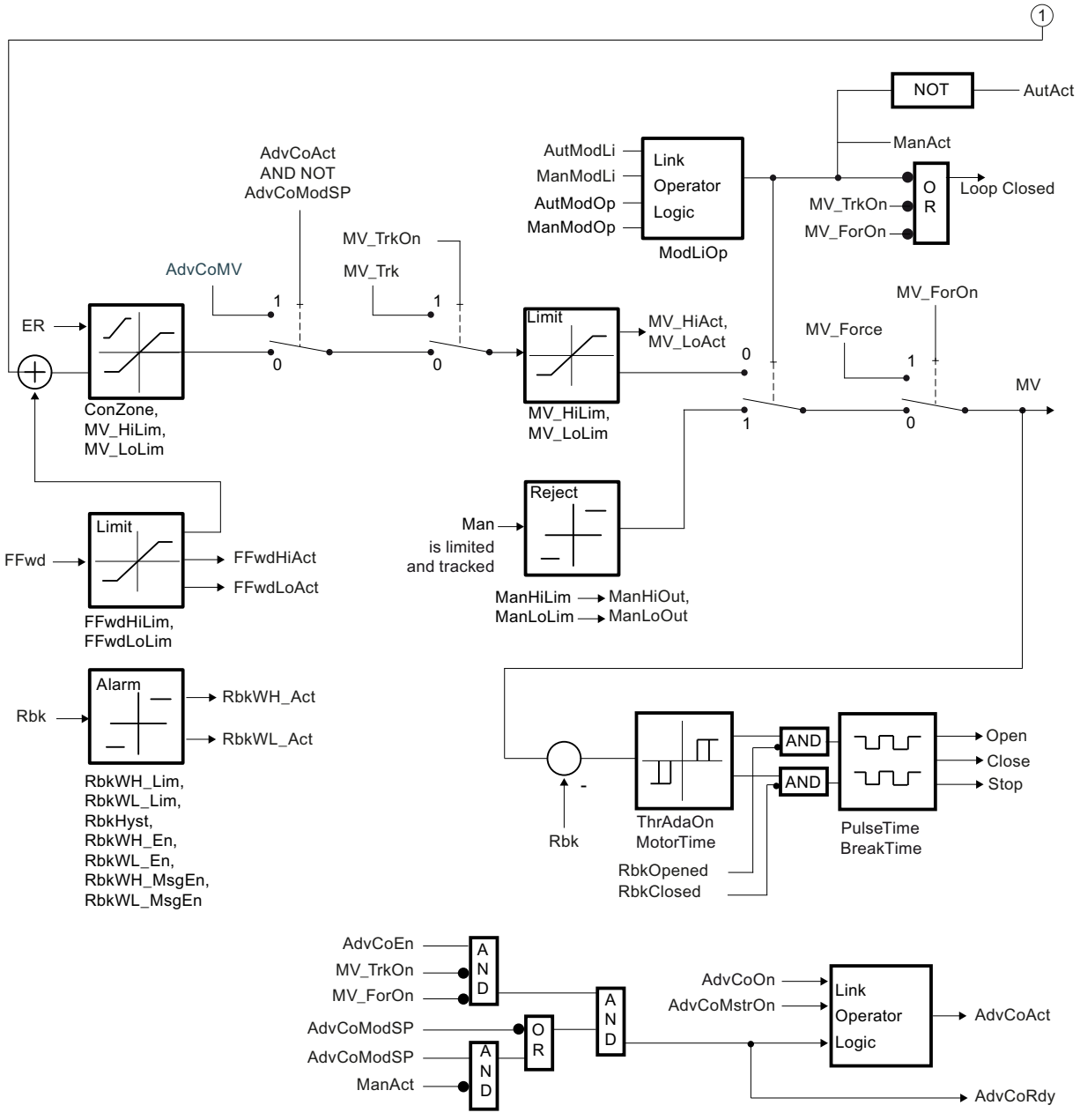
5.10.7 PIDStepL block diagram

PIDStepL block diagram

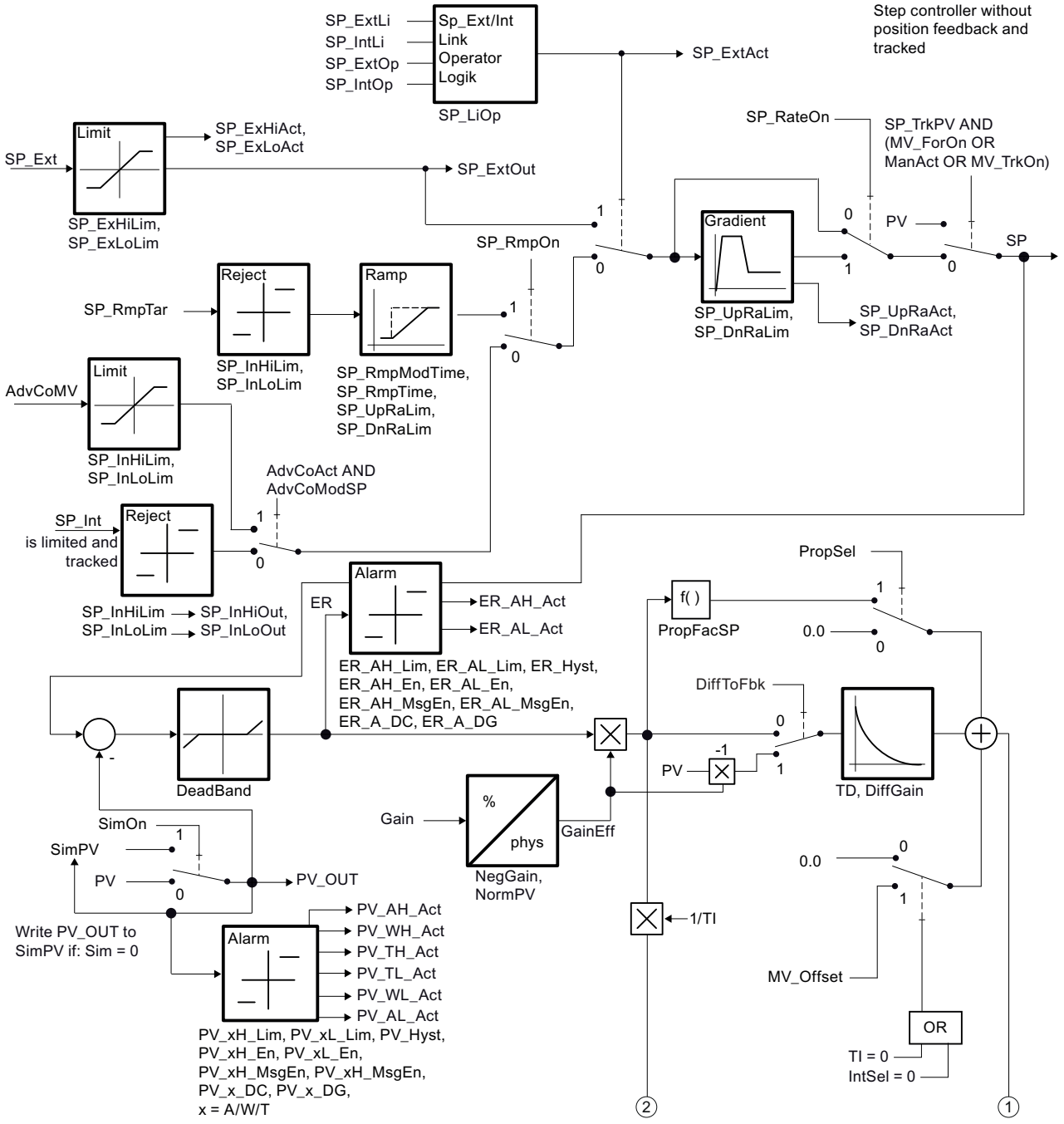
There are two block diagrams for this block: the step controller with and without position feedback.

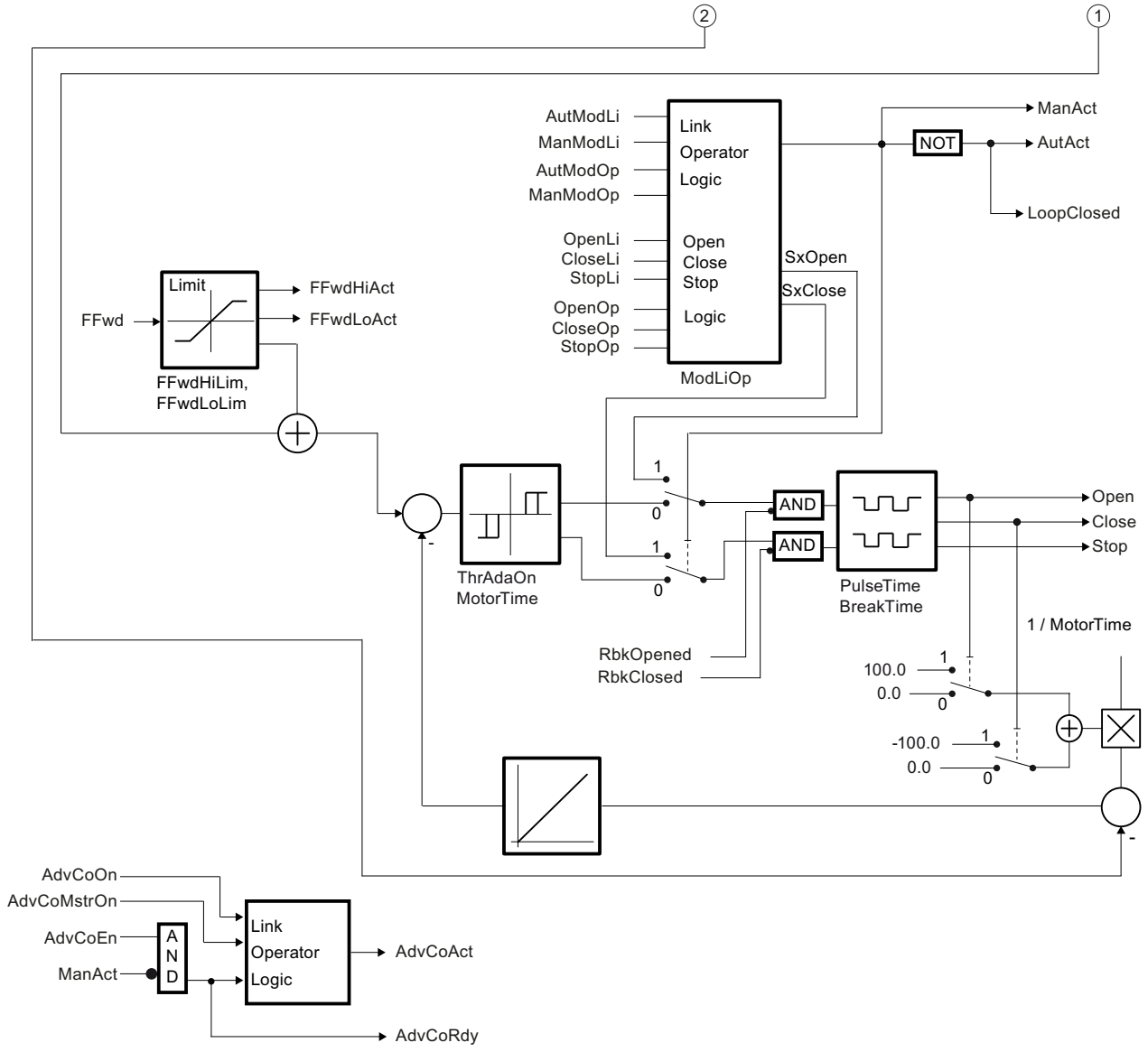
Step controller with position feedback




1
NOT
AutAct
ManAct
AdvCoAct
AND NOT
AdvCoModSP
AutModLi
ManModLi
AutModOp
ManModOp
Link
Operator
Logic
ModLiOp
O
R
Loop Closed
MV_TrkOn
MV_ForOn
AdvCoMV
MV_TrkOn
MV_Trk
MV_ForOn
MV_Force
ER
Limit
MV_HiAct,
MV_LoAct
MV_HiLim,
MV_LoLim
ConZone,
MV_HiLim,
MV_LoLim
MV
Reject
Man
is limited
and tracked
ManHiLim
ManLoLim
ManHiOut,
ManLoOut
FFwd
FFwdHiAct
FFwdLoAct
FFwdHiLim,
FFwdLoLim
Rbk
Alarm
RbkWH_Act
RbkWL_Act
RbkWH_Lim,
RbkWL_Lim,
RbkHyst,
RbkWH_En,
RbkWL_En,
RbkWH_MsgEn,
RbkWL_MsgEn
AND
Open
Close
Stop
ThrAdaOn
MotorTime
PulseTime
BreakTime
RbkOpened
RbkClosed
AdvCoEn
MV_TrkOn
MV_ForOn
AdvCoModSP
AdvCoModSP
ManAct
AdvCoOn
AdvCoMstrOn
AdvCoAct
AdvCoRdy

Step controller without position feedback





See also

Description of PIDStepL (Page 835)

PIDStepL modes (Page 839)

PIDStepL functions (Page 840)

PIDStepL error handling (Page 851)

PIDStepL messaging (Page 853)

PIDStepL I/Os (Page 855)

5.10.8 Operator control and monitoring

5.10.8.1 PIDStepL views

Views of the PIDStepL block

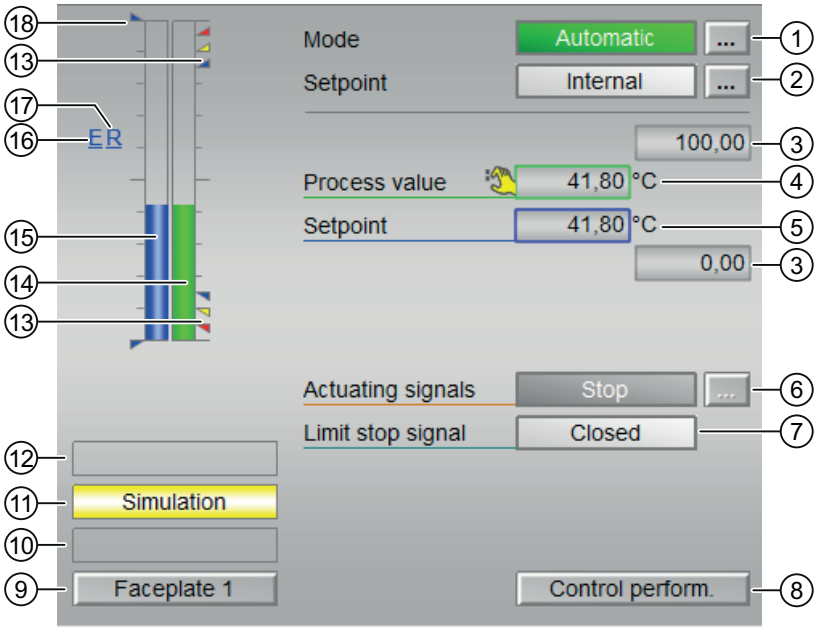
The block PIDStepL provides the following views:

- PIDStepL standard view without position feedback (Page 875)
- PIDStepL standard view with position feedback (Page 878)
- Alarm view (Page 296)
- Limit view of PID controllers (Page 285)
- Trend view (Page 299)
- Ramp view (Page 294)
- Parameter view of PID controllers (Page 275)
- PIDStepL preview (Page 882)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icons for PID and FM controller (Page 235)

Refer to the Structure of the faceplate (Page 242) and Block icon structure (Page 226) sections for general information about the faceplate and block icon.

5.10.8.2 PIDStepL standard view without position feedback

PIDStepL standard view without position feedback



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 72)
- Automatic mode (Page 72)
- Program mode for controllers (Page 78)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) Displaying and switching the setpoint

This area shows how to specify the setpoint. The setpoint can specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the setpoint specification.

You can find additional information on this in the Setpoint specification - internal/external (Page 127) section.

(3) High and low scale range for the process value

These values provide information on the display range for the bar graph of the process value. The scale range is defined in the Engineering System (ES).

(4) Display of the process value including signal status

This area shows the current process value with the corresponding signal status.

(5) Displaying and changing the setpoint including signal status

This area shows the current setpoint with the corresponding signal status.

Refer to the Changing values (Page 253) section for information on changing the setpoint. The setpoint specification also needs to be set to "Internal" for this block.

(6) Displaying and changing manipulated variables

This area shows you the currently valid manipulated variable. Refer to the Switching operating states and operating modes (Page 251) section for information on changing the manipulated variable.

The following manipulated variables can be selected:

- "Open"
- "Stop"
- "Close"

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in the Section Labeling of buttons and text (Page 205).

(7) Displaying the feedback

The following feedback can be displayed:

- "Open"
- "Closed"

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in the Section Labeling of buttons and text (Page 205).

(8) Navigation button for switching to the standard view of the ConPerMon block

Use this navigation button to reach the standard view of the ConPerMon block. The visibility of this navigation button depends on the configuration in the engineering system (ES).

See also the Opening additional faceplates (Page 203) section for more on this.

(9) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the Engineering System (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the Opening additional faceplates (Page 203) section.

(10) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on the display area for states of the block is available in section Release for maintenance (Page 64).

(11) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in the Simulating signals (Page 58) section.

(12) Display area for block states

This area provides additional information on the operating state of the block (from high to low according to priority):

- "Optimization"
- "Tracking"
- "Forced tracking"
- "SP ramp active"

(13) Limit display

These colored triangles show you the configured limits in the respective bar graph.

(14) Bar graph for the process value

This area shows the current process value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(15) Bar graph for the setpoint

This area shows the current setpoint in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(16) Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

(17) Display for the target setpoint of the setpoint ramp

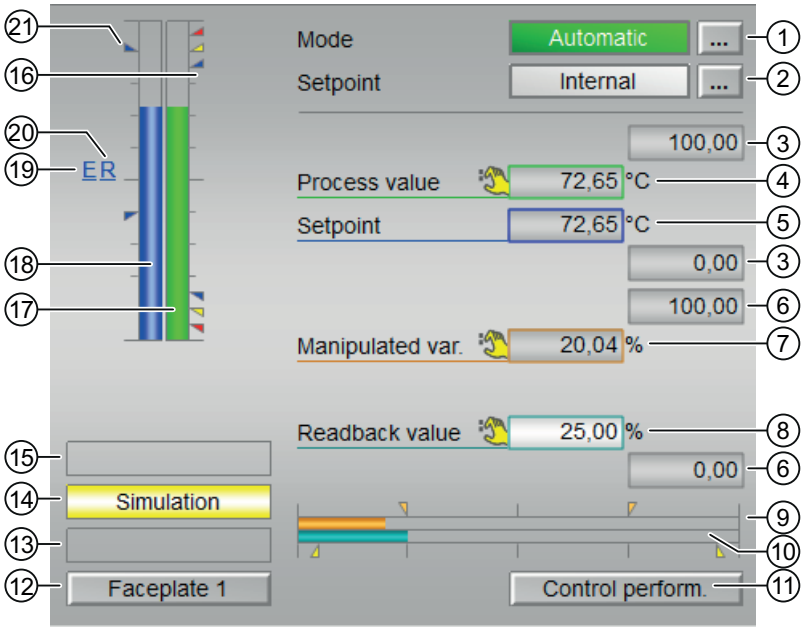
This display [R] shows you the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

(18) Displaying the limits

These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the Engineering System (ES).

5.10.8.3 PIDStepL standard view with position feedback

PIDStepL standard view with position feedback



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 72)
- Automatic mode (Page 72)

- Program mode for controllers (Page 78)
- Out of service (Page 71)

Refer to chapter Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Display and switch the setpoint specification

This area shows how to specify the setpoint. The setpoint can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) chapter for information on switching the setpoint specification.

You can find additional information on this in section Setpoint specification - internal/external (Page 127).

(3) High and low scale range for the process value

These values provide information on the display range for the bar graph of the process value. The scale range is defined in the Engineering System (ES).

(4) Display of the process value including signal status

This area shows the current process value with the corresponding signal status.

(5) Display and change the setpoint including signal status

This area shows the current setpoint with the corresponding signal status.

Refer to the Changing values (Page 253) chapter for information on changing the setpoint. The setpoint specification also needs to be set to "Internal" for this block.

(6) High and low limits of the manipulated variable

You can only display and change a manipulated variable in "manual mode".

(7) Displaying and changing the manipulated variable

You can only display and change a manipulated variable in "manual mode".

(8) Displaying the feedback

This display is only visible when the corresponding block input is interconnected.

The following feedback can be displayed:

- "Open"
- "Closed"

(9) Bar graph for the manipulated variable

The bar graph for the manipulated variable is only available in manual mode.

(10) Bar graph for the readback value

This display is only visible when the corresponding block input is interconnected.

The bar graph for the readback value is only available in manual mode.

(11) Button for switching to the standard view of the ConPerMon block

Use this button for the standard view of the ConPerMon block. The visibility of this button depends on the configuration in the engineering system (ES).

Refer also to chapter Opening additional faceplates (Page 203) for more on this.

(12) Button for switching to the standard view of any faceplate

Use this button for the standard view of a block configured in the Engineering System (ES). The visibility of this button depends on the configuration in the engineering system (ES).

You can find additional information on this in section Opening additional faceplates (Page 203).

(13) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in chapter Release for maintenance (Page 64) Display area for block states.

(14) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in section Simulating signals (Page 58).

(15) Display area for block states

This area provides additional information on the operating state of the block (from high to low according to priority):

- "Optimization"
- "Tracking"
- "Forced tracking"
- "SP ramp active"

(16) Limit display

These colored triangles show you the configured limits in the respective bar graph.

(17) Bar graph for the process value

This area shows the current process value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(18)Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

(19) Display for the target setpoint of the setpoint ramp

This display [R] shows the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

(20) Bar graph for the setpoint

This area shows the current setpoint in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(21) Displaying the limits

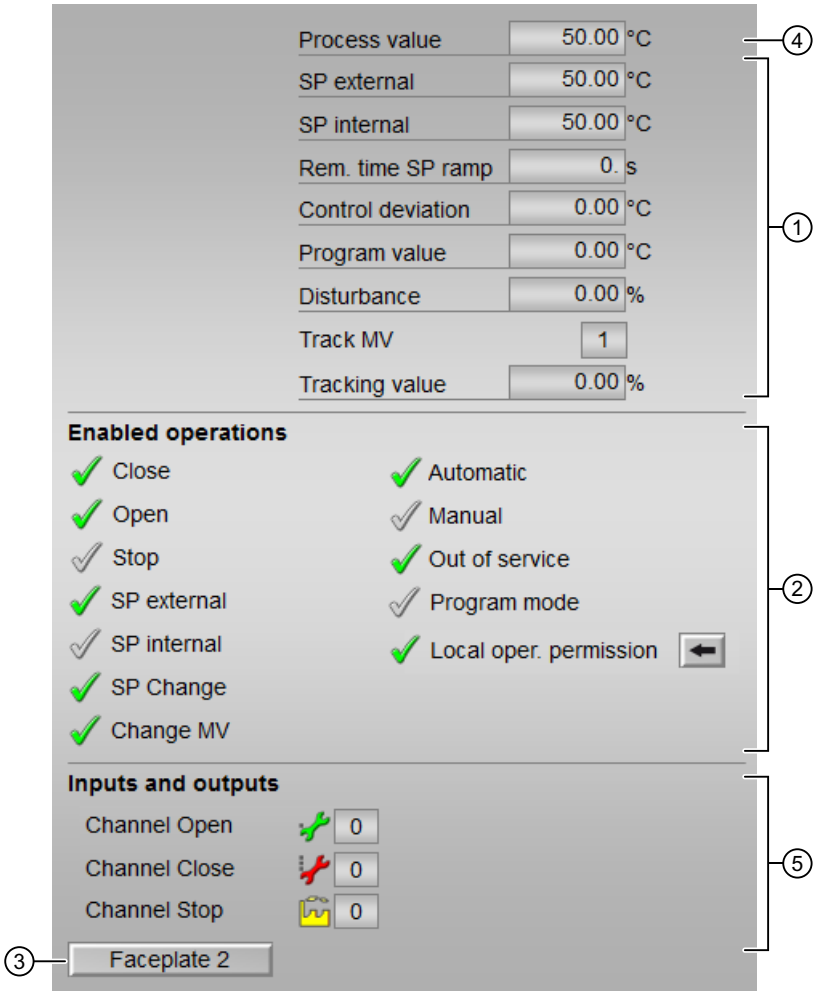
These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the Engineering System (ES).

See also

Labeling of buttons and text (Page 205)

5.10.8.4 PIDStepL preview

Preview of PIDStepL



(1) Preview area

This area shows you a preview for the following values:

- "SP external": currently applicable external setpoint
- "SP internal": currently applicable internal setpoint
- "Rem. time SP ramp": Remaining time to reach the ramp target value.
- "Error signal": Current control deviation
- "Program value": specified value for program mode
- "Disturbance variable": additive value for feedforward control

- "Track MV": Track manipulated variable (value is 1)
- "Tracking value": effective manipulated variable for "Track manipulated variable"

(2) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Close": You can select the manipulated variable "Close". If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205) .
- "Open": You can select the manipulated variable "Open". If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205) .
- "Stop": You can select the manipulated variable "Stop". If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205) .
- "SP external": You can feedforward the external setpoint.
- "SP internal": You can feedforward the internal setpoint.
- "Change SP": You can change the setpoint.
- "Change MV": You can change the manipulated variable.
- "Program mode": You can switch to "program mode".
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248) .

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

(4) Process value

This area displays the real process value (PV).

(5) Display of the current control signal

This area shows the most important parameters for this block with the current selection

- "Channel Open": Signal from the output channel block for "Open"
- "Channel Close": Signal from the output channel block for "Close"
- "Channel Stop": Signal from the output channel block for "Stop"

5.11 Ratio - Ratio controlling

5.11.1 Description of Ratio

Object name (type + number) and family

Type + number: FB 1883

Family: Control

Area of application for Ratio

The block is used for the following applications:

- Forming a ratio

How it works

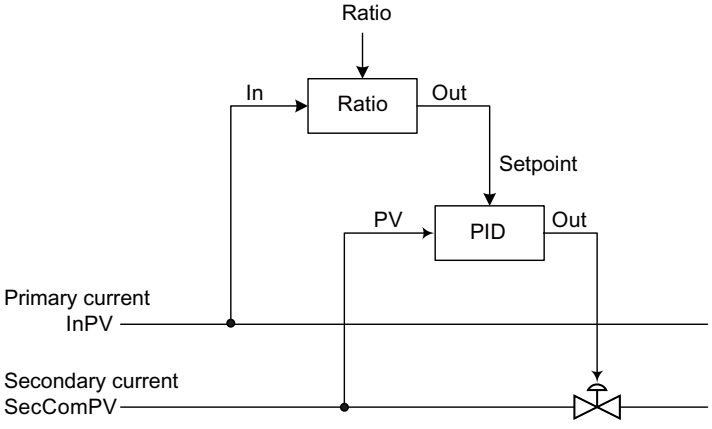
The block is used to create a ratio, for example, in a ratio control. It is also used to set elements (for example, synchronization control loop), or to influence the reference variable of a cascade.

The block operates according to the equation: `Out` = `In` · `RatioOut` + `Offset`

Where:

- `RatioOut` is either the value from `RatioInt` or `RatioExt`
- `Offset` is the offset value that should be added to the output value

`In` is based on the interconnection whereas `RatioOut` is selected based on the internal/ external selection.



Also refer to the Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312) section for information on calculating the current ratio.

Configuring OBs

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status` parameter

The description for each parameter can be found in the Ratio I/Os (Page 891) section.

Status bit	Parameter
0 – 1	Not used
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`
7	Not used
8	`RatExtAct.Value`
9	`RatLoAct.Value`
10	`RatHiAct.Value`
11	`OutLoAct.Value`
12	`OutHiAct.Value`
13	`RatTrkExt`
14	`SimLiOp.Value`
15 - 31	Not used

See also

Ratio functions (Page 887)

Ratio messaging (Page 891)

Ratio block diagram (Page 895)

Ratio error handling (Page 890)

Ratio modes (Page 887)

5.11.2 Ratio modes

Ratio operating modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the Out of service (Page 71) section.

The last valid value is provided at the `Out` output in this operating mode. The `OutHiAct`, `OutLoAct`, `RatHiAct`and `RatLoAct`output parameters are reset.

See also

Ratio block diagram (Page 895)

Ratio I/Os (Page 891)

Ratio messaging (Page 891)

Ratio error handling (Page 890)

Ratio functions (Page 887)

Description of Ratio (Page 885)

5.11.3 Ratio functions

Functions of Ratio

The functions for this block are listed below.

Internal or external ratio

You can use the RatLiOp parameter to specify if the ratio should be specified internally or externally:

- RatLiOp = 0: The ratio (RatioInt) is specified by the operator. The operator can decide if the specification should be made internally (RatIntOp = 1) or externally (RatExtOp = 1) .
- RatLiOp = 1: The ratio (RatioExt) is specified by an interconnection. The interconnection is used to determine if the specification should be made internally (RatIntLi = 1) or externally (RatExtLi = 1) .

Bumpless switchover from external to internal ratio

The parameter RatTrkExt = 1 is used so that the internal ratio tracks the external setpoint to achieve a bumpless switchover from the external to the internal ratio. This allows unwanted jumps at the output parameter to be avoided.

Limiting the ratio

The ratio is limited by the RatHiLim parameter (high) or RatLoLim parameter (low).

The external ratio RatioExt is limited to the value you have specified if a limit is violated. This is then also used to form the output value Out. If these limits are reached or violated RatioHiAct or RatioLoAct are also set to 1.

The internal ratio RatioInt is checked against the value you have specified. If a value is outside the specified limit, it is reset to the most recent valid value.

Limiting the output value

The output value is limited by the OutHiLim parameter (high) or OutLoLim parameter (low).

The OutHiAct or OutLoAct output parameter is set to 1 as soon as the output value has reached or exceeded the limits

Simulating signals

This block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Analog input value (SimIn, SimInLi)

Display and operator input area for process values and setpoints

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Selecting a unit of measure

The block provides the standard function Selecting a unit of measure (Page 207).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can switch to external
5	1 = Operator can switch to internal
6	1 = Operator can specify the internal ratio
7 - 9	Not used
10	1 =Operator can change the simulation value `SimIn`
11	1 = Operator can switch to "Simulation" mode
12	1 = Operator can enable bumpless switchover
13	Not used
14	1 = Operator can change `RatHiLim`
15	1 = Operator can change `RatLoLim`
16	1 = Operator can change `OutHiLim`
17	1 = Operator can change `OutLoLim`
18 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Forming and outputting signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` is formed from the following parameters:

- `Out.ST`
- `SecComPV.ST`
- `InPV.ST`
- `RatioExt.ST`
- `Offset.ST`

The signal status of the `Out` output parameter corresponds to the input parameter `In`.

The signal status of the current ratio calculation corresponds to the status of the input parameter `SecComPV`.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the Feature parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
24	Enabling local operator authorization (Page 157)

See also

Description of Ratio (Page 885)

Ratio messaging (Page 891)

Ratio I/Os (Page 891)

Ratio block diagram (Page 895)

Ratio error handling (Page 890)

Ratio modes (Page 887)

5.11.4 Ratio error handling

Ratio error handling

Refer to section Error handling (Page 119) for basic instructions on how to troubleshoot all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when implementing the block; block will not be processed.
0	No active fault
30	The input value of `In` can no longer be displayed in the REAL number field.
51	`RatLiOp` = 1 and `RatExtLi` = 1 and `RatInLi` = 1

See also

Ratio block diagram (Page 895)
Ratio I/Os (Page 891)
Ratio messaging (Page 891)
Ratio functions (Page 887)
Ratio modes (Page 887)
Description of Ratio (Page 885)

5.11.5 Ratio messaging

Messaging

This block does not offer messaging.

See also

Description of Ratio (Page 885)
Ratio functions (Page 887)
Ratio I/Os (Page 891)
Ratio block diagram (Page 895)
Ratio error handling (Page 890)
Ratio modes (Page 887)

5.11.6 Ratio I/Os

I/Os of Ratio

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 887)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0

Parameter	Description	Type	Default
`In`	Analog input	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`InPV`	Input for process variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`InUnit`	Unit of measure for input parameter `In`	INT	1001
`Offset`	Offset	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`OnOp*`	1 = "On" mode via operator	BOOL	0
`OosLi`	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`OosOp*`	1 = "Out of service", via OS operator	BOOL	0
`OpSt_In`	Input parameter for local operating permission, connected with the `Out` output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
`OS_Perm`	I/O for operator control permissions (Page 887)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
`OutHiLim`	High limit for output value	REAL	100.0
`OutLoLim`	Low limit for output value	REAL	0.0
`RatExtLi`	1 = Select external ratio (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`RatExtOp*`	1 = Select external ratio (via operator)	BOOL	0
`RatHiLim`	High limit	REAL	100.0
`RatIntLi`	1 = Select internal ratio (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`RatIntOp*`	1 = Select internal ratio (via operator)	BOOL	1
`RatioExt`	External ratio	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
`RatioInt*`	Internal ratio	REAL	1.0
`RatioUnit`	Unit of measure for the `RationInt`, `RatioExt` input parameter or `RatioPV` (output parameter)	INT	0

Parameter	Description	Type	Default
RatLiOp	Select ratio source (internal/external): 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RatLoLim	Low limit	REAL	0.0
RatOpScale	Limit for scale in bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
RatTrkExt	1 = Bumpless switchover from external to internal ratio active	BOOL	0
SecComPV*	Process value of secondary component	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SecComUnit	Unit of measure for input parameter SecComPV	INT	1001
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
SimIn*	Analog input value used for SimOn = 1	REAL	0.0
SimInLi	Value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see Ratio error handling (Page 890)	INT	-1

Parameter	Description	Type	Default
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
Out	Output	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
OutHiAct	1 = High limit overshoot	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutHiLmOut	Output of high limit	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
OutLoAct	1 = Low limit overshoot	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutLoLmOut	Output of low limit	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RatExtAct	1 = External ratio used 0 = Internal ratio used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RatHiAct	1 = Limit (high) for ratio active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RatioPV	Current ratio	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RatioOut	Applied ratio (RatioInt or RatioExt)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RatLoAct	1 = Limit (low) for ratio active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
ST_Worst	Worst signal status	BYTE	16#80
Status	Status word (Page 885)	DWORD	16#00000000

See also

Ratio messaging (Page 891)

Ratio block diagram (Page 895)

Ratio modes (Page 887)

5.11.7 Ratio block diagram

Ratio block diagram

A block diagram is not provided for this block.

See also

Ratio I/Os (Page 891)

Ratio messaging (Page 891)

Ratio error handling (Page 890)

Ratio functions (Page 887)

Ratio modes (Page 887)

Description of Ratio (Page 885)

5.11.8 Operator control and monitoring

5.11.8.1 Ratio views

Views of the Ratio block

The block Ratio provides the following views:

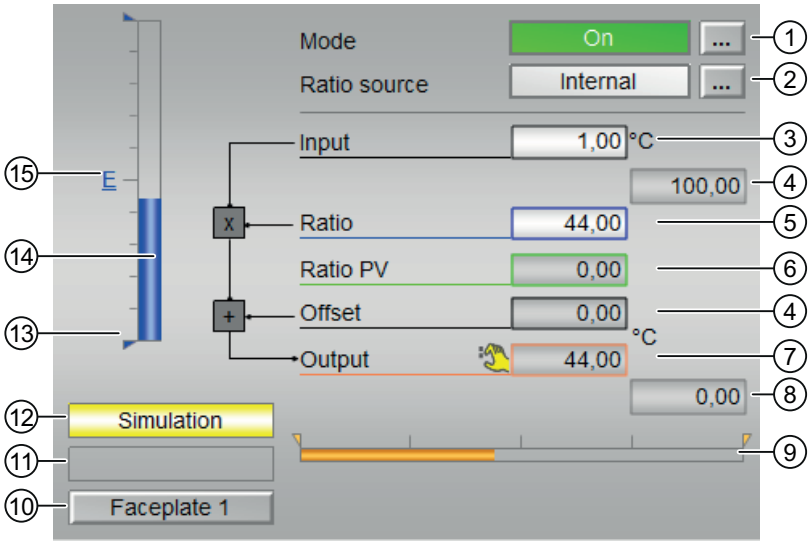
- Ratio standard view (Page 896)
- Trend view (Page 299)
- Ratio parameter view (Page 899)
- Ratio preview (Page 900)

- Memo view (Page 298)
- Block icon for Ratio (Page 901)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

5.11.8.2 Ratio standard view

Ratio standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) Displaying and switching the ratio source

This area shows you the currently valid signal source for the ratio setpoint. The following signal sources can be shown here:

- "External"
- "Internal"

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the signal source.

(3) Displaying the input value

The faceplate is a schematic representation of the function of the Ratio block as a signal flow chart:

Output = Input · Ratio + `Offset`

The input is typically the flow setpoint or actual value of the primary component of a ratio controller.

(4) High and low scale range for the ratio

The scale range is based on the bar graph for the ratio specification.

(5) Displaying and switching the default ratio

This area shows you the currently valid specification for the ratio.

The ratio source **(2)** needs to be set to "internal" in order to change this value.

Refer to the Changing values (Page 253) section for information on changing this value.

(6) Displaying the ratio `PV`

This area shows you the current ratio actual value with the corresponding signal status, i.e. the ratio of the actually measured `PV` from the active controller. The task of the ratio controller is to set the flow of all components so that the actual ratio approximates the specified ratio as closely as possible.

(7) Display of the `Offset`

This area shows the current `Offset`.

(8) Displaying the output value

This area shows the current output value `Out`, which typically serves as the setpoint for the flow of the secondary component.

(9) Bar graph for the output

This display graphically represents the output value with the limits set in the Engineering System (ES) (orange triangles, output parameters `OutHiLmOut` and `OutLoLmOut`).

(10) Displaying the limit

This status display is based on the limit of the output value `Out`.

(11) Display area for block states

This area shows if the output value has violated the range limits:

- "Output ≥ HL"
- "Output ≤ LL"

You can set the range limits in the parameter view (Page 899) of the block.

(12) Display area for block states

This area provides additional information on the operating state of the block:

- Simulation

You can find additional information on this in the Simulating signals (Page 58) section.

(13) Display of the limits for the ratio

This blue triangle shows the configured range limits for the ratio.

(14) Bar graph for the default ratio

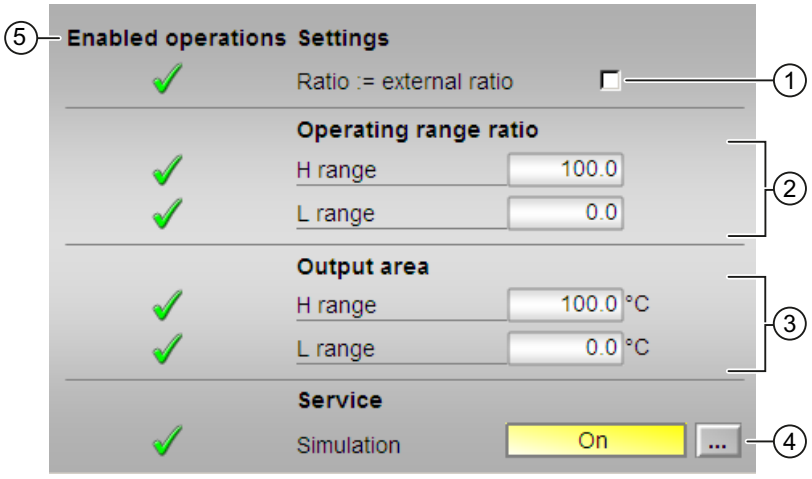
This area shows you the currently valid specification for the ratio in the form of a bar graph. The visible area in the bar graph depends on the configuration of the ratio in the engineering system (ES).

(15) Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

5.11.8.3 Ratio parameter view

Parameter view of Ratio



(1) Settings Ratio := external ratio

When the check box is selected ☑, the ratio is bumplessly switched from external to internal. The internal ratio value is tracked to the external one.

(2) Operating range ratio

This is where you specify the operating range for the ratio (input parameters `RatHiLim` or `RatLoLim`). The range is indicated by a blue triangle in the standard view of the bar graph.

(3) Range for output value

This is where you specify the operating range for the output value (input parameters `OutHiLmOut` or `OutLoLmOut`).

(4) Service

You can select the following functions in this area:

- "Simulation"

You can find information about this in the following chapters:

- Switching operating states and operating modes (Page 251)
- Simulating signals (Page 58)
- Release for maintenance (Page 64).

(5) Enabled operations

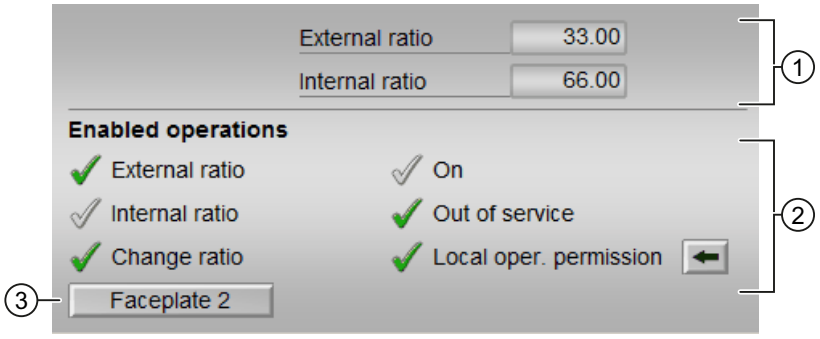
This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

5.11.8.4 Ratio preview

Preview of Ratio



(1) Preview area

This area shows you a preview for the following values:

- "External ratio": currently applicable external ratio
- "Internal ratio": currently applicable internal ratio

(2) Enabled operations

This area shows all operations for which special operating permissions are assigned. They depend on the configuration in the Engineering System (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter at all due to the configured AS operating permissions (`OS_Perm` or `OS1Perm`).

The following enabled operations are shown here:

- "Ratio external": You can change the external ratio.
- "Ratio internal": You can change the internal ratio.
- "Change ratio": You can change the ratio.
- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in chapter Operator control permissions (Page 248).

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the Engineering System (ES). The visibility of this navigation button depends on the configuration in the Engineering System (ES).

You can find additional information on this in chapter Opening additional faceplates (Page 203).

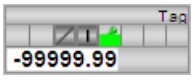
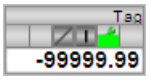
5.11.8.5 Block icon for Ratio

Block icons for Ratio

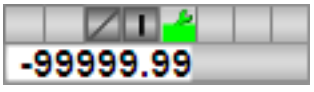
A variety of block icons are available with the following functions:

- Process tag type
- Display of the ratio
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits
- Operating modes
- Internal and external setpoint specification
- Signal status, release for maintenance
- Displays for bypassing interlocks
- Interlocks
- Memo display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

5.12 SplRange - Signal splitter

5.12.1 Description of SplRange

Object name (type + number) and family

Type + number: FC 372

Family: Control

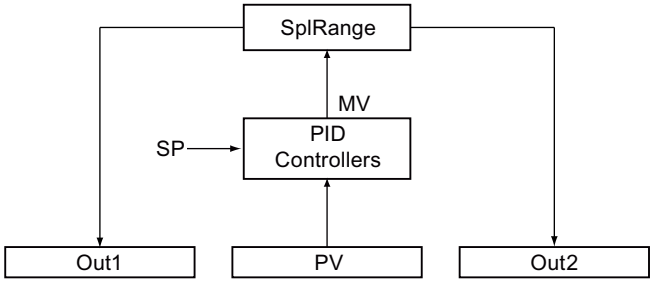
Area of application for SplRange

The block is used for the following applications:

- Splitting the output signal of a PID controller

How it works

The block is used to split signals output coming from a PID controller. You can use this controller to control two valves, for example.



The output parameters are calculated as follows:

`Out1` = `Out1Scale.Low` + ((`NeutPos` - `DeadBand` - `In`) / (`NeutPos` - `DeadBand` - `InScale.Low`)) · (`Out1Scale.High` - `Out1Scale.Low`)

`Out2` = `Out2Scale.Low` + ((`NeutPos` + `DeadBand` - `In`) / (`NeutPos` + `DeadBand` - `InScale.High`)) · (`Out2Scale.High` - `Out2Scale.Low`)

The neutral position (`NeutPos`) forms the reference point for selecting the individual splitter profiles. For details, refer to the SplRange functions (Page 906) section.

Refer to SplRange I/Os (Page 910) to learn the meaning of the parameters.

The block is installed in the run sequence downstream of the controller block. The manipulated variable output (`MV`) of the controller block is interconnected to the input `In` of the SplRange block.

The neutral position (`NeutPos`) and the dead band zone (`DeadBand`) can be set by means of the corresponding parameters. `DeadBand` must be configured to be less than `NeutPos`.

`Out1` and `Out2` are adapted to the physical variable by configuring the high/low limits of `Out1` and `Out2`.

The `Out1Act` or `Out2Act` output parameter indicates (= 1) that the corresponding `Out1` or `Out2` output parameter is enabled if the `In` input value is less than (for `Out1`) the neutral position (`NeutPos`) or the `In` input value is greater than the neutral position (`NeutPos`) depending on the dead band (`Deadband`).

Configuration

Use CFC editor to install the block in the OB in which the controller block runs whose manipulated variable is being processed.

There are templates (Templates) for the SplRange block for process tag types in the Advanced Process Library, one example (APL_Example_xx, xx refers to the language variant) with an application scenario for this block being:

- Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl) (Page 2309)

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

SplRange messaging (Page 909)

SplRange block diagram (Page 911)

SplRange error handling (Page 909)

SplRange modes (Page 905)

5.12.2 SplRange modes

SplRange operating modes

This block does not have any modes.

See also

SplRange I/Os (Page 910)

SplRange messaging (Page 909)

SplRange error handling (Page 909)

SplRange functions (Page 906)

Description of SplRange (Page 904)
SplRange block diagram (Page 911)

5.12.3 SplRange functions

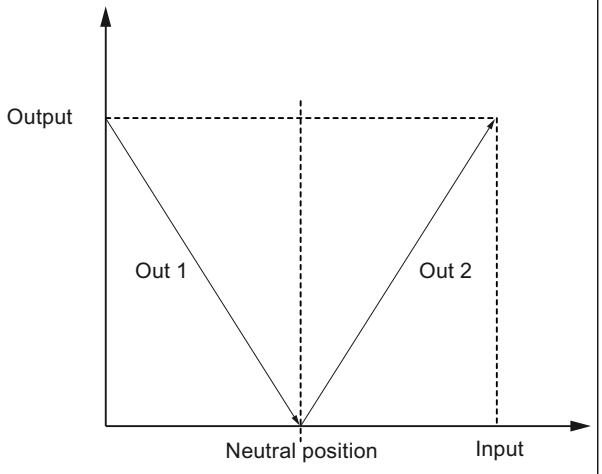
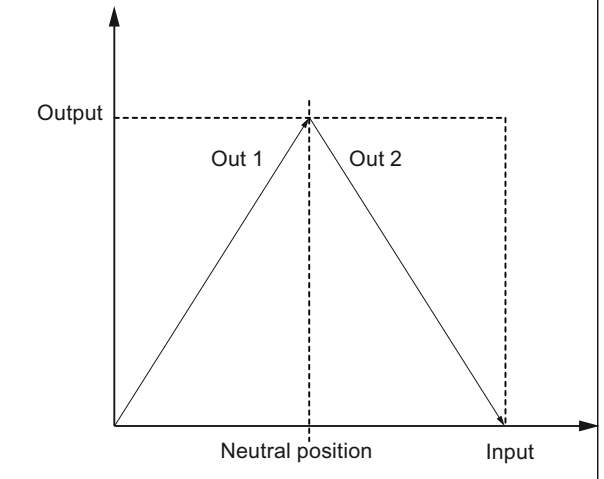
Functions of SplRange

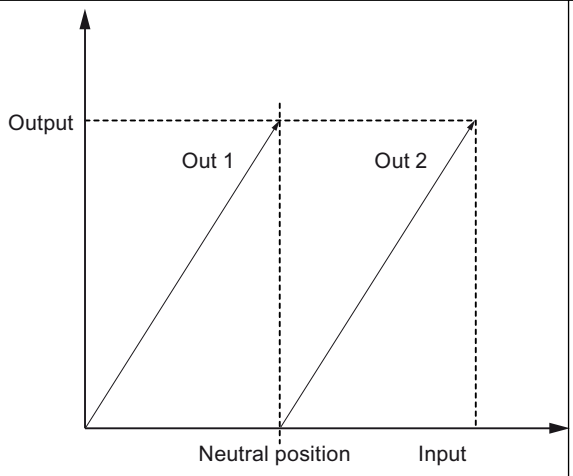
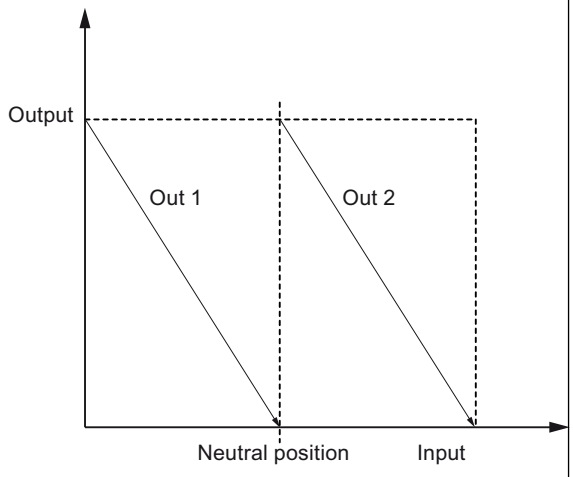
The functions for this block are listed below.

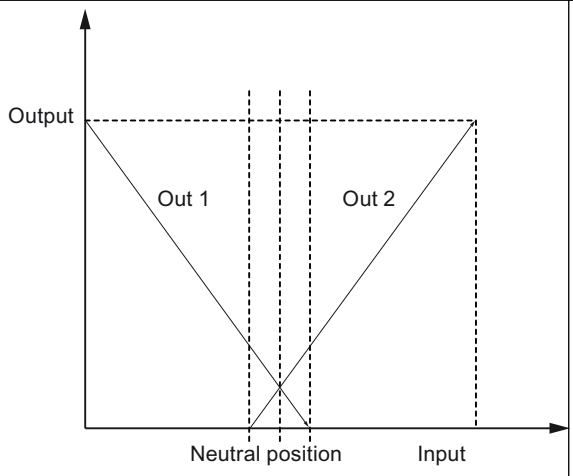
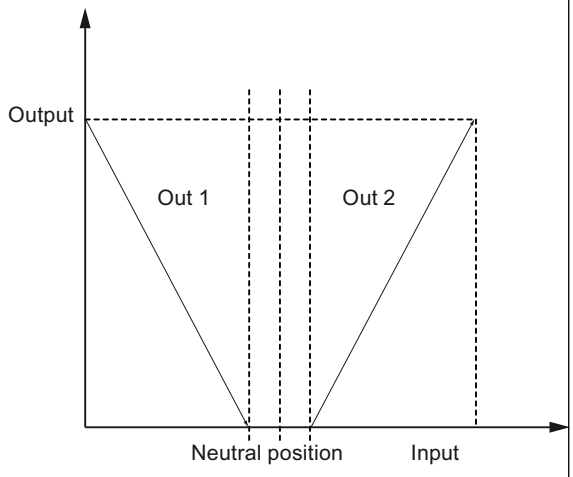
Splitting of the output signal of a controller

The `In` block input can be used to split the output signal of a controller.

The table below shows the six types of signal splitting that are available:

Case	Signal splitting	Setting the I/Os
1		`Out1Scale.Low` = 0.0 and `Out1Scale.High` = 100.0 `Out2Scale.Low` = 0.0 und `Out2Scale.High` = 100.0 `DeadBand` = 0.0 and `NeutPos` = 50.0
2		`Out1Scale.Low` = 100.0 and `Out1Scale.High` = 0.0 `Out2Scale.Low` = 100.0 and `Out2Scale.High` = 0.0 `DeadBand` = 0.0 and `NeutPos` = 50.0

Case	Signal splitting	Setting the I/Os
3		`Out1Scale.Low` = 100.0 and `Out1Scale.High` = 0.0 `Out2Scale.Low` = 0.0 and `Out2Scale.High = 100.0` `DeadBand` = 0.0 and `NeutPos` = 50.0
4		`Out1Scale.Low` = 0.0 and `Out1Scale.High` = 100.0 `Out2Scale.Low` = 100.0 and `Out2Scale.High` = 0.0 `DeadBand` = 0.0 and `NeutPos` = 50.0

Case	Signal splitting	Setting the I/Os
5		`Out1Scale.Low` = 0.0 and `Out1Scale.High` = 100.0 `Out2Scale.Low` = 0.0 and `Out2Scale.High` = 100.0 `DeadBand` = -10.0 and `NeutPos` = 50.0
6		`Out1Scale.Low` = 0.0 and `Out1Scale.High` = 100.0 `Out2Scale.Low` = 0.0 and `Out2Scale.High` = 100.0 `DeadBand` = 10.0 and `NeutPos` = 50.0

See also

Description of SplRange (Page 904)
SplRange messaging (Page 909)
SplRange I/Os (Page 910)
SplRange block diagram (Page 911)
SplRange error handling (Page 909)
SplRange modes (Page 905)

5.12.4 SplRange error handling

Error handling of SplRange

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
30	The value of `In` can no longer be displayed in the REAL number field.

See also

SplRange block diagram (Page 911)

SplRange I/Os (Page 910)

SplRange messaging (Page 909)

SplRange functions (Page 906)

SplRange modes (Page 905)

Description of SplRange (Page 904)

5.12.5 SplRange messaging

Messaging

This block does not offer messaging.

See also

Description of SplRange (Page 904)

SplRange functions (Page 906)

SplRange I/Os (Page 910)

SplRange block diagram (Page 911)

SplRange error handling (Page 909)

SplRange modes (Page 905)

5.12.6 SplRange I/Os

I/Os of SplRange

Input parameters

Parameter	Description	Type	Default
DeadBand	Width of dead band	REAL	0.0
EN	1 = Called block will be processed	BOOL	1
In	Input value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
InScale	Limit range for the input signal	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
NeutPos	Neutral position	REAL	50.0
Out1Scale	Limit range of output 1	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
Out2Scale	Limit range of output 2	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see SplRange error handling (Page 909).	INT	-1
Out1	Output value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out1Act	1 = Output 1 is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Out2	Output value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out2Act	1 = Output 2 is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Description of SplRange (Page 904)

SplRange functions (Page 906)

SplRange messaging (Page 909)

SplRange block diagram (Page 911)

SplRange modes (Page 905)

5.12.7 SplRange block diagram

SplRange block diagram

A block diagram is not provided for this block.

See also

SplRange I/Os (Page 910)

SplRange messaging (Page 909)

SplRange error handling (Page 909)

SplRange functions (Page 906)

Description of SplRange (Page 904)

SplRange modes (Page 905)

5.13 AutoExcitation - Process trigger for predictive controller

5.13.1 Description of AutoExcitation

Object name (type + number) and family

Type + number: FB 1842

Family: Control

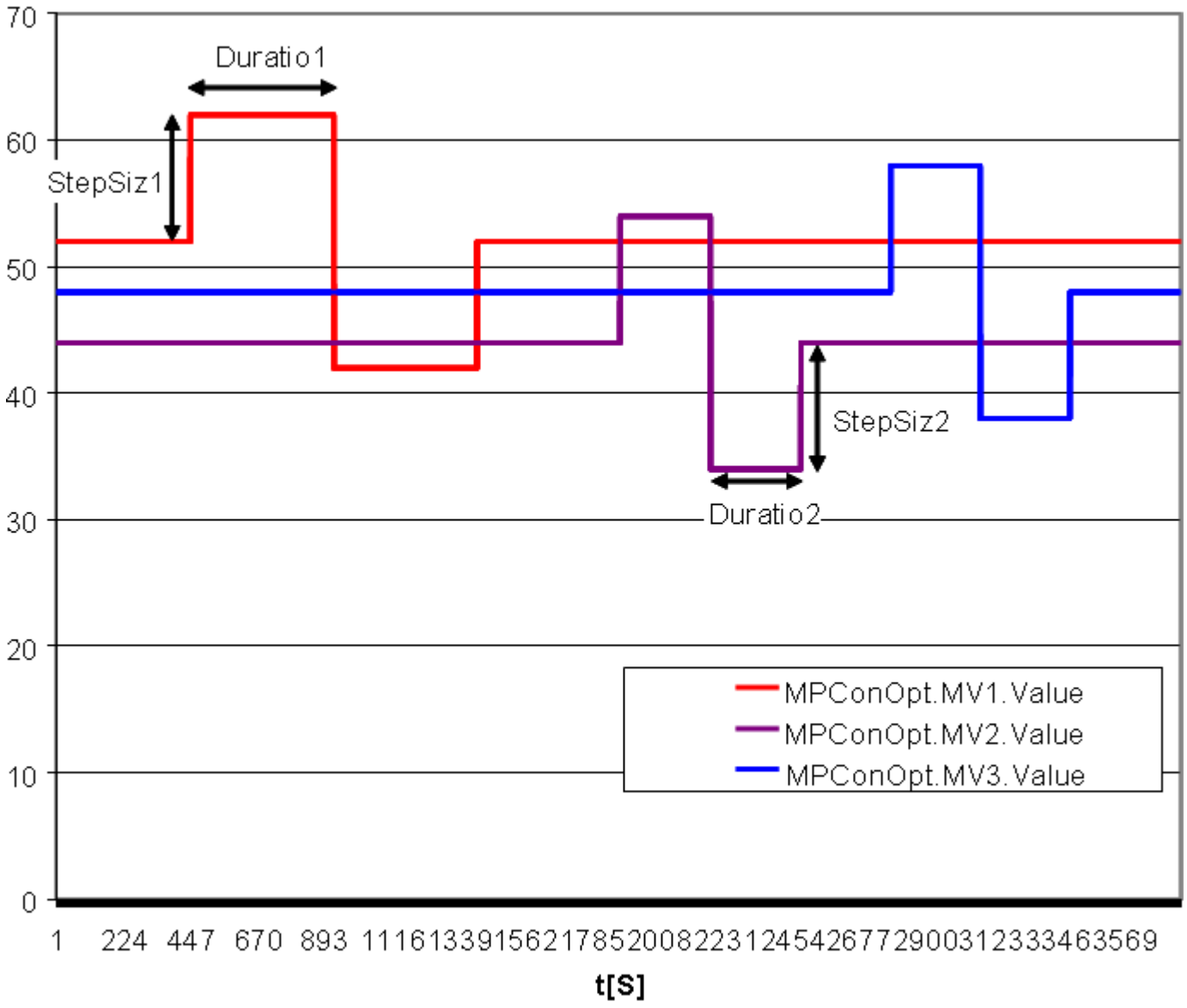
Area of application of AutoExcitation

This block is used to generate suitable excitation signals for the identification of dynamic multi-variable process models for the model-based predictive controllers ModPreCon and the large predictive controller MPC10x10. It is interconnected with aModPreCon or MPC10x10 block for this purpose.

How it works

The AutoExcitation block generates a sequence of abrupt trigger signals for the selected manipulated variables, MV1, MV2 to NumberMVs. A jump is initially made up, down, and then back to the starting point for each manipulated variable, so that the signals is symmetrical to the operating point.

The operating point taken from assigned controller via the MV1Actual...MV10Actual inputs. You set the jump height for each manipulated variable with the StepSiz1... StepSiz10 parameter and the jump duration via the Duratio1...Duratio10 parameter.



Configuration

The AutoExcitation block is part of the new process tag type, ModPreCon and MPC10x10. Only the new controller version has the necessary additional inputs for automatic triggering.

If the block retro-installed in an existing CFC, make sure that AutoExcitation andModPreCon or MPC10x10 are in the same OB cycle.

- Supply the AutoExcitation.MV1Actual...MV10Actual input variables with the appropriate ModPreCon.MV1...MV4 or MPC10x10.MV1...MV10 values from the controller.
- Specify the input parameter AutoExcitation.NumberMVs according to the number of manipulated variables you want to use in your MPC application. Exception: If you do not want to automatically trigger all manipulated variables of the controller, specify the number of manipulated variables to be directly triggered at the AutoExcitation.NumberMVs input parameter .
- Connect the AutoExcitation.MV1Excite...MV10Excite output variables to the appropriate ModPreCon.MV1Excite... MV4Excite or MPC10x10.MV1Excite…MV10Exciteinput variables
  on the controller.
- Connect the AutoExcitation.ExciteAct output variable to the ModPreCon.ExciteOn or MPC10x10.ExciteOn input variables of the controller.

5.13.2 AutoExcitation modes

Operating modes of AutoExcitation

This block does not have any modes.

5.13.3 AutoExcitation functions

Using AutoExcitation

Start triggering by setting the StartExcite binary input variable to the value 1 .

If you cancel triggering during ongoing operation via StartExcite =0 , the manipulated variables are reset to the original operating point.

When the Next binary input parameter = 1 , you can start the next jump manually before the planned time has expired.

The StepPhase output variable shows the current phase of the attempted jump:

0: Constant

1: Jump up

2: Jump down

3: Jump back to the operating point

The StepTime output variable indicates the remaining time in [s] until the next jump.

5.13.4 AutoExcitation error handling

Overview of error numbers

The ErrorNum output parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Default value when the block is installed, the block is not executed
0	There is no error.
60	One of the input parameters, Duratio1,2,3,4...10 < SampleTime or SampleTime < 0.001
61	The trigger signals are not adopted correctly by the assigned controller.

5.13.5 AutoExcitation messaging

Messaging of AutoExcitation

This block does not offer messaging.

5.13.6 AutoExcitation I/Os

I/Os of AutExcite

Input parameters

Parameter	Description	Type	Default
Duration1...Duration10	Duration for jump in MV1...MV10 in s	REAL	20
MV1Actual...MV10Actual	Current value of MV1...MV10	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Next*	1 = Manual start of the next trigger change	BOOL	0
SampleTime	Sampling time in s	REAL	1
StartExcite*	1 = Start of automatic triggering; 0= Cancel automatic triggering	BOOL	1
StepSize1...StepSize10	Step height of MV1...MV10	REAL	10

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ErrorNum	Output of pending error number	INT	0
ExciteAct	1 = Automatic triggering is active	BOOL	0
MV1Excitation...MV10Excitation	Value of the trigger for MV1...MV10	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
StepPhase	Phase of the jump; 0 = Constant, 1= Jump up, 2 = Jump down, 3 = Return	INT	0
StepTime	Time remaining until the next jump phase in s	REAL	0

5.13.7 AutoExcitation block diagram

Block diagram of AutoExcitation

A block diagram is not provided for this block.

5.14 LPOptim - Optimization after traversing the linear programming

5.14.1 Description of LPOptim

Object name (type + number) and family

Type + number: FB1844

Family: Control or system

Area of application of LPOptim

Optimization of a linear performance function taking into account constraints after traversing the linear programming.

The LPOptim block is used by the ModPreCon block for static operating point optimization (Page 682). However, it can also be used separately.

Operating principle

The performance function has a linear relationship to up to four decision variables, x1...x4, with corresponding weighting factors, g1...g4 .

J = g1*x1 + g2*x2 + g3*x3 + g4*x4

The performance function is a hyper-level in a five dimensional space whose gradient direction is described by the gradient vector, Gradient = [g1, g2, g3, g4].

The block can determine both the maximum as well as the minimum of the J performance function. A minimization problem is transformed into an equivalent maximization problem by inverting the gradient vector.

The decision variables are positive and limited by a system of up to eight linear non-equivalence constraints .

The first constraint is, for example

a11*x1+a12*x2+a13*x3+a14*x4 <= b1

All four elements of the first line from the A matrix are combined in the A1n data structure. All eight elements of the B vector are combined in the Begrenzungen data structure.

How it works

The block operates according to the simplex procedure, i.e. based on an iterative algorithm. Due to the linear performance criterion, the solution always lies at an intersection of multiple restrictions.

Configuration

The block can be installed in OB1 because no time-critical calculations are performed. If the block is called by ModPreCon , the user does not have to set parameters directly at LPOptim block.

5.14.2 LPOptim modes

Operating modes of LPOptim

This block does not have any modes.

5.14.3 LPOptim functions

Use of LPOptim

The algorithm is started with a positive edge at the Start parameter or if there is a change in one of the input variables (gradient vector and limitations), and displays the optimum variable found at the XOpt1...XOpt4 output parameters when execution is completed.

If no solution is found (non-plausible coefficient), execution is aborted with an error code.

5.14.4 LPOptim error handling

Overview of error numbers

The Status output parameter can be used to display the following states:

Status	Meaning
0	There is no error.
10	Unrestricted target function
15	No XOpt found
20	Infinite number of solutions
35	Unlimited permissible range
40	Configuration error

5.14.5 LPOptim messaging

Messaging

This block does not offer messaging.

5.14.6 LPOptim I/Os

I/Os of LPOptim

Input parameters

Parameter	Description	Type	Default
J_Mini	1= Search minimum 0= Search maximum	BOOL	0
Start*	1= Start optimization calculation immediately 0= Start optimization calculation only if one of the input variables has changed	BOOL	0
X1	Decision variable 1 for calculation of J_Act	REAL	0
X2	Decision variable 2 for calculation of J_Act	REAL	0
X3	Decision variable 3 for calculation of J_Act	REAL	0
X4	Decision variable 4 for calculation of J_Act	REAL	0
J_0	Constant part of the performance criterion, irrespective of the decision variables	REAL	0
N_X	Number of decision variables, N_X <= 4	INT	4
N_b	Number of constraints, N_b <= 8	INT	8
Gradient.g1* ... Gradient.g4*	Gradient vector with up to 4 REAL elements	STRUCT	[1, 1, 1, 1]
Constraints.b1 ... Constraints.b8	Vector B of the limits with up to 8 REAL elements	STRUCT	[0, 0, 0, 0, 0, 0, 0, 0]
A1n.a11 A1n.a12 A1n.a13 A1n.a14	1. Line of the A matrix contains factors for the elements of vector X in the first constraint	STRUCT	[0, 0, 0, 0]
A2n.a21 A2n.a22 A2n.a23 A2n.a24	2. Line of the A matrix contains factors for the elements of vector X in the second constraint	STRUCT	[0, 0, 0, 0]
A3n.a31 A3n.a32 A3n.a33 A3n.a34	3. Line of the A matrix contains factors for the elements of vector X in the third constraint	STRUCT	[0, 0, 0, 0]

Parameter	Description	Type	Default
A4n.a41 A4n.a42 A4n.a43 A4n.a44	4. Line of the A matrix contains factors for the elements of vector X in the fourth constraint	STRUCT	[0, 0, 0, 0]
A5n.a51 A5n.a52 A5n.a53 A5n.a54	5. Line of the A matrix contains factors for the elements of vector X in the fifth constraint	STRUCT	[0, 0, 0, 0]
A6n.a61 A6n.a62 A6n.a63 A6n.a64	6. Line of the A matrix contains factors for the elements of vector X in the sixth constraint	STRUCT	[0, 0, 0, 0]
A7n.a71 A7n.a72 A7n.a73 A7n.a74	7. Line of the A matrix contains factors for the elements of vector X in the seventh constraint	STRUCT	[0, 0, 0, 0]
A8n.a81 A8n.a82 A8n.a83 A8n.a84	8. Line of the A matrix contains factors for the elements of vector X in the eighth constraint	STRUCT	[0, 0, 0, 0]

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
Bad	1= Status bad, optimization has failed	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see the error handling of LPOptim.	INT	0
J_Opt	Value of the performance criterion at the optimum	REAL	0
J_Act	Value of the performance criterion for the current assignment of the decision variables in accordance with input parameters X1...X4	REAL	0
X1Opt	Optimum value of the 1st decision variable	REAL	0
X2Opt	Optimum value of the 2nd decision variable	REAL	0
X3Opt	Optimum value of the 3rd decision variable	REAL	0
X4Opt	Optimum value of the 4th decision variable	REAL	0

See also

LPOptim error handling (Page 918)

5.14.7 LPOptim block diagram

LPOptim block diagram

A block diagram is not provided for this block.

5.15 MPC10x10 - Large predictive controller

5.15.1 Description of MPC10x10

Object name (type + number) and family

Type + number: FB 1920

Family: Control

Area of application for MPC10x10

The block is used for multivariable control, similar to the ModPreCon, typically as master controller in cascade structures.

Compared to the lean multivariable controller ModPreCon, the MPC10x10 has the following functional expansions:

- Greater capacity: up to 10x10 interacting manipulated and controlled variables (compared with ModPreCon: 4x4), up to 4 measurable disturbances for a dynamic disturbance compensation (feedforward) (ModPreCon: 1). Unused MV channels can be rededicated to disturbance variables, whereby the total number of manipulated variables and disturbances is limited to 14 (ModPreCon: 5).
- Dynamic online optimization considering constraints in each sampling step. The optimization problem is resolved without constraints with ModPreCon and the manipulated variables are limited afterward. This may result in solutions that are less than optimal if several constraints are active simultaneously in automatic mode.
- Processing of (low-priority) target values for the manipulated variables, which is mainly of interest for control tasks in which there are more manipulated variables than controlled variables or in which some controlled variables do not have to be kept at an exact setpoint.
- Integrated steady state operating point optimization even for control tasks in which the number of manipulated variables does not match the number of controlled variables.

Method of operation and area of application

The block is used for Multivariable controller (Page 2364) dynamic processes. It can handle up to ten interacting manipulated and controlled variables as well as up to four measurable disturbances.

The MPC10x10 algorithm only works for stable processes with a step response that settles to a fixed value in a finite time. If the process is unstable in one of the main transfer functions or includes an integrator (level control, for example), the corresponding partial transfer function must be stabilized with a secondary controller.

A simple P controller (proportional component only) is sufficient as a slave controller for integrating processes. It is inserted between the manipulated variable output of MPC10x10 and the input of the unstable process section, and receives the output of the integrating process section as a manipulated variable. (Unstable data links are stabilized with this approach.)

You can find an explanation of "multivariable controls" and "non-phase-minimum response" in the Help on the Advanced Process Library > Definitions.

Note on the area of application of the controller: Longer run times

Due to the principle on which it works, the runtime of MPC10x10 is significantly longer than that of ModPreCon and obviously much longer than that for PID controllers, because very large matrices have to be multiplied in the algorithm. The runtime is also determined by the number of the process and manipulated variables in the control algorithm. This is why the MPC10x10 is unsuitable for high-speed controls and is usually used for slow, complex control tasks in which controller sample times of more than 4 s are permitted.

The computing time required on the CPU is compensated for by the fact that very slow sample times of > 20 s are used for the typical MPC10x10 applications (see Advanced control templates). The MPC10x10 is then typically located in OB30 and can be interrupted by faster OBs. Some parts of the MPC10x10 algorithm have been outsourced to auxiliary blocks due to the limitations of the address space for SIMATIC function blocks and data blocks. The MPC10x10 uses two outsourced optimization methods:

- MPC_LP_OptiFB for operating point optimization which, in turn, needs an auxiliary block MPC_LPOptiFC .
- MPC_ActSetFB

as well as auxiliary blocks for matrices that are created by the MPC Configurator and can all be identified by the prefix MPC_* in their name.

There are different forms of implementation for the matrices:

- User DBs that are filled with data by one or more FCs during initialization because large matrices cannot be pre-assigned with values in the source code for the DB:
- Data block MPC_g with initialization FCs MPC_g_FC, MPC_gb…ge;
- Data blocks MPC_ESP, MPC_Hu_FB, MPC_Hinv_FB, MPC_Omega_FB and MPC_M_FB each with a dedicated initialization function MPC_Esp_FC... as well as
- a data block MPC_RgMVi (i=1..10) for each MV with initialization FCs MPC_RgMVia_FC and MPC_RgMVib_FC.
- User DBs filled with data by the MPC10x10 during runtime: MPC_Nn, MPC_Ms, MPC_MsOld.

As user, you only have to interconnect the main function block in the CFC and need not worry about the auxiliary blocks.

The static operating point optimization is called within the MPC10x10 block in the program section in which OB1 is executed. In cyclic operation OB3x, this avoids the additional computing time required by the optimizer, the acyclic time, i.e. which is only relevant for changes to the optimizer inputs.

Operating principle

The MPC10x10 block is a model-based predictive multivariable controller. It uses a mathematical model of the process dynamics including all interactions as part of the controller.

This model allows the process response to be predicted over a defined period in the future, also known as the prediction horizon.

Based on this prediction, a performance index

$$J = \left(\vec{w} - \vec{y}\right)^T R\left(\vec{w} - \vec{y}\right) + \Delta\vec{u}^T Q \Delta\vec{u} + \left(\vec{u}_{soll} - \vec{u}\right)^T S\left(\vec{u}_{soll} - \vec{u}\right)$$

is optimized (minimized) where the following applies:

- *w* contains the time series of the future setpoints within the prediction horizon,
- *y* contains the vector of the controlled variables in the future,
- *Δu* contains the future changes of the manipulated variables within the control horizon,
- u_{soll} contains the target values for the manipulated variables *u*

The control horizon is generally significantly shorter than the prediction horizon. Changes of future manipulated variables are only planned within the control horizon.

If you increase the weighting in the Q diagonal matrix, the controller moves its manipulated variables more cautiously resulting in a slower but more robust control response. Using the weighting factors in the R diagonal matrix, you specify the relative significance of the individual controlled variables. A higher weighting (priority) for a controlled variable means that this CV moves more quickly towards the setpoint and remains more accurately at the setpoint in steady state if it is not possible to achieve all setpoints precisely.

The weightings for the target values of the manipulated variables in the S matrix are typically set much lower than the weightings of the controlled variables in the R matrix so that the controller must only handle the manipulated variable target values once it has reached all primary control aims regarding the controlled variables.

The algorithm is a variant of the DMC method (**D**ynamic **M**atrix **C**ontrol), comparable to the control algorithms in so-called "full-blown" MPC software packages that were previously linked to the process control system by external PCs.

The optimization problem is solved in ongoing automatic mode in each sampling step of the controller by initially neglecting the restrictions. If a violation of the limitations is detected within the control horizon during the process, an iterative optimization with the so-called "active set" method takes place. Only the currently active limitations are taken into consideration during each iteration step of the optimization. Manipulated variable limitations, both absolute and relating to the gradients, are treated in the algorithm of the function block as hard constraints that must not be violated. This means that precise setpoints or target zones for the controlled variables are taken into account during optimization as well as possible. The target zones for the controlled variables are therefore soft limits, which are maintained as well as possible although they cannot be guaranteed. Using a reference variable filter for future setpoint settings, the control response of the controller can be fine-tuned during operation.

You can achieve significant improvements in control performance when individual disturbances can be measured, for example, variations in throughput. In this case, it is a good idea to take into account a model of the influence of this disturbance on the controlled variables when predicting the controlled variables so that the controller can react preemptively to such disturbances.

Operating point optimization

The integrated static operating point optimization can be used when at least one controlled variable provides certain degrees of freedom. No exact setpoint is specified for such controlled variables. Instead there is a tolerance band, e.g.

- SP2OptHiLim…SP2OptLoLim

within which the process value, CV2, must remain. These areas can be defined for any subset of the relevant controlled variables. From an economic perspective, different values within the tolerance range can be more or less favorable. With the help of the optimization function, the optimal economic point can be found within the tolerance range. This is done by defining a target function (performance criterion), which depends on the manipulated variable and controlled variable of the predictive controller. This can be, for example, the economic yield of plant operation per time unit or it may involve specific costs or energy consumption.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100) and in OB 1.

MPC10x10 and all its auxiliary blocks are provided in the block folder "MPC10x10 blocks" of the Advanced Process Library. After you have placed the block in the CFC, you still need to copy the fixed addressed global data blocks DB1 to DB22 into the project. This will make all auxiliary blocks of MPC10x10 available in the project, thereby making it executable and allowing the SCL source of the MPC configurator to be later compiled.

In contrast to the ModPreCon, only one instance of the MPC10x10 can be configured per SIMATIC CPU. The addresses of the MPC auxiliary blocks are determined during compilation of the main block by the SCL Compiler and cannot be modified afterwards. The addresses for the auxiliary blocks (DBs and FCs) used by the MPC10x10 are defined in the following symbol table. If you already have other blocks in the project assigned to the specified addresses, you need to move them.

Icon	Address	Data type	Comment
MPC10x10	FB 1920	FB 1920	Model Predictive Controller with online optim.
MPC_LP_OptiFB	FB 1922	FB 1922	Linear Optimization
MPC_ActSetFB	FB 1923	FB 1923	Modified Simplex Optimization with Active Set Problem Reduction for ModPreConL
MPC_LPOptiFC	FC 400	FC 400	Linear Optimization
MPC_g_FC	FC 401	FC 401	MPC Configurator
MPC_gb_FC	FC 402	FC 402	MPC Configurator
MPC_gc_FC	FC 403	FC 403	MPC Configurator
MPC_gd_FC	FC 404	FC 404	MPC Configurator
MPC_ge_FC	FC 405	FC 405	MPC Configurator
MPC_gz_FC	FC 406	FC 406	MPC Configurator
MPC_Hu_FC	FC 407	FC 407	MPC Configurator
MPC_Hinv_FC	FC 408	FC 408	MPC Configurator
MPC_Omega_FC	FC 409	FC 409	MPC Configurator
MPC_M_FC	FC 410	FC 410	MPC Configurator

MPC_RgMV1a_FC	FC 411	FC 411	MPC Configurator
MPC_RgMV1b_FC	FC 412	FC 412	MPC Configurator
MPC_RgMV2a_FC	FC 413	FC 413	MPC Configurator
MPC_RgMV2b_FC	FC 414	FC 414	MPC Configurator
MPC_RgMV3a_FC	FC 415	FC 415	MPC Configurator
MPC_RgMV3b_FC	FC 416	FC 416	MPC Configurator
MPC_RgMV4a_FC	FC 417	FC 417	MPC Configurator
MPC_RgMV4b_FC	FC 418	FC 418	MPC Configurator
MPC_RgMV5a_FC	FC 419	FC 419	MPC Configurator
MPC_RgMV5b_FC	FC 420	FC 420	MPC Configurator
MPC_RgMV6a_FC	FC 421	FC 421	MPC Configurator
MPC_RgMV6b_FC	FC 422	FC 422	MPC Configurator
MPC_RgMV7a_FC	FC 423	FC 423	MPC Configurator
MPC_RgMV7b_FC	FC 424	FC 424	MPC Configurator
MPC_RgMV8a_FC	FC 425	FC 425	MPC Configurator
MPC_RgMV8b_FC	FC 426	FC 426	MPC Configurator
MPC_RgMV9a_FC	FC 427	FC 427	MPC Configurator
MPC_RgMV9b_FC	FC 428	FC 428	MPC Configurator
MPC_RgMV10a_FC	FC 429	FC 429	MPC Configurator
MPC_RgMV10b_FC	FC 430	FC 430	MPC Configurator
MPC_Esp_FC	FC 432	FC 432	MPC Configurator
MPC_Ms	DB 1	DB 1	MPC Configurator
MPC_Nn	DB 2	DB 2	MPC Configurator
MPC_MsOld	DB 3	DB 3	MPC Configurator
MPC_g	DB 4	DB 4	MPC Configurator
MPC_gz	DB 5	DB 5	MPC Configurator
MPC_Hu	DB 6	DB 6	MPC Configurator
MPC_Hinv	DB 7	DB 7	MPC Configurator
MPC_Om	DB 8	DB 8	MPC Configurator
MPC_M	DB 9	DB 9	MPC Configurator
MPC_RgMV1	DB 10	DB 10	MPC Configurator
MPC_RgMV2	DB 11	DB 11	MPC Configurator
MPC_RgMV3	DB 12	DB 12	MPC Configurator
MPC_RgMV4	DB 13	DB 13	MPC Configurator
MPC_RgMV5	DB 14	DB 14	MPC Configurator
MPC_RgMV6	DB 15	DB 15	MPC Configurator
MPC_RgMV7	DB 16	DB 16	MPC Configurator
MPC_RgMV8	DB 17	DB 17	MPC Configurator
MPC_RgMV9	DB 18	DB 18	MPC Configurator
MPC_RgMV10	DB 19	DB 19	MPC Configurator
MPC_Esp	DB 20	DB 20	MPC Configurator
MPC_LP_Opti	DB 21	FB 1922	Instance DB FB 1922
MPC_ActSet	DB 22	FB 1923	Instance DB FB 1923

After installation in CFC, follow the steps outlined below:

1. Excite the process with the controller in manual mode by applying a series of manipulated variable step changes. We recommend using the auxiliary function block, AutoExcitation, to generate excitation signals.
2. Record the measured data with the CFC trend display and export it to an archive file. You can also use a WinCC TagLogging TrendControl to record the data and to export the *.csv file. If the total number of manipulated and controlled variables exceeds the maximum number of trends in the CFC, you will have to use the WinCC TrendControl anyway.
3. Select the MPC10x10 instance in the CFC. Start the MPC Configurator under Edit > MPC Configurator.
   The working steps process identification, controller design and simulation of the closed control loop are executed in the MPC Configurator. You can find a detailed description of this procedure in the MPC Configurator help. You can find the help with a button in the MPC Configurator or directly under …\Program Files (x86)\SIEMENS\STEP7\S7JMPC \s7jmpctb.chm.
4. Using the Configurator, create an SCL source code for the user data blocks (DBs) and other auxiliary blocks (FCs). The source contains all models and matrices required for an MPC10x10 instance.
5. Compile the SCL source with the S7-SCL compiler and archive the SCL source with the controller data.

Load the changes to the AS (with the option: Include user data blocks)

The values in the auxiliary blocks are applied in the controller by restarting the block by means of the input variable Restart:= true at the CFC block instance.

Note

During controller design in the MPC Configurator, a controller cycle time and an OB sampling time are calculated, displayed, and stored in the user data block. You yourself are responsible for ensuring that the MPC10x10 block is called in the cyclic interrupt level suitable for the OB sampling time. This is checked in the current MPC10x10 version during initialization. If the SampleTime parameter of the function block does not match the OB_SampleTime parameter of the user data block, a parameter assignment error (ErrorNum=3) is displayed. For controller cycle times greater than 5s, insert the MPC10x10 block in the OB30 and specify the appropriate cycle time for the OB30 in the hardware configuration of the SIMATIC CPU. Controller cycle times slower than 20 s cannot be set in the hardware configuration. The block would then be called every 20s and the slower sampling time automatically realized by an internal cycle reduction ratio in the block.

There are no templates for the MPC10x10 block for a process tag type in the Advanced Process Library because the controller is typically configured individually as master controller for several slave controllers.

Startup characteristics

When the CPU starts up, the block always starts in manual mode. It is only possible to change to automatic mode when a user data block is loaded and the internal measured value memory in MPC10x10 is filled with data.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section I/Os of MPC10x10 I/Os (Page 943)

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutAct.Value`
6	Not used
7	`ManAct.Value`
8 - 9	Not used
10	`MV1TrkOn.Value` AND NOT (`ManAct.Value` OR `OosAct.Value`)
11	`MV2TrkOn.Value` AND NOT (`ManAct.Value` OR `OosAct.Value`)
12	`MV3TrkOn.Value` AND NOT (`ManAct.Value` OR `OosAct.Value`)
13	`MV4TrkOn.Value` AND NOT (`ManAct.Value` OR `OosAct.Value`)
14	Not used
15	`DB_Loaded`
16	`DV_Model Available`
17	OptimAct
18	`NOT(OptimAct)`
19	Not used
20	J_Mini
21	NOT(J_Mini)
22	ExciteOn AND ManAct.Value
23	`MV5TrkOn.Value AND NOT (ManAct.Value OR OosAct.Value)`
24	`MV6TrkOn.Value` AND NOT (`ManAct.Value OR OosAct.Value`)
25	`MV7TrkOn.Value AND NOT (ManAct.Value OR OosAct.Value)`
26	`MV8TrkOn.Value` AND NOT (`ManAct.Value OR OosAct.Value`)
27	`MV9TrkOn.Value AND NOT (ManAct.Value OR OosAct.Value)`
28	`MV10TrkOn.Value` AND NOT (`ManAct.Value OR OosAct.Value`)
29	Tracking mode; i.e. at least one of the original manipulated variables (without pseudo MVs, which are used as DVs) is tracked.
30	Not allocated
31	Feature.Bit31: Display of the predictions in the faceplate

Status word allocation for `Status2` parameter

Status bit	Parameter
0 - 30	Not used
31	`MS_RelOp`

5.15.2 MPC10x10 modes

Operating modes of MPC10x10

The block can be operated using the following modes:

- Automatic mode (Page 72)
- Manual mode (Page 72)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

The aforementioned operating modes are valid for the block with all control channels (`MV1` ... `MV10`). Moreover, individual control channels can be tracked; see section MPC10x10 functions (Page 930) for more information.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and bumpless switchover in the section Manual and automatic mode for control blocks (Page 72) .

Note

In contrast to PID controllers, it is permitted to run the MPC10x10 block in "automatic mode" without its actuating signals affecting the process because there is no risk of integrator windup.

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for control blocks (Page 72) .

"Out of service"

You can find general information about the "Out of service" mode in the Out of service (Page 71) .

5.15.3 MPC10x10 functions

Functions of MPC10x10

The functions for this block are listed below.

Generating and limiting the manipulated variable

The manipulated variable `MV1` ... `MV4` (hereinafter referred to as `MVx`, x = 1 ... 10) can be generated as follows:

ManAct	MVxTrkOn	MVx	Limit monitoring	State
1	-	`Manx`	`ManxHiLim` `ManxLoLim`	"Manual mode", set by the operator
0	1	`MVxTrk`	`MVxHiLim` `MVxLoLim`	Tracking with limitation
0	0	Automatic manipulated variable	`MVxHiLim` `MVxLoLim`	"Automatic mode": Predictive controller algorithm

Remark

If the controller is in "Out of service" mode, the output parameters `MV1` ... `MV10`, depending on the `Feature` Bit (Neutral position manipulated variable takes effect at startup (Page 165)), are set to the last valid value in manual mode or the corresponding neutral position manipulated variable (`SafePos1` ... `SafePos10`). Refer to the Out of service (Page 71) section for more on this.

The limited operating range (between `MVxHiLim` and `MVxLoLim`) is typically smaller in automatic mode than in manual mode. With regard to the limited range of validity of a linear process model for approximating a non-linear process response, this allows the stability of the closed control loop to be guaranteed within the control range in automatic mode.

The gradients of the manipulated variable (changes per second) are limited to `MV1RaLim` to `MV10RaLim` in "automatic mode". Gradient limitation applies both to the positive and negative directions.

Tracking and limiting a manipulated variable

The block provides the standard function Tracking and limiting a manipulated variable (Page 192).

In contrast to PID controllers, tracking the manipulated variables (`MV1` ... `MV10`) is enabled for specific channels via one of the input parameters `MV1TrkOn` ... `MV10TrkOn`. The corresponding manipulated variable is then tracked by the interconnectable input parameters `MV1Trk` ... `MV10Trk`.

Setting the setpoint internally

With this block, the setpoint must always be set internally at the I/Os `SP1` ... `SP10`. These are normally operated in the faceplate. In special situations, you can interconnect the setpoints but they can then no longer be changed using the faceplate.

Setpoint tracking in manual mode

In this situation (SP_TrackCV = 1), the internal setpoints SP1 ... SP10 are tracked to the assigned process values CV1 ... CV10 in "manual mode". This function allows a bumpless transfer to "automatic mode". After the transfer, the setpoints can be changed by the operator again.

Setpoint filters

The setpoint filter is the only way of changing the action of the predictive controller without having to create a new user data block with the MPC Configurator and reinitialize the controller. The specified time constant PreFilt1 ... PreFilt10 of the setpoint filter can be interpreted as the required settling time of this CV channel following a setpoint step change. As the time constant setting increases, the controller works more slowly and less aggressively. In particular, this reduces the influence of a setpoint step change in one control channel on neighboring control channels.

Internally, the MPC10x10 block works with sets of future setpoints that are compared with the predicted movements of the controlled variables. Without the setpoint filter, it is assumed that the current setpoint will continue to remain valid in the future within the prediction horizon. If there is a setpoint step change, this means that the full value of the new setpoint will be required in the near future although the process cannot achieve this (according to the prediction). With the setpoint filter, an asymptotic setpoint trajectory (first order) is calculated from the current process value to the required setpoint so that the required setpoint is reached in the specified time.

Note

The setpoint filter also comes into effect without a setpoint step if the process value deviates significantly from the setpoint due to disturbances. This means that the filter not only slows down the control response but indirectly also the response to disturbances.

The control response can only be slowed down by the setpoint filter and not accelerated; when the value is 0, the prefilter is deactivated. It is therefore advisable to set the basic controller action in the MPC Configurator with the "Manipulated variable change penalty" parameter and then to optimize this in the software using the function for simulation of the closed control loop. The setpoint filter should then only be used for fine tuning of the action in the operational system.

Simulating signals

The block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Controlled variable (SimCVx , SimCVxLi)

Selecting a unit of measure

The block provides the standard function Selecting a unit of measure (Page 207).

Error signal generation and dead band

The block provides the standard function Error signal generation and dead band (Page 188).

In the predictive controller, the error signal is generated over the entire prediction horizon for each control channel as the deviation between the predicted movement of the process (starting at the current process value CV1 ... CV4) and future setpoint settings (ending at SP1 ... SP10) and used to calculate the manipulated variable.

In principle, the effect of the dead bands SP1DeadBand to SP10Deadband is the same as in a PID controller, but extends over the entire future prediction horizon. In other words, if, for example, the predicted controlled variable CV1 in the entire prediction horizon is within the band SP1 ± SP1DeadBand, the controller sees no reason whatsoever to change any manipulated variable. These are therefore also known as CV bands. In contrast to the manipulated variable limits, these are not hard constraints that need to be adhered to at all costs.

In multivariable controllers, it is advisable to make use of the fact that from the perspective of the application only some of the controlled variables need to move to a specified setpoint exactly while others only need to remain within a defined range.

A typical example would be quality characteristics for which a tolerance band is specified. While a dead band in a PID controller tends to put stability at risk, CV bands in individual controller channels generally relieve the multivariable controller overall.

Using CV bands, the action of a soft override control can be achieved.

Use case for error signal generation with dead band

As long as the pressure in a reactor remains within the set safety limits, the controller is interested only in product quality. However, as soon as the pressure threatens to leave the permitted range (in other words, in the prediction it moves towards an illegal value in the future), the pressure control cuts in. By weighting the controlled variables in the performance index (see MPC Configurator), the user can specify that predicted violations of the pressure limits are given a particularly high weighting.

Predictive controller algorithm

The MPC10x10 block is derived from the familiar DMC algorithm (**D**ynamic **M**atrix **C**ontrol) .

Future changes to the manipulated variable within the control horizon are calculated from future predictive control deviations. To accomplish this, first the ideal solution is determined in each sampling step using an analytical formula without consideration of constraints. If the ideal solution comes into conflict with a constraint, an iterative optimization process is used to find a solution of the dynamic quadratic optimization problem with secondary conditions.

Based on the principle of the receding horizon, only the first value is taken from the calculated vector of the optimum manipulated variable changes over the entire control horizon and applied to the process. In the next step, the newly arrived process values are taken into account and the calculation repeated over the entire prediction horizon.

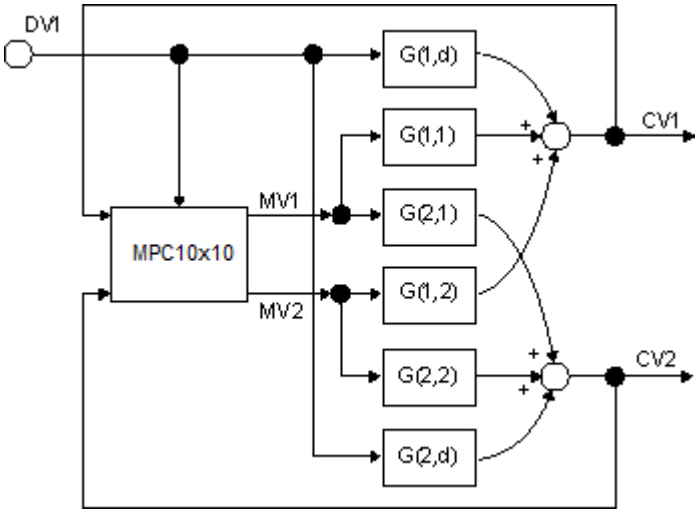
With predictive controllers, the manipulated variable changes are based on the control deviations predicted in the future, while with a PID controller, they are based on error signal of the past (possibly also integrated). This can be interpreted as a "looking ahead" strategy.

Anti-windup

When manipulated variable limits are active, anti-windup measures are taken automatically within the controller. The prediction equations use the real limited values of the manipulated variable instead of theoretically calculated values.

Model-based disturbance compensation

Model-based disturbance compensation can and should be used if one or more known disturbances have a strong influence on the process and their causes can be measured.



The effects of a measurable disturbance (`DV1..4` I/O) on all controlled variables `CV1 ... CV10` can be estimated when the controller is taken into manual mode. This means that no movements of the controlled variables whatsoever result from changes to the manipulated variable and all movements result from the disturbance. If the disturbance (e.g. ambient temperature) can be measured but cannot be actively adjusted, it may be necessary to search through a data archive to find the time segments in which the disturbance changed.

The identification of the transfer functions from disturbances `DV1…DV4` to all controlled variables `CV1 ... CV10` (disturbance models in the graphic above G(1,d) and G(2,d)) is performed with the MPC Configurator and is analogous to the identification of the main transfer functions (G(1,1) to G(2,2)). The measured disturbances are then switched to the `DV1…DV4` inputs of the MPC10x10 block and disturbance compensation is activated with `DV_On = 1`. As a result, the effect of the measurable disturbances is taken into account in the prediction and the controller can start counter measures in advance before the disturbance can have a massive influence on the controlled variables.

Such disturbance compensation is especially effective when the disturbance is constant at times and changes from time to time. If a disturbance changes constantly or oscillates, however, the feedforward control is not enabled during operation of the controller to avoid constant oscillation of the manipulated variables, although it should be taken into account in the MPC Configurator when creating the process model.

If there is no disturbance model in the user data block, the signals at the `DV1...DV4` input are ignored.

Typical examples of measurable disturbances are inlet volumes in distillation columns or throughput of continuous reactors.

Predictive controller with more than four measurable disturbances

If you want to plan for more than four measurable disturbances in an application, but do not need all ten manipulated variable channels of the MPC10x10 block, you can dedicate the first of the previously unused MV channels for disturbance feedforwarding.

Example: You only have two control actions available, so only use `MV1` and `MV2`. Then connect the additional measurable disturbance `DV5` to the `MV3Trk` input parameter and set `MV3TrkOn` = 1. For recording of training data for the predictive controller, declare `MV3Trk` as the third disturbance and also use the CFC trend recorder to record the effect of changes to `DV5` on all controlled variables.

Use the MPC Configurator to then determine a process model that describes the effects of `DV5`. However, if the disturbance variable `DV5` in "automatic mode" of the controller fluctuates due to external influences, the effect of such changes is taken into consideration for predicting future process reactions through `MV3Trk` and predictive methods can be used to compensate this disturbance. The performance of the disturbance compensation is exactly the same as that for regular feedforward control via the input parameters `DV1...DV4` and `DV_On` = 1.

If you want to disable this disturbance compensation in runtime with the unused `MV5` manipulated variable, you need to insert a `SelA02In` selector block in front of the `MV3Trk` input. This allows you to set a constant zero for `MV3Trk` instead of the measured value `DV5`, which stops the effect of `MV3` on prediction. (Due to this reassignment, `MV3TrkOn` must always remain 1 to prevent the controller from changing the value of `MV3`.)

This way, additional measurable disturbances can be selected. However, the sum of the manipulated variables and disturbances may not exceed a total number of 14.

At the output variables `NumberOrigMVs`, the number of original manipulated variables is displayed that can actively be influenced by the controller. The other manipulated variables (in the above example `MV3`) are only used for disturbance feedforward and are therefore always in tracking mode. The are referred to as pseudo MVs.

Control of square and non-square systems

In multivariable controllers, the number of manipulated variables should ideally be the same as the number of controlled variables. This is known as a "square system". As long as constraints do not influence operation, the controller can, in principle, track all controlled variables exactly to the selected setpoints.

If there are less manipulated variables than controlled variables or if individual manipulated variables have reached their limits, the degrees of freedom are lacking in the control problem. This means that it is not possible for all setpoints to be reached exactly.

The MPC10x10 algorithm then finds a compromise that can be influenced by the selection of controlled variable weightings (priorities) in the MPC Configurator: Controlled variables with higher priority will have lower control deviations.

If there are more manipulated variables than controlled variables or if some of the controlled variables are already within their setpoint bands, there are surplus degrees of freedom in the control problem.

The MPC10x10 algorithm can use these degrees of freedom to move the manipulated variable in the direction of the specified target value, whereby the weighting of the manipulated variable target value in the performance index is typically substantially less than the weighting of setpoints for the controlled variables. If the manipulated variables result in costs (e.g. mass flows of steam or other forms of energy consumption), the manipulated variable target values

can be set somewhat lower than the values previously observed at the operating point. The controller then attempts to reduce the manipulated variables (the costs) in the direction of the target values as long as the achievement of the primary control target provides it some degree of freedom. In general, the advantages of the dynamic online optimization in the MPC10x10 are especially attractive for non-quadratic systems, because the restrictions of the lean algorithm from ModPreCon are overcome.

Control of linear and non-linear systems

The MPC10x10 algorithm is based on a linear, time invariant process model. As a result, in much the same way as a PID controller, it is suitable above all for controlling non-linear systems around a fixed operating point.

Again analogous to the PID controller, there are, however, several possibilities with which the area of application can be extended with non-linear systems:

Compensation functions between controller and controlled system:

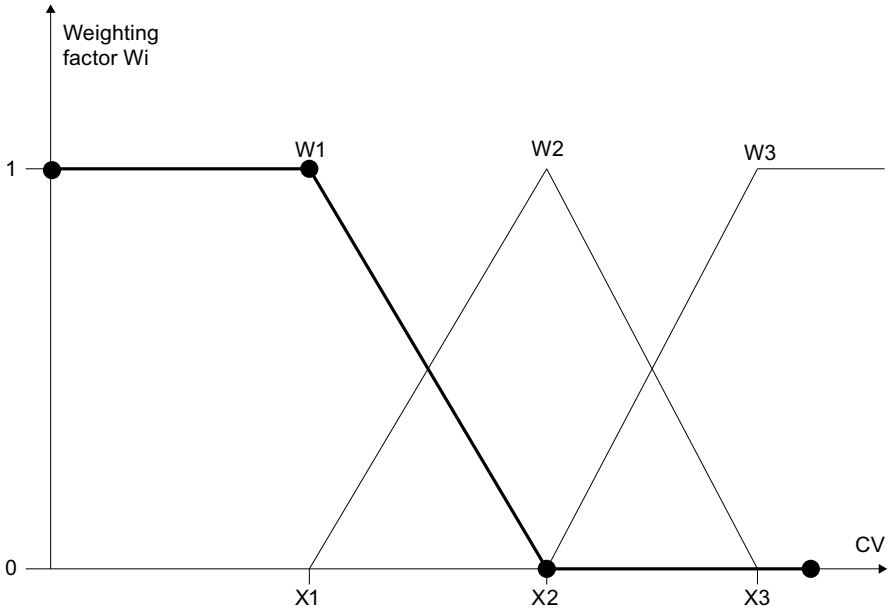
It is, for example, possible to compensate the effect of a non-linear valve characteristic curve using a polygon block between the `MV` output and the control input of the valve block. Care must be taken when implementing the manipulated variable limits. In the same way, the effect of a non-linearity at the output of the controlled system (for example a sensor characteristic curve) can be compensated by a polygon block before the `CV` input of the controller. Remember that the corresponding SP must also be equivalently transformed. In both cases, the compensation functions become part of the controlled system from the perspective of the controller. The aim is always to keep the overall response of the controlled system consisting of process and compensation elements as linear as possible.

Multimodel control:

This approach is related to the basic idea of operating-point-based parameter control with PID controllers. Because the model parameters of the MPC10x10 block cannot be modified in runtime, however, the control strategy for selecting the suitable parameter set becomes a control strategy for selecting the suitable model.

Several MPC10x10 instances with different models for different operating points run at the same time. The local optimal models are determined by starting the process at the various operating points with small amplitudes, so that only the reaction of the non-linear process in the ambience of this operating point is registered.

The final manipulated variable for each manipulated variable is formed as a weighted mean value of the manipulated variables proposed by the controller instances. (It is recommended that experiments for starting the process for the MPC Configurator are only performed after implementing the functions for adding the manipulated variables, in order to ensure that the same conditions applicable when the model is in operation actually take effect.)



The weighting factors 0 ... 1 are formed in the same way as the membership functions known from fuzzy logic so that the sum of all weights is always one and each controller has the highest weighting at its own operating point. A polygon with 4 or 5 interpolation points is used to calculate each individual weighting factor. The weighting factors are calculated based on a specific measurable `PV` variable of the process, which is representative for the operating point of the process. This can be one of the `CVx` controlled variables, although it does not have to be. The abscissa of the interpolation points of all polygons is selected in such a way that they cover the entire value range of `PV` in order to avoid extrapolation errors.

One should note in this regard that the only non-linear effects in the full multi-variable control loop that can be modeled are those that correlate exactly to a representative `PV` variable. This approach is therefore not suitable for cases in which individual partial transfer functions demonstrate non-linear effects that depend on various, totally independent variables.

To ensure the stability of the overall control loop, all subcontrollers must be at least stable at all operating points. In contrast to PID controllers however, an MPC is not affected by windup problems if it temporarily runs in "automatic mode" but cannot intervene in the real process (weighting factor zero).

One of the controller instances is defined as the main controller and shown in the operator faceplate on the OS. All others are connected in such a way that they adopt the operating mode ("manual"/"automatic mode") and the setpoints from the main controller. The manual manipulated variables are passed to the secondary controller via the tracking inputs. This means that no operator intervention is required on secondary controllers.

Note

The manipulated variables of the main controller can be used when switching from automatic to manual mode. If the process was in the operating range of a secondary controller beforehand, the current manipulated variables in the process can deviate significantly. In this case, you should apply the actual manipulated values in effect by manual input in the faceplate of the primary controller.

You can find an example of multi-model control with the ModPreCon block in the example project of the Advanced Process Library in the section "Predictive control of a non-linear process (Page 2355)" (`MPC10x10NonLinSim`). The approach is also applicable to the MPC10x10 block for the most part.

Trajectory control:

This approach neatly combines the advantages of an open loop controller (Feedforward Control) with those of a closed loop controller with process value feedback (Closed Loop Control). The controller follows a previously optimized trajectory of setpoints and manipulated variables; in other words, it only needs to compensate small deviations between the stored trajectory and the current plant state. A trajectory is an optimum series of manipulated variables over time and the process values that match them. The required manipulated variables are read by means of the `MV10Traj` inputs into the MPC10x10 block and added to the values of the manipulated variable calculated by the algorithm (in automatic mode only). Among other things, the advantage of this is that the effective manipulated variable acting on the process can be configured and is limited to the sum of the trajectory and controller action. The process values from the trajectory are switched to the corresponding setpoint specifications `SP1` ... `SP10` of the controller. As long as the process reacts exactly as planned in the trajectory, it will respond to the series of manipulated variables from the trajectory with the corresponding series of process values and the control deviation is zero. It is generally known that a non-linear dynamic process can linearized around a fixed operating point or a steady state of the system. It is also possible to linearize it around a trajectory.

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `DV1.ST`
- `DV4.ST`
- `CV1.ST`
- `CV2.ST`
- ...
- `CV10.ST`

Configurable reactions using the `Feature` parameter

An overview of all the reactions that are provided by the `Feature` parameter is available in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
4	Setting switch or button mode (Page 166)
15	Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165)

Bit	Function
16	Neutral position manipulated variable takes effect at startup (Page 165)
24	Enabling local operator authorization (Page 157)

Operator permissions

The block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	Not used
2	1 = Operator can switch to "Out of service" mode
3	Not used
4	Not used
5	1 = Operator can change the setpoint 1
6	1 = Operator can change the manipulated variables of all channels
7	1 = Operator can change operating high limits of the setpoints for all channels
8	1 = Operator can change operating low limits of the setpoints for all channels
9	1 = Operator can change the setpoint 2
10	1 = Operator can change the setpoint 3
11	1 = Operator can change the setpoint 4
12	1 = Operator can change the setpoint filter of all channels
13 - 16	1 = Operator can change the setpoint 5…8
17	1 = Operator can enable the track setpoint in "manual mode" function
18	1 = Operator can activate the model-based disturbance compensation function
19	1 = Operator can change the setpoint 9
20	1 = Operator can change the setpoint 10
21	1 = Operator can only activate the "Prediction Mode" function
22	1 = Operator can change the target values of all channels
23	1 = Operator can change the dead band parameter of all channels
24	1 = Operator can change all parameters of the operating point optimization
25	1 = Operator can change the simulation value `SimCV1..10`
26	1 = Operator can activate the Simulation function
27	1 = Operator can activate the Release for maintenance function
28	1 = Operator can change the manipulated variable limits of all channels
29	1 = Operator can change the gradient limits of manipulated variables of all channels
30	1 = Operator can activate the operating point optimization
31	1 = Operator can deactivate the operating point optimization

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Release for maintenance

The block provides the standard function Release for maintenance (Page 64).

Specifying the display area for process and setpoint values as well as operations

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

In contrast to PID controllers, there are no separate parameters for bar limits. The setpoints limits are used for all setpoint and actual value bars; manual limits are used as bar limits for all manipulated variable bars.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Integrated static operating point optimization

The integrated static operating point optimization can be used if no exact `SPi` setpoint is specified for at least one controlled variable (index i =1...10), but rather the `CVi` process value must remain within a tolerance range `SPiOptHiLim`…`SPiOptLoLim`. The tolerance range must, of course, be within the valid setpoint limits `SPiHiLim`…`SPiLoLim` for this control channel. The tolerance range `SPiOptHiLim`…`SPiOptLoLim` is not coupled to the operator-controlled setpoint `SPi`. If the setpoint is changed, the tolerance range is not automatically shifted. If you want to do this nevertheless, interconnect the `SPiOpOut` output parameters via two adders to the width of the tolerance range at the inputs `SPiOptHiLim` and `SPiOptLoLim`.

From an economic perspective, different values within the tolerance range can be more or less favorable. With the help of the optimization function, the optimal economic point can be found within the tolerance range.

This is done by defining a target function (performance criterion), which depends on the manipulated variable and controlled variable of the predictive controller. This can be, for example, the economic yield of plant operation per time unit or it may involve specific costs or energy consumption.

```
J= GradMV1*MV1 +GradMV2*MV2 +… +GradMV10*MV10
+GradCV1*CV1 +GradCV2*CV2 +…+GradCV10*CV10
+J0
```

You specify the individual `GradXVi` coefficients of the gradient vector as input variables at the MPC10x10 function block in the CFC or in the parameter view of the faceplate. If individual coefficients vary with time, e.g. they are dependent on current market prices, you can also interconnect these input variables. If individual manipulated or controlled variables have no influence on the performance criterion, leave the corresponding coefficients at the default value, zero.

You can use the binary `J_Mini` input parameter to specify whether the target function is to be maximized or minimized, based on whether this involves yields or costs (`J_Mini = 1`: minimization).

The term `J0` combines all contributions to the target function that do not depend on manipulated variables and controlled variables. These contributions have no effect on the optimum values in the decision variables, but are applied for calculating the current value of employed performance criterion similar to the above-mentioned formula.

Within the controller, the terms of the target function that depend on controlled variables are converted to make their dependence on the manipulated variables visible. To do this, the inverse stationary process model from the MPC Configurator is used. This requires that the number of manipulated variables matches the number of controlled variables. If the number of manipulated variables does not match the number of controlled variables, the largest possible square submodel in the matrix of the transfer functions is truncated from the top left. If there are more manipulated variables than controlled variables, for example, only the first manipulated variables are used, in accordance with the number of controlled variables.

Constraints for the controlled variables take the form of the above-mentioned tolerance ranges for setpoints. The controller takes care of adhering to the manipulated variable limits, in any case; they do not have to separately specified as constraints for the optimization.

Enable the optimization using the binary input variable `OptimizeOn` in the controller faceplate. The optimizer then returns setpoints within the tolerance ranges that are optimal for the performance criterion. These setpoints are then sent to the control algorithm, which handles them in the same way as conventionally specified setpoints (with or without dead band). The operable `SP1...SP10` setpoints are not tracked to the optimized setpoints; when optimization is disabled, the old setpoints from the faceplate take effect once again. When selecting variables for archiving and graphic plotter, ensure you use the `SP1Out...SP10Out` setpoint actually in effect and not the `SP1...SP10`. input variables.

The current value of the performance criterion is displayed at the `J_Actual` output variables.

You can find additional information about the topic of static operating point optimization in the online help for the MPC configuration editor.

Display of the prediction of free movement

The prediction of free motion is a forecast for the future behavior of the process within the overall prediction horizon, under the assumption that all manipulated variables are frozen at their current values. The time length of the prediction horizon is indicated in the output parameter `PrediHorizon` in the [s] unit.

The prediction of free motion is recalculated in each sampling step within the control algorithm. If the manipulated variables is to a constant value in manual mode, the prediction of the free movement is actually a realistic prediction for the future process response. It can therefore be represented graphically in the faceplate at least in terms of quality. For this purpose, five equidistant interpolation values are copied from the prediction horizon, and displayed in the standard view of the faceplate as a vertical bar next to the current process value.

Example: The prediction horizon is 1800s=30min and the current time is labeled with the index k. The prediction for k+6min, and next to it k+12min, up to k+30min then appears on the right next to the bar of the current process value. If the upper border of the bar is conceptually connected with a line (red in the picture red), you can imaging the curve created by the future course of the process value over the next half hour.

In automatic mode, the value of the manipulated variables changes with each sampling step. The prediction of free movement is then only a fictional mathematical formulation within the algorithm, and not a realistic prediction for the future process response. This is why the prediction is only displayed in manual mode. The display can be generally suppressed using Feature.Bit31.

Prediction without control response

In this special "operating mode" (comparable to the block-internal simulation), the controller only monitors the process and indicates what it would like to do in the next sampling step without actively intervening in the process. This allows you as the user to build trust before "switching active" the controller the first time, i.e. actively intervening in the process.

"Prediction Mode" is activated via the binary input variable `PredictMode` or in "Parameters" in the faceplate view. Setpoints and process values are read as in normal automatic mode. The prediction of the free movement and manipulated variable change for the next sampling step are calculated as in normal automatic mode. The starting point for the prediction of the manipulated variable for the next sampling step, however, is the current process value of the follow-up control loop at the `MV1Trk`…`MV10Trk` tracking inputs. The predicted manipulated variables are not output at the normal outputs `MV1...10` but rather at the `MV1Pred...MV10Pred` outputs, which were especially introduced for this purpose, and are displayed in the standard view of the faceplate on the left next to `MV1...10` as long as "Prediction Mode" is active.

- When the controller is in automatic mode, in "Prediction Mode" all `MVi (i=1..10)` are set to match the assigned MViTrk input parameters, similar to tracking mode.
- When the controller in manual mode, all `MVi` are set to the desired manual values regardless of "Prediction Mode".
- When "Prediction Mode" is disabled, all `MViPred` always equal the assigned `MVi`.

Automatic process trigger for model identification

In order to determine the process model for the model predictive controller, the process must be artificially triggered in order to observe its dynamic response and record it in the form of training data. This trigger can be specified manually in manual mode of the controller.

Alternatively, a suitable trigger signal can be generated automatically in the form a defined, symmetrical sequence of manipulated variable jumps. The trigger signals are calculated by an auxiliary function block, "`AutoExcitation`", which is built into the process tag type and interconnected with MPC10x10 .

Additional `MV1Excite`…`MV10Excite` input variables are required for this on the controller. Process triggering performed in manual mode of the controller, because automatic mode cannot be activated before modeling. The new "Operating mode" process trigger can only be controlled via the `ExciteOn` input bit on the engineering system, and not on the operator station, because the CFC is needed for configuration of the AutoExcitation block. However, process triggering must be displayed on the OS in standard view at the lower left.

Manual intervention per faceplate remains possible even during the triggering. The values of the `MV1Excite...MV10Excite` input parameters are therefore only written to the `MV1Man...MV10Man` manual values event-based, but only if they change.

You can find additional details on automatic process triggering in the online help for the MPC configuration editor.

5.15.4 MPC10x10 error handling

Error handling of MPC10x10

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` output parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Default value when the block is installed; this message is irrelevant.
0	There is no error.
3	Configuration error: Block is inserted in the wrong OB, cycle time of the OB is not suited for controller configuration.
31	The value of `CV1` can no longer be displayed in the real number field or is not a number.
32	The value of `CV2` can no longer be displayed in the real number field or is not a number.
...	..
40	The value of `CV10` can no longer be displayed in the real number field or is not a number.
41	The `MV_Trk1` value can no longer be displayed in the real number field or is not a number.
42	The `MV_Trk2` value can no longer be displayed in the real number field or is not a number.
...	...
50	The `MV_Trk10` value can no longer be displayed in the real number field or is not a number.
51	`AutModLi` = 1 and `ManModLi` = 1
90	The controller matrix could not be loaded from the user data block.

The ErrorOpt output parameter is used to output the status of the lower-level LPOptim block.

5.15.5 MPC10x10 messaging

Messaging

This block does not offer messaging.

5.15.6 MPC10x10 I/Os

Input parameters

Parameter	Description	Type	Default
AutModLi	1 = Automatic mode via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp	1 = Automatic mode via operator (controlled by ModLiOp = 0)	BOOL	0
BatchEn	1 = Enabled for allocation by batch control	BOOL	0
BatchID	Batch number	DWORD	16#00000000
BatchName	Batch name	S7-String	
CV1...CV10	Controlled variable 1...10 (process value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
CV1_Unit...CV10_Unit	Unit of measure for controlled variable 1...10	INT	1001
DV_On	1 = Activate the disturbance variable feedforward using DV	BOOL	1
DV1...DV4	Disturbance variable 1...4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DV1_Unit...DV4_Unit	Unit of measure for DV1...DV4	INT	1342
DV1DeadBand...DV4DeadBand	Dead band for DV1...DV4	REAL	1.0
DynOptOn	1 = Activate dynamic optimization with constraints. 0 = Ignore constraints for optimization.	BOOL	1
ExciteOn	1 = Automatic process trigger; MViExite input parameters are written to the MVi outputs	BOOL	0

Parameter	Description	Type	Default
Feature	I/O for additional functions (Page 930)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 1
GradCV1… GradCV10	Gradient vector for performance criterion, element (factor) for CV1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
GradMV1… GradMV10	Gradient vector for performance criterion, element (factor) for MV1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
J_Actual_Unit	Physical unit of performance J_Actual	INT	0
J_Mini	1 = minimize J, 0= maximize J	BOOL	0
J0	Value of the performance criterion in the operating point	REAL	0.0
ManModLi	1 = Manual mode via interconnection or SFC (effective if ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp	1 = Manual mode via OS operator (effective if ModLiOp = 0)	BOOL	0
ModLiOp	Operating mode switchover by: 1 = Interconnection or SFC 0 = Operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp	Operator input for release for maintenance 1 = MS release requested	BOOL	0
MV1_Unit...MV10_Unit	Unit of measure for manipulated variable 1...10	INT	1342
MV1Excite...MV10Excite	Default value for MV1..10 during the automatic process trigger for model generation	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV1HiLi	High limit of manipulated variable MV1	REAL	100.0
MV1LoLi	Low limit of manipulated variable MV1	REAL	0.0
MV1Man	Manual value: Operator input for setting the manipulated variable MV1 in manual mode	REAL	0.0
MV1ManHiLim	High limit of manipulated variable MV1 in manual mode	REAL	100.0

Parameter	Description	Type	Default
MV1ManLoLim	Low limit of manipulated variable MV1 in manual mode	REAL	0.0
MV1RaLim	Gradient limiting of the manipulated variable MV1 [MV1_Unit/ second]	REAL	100.0
MV1Target...MV10Target	Target value (optimum value) for manipulated variable MV1...MV10	REAL	0.0
MV1Trk	Tracking value for the manipulated variable MV1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV1Traj	Trajectory value that is added to the manipulated variable MV1	REAL	0.0
MV1TrkOn...MV10TrkOn	1 = Tracking of manipulated variable MV1...MV10	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = Allocated by SIMATIC BATCH	BOOL	0
OosLi	1 = Out of service, via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp	1 = Out of service, via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, to be connected with the Out output parameter of the upstream block OpStations	DWORD	16#00000000
OptimOffOp	1 = Disable operating point optimization, normal setpoints SP1...SP10 in effect	BOOL	0
OptimOnOp	1 = Enable operating point optimization, optimized setpoints SP1Out...SP10Out in effect	BOOL	0
OS_Perm	I/O for operating permissions (Page 930)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
PredictMode	1 = "Prediction only mode" enabled, prediction only, no intervention in the process	BOOL	0
PreFilt1...PreFilt10	Settling time [s] of the setpoint filter for setpoint SP1...SP10	REAL	0.0
Restart	1 = Restart of the block and adoption of the data from the user block of the MPC Configurator	BOOL	0
SafePos1...SafePos10	1 = Safety setting for MV1...MV10 is Man1HiLim...Man10HiLim, 0 = Safety setting for MV1...MV10 is Man1LoLim...Man1LoLim,	BOOL	0

Parameter	Description	Type	Default
SampleTime	Sampling time [s] (assigned automatically)	REAL	1.0
SelFp1	1 = Call a block saved in this parameter as an additional faceplate in the standard view	ANY	
SelFp2	1 = Call a block saved in this parameter as an additional faceplate in the preview	ANY	
SimCV1… SimCV10	Simulation value for control variable CV1...CV10, which is used for SimOn = 1	REAL	0.0
SimCV1Li… SimCV10Li	Simulation value for control variable CV1...CV10, which is used for SimOnLi.Value = 1 (and SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimLiOp	Activation/deactivation of the simulation by: 1 = Interconnection or SFC 0 = Operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn	1 = Enable simulation	BOOL	0
SimOnLi	1 = Enable simulation via interconnection or SFC (if SimLiOp = 1))	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_TrkCV	1 = Setpoints follow the CVs in manual mode and in tracking	BOOL	0
SP1...SP10	Setpoint 1...10	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
SP1DeadBand... SP10DeadBand	Width (radius) of the dead band control of CV1...CV10	REAL	0.0
SP1HiLim...SP10HiLim	Setpoint high limit 1...10	REAL	100.0
SP1LoLim...SP10LoLim	Setpoint low limit 1...10	REAL	0.0
SP1OptHiLim... SP10OptHiLim	High limit for optimization of setpoint 1...10	REAL	0.0
SP1OptLoLim... SP10OptLoLim	Low limit for optimization of setpoint 1...10	REAL	0.0
StepNo	Batch step number	DWORD	16#00000000
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

Output parameters

Parameter	Description	Type	Default
AutAct	1 = Automatic mode is active 0 = Manual mode is active (applies to all manipulated variable channels)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CV1Out...CV10Out	Copy CV1...CV10, preferably as output parameter	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
ErrorEndless	Error: infinite iteration loop in the dynamic optimization	BOOL	0
ErrorNum	Output of pending error number. You can find information about the error numbers that are output by this block at MPC10x10 error handling (Page 942)	INT	-1
ErrorOpt	Error number of the integrated operating point optimization function, see MPC10x10 error handling (Page 942)	INT	0
Fut1_y1… Fut10_y5	Prediction of free movement of CV1...CV10 for five future points in time within the prediction horizon	REAL	0.0
J_Actual	Current value of the performance criterion	REAL	0.0
Loop1Closed… Loop10Closed	1 = MV1 is used by the algorithm for closed loop control, i.e. AutAct AND NOT MV1TrkOn 0 = Control loop for MV1 open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = Manual mode enabled, for all control channels	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Release	Release for maintenance: 1 = Release by OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV1...MV10	Manipulated variable 1..10 (controller output signal)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
MV1HiAct	1 = High limit of manipulated variable 1 reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV1LoAct	1 = Low limit of manipulated variable 1 reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV1Pred	One-step prediction for MV1 in the "Prediction without control response" mode	REAL	0.0
MV1TargOut... MV10TargOut	Target value output (optimum value) for manipulated variable MV1...MV10	REAL	0.0
NumberCVs	Number of control variables used	INT	0

Parameter	Description	Type	Default
NumberDVs	Number of disturbances used	INT	0
NumberMVs	Number of manipulated variables used	INT	10
NumberOrigMVs	Number of original manipulated variables that are actively influenced by the controller (without pseudo MVs that are used as DVs	INT	0
OosAct	1 = Block is "out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feed-forwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OptimAct	1 = Operating point optimization enabled, setpoints calculated by the optimization are used 0 = Optimization off	BOOL	0
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
PrediHorizon	Prediction horizon [s]	REAL	0.0
SP1OpOut...SP10OpOut	Copy of the operable setpoint 1 for step-enabling	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
SP1Out...SP10Out	Setpoint 1 actually used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 922)	DWORD	16#00000000
Status2	Status word 2 (Page 922)	DWORD	16#00000000

5.15.7 MPC10x10 block diagram

MPC10x10 block diagram

A block diagram is not provided for this block.

5.15.8 Operator control and monitoring

5.15.8.1 MPC10x10 views

Views of the MPC10x10 block

The block MPC10x10 provides the following views:

- MPC10x10 standard view (Page 949)
- Trend view (Page 299)
- MPC10x10 parameter view (Page 955)
- MPC10x10 CV parameter view (Page 957)
- MPC10x10 MV parameter view (Page 958)
- MPC10x10 preview (Page 959)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MPC10x10 (Page 962)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

See also

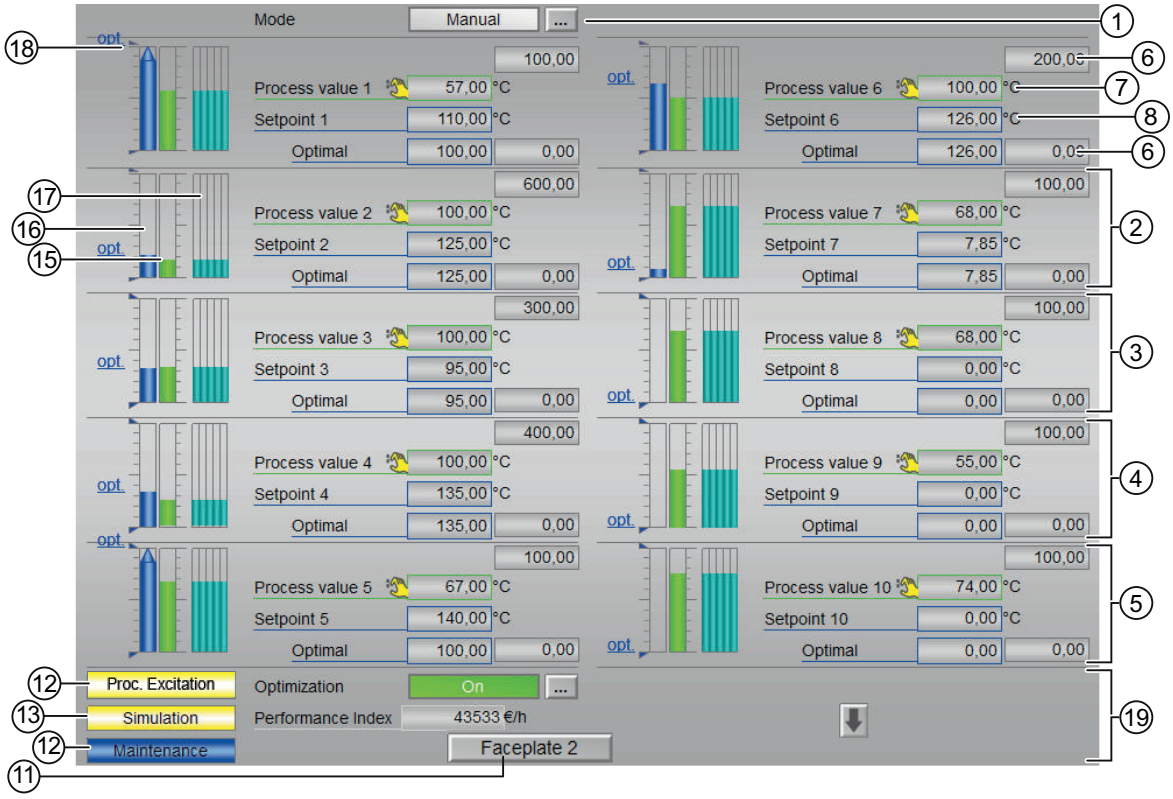
MPC10x10 trend view (Page 961)

5.15.8.2 MPC10x10 standard view

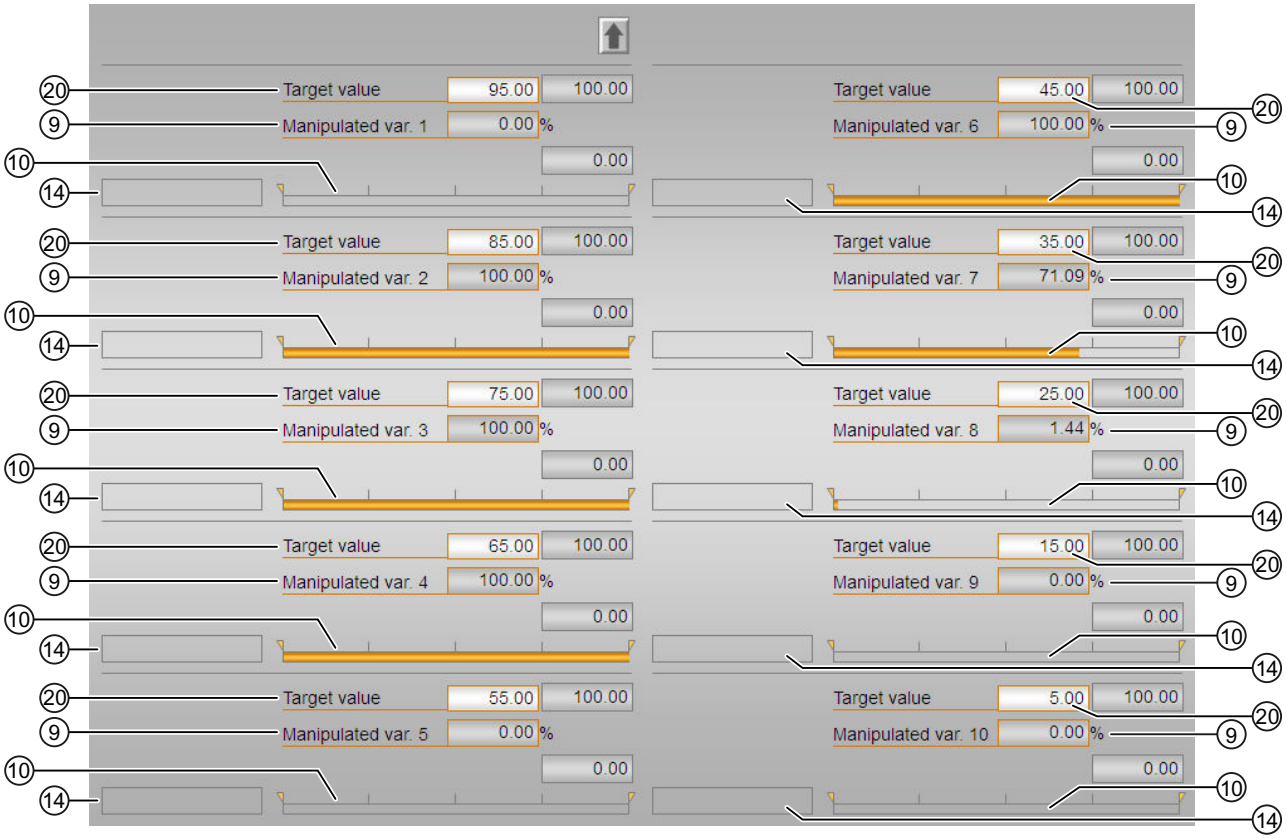
Standard view of MPC10x10

The standard view has an upper half and a lower half. You can change between the two halves with the arrow keys. The upper half shows all available controlled variable channels with their setpoints, while the lower half shows all available manipulated variable channels.

Upper screen half (controlled variables)



Lower screen half (manipulated variables)



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 72)
- Automatic mode (Page 72)
- Out of service (Page 71)

For information on switching the operating mode, refer to the section

Switching operating states and operating modes (Page 251).

(2), (3), (4) and (5) Displaying and switching for values for channels 1 to 10

This area always has the same layout for channels 1 to 10:

(6) High and low scale range for the process value

These values provide information on the display range for the bar graph of the process value. The scale range is defined in the engineering system.

(7) Displaying and changing the process value including signal status

This area shows the current process value with the corresponding signal status.

The process value is normally displayed and cannot be operated. Process values in the faceplate can only be changed within the context of internal block simulation.

(8) Displaying and changing the setpoint including signal status

This area shows the current setpoint with the corresponding signal status. Refer to the Changing values (Page 253) section for information on changing the setpoint.

(11) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

(12) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on the display area for states of the block is available in the Release for maintenance (Page 64) .

- "Process excitation"
  The automatic process trigger is fed forward using the upstream block AutoExcitation for recording learning data for the MPC configurator. The manipulated variable step changes are added to the manipulated values 1 to 10 according to schedule. Avoid external disturbances to the process while the process trigger is running. The manipulated variables can be changed manually while the process trigger is running.

(13) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in the section Simulating signals (Page 58) .

(15) Bar graph for the process value 1

There is a bar graph for the process value for every channel 1 to 10.

This area shows the current process value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(16) Bar graph for the setpoint 1

There is a bar graph for the setpoint for every channel 1 to 10.

This area shows the current setpoint in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(17) Prediction of free movement

This area shows you the prediction of free movement in the form of a bar graph. For each channel from 1 to 10, there is a bar graph for the prediction of free movement, that is, for the future behavior of the process within the overall prediction horizon, under the assumption that all manipulated variables are frozen at their current values.

This is why the prediction of free movement is only displayed in manual mode.

The value range of the bar graph matches the value range of the assigned setpoint and current value bar.

Functions of MPC10x10 (Page 930)

(18) Displaying the limits

These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the engineering system (ES).

(19) Static operating point optimization

Activate the optimization using the button at bottom right. Activation means that the optimized setpoints SP1Out...SP4Out are actually used instead of the SP1...SP4 setpoints specified in the faceplate for the closed-loop control. (The actual calculation of the optimum setpoints depends on this, and is only performed if one of the input variables for the optimization has changed.) The current value the economic performance criterion J appears in the display field below.

When optimization is enabled, the optimum setpoints are displayed on the setpoint bar as small, horizontal lines and highlighted with the abbreviation "opt.". The numerical values of the optimum setpoints are then displayed left of the input fields for the setpoints.

Lower screen half (manipulated variables)

(9) Displaying and changing the manipulated variable including signal status

This area shows the current manipulated variable with the corresponding signal status. Refer to the Changing values (Page 253) section for information on changing the manipulated variable. You can only make a change in manual mode.

(10) Bar graph for the manipulated variable with limit display

This area shows the current manipulated variable in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES):

- Limits: `MVxHiLim` and `MVxLoLim`
- Display area: `MVxManHiLim` and `MVxManLoLim`

(14) Display for states of the manipulated channel

There is a display for the state of the manipulated channel for channels 1 to 10: the status display only appears if the respective channel is tracked:

- "Tracking"

(20) Displaying and changing the target value for the manipulated variable

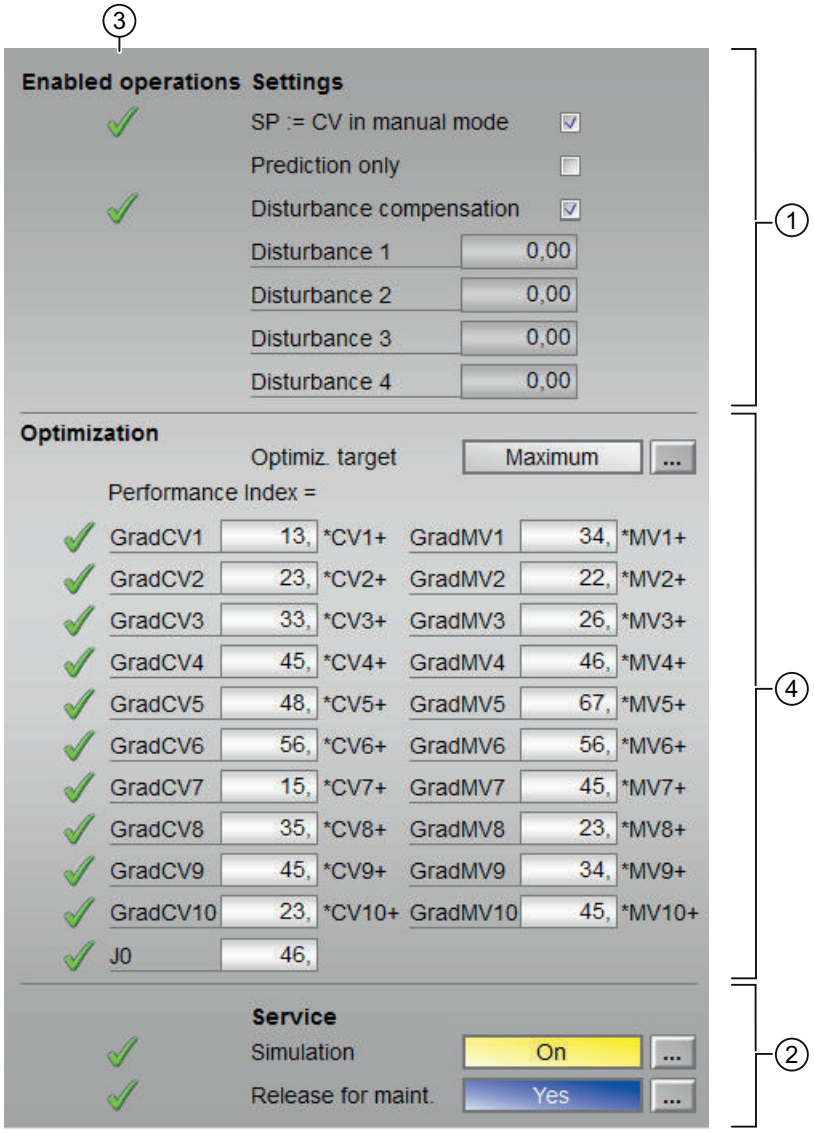
This area shows the current target value for the manipulated variable. Refer to the section Changing values (Page 253) for more information on changing the target value. You can make a change only in the "Automatic" mode and also only if `OptimAct` = 0. This field is not displayed in the "Manual" mode.

See also

MPC10x10 preview (Page 959)

5.15.8.3 MPC10x10 parameter view

Parameter view of MPC10x10



(1) Settings

You can activate the following functions for the controller in this area:

- "SP := CV in manual mode": ☑ Bumpless switchover from "manual mode" to "automatic mode"
- "Prediction only" activate this special "operating mode" by selecting the check box. The controller then only listens in on the process and indicates what it would like to do in the next sampling step (i.e. which manipulated variables it would output in the next sampling step) without actively intervening in the process
- "Disturbance compensation": ☑ Select disturbance feedforward
- "Disturbance" DV1…DV4 depending on the number of configured measurable disturbances

You cannot change the disturbance, it can only be displayed.

(2) Service

You can select the following functions in this area:

- "Simulation"
- "Release for maintenance"

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Simulating signals (Page 58)
- Release for maintenance (Page 64)

(3) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

(4) Optimization

Direction of the optimization (minimize or maximize)

By default, the optimizer seeks to maximize the performance function, in the assumption that it is dealing with economic yield. If you want to search a minimum, however, because you are dealing with costs or consumption values, click this button.

Specification of performance criterion for the operating point optimization

The performance criterion consists of a weighted sum of all manipulated and controlled variables. For each manipulated variable and controlled variable, enter the appropriate weighting factor, i.e. the coefficient of the gradient vector. Zero means that the value of the corresponding manipulated variable or controlled variable no direct influence on the economic yield. If the controller has less than 10 manipulated variables or controlled variables, the irrelevant variables are hidden automatically.

J0 is the proportion of the performance criterion which does not depend on the controlled and manipulated variables of the MPC, e.g. fixed costs or costs calculated otherwise. Although this proportion of costs cannot be influenced during internal controller optimization, including it in the summation allows realistic numerical values to be displayed for the total costs.

See also

MPC10x10 views (Page 949)

MPC10x10 standard view (Page 949)

MPC10x10 trend view (Page 961)

MPC10x10 preview (Page 959)

5.15.8.4 MPC10x10 CV parameter view

CV parameter view for MPC10x10

The CV parameter view is a table which contains a row for each control channel:

Channel setpoint	H range	H range optimiz.	Operator specific	Optimal setpoint	L range optimiz.	L range	Dead band	Unit	Prefilter [s]
1	100,00	0,00	110,00	100,00	0,00	0,00	0,00	°C	0,00
2	600,00	0,00	125,00	125,00	0,00	0,00	0,00	°C	0,00
3	300,00	0,00	95,00	95,00	0,00	0,00	0,00	°C	0,00
4	400,00	0,00	135,00	135,00	0,00	0,00	0,00	°C	0,00
5	100,00	0,00	140,00	100,00	0,00	0,00	0,00	°C	0,00
6	200,00	0,00	126,00	126,00	0,00	0,00	0,00	°C	0,00
7	100,00	0,00	7,85	7,85	0,00	0,00	0,00	°C	0,00
8	100,00	0,00	0,00	0,00	0,00	0,00	0,00	°C	0,00
9	100,00	0,00	0,00	0,00	0,00	0,00	0,00	°C	0,00
10	100,00	0,00	0,00	0,00	0,00	0,00	0,00	°C	0,00

The channel number is at the far left. The physical unit applies to all values of this control channel except for the prefilter. The time constant of the prefilter is specified for all channels in seconds.

(1) Displaying and changing the limit parameters for the setpoint

You can change the following setpoint parameters for the relevant control channel in each row:

- "H range": High limit for setpoint operation
- "H range optimization": High limit for optimizing the setpoint
- "Operator specification": Display of the setpoint specified in the standard view, cannot be operated here.
- "Dead band": Error signal generation and dead band (Page 188), Error signal generation and dead band section
- "Optimal setpoint": Calculated by the optimization, cannot be operated
- "L range optimization": Low limit for optimizing the setpoint
- "L range": Low limit for setpoint operation
- "Prefilter": TimeTrig functions (Page 1767), Setpoint filter section

The numerical values for limits are arranged from left to right in descending order. The three values per channel relevant for the optimization are highlighted in blue.

You can find additional information on this in the section Changing values (Page 253) .

The enabled operations for the setpoint parameters are displayed in the preview.

5.15.8.5 MPC10x10 MV parameter view

MV parameter view for MPC10x10

The MV parameter view is a table which contains a row for each manipulated variable channel:

MV channel	H range	L range	Unit	Gradient [Unit/s]
1	100,00	0,00	%	100,00
2	100,00	0,00	%	100,00
3	100,00	0,00	%	100,00
4	100,00	0,00	%	100,00
5	100,00	0,00	%	100,00
6	100,00	0,00	%	100,00
7	100,00	0,00	%	100,00
8	100,00	0,00	%	100,00
9	100,00	0,00	%	100,00
10	100,00	0,00	%	100,00

①

The channel number is at the far left. The physical unit applies to the high and low limit of the respective channel. The gradient is limited in the respective manipulated variable unit per second.

(1) Displaying and changing the limit parameters for the manipulated variable

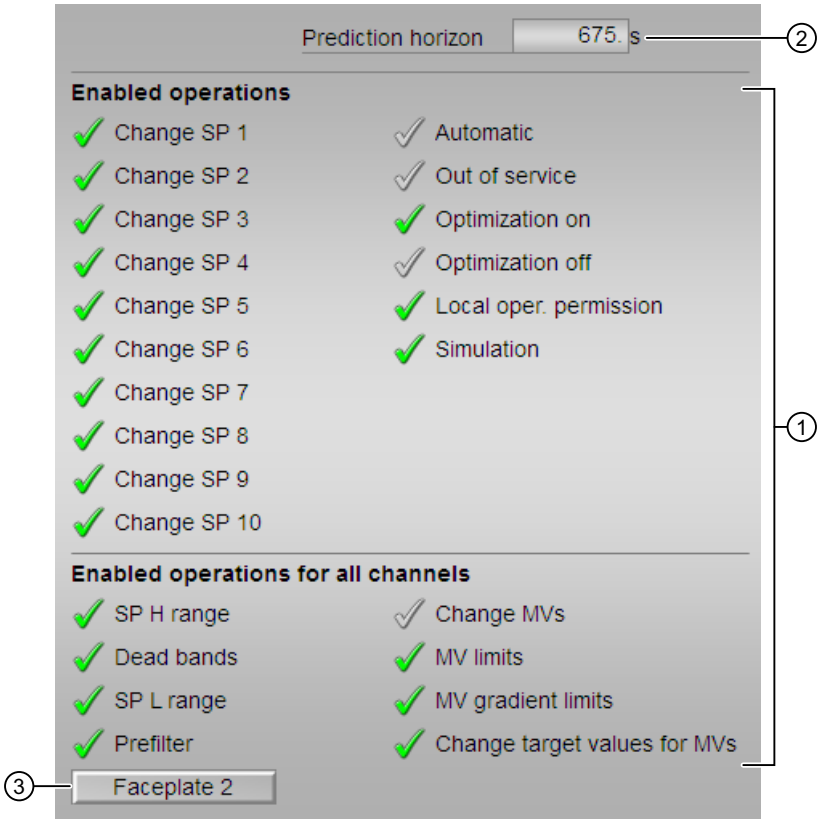
You can change the following parameters for the manipulated variable in this area:

- "H range": Upper limit of the manipulated variable for automatic mode
- "L range": Low limit of manipulated variable for automatic mode
- "Gradient limit": Maximum (absolute) change in the manipulated variable per second

The enabled operations for the setpoint parameters are displayed in the preview.

5.15.8.6 MPC10x10 preview

Preview for MPC10x10



(1) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block. The lower screen half shows operator permissions which apply to all manipulated and control channels.

The upper screen half shows operator permissions which are not channel-based on the left, and channel-specific operator permissions for the individual setpoints on the right.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

The following enabled operations for parameters are shown here:

- "Automatic": You can switch to "automatic mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Optimization on": You can switch on the optimization.
- "Optimization off": You can switch off the optimization.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator permissions (Page 248) .
- "Change SP1..10": You can change the setpoint 1..10
- "Change MVs": You can change the manipulated variables

Note

The OS operator must always be able to switch to "manual mode". For this reason, there is no special operator permission for switching to "manual mode" in the faceplate.

The following operator permissions apply to all manipulated channels.

- "Change MVs": You can change the manipulated variables in manual mode
- "MV limits": You can change the high and low limits for all manipulated variables.
- "MV gradient limits": You can change the gradient limits for all manipulated variables. The following operator permissions apply to all control channels.
  - "SP high limits": You can change the high limits for all setpoints.
  - "Dead bands": You can change the dead bands for all controlled variables.
  - "SP low limits": You can change the low limits for all setpoints.
  - "Prefilter": You can change the time constants of the prefilter for all setpoints.
- "Change target values for MVs": You can change the target values for all manipulated variables.

(2) Prediction horizon

The prediction horizon specifies how far the controller looks into the future in its calculations. The value is set in the MPC Configurator and displayed in the faceplate for informational purposes.

(3) Navigation button for switching to the standard view of any faceplate

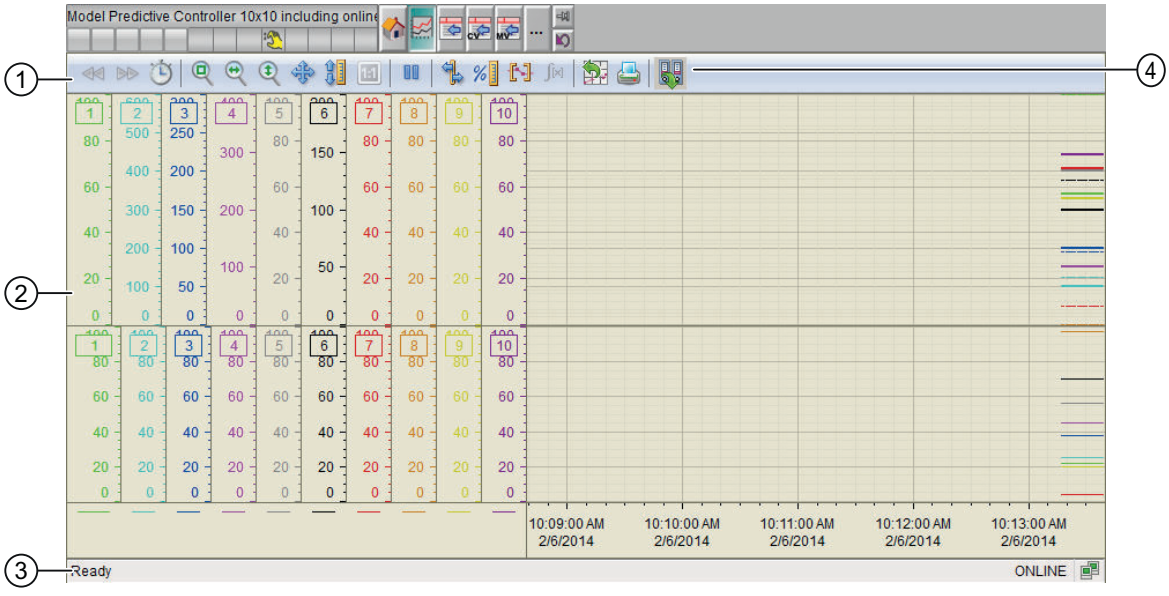
Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

5.15.8.7 MPC10x10 trend view

MPC10x10 trend view

There is a block-specific trend view for MPC10x10 which is supplied as file @PG_APL_TrendMPC_L.pdl and which you can modify if necessary.



(1) Toolbar

(2) Display area for trends

(3) Status bar

(4) Button for switching between archive tags and online tags. The status bar shows if the trend view is working with online data or archive data.

The Export button is only visible and operable with the "Higher-level process control" operating permission.

For additional information about the trend view, refer to the *WinCC Information System* Online Help.

The trend view is divided into two screen halves.

The upper screen half shows all controlled variables with their associated setpoints. The setpoint is shown in the same color as the associated process value to allow the assignment

to be identified straight away. Setpoints are dashed lines, process values are bold lines. If a controlled variable is exactly on the setpoint, it hides the setpoint.

The lower screen half shows all manipulated variables.

Both screen halves use the same color sequence for the individual channels. The sequence starts at channel 1 with green (standard color for the process value with the PID controller) and then goes through the color spectrum from top to bottom as far as gray and black. From channel 7 (red), the rest of the spectrum is run through from top to bottom. Each channel has its own y-axis in the corresponding color.

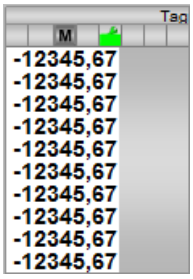
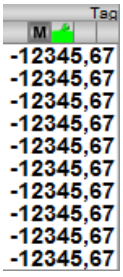
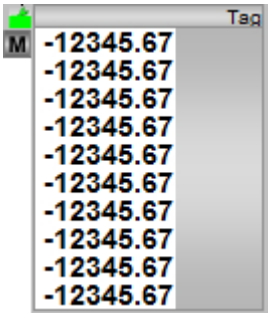
5.15.8.8 Block icon for MPC10x10

Block icons for MPC10x10

A variety of block icons are available with the following functions:

- Process tag type
- Operating modes
- Signal status, release for maintenance
- Tracking
- Memo display
- Process value (black, with and without decimal places)

The display of the tag name can be activated centrally for all block instances with the WinCC variable @APLShowTag:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	Narrow, without units
	3	Block icon in the full display

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Operation via the block icon (Page 234)
- Block icon structure (Page 226).

5.16 KalFilt - State estimator

5.16.1 Description of KalFilt

Object name (type + number) and family

Type + number: FB 1925

Family: Control

Area of application for KalFilt

The Kalman filter is a non-linear stochastic state observer. It forms a framework for parallel operation of simulation of a dynamic process model, whose states are compared with real data in each sampling. During runtime, all internal state variables may be read from the model and used as a substitute for real measured values.

The state observer thus makes it possible to determine the physical parameters of a process unit that is not accessible or only accessible with a very large metrological effort from other readily available metrics. For example, temperatures and mass flows.

It can therefore be used as a soft sensor, that is it works as a software that replaces a real sensor.

In addition to its use as a state observer, it is also used as a model-based filter. The measured variables calculated from the estimated states are not subjected to measurement noise and therefore generally run quieter than noisy measurements.

How it works

The extended Kalman filter

The block operates as a state observer, in which the algorithm of the extended Kalman filter is implemented and the online calculation of the internal states of non-linear dynamic systems is performed. Both the mathematical structure of the underlying dynamic system as well as the properties of the stochastic disturbances must be known.

The term "filter" indicates that the algorithm is able to reduce measurement noise and other disturbances of stochastic data.

The Kalman filter is an approach to solve the general problem of estimating the states of a time-discrete process, which is described by linear stochastic differential equations, and is a generalization of the classical state observer (Luenberger observer) in this respect. The estimation problem is extended to non-linear processes that result from a non-linear state space model.

$$\underline{x}_{k+1} = \underline{f}(\underline{x}_k, \underline{u}_k) + \underline{w}_k$$

$$\underline{y}_k = \underline{h}(\underline{x}_k) + \underline{v}_k$$

Symbols	Meaning
$\underline{x}_k$	Vector of the (inner) state variables at the time t_k
$\underline{u}_k$	Vector of the input variables at the time t_k
$\underline{y}_k$	Vector of the measurable output variables at the time t_k
$\underline{\Phi}_k$	Dynamic matrix (Jakobi matrix) => see FC KalMod
$\underline{H}_k$	Output matrix (Jakobi matrix) => see FC KalMod
$\underline{P}_k$	Estimation error covariance at the time t_k
$\underline{K}_k$	Kalman gain at the time t_k
$\underline{Q}_k$	Covariance of the process noise at the time t_k
$\underline{R}_k$	Covariance of the measurement noise at the time t_k
w_k, v_k	Stochastic interference

For the Kalman algorithm, knowledge about the observed process must be provided to the block *a priori*. This includes the following:

- The covariance of the process noise:
  $\underline{Q}_k$
- The covariance of the measurement noise:
  R_k
- The initial values (initialization) for the *a priori* state estimate:
  $\hat{\underline{x}}_0^-$
- The a priori estimate error covariance:
  $\underline{P}_0^-$

For more details on the configuration of the filter, refer to the "*Kalman Configurator User Documentation*". The setting, identification, simulation, and validation of the filter are performed in the Kalman Configurator.

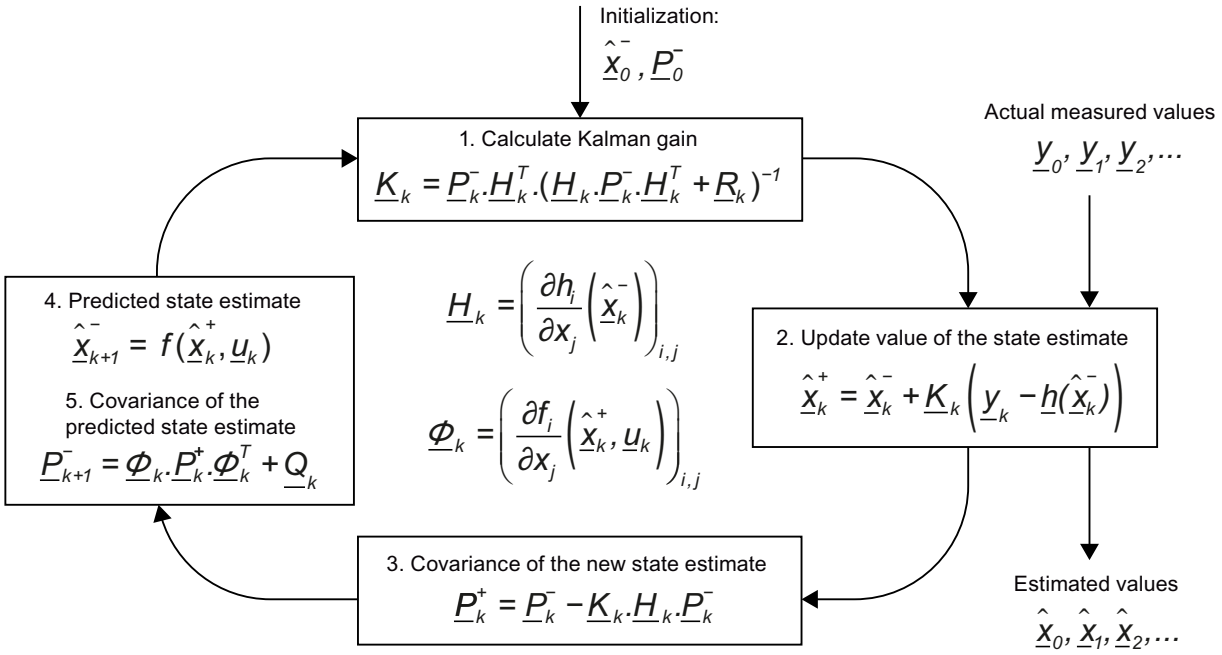


Figure 5-1 Implemented sequence of the extended Kalman filter

The sequence of the extended Kalman filter (EKF) corresponds to the linear filter. This linear filter is aided by

- the Kalman Gain
  $\underline{K}_k$
- the predicted state estimate (a priori)
  $\hat{\underline{x}}_k^-$

The predicted state estimate (a priori) are corrected with the actual measured values:

$\underline{y}_k$

This results in the corrected (a posteriori) state estimate:

$\hat{\underline{x}}_k^+$

The two additional intermediate steps for calculating the Jacobi matrices are:

- Output matrix:
  $\underline{H}_k$
- Dynamic matrix:
  $\underline{\Phi}_k$

The calculations of the Jacobi matrices is performed in the FC block KalMod. For more details about the Kalman filter, refer to /3/.

Consideration of sporadic laboratory measurements

When sporadic laboratory measurements of relevant state variables are taken in a system, the inclusion of these laboratory measurements in the extended Kalman filter can lead to significant improvements in estimation accuracy, since the unknown state variables can be compared with the laboratory values. The sample is taken sporadically. Once laboratory results are received, the laboratory values are provided to the EKF. The algorithm takes into account that the laboratory result is associated with a process state which is already in the past.

Operating principle

The following corrected states are used for application as a state observer:

$$\hat{\underline{x}}_k^+$$

(X1...X15)

These can be represented for visualization, interconnections for further calculations, or used for control.

The following output variables calculated from the corrected states are used for application as a model-based filter:

$$\hat{\underline{y}}_k^+ = \underline{h}(\hat{\underline{x}}_k^+)$$

(Y1Filt...Y7Filt)

The advantage over a simple low-pass filter is that the influences of the inputs are provided at no additional delay to the output variables via the model. The signal status of the filtered output values YiFilt are based on the worst signal status of the states.

To assess the quality of the Kalman filter, the following deviations should be used:

- The deviations of the predicted output values from the measured variables:

  $$\underline{e}_k^- = \underline{y}_k - \underline{h}(\hat{\underline{x}}_k^-)$$

  (EY1...EY7)

- The deviations of the predicted state variables from the laboratory measurements:

  $$\underline{e}_{nk}^- = \underline{y}_{nk} - \underline{h}_n(\hat{\underline{x}}_k^-)$$

  (EYn1...EYn7)

The deviations are usually caused by modeling inaccuracies and unknown or unconsidered disturbances in the model.

In many applications, there are secondary variables that can be calculated using algebraic equations from state and input variables that are relevant for the practical use of the Kalman filter. For example, exothermicity with calorimetric reaction observers and specific conversion rates with the Kalman filter for organic fermenter. These secondary variables can be defined as additional process output variables in the Kalman configurator, for which there is a respective output equation, but no measured values. The associated entries in the R matrix

are assigned the value "inf" (infinity) for the uncertainty, so that the Kalman filter will not use the measured values to correct the states. The associated input variables `Yj` at the function block are supplied through interconnection with the values from output variables `YjFilt` calculated in the last sampling so that no great fictitious estimation error `EYj` is displayed.

A similar procedure is recommended for application of the filter block as a process simulator: the input variables `Ui` are interconnected to the actual input values to simulate the characteristics of the process and get simulated measured values at the output variables `YjFilt`. Since there are no measurements for the process output variables `Yj`. The corresponding inputs at the block are connected to the output variables `YjFilt` and large values are set in the associated elements of the R matrix, so that the Kalman filter will not use the measured values for the correction of the states. The elements of the Q matrix are all set to zero, that is no model uncertainty is assumed for the simulation.

Kalman Configurator

The Kalman Configurator is implemented in Matlab software tool. Using this tool, the user can conveniently configure the process model and make all model specific settings for the Kalman filter. As a result, the tool generates the source code of the FC KalMod. Once the user has compiled the block, it can be downloaded to the automation system. For more details regarding the Kalman Configurator, refer to the "*Kalman Configurator User Documentation*".

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Perform the following configuration tasks:

- Determining the number of configured input variables (`NumberU`)
- Determining the number of configured output variables (`NumberY`)
- Determining the number of configured state variables (`NumberX`)
- Interconnection of the input and output variables `U1`...`U7` or `Y1`...`Y7`
- Determining the designations for all configured input, output, and state variables at the attributes (identifiers)
- Selection of the associated process model in FC KalMod through `ModelNo` (1…5)
- Possible interconnection of the initial values for the configured state variables (if these are not always the same but one signal source is known)
- Recording of measured data and configuration of the Kalman filter through the Kalman Configurator (refer to the *Kalman Configurator User Documentation*)

To compile the SCL sources generated by the configurator, the following user defined data types are needed:

- UDT???: Kal_DB
- UDT???: Matrix15
- UDT???: Vector15

Data types declared by user himself may not be within this number range.

Additional configuration tasks for inclusion of the laboratory measurements:

- General enable for lab measurements (`SampleEn` = 1)
- Determining the amount of lab data (`NumberYn`)
- Selection of the internal or external lab sampling (`SaPo_LiOp` = 1, through interconnection)
- Selection of the internal or external lab measurement (`SaRe_LiOp` = 1, through interconnection)
- The following data must be available at the time `SaRe_Ext` = 1 and `SaRe_LiOp` = 1 or `SaRe_Int` = 1 and `SaRe_LiOp` = 0:
  - Lab measurements `Yn1_Ext`...`Yn5_Ext` or `Yn1_Int`...`Yn5_Int`
  - Diagonal elements of the covariance matrix of the measurement noise `Rn11`...`Rn55` from the Kalman Configurator
- Determining the monitoring time `TimeMon`:
  When this time has expired, the wait on the laboratory test results is terminated.

The library contains the following blocks:

Icon	Address	Data type	Comment
KalFilt	FB 1925	FB 1925	State observer based on the algorithm of the extended Kalman filter.
KalFunct	FB1926	FB1926	Multiplexer for calling the correct model
KalMod1…KalMod5	FC 461…FC 465	FC 461…FC 465	Process model including Jacobi matrices and filter parameters, created in the configurator
MxAdd	FC 466	FC 466	Adds two matrices (used internally)
MxCBnd	FC 467	FC 467	Merging by column (columnbind) of two matrices (used internally)
MxInv	FC 468	FC 468	Returns an inverse of a matrix (used internally)
MxMul	FC 469	FC 469	Multiplies two matrices (used internally)
MxRBnd	FC 470	FC 470	Merging by row (rowbind) of two matrices (used internally)
MxSub	FC 471	FC 471	Subtracts two matrices (used internally)
MxTrans	FC 472	FC 472	Returns a transpose of a matrix (used internally)
MxUnity	FC 473	FC 473	Generates an identity matrix (used internally)
MxZero	FC 474	FC 474	Generates a zero matrix (used internally)
VECTOR	UDT	UDT	Data type for vector calculations
MATRIX	UDT	UDT	Data type for matrix calculations
Kal_DB	UDT	UDT	Data type for user data

Startup characteristics

At the start of a run or a batch, the algorithm must be initialized and all states set to the initial values. This initialization is triggered by the binary input variable `Reset`, which is automatically reset after executing the initialization and is also accessible in the faceplate.

If the values are read from the FC generated by Kalman Configurator during initialization, the binary input variable `LoadParam` should be used instead, which is automatically reset after a successful read operation as well, but is not accessible in the faceplate. The current values

set at the corresponding input variables or in the faceplate are overwritten (for example, initial values of the states) when the parameters are loaded from the FC.

When the CPU starts up, the block is reinitialized and the parameters are loaded from the FC (in accordance with `LoadParam`).

Status word allocation for `Status1` parameter

You can find description of each parameter in the section KalFilt I/Os (Page 974).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct`
4	`OosLi`
5	Not used
6	`OnAct`
7	`SampleEn`, enable for introducing the laboratory values
8	`SamplePoint`, laboratory sampling performed
9	`SampleResult`, lab results incorporated
10	Laboratory results bad → lab results are ignored
11	Waiting period for laboratory measurement results has expired
12	Laboratory measurement results adapted to limits
13	`SaPo_ExtAct`, external/internal lab sampling
14	`SaRe_ExtAct`, external/internal lab measurement
15	Not used
16 - 25	`UserAna1` to `UserAna10` interconnected
26 - 31	Not used

See also

KalFilt modes (Page 971)

KalFilt functions (Page 971)

KalFilt error handling (Page 973)

KalFilt messaging (Page 980)

KalFilt block diagram (Page 981)

5.16.2 KalFilt modes

Operating modes of KalFilt

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

You can find general information about the "On" mode in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of KalFilt (Page 964)

KalFilt functions (Page 971)

KalFilt error handling (Page 973)

KalFilt I/Os (Page 974)

KalFilt messaging (Page 980)

KalFilt block diagram (Page 981)

5.16.3 KalFilt functions

Functions of KalFilt

The functions for this block are listed below.

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

- Signal status for the state estimated values `X1`...`X15`:
  - When limited (to the high or low limit), the signal status 16#78 (uncertain, process-related) is outputted at the relevant state estimate.
  - If the REAL value of the input or output variables is "infinity" (+#INF and -#INF) or "not a number" (+#NAN and -#NAN), the signal status 16#28 (Bad) is assigned to all state variables.
  - If a signal status of the input variables (`Ui`) is bad, all state variables obtain the signal status `ST_Worst`.
  - If a signal status of the output variables (`Yi`) is bad, the associated covariance element of the measurement noise `Rii` is set to 1.0e+10 and all state variables obtain the signal status 16#68 (uncertain, device related).
  - If a signal status of the lab measured values (`Yni`) is bad or the lab value cannot be shown in the REAL number field, all lab values are ignored. The signal status of the lab measured values has no affect on the signal status `ST_Worst`.
  - Otherwise, the state estimated values have the signal status 16#80 (Good).
- The worst signal status `ST_Worst` for the block is formed from the parameters (`U1.ST` to `U7.ST`).

Configurable reactions using the `Feature` parameter

An overview of all the reactions that are provided by the `Feature` parameter is available in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode
2	Resetting the commands for changing the mode (Page 160)

Operator permissions

The block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch the block to "On" mode
2	Not used
3	1 = Operator can switch the block to "Out of service" mode
4	Not used
5	1 = Operator can perform a block restart

Bit	Function
6	1 = Operator can change SimUi simulation values
7	1 = Operator can change SimYi simulation values
8	1 = Operator can enter the initial values for XiStart states
9	1 = Operator can change the high limit for XiHiLim states
10	1 = Operator can change the low limits for XiLoLim states
11	1 = Operator can enable the simulation function
12	Not used
13	1 = Operator can enter lab measured values and operate SaPo and SaRe
14	1 = Operator can specify the maximum duration for lab data and limits for errors in lab values
15	1 = Operator can change the scaling of the bar graph display for the estimation error in the standard view of the faceplate
16 - 31	Not used

See also

Description of KalFilt (Page 964)
KalFilt modes (Page 971)
KalFilt error handling (Page 973)
KalFilt I/Os (Page 974)
KalFilt messaging (Page 980)
KalFilt block diagram (Page 981)

5.16.4 KalFilt error handling

Error handling of KalFilt

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers in ErrorNum
- Output bit ErrorPara = 1

Overview of error numbers

The ErrorNum output parameter can be used to output the following error numbers:

Error number	Meaning of the error number
–1	Predefined value when the block is inserted; the block is not executed.
0	There is no error.

Error number	Meaning of the error number
2	`SampleTime` < 0.001
3	Block FB KalFunct called internally returns an error
4	Parameter assignment error with `NumberU`/`NumberY`/`NumberX`/`NumberYn`
5	Block requires a restart due to a change of the matrix dimensions
6	Error loading model FC from the Kalman Configurator. The following sources of error are possible: • FC is not available or too short • FC is not Kalman Configurator FC • Required matrix dimensions are not supported • Required sampling time for Kalman filter is less than the cycle time of the cyclic interrupt OB in which the block is integrated
7	Only during startup: Initial values of the state variables are out of range
21 - 27	Signal status of the input variable `U1`...`U7` is bad
30	The value of an input variable `U1`…`U7` cannot be displayed in the REAL number field

See also

Description of KalFilt (Page 964)
KalFilt modes (Page 971)
KalFilt functions (Page 971)
KalFilt I/Os (Page 974)
KalFilt messaging (Page 980)
KalFilt block diagram (Page 981)

5.16.5 KalFilt I/Os

Input parameters

Parameters	Description	Type	Default
`BatchEn`	1 = Enable allocation for Batch control	BOOL	0
`BatchID`	Current batch ID	DWORD	16#00000000
`BatchName`	Current batch designation	STRING[32]	''
`EN`	1 = Called block will be processed	BOOL	1
`EYn1Lim...EYn5Lim`	Limits for the difference between lab measurement results and associated state values	REAL	0.0
`EYPScale`	Scaling of the bar graph display of the estimation error in the faceplate	REAL	100.0

Parameters	Description	Type	Default
Feature	I/O for additional functions (Page 971)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
LoadParam	1 = Restart the device with loading of the parameters from the FC	BOOL	Normal oper.
ModelNo	Process model number (assignment of the appropriate process model in KalFunct)	INT	1
Occupied	1 = Occupied by batch control	BOOL	0
OnOp	1=On Mode: On Mode by Operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp	1 = "Out of service", via OS operator	BOOL	0
OS_Perm	I/O for operator permissions (Page 971)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
Q1010	Diagonal elements of the covariance matrix of the process noise Q1010	REAL	0.0
Q1111	Diagonal elements of the covariance matrix of the process noise Q1111	REAL	0.0
Q1212	Diagonal elements of the covariance matrix of the process noise Q1212	REAL	0.0
Q1313	Diagonal elements of the covariance matrix of the process noise Q1313	REAL	0.0
Q1414	Diagonal elements of the covariance matrix of the process noise Q1414	REAL	0.0
Q1515	Diagonal elements of the covariance matrix of the process noise Q1515	REAL	0.0
Q11	Diagonal elements of the covariance matrix of the process noise Q11	REAL	0.0
Q22	Diagonal elements of the covariance matrix of the process noise Q22	REAL	0.0
Q33	Diagonal elements of the covariance matrix of the process noise Q33	REAL	0.0
Q44	Diagonal elements of the covariance matrix of the process noise Q44	REAL	0.0
Q55	Diagonal elements of the covariance matrix of the process noise Q55	REAL	0.0
Q66	Diagonal elements of the covariance matrix of the process noise Q66	REAL	0.0
Q77	Diagonal elements of the covariance matrix of the process noise Q77	REAL	0.0

Parameters	Description	Type	Default
Q88	Diagonal elements of the covariance matrix of the process noise Q88	REAL	0.0
Q99	Diagonal elements of the covariance matrix of the process noise Q99	REAL	0.0
Reset	1 = Reset the states X1...X15 to their initial values	BOOL	Normal oper.
R11	Diagonal elements of the covariance matrix of the measurement noise R11	REAL	0.0
R22	Diagonal elements of the covariance matrix of the measurement noise R22	REAL	0.0
R33	Diagonal elements of the covariance matrix of the measurement noise R33	REAL	0.0
R44	Diagonal elements of the covariance matrix of the measurement noise R44	REAL	0.0
R55	Diagonal elements of the covariance matrix of the measurement noise R55	REAL	0.0
R66	Diagonal elements of the covariance matrix of the measurement noise R66	REAL	0.0
R77	Diagonal elements of the covariance matrix of the measurement noise R77	REAL	0.0
Rn11	Diagonal elements of the covariance matrix of the measurement noise for lab measured values Rn11	REAL	0.0
Rn22	Diagonal elements of the covariance matrix of the measurement noise for lab measured values Rn22	REAL	0.0
Rn33	Diagonal elements of the covariance matrix of the measurement noise for lab measured values Rn33	REAL	0.0
Rn44	Diagonal elements of the covariance matrix of the measurement noise for lab measured values Rn44	REAL	0.0
Rn55	Diagonal elements of the covariance matrix of the measurement noise for lab measured values Rn55	REAL	0.0
SampleEn	Enable for sporadic laboratory measurements	BOOL	0
SampleTime	Sample time in s (assigned automatically)	REAL	0.1
SaPo_LiOp	Select lab sampling (internal/external): 1 = Via interconnection (external) 0 = Via operator (internal)	BOOL	0
SaPo_Int	1 = Lab sampling internal (via operator)	BOOL	0
SaPo_Ext	1 = Lab sampling external (via interconnection)	BOOL	0
SaRe_LiOp	Select lab result (internal/external): 1 = Via interconnection (external) 0 = Via operator (internal)	BOOL	0
SaRe_Int	1 = Lab result internal (input by operator)	BOOL	0
SaRe_Ext	1 = Lab result external (values comes via interconnection)	BOOL	0
SelFp1	1 = Call a block saved in this parameter as an additional faceplate in the standard view	ANY	
SelFp2	1 = Call a block saved in this parameter as an additional faceplate in the preview	ANY	

Parameters	Description	Type	Default
SimOn	Switch on/off of the simulation	BOOL	0
SimU1...SimU7	Simulation values for U1...U7	REAL	0.0
SimY1 ... SimY7	Simulation values for Y1...Y7	REAL	0.0
StepNo	Batch step number	DWORD	16#00000000
TimeMon	Time monitoring for sampling in min	INT	60
U1...U7	Input variables	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
U1Unit...U7Unit	Unit for input variables U1...U7	INT	1342
U1LimScale...U7LimScale	Limit of the value range and scaling for faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
UserAna1...UserAna10	Arbitrary analog values for display in the block icon	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA1Unit...UA10Unit	Unit for UserAna1...UserAna3	INT	1001
UserStatus	Freely assignable user area for status word (bits 24-31)	BYTE	16#00
X1LimScale...X15LimScale	High and low limit for X1...X15	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
X1Start...X15Start	Initial value for the state variable X1...X15	REAL	0.0
X1Unit...X15Unit	Unit for the state variable for X1...X15	INT	1001 (°K)
Y1...Y7	Output variables	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Y1LimScale...Y7LimScale	Limitation of the value range of Y1...Y7 and scaling for faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
Y1Unit...Y7Unit	Unit for the output variables for Y1...Y7	INT	1001
Yn1_Ext...Yn5_Ext	External lab measured value (input for interconnection)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Yn1_Int...Yn5_Int	Internal lab measured value Yn1_Int...Yn5_Int (input by operator)	REAL	0.0
Yn1Unit...Yn5Unit	Unit for the lab measured values Yn1...Yn5	INT	1342 (%)

Parameters	Description	Type	Default
Yn1LimScale...Yn5 LimScale	Limitation of the value range of Yn1...Yn5 and scaling for faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
Z1Unit...Z5Unit	Unit of measurement for Z1...Z5	INT	1001

Output parameters

Parameters	Description	Type	Default
DTimeMon	Current period since the lab sampling	REAL	0.0
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see KalFilt error handling (Page 973).	INT	-1
ErrorPara	1= Parameter assignment error	BOOL	0
EY1...EY7	Deviation between measured value and prediction of the Kalman filter	REAL	0.0
EYn1...EYn5	Deviation between lab measurement and prediction of the Kalman filter	REAL	0.0
EY1P_k1...EY1P_k5	Percent deviation between measured value and prediction of the Kalman filter	REAL	0.0
EY2P_k1...EY2P_k5	Percent deviation between measured value and prediction of the Kalman filter	REAL	0.0
EY3P_k1...EY3P_k5	Percent deviation between measured value and prediction of the Kalman filter	REAL	0.0
EY4P_k1...EY4P_k5	Percent deviation between measured value and prediction of the Kalman filter	REAL	0.0
EY5P_k1...EY5P_k5	Percent deviation between measured value and prediction of the Kalman filter	REAL	0.0
EY6P_k1...EY6P_k5	Percent deviation between measured value and prediction of the Kalman filter	REAL	0.0
EY7P_k1...EY7P_k5	Percent deviation between measured value and prediction of the Kalman filter	REAL	0.0
EYPScaleOut	Calculated scaling of the bar representation estimation error in the standard view of the faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
LabResIncluded	Available laboratory results	BOOL	0
NumberU	Number of input variables U, max. number before loading model	INT	7
NumberX	Number of state variables X	INT	15
NumberY	Number of output variables Y	INT	7
NumberYn	Number of lab sample variables Yn, default: SampleEn = 0	INT	1
NumberZ	Number of lab sample variables Yn, default: SampleEn = 0	INT	0

Parameters	Description	Type	Default
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with the settings changed by the block algorithm block algorithm	DWORD	16#FFFFFFFF
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
P1010	Estimation error covariance 1010	REAL	0.0
P1111	Estimation error covariance 1111	REAL	0.0
P1212	Estimation error covariance 1212	REAL	0.0
P1313	Estimation error covariance 1313	REAL	0.0
P1414	Estimation error covariance 1414	REAL	0.0
P1515	Estimation error covariance 1515	REAL	0.0
P11	Diagonal element of the estimation error covariance 11	REAL	0.0
P22	Diagonal element of the estimation error covariance 22	REAL	0.0
P33	Diagonal element of the estimation error covariance 33	REAL	0.0
P44	Diagonal element of the estimation error covariance 44	REAL	0.0
P55	Diagonal element of the estimation error covariance 55	REAL	0.0
P66	Diagonal element of the estimation error covariance 66	REAL	0.0
P77	Diagonal element of the estimation error covariance 77	REAL	0.0
P88	Diagonal element of the estimation error covariance 88	REAL	0.0
P99	Diagonal element of the estimation error covariance 99	REAL	0.0
SampExtracted	Laboratory sample extracted	BOOL	0
SaPo_ExtAct	1 = External lab sampling is active 0 = Internal lab sampling is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SaRe_ExtAct	1 = External lab results active 0 = Internal lab results active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Status1	Status word 1	DWORD	16#00000000
Status2	Status word 2 (not used)	DWORD	16#00000000
ST_Worst	Worst signal status	BYTE	16#80
X1...X15	Corrected state estimation value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
U1Out...U7Out	Input variables set at the block (copy of U1...U7)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameters	Description	Type	Default
`Y1Filt...Y7Filt`	Process output variables calculated based on the corrected states `X1...X15`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Y1Out...Y7Out`	Process output variables set at the block (copy of `Y1...Y7`)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Yn1Out...Yn5Out`	Lab results set at the block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Z1Out...Z5Out`	Additonal output variable calculated from a posteriori state estimates	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of KalFilt (Page 964)
KalFilt modes (Page 971)
KalFilt messaging (Page 980)
KalFilt block diagram (Page 981)

5.16.6 KalFilt messaging

Messaging

This block does not offer messaging.

See also

Description of KalFilt (Page 964)
KalFilt modes (Page 971)
KalFilt functions (Page 971)
KalFilt error handling (Page 973)
KalFilt I/Os (Page 974)
KalFilt block diagram (Page 981)

5.16.7 KalFilt block diagram

KalFilt block diagram

A block diagram is not provided for this block.

See also

Description of KalFilt (Page 964)

KalFilt modes (Page 971)

KalFilt functions (Page 971)

KalFilt error handling (Page 973)

KalFilt I/Os (Page 974)

KalFilt messaging (Page 980)

5.16.8 Operator control and monitoring

5.16.8.1 KalFilt views

Views of the KalFilt block

The block KalFilt provides the following views:

- KalFilt standard view (Page 982)
- KalFilt parameter view (Page 985)
- KalFilt parameter view 2 (Page 986)
- KalFilt preview (Page 987)
- KalFilt measurements view (Page 988)
- KalFilt trend view (Page 989)
- Batch view (Page 296)
- Memo view (Page 298)
- Block icon for KalFilt (Page 989)

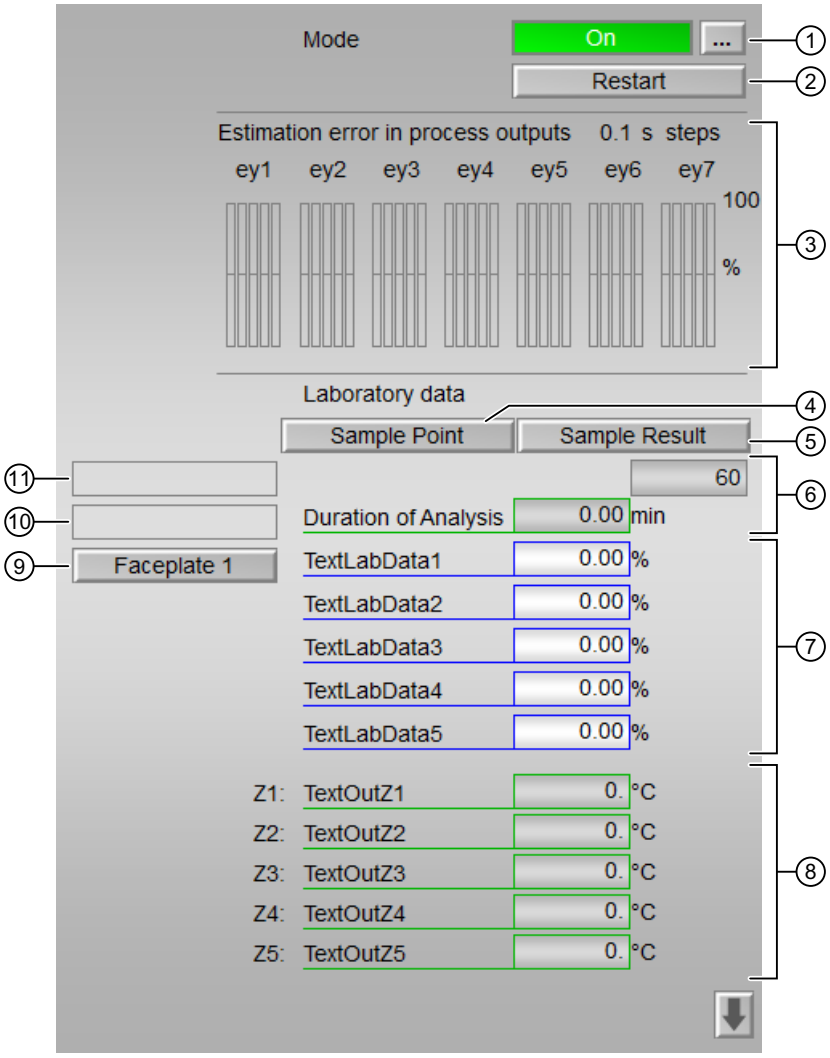
Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

5.16.8.2 KalFilt standard view

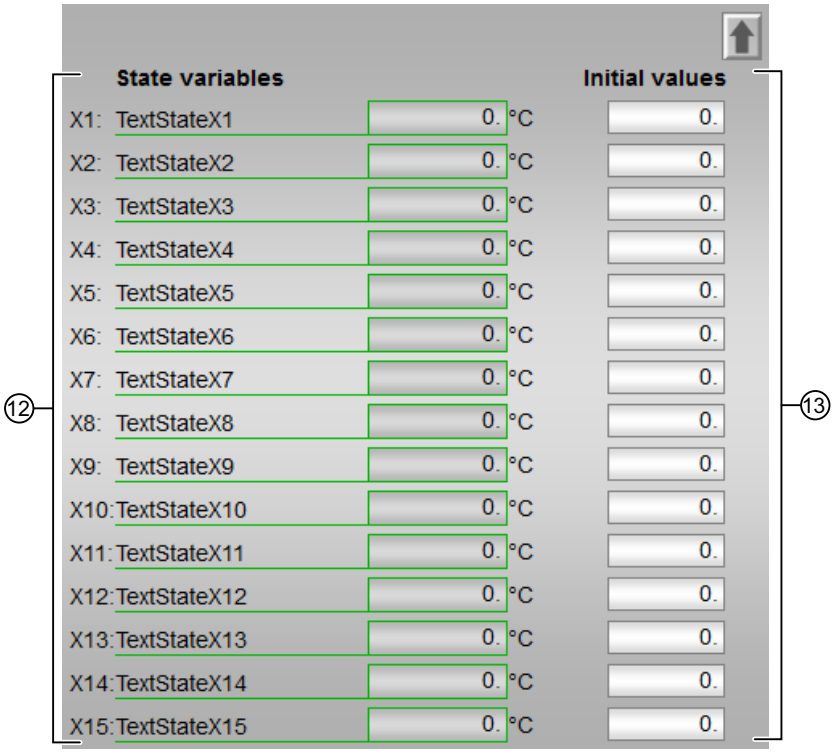
Standard view of KalFilt

The standard view has two views; view 1 and view 2. You can change between the two views with the arrow keys.

View 1



View 2



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating mode can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to chapter Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Restarting the block

Click "Restart" to restart the Kalman filter block.

Either parameters of the associated FCs can be loaded via `LoadParam` or the initial values specified at the block inputs can be set with `Reset`.

(3) Display of the estimation error in process outputs

In this area, the percentage estimation errors are displayed in the existing measurement channels. These are based on the particular value range and are calculated for 5 consecutive sampling tasks in the recent past. The estimation error at the current time is displayed on the far right and the most distant error in the past (5 sampling steps previous) can be seen at the far left. You can change the scaling of the bar in the parameter view (Page 985) of the faceplate.

(4) Sample Point

You can mark the time of the sampling using "Sample Point" button (only if `SaPo_LiOp` = 0).

(5) Sample Result

Click this button to confirm the laboratory results entered in the "TextLabData1" field. Refer to **(7) Laboratory results**.

(6) Duration of Analysis

This field displays the elapsed time since the sampling time in the display. If this is greater than the specified limit (`TimeMon`, can be changed in the parameter view), which is displayed at the upper right field, the result of the laboratory measurement is no longer used.

(7) Laboratory results

You can enter the laboratory results (`Yni_Int`) directly in the "TextLabData1" field. Once they all fully entered, the entry must be confirmed with the "Sample Result" button. The counter stops and the laboratory data are incorporated in the calculations of the Kalman filter.

(8) Additional output variables

If you have output parameters which cannot be measured and are calculated from a posteriori state estimates, you can define and enter the algorithm of maximal 5 additional output parameters (`Z1Out`...`Z5Out`) in the Kalman Configurator.

(9) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system (ES).

The visibility of this navigation button depends on the configuration in the engineering system (ES).

For more information, refer to the section Opening additional faceplates (Page 203).

(10) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You will find more detailed information on this in the sections Simulating signals (Page 58).

(11) Display area for block states

This area provides additional information on the operating state of the block:

- "Sample extracted"
- "Lab results included"

- "Error in lab results"
- "Time out in analysis"
- "Lab results limited"

(12) State variables

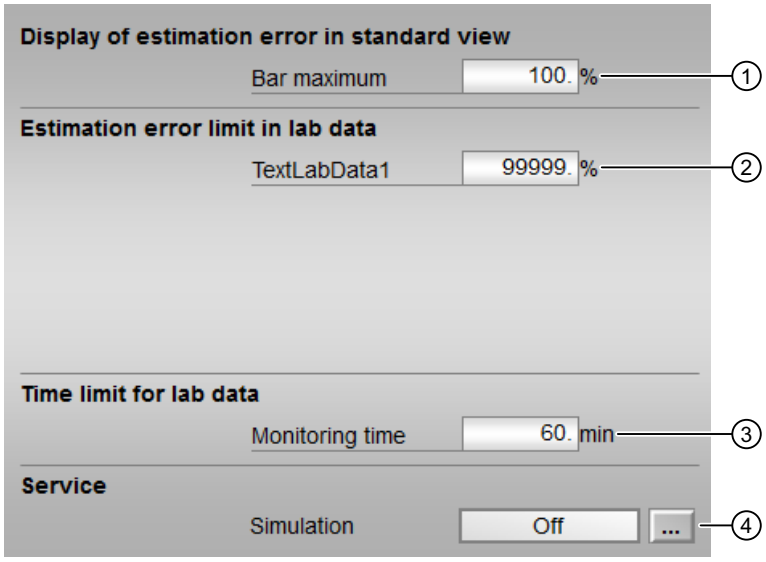
This area shows the current values of the estimated state variables.

(13) Initial values

You can specify intial values for the state variables.

5.16.8.3 KalFilt parameter view

Parameter view of KalFilt



(1) Display of estimation error in standard view

You can set the scale of the bar diagram in the "Bar maximum" (`EYPScale`) field. The bar diagram is displayed in the KalFilt standard view (Page 982).

(2) Estimation error limits in lab data

You can change the maximum permissible deviation of the laboratory data from the estimated variable in the "Text LabData1" field (`EYniLim`).

If the actual error is above this value, the lab results are regarded as implausible and not used for the Kalman filter.

(3) Duration limits in lab data

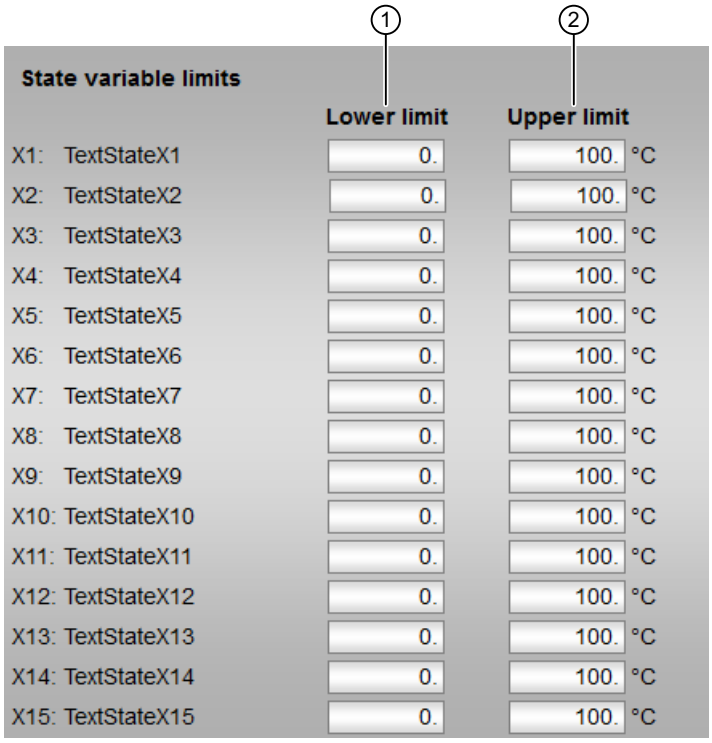
You can adjust the maximum duration for a laboratory measurement in the "Monitoring time" field.

(4) Service

You can enable and disable simulation of the block using the "Simulation" button.

5.16.8.4 KalFilt parameter view 2

Parameter view 2 of KalFilt



(1) Lower limit of state variables

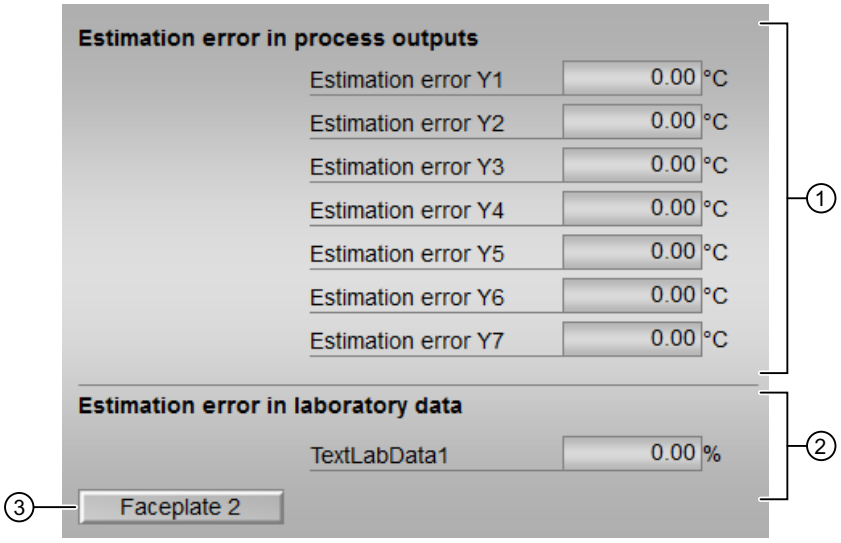
You can specify lower limit of the state variables.

(2) Upper limit of state variables

You can specify upper limit of the state variables.

5.16.8.5 KalFilt preview

Preview of Kalfilt



(1) Estimation error in process variables

This area shows the estimation error in the output variables. The values of the estimation error are shown as physical unit in preview and as percentage in the standard view (Page 982).

(2) Estimation error in laboratory data

This area shows the estimation error in the laboratory results. The values of the estimation error are shown as physical unit in preview and as percentage in the standard view (Page 982).

(3) Navigation button for switching to the standard view of any faceplate

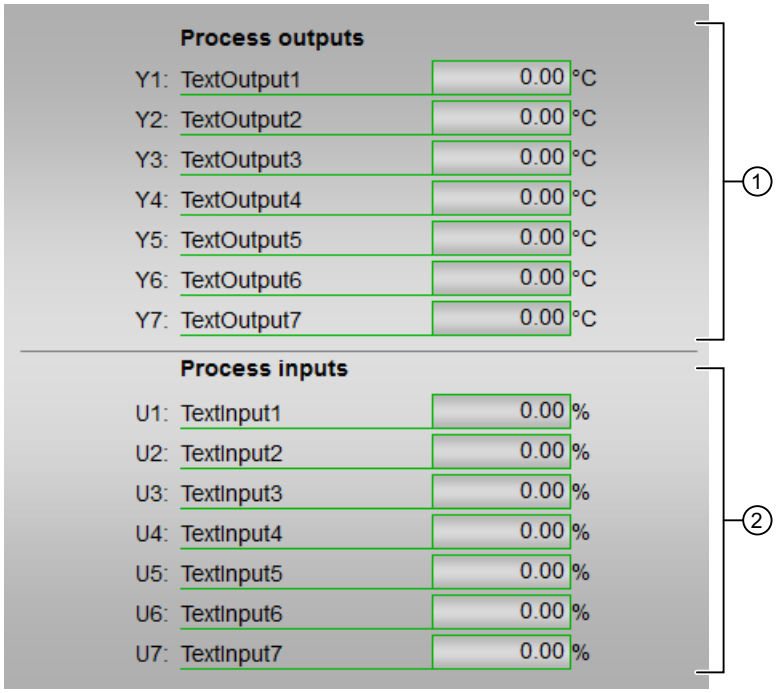
Use this navigation button to open the standard view of a block configured in the engineering system (ES).

The visibility of this navigation button depends on the configuration in the engineering system (ES).

For more information, refer to the section Opening additional faceplates (Page 203).

5.16.8.6 KalFilt measurements view

Measurements view of KalFilt



(1) Process outputs

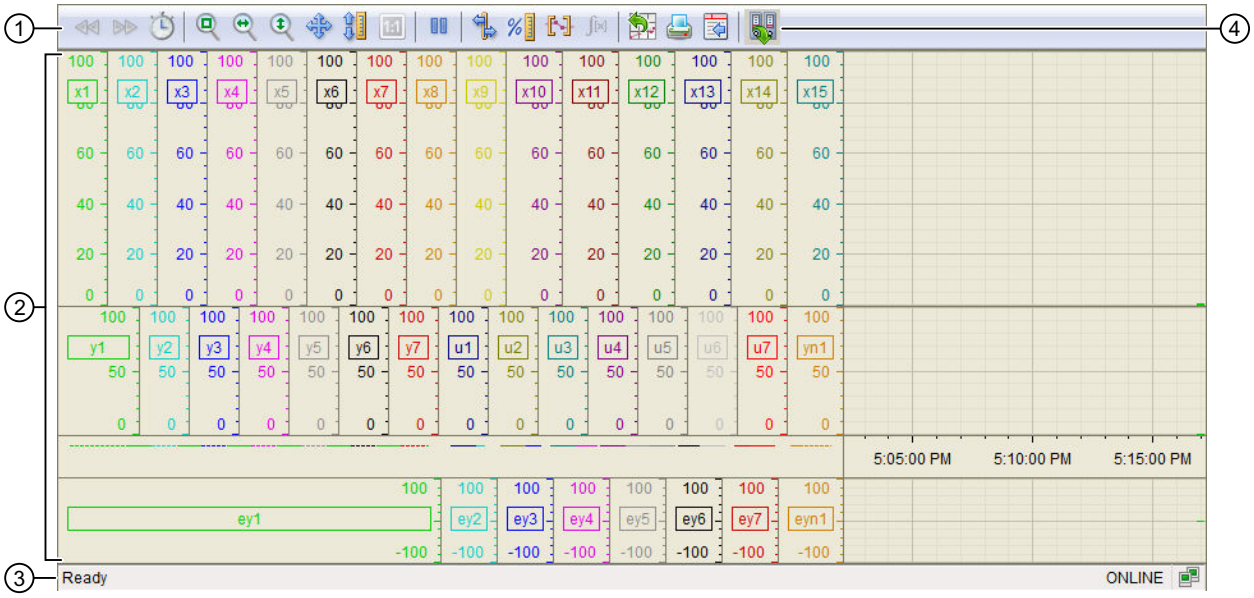
This area displays current values of the process outputs.

(2) Process inputs

This area displays current values of the process inputs.

5.16.8.7 KalFilt trend view

Trend view of KalFilt



(1) Toolbar

(2) Display area for trends

(3) Status bar

(4) Button for switching between archive tags and online tags. The status bar shows if the trend view is working with online data or archive data.

For additional information about the trend view, refer to the *WinCC Information System* Online Help.

The trend view is divided into three areas. The top area shows the state variables, the middle area shows the input and output variables, and the lower area shows the estimation error.

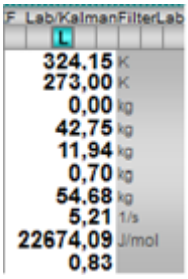
5.16.8.8 Block icon for KalFilt

Block icons for KalFilt

A variety of block icons are available with the following functions:

- Process tag type
- Operating modes
- Signal status, release for maintenance
- Tracking
- Memo display
- Process value (black, with and without decimal places)

The display of the tag name can be activated centrally for all block instances with the WinCC variable @APLShowTag:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon with full display for 10 freely assignable measured values: `UserAna1...UserAna10`

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Operation via the block icon (Page 234)
- Block icon structure (Page 226).

Dosing blocks

6

6.1 DoseL - Dosing device

6.1.1 Description of DoseL

Object name (type + number) and family

Type + number: FB 1809

Family: Dosage

Area of application for DoseL

The block is used for the following applications:

- Single-component dosing using flow measurement
- Weighing of fill/removal volume using dosing scales

How it works

Processing is performed discretely via a coarse/fine flow control with flow monitoring and setpoint specification. The dosing flow can be determined via a maximum of 16 cycles.

A dribbling phase and post dosing can be implemented for both processes. The input value can also be supplied from a pulse module via the `PV1CycLi` output of the Pcs7Cntx (x=1...3) channel block.

Note

When using a pulse module, the counted pulses per cycle must be normalized via the I/Os "Gain" and "Ti" at DoseL .

Example:

A pulse of 35 Kg per hour means that I/O "Gain" must be configured to 35 and I/O "Ti" to 3600.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the DoseL block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Dosing (Dose_Lean) (Page 2332)
- Dosing with PA/FF devices (Dose_Lean_Fb) (Page 2333)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section DoseL I/Os (Page 1017).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutoAct.Value`
6	`LocalAct.Value`
7	0: Open padlock in the block icon 1: Closed padlock in the block icon
8	`SP_ExtAct`
9	`Ctrl`
10	`Ctrl2`
11	1 = Dosing by scale
12	0 = Dosing by scale for filling 1 = Dosing by scale for removal
13	`BypProt` active
14	Invalid signal status
15	Mode switchover error
16	1 = `Intlock` is active
17	1 = `Permit` is active
18	1 = `Protect` is active
19	`DosOn`
20	`DosRelax`
21	`DosEnd`
22	`DosOff`
23	`DosPause`
24	`DosStart`
25	1 = Post dosing
26	"Start" command

Status bit	Parameter
27	"Pause" command
28	"Continue" command
29	"Cancel" command
30	`UserAna1` interconnected
31	`UserAna2` interconnected

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	`DQ_AH_Act xx_Act` taking into account `Feature Bit` 29
2	`DQ_AL_Act xx_Act` taking into account `Feature Bit` 29
3	`DQ_AH_En`
4	`DQ_AL_En`
5	`DQ_AH_MsgEn`
6	`DQ_AL_MsgEn`
7	`PV_AH_Act xx_Act` taking into account `Feature Bit` 29
8	`PV_AL_Act xx_Act` taking into account `Feature Bit` 29
9	`PV_AH_En` and `Feature` Bit 7 = 1 or `PV_AH_En` and flow mode
10	`PV_AL_En` and `Feature` Bit 7 = 1 or `PV_AL_En` and flow mode
11	`PV_AH_MsgEn`
12	`PV_AL_MsgEn`
13	`PV_AH2_Act xx_Act` taking into account `Feature Bit` 29
14	`PV_AL2_Act xx_Act` taking into account `Feature Bit` 29
15	`PV_AH2_En` and `Feature` Bit 7 = 1 or `PV_AH2_En` and flow mode
16	`PV_AL2_En` and `Feature` Bit 7 = 1 or `PV_AL2_En` and flow mode
17	`PV_AH2_MsgEn`
18	`PV_AL2_MsgEn`
19	`CR_AH_Act xx_Act` taking into account `Feature Bit` 29
20	`CR_AH_En` and `Feature` Bit 7 = 1 or `CR_AH_En` and flow mode
21	`CR_AH_MsgEn`
22	Display for interlocks in block icon
23	1 = Scales tared
24	Automatic preview 1 = Dosing "On"
25	Automatic preview 1 = Dosing "Dribbling"
26	Automatic preview 1 = Dosing "End"
27	Automatic preview 1 = Dosing "Off"
28	Automatic preview 1 = Dosing "Pause"
29	Forcen active
30	Bypass information from previous function block
31	`MS_RelOp`

Status word allocation for `Status3` parameter

Status bit	Parameter
0	"Interlock" button is enabled
1	"Permission" button is enabled
2	"Protection" button is enabled
3	`DosCancelMsgEn`
4	`Feature` Bit 7 = 1 (Calculation of the flow rate for dosing by scale)
5	`Feature` Bit 7 = 1 or flow mode
6	Display flow setpoint in percent
7	Display flow setpoint bar
8	Reset request in automatic mode
9	`Feature` bit 8 = 1 (Fine dosing quantity setpoint absolute)
10	`SimLiOp.Value`
11	External error generated by FaultExt or external control system fault CSF with set Feature bit 18 Activating error state for external process control error CSF (Page 150)
12	0 = Button text "Ack underdose" is visible 1 = Button text "Ack overdose" is visible
13	Display "Ack Dos End" block state in the standard view
14 - 18	Not used
19	Display "Forced start"
20	Display "Force abort"
21	Display "Forced pause"
22	Display "Forced continue"
23 - 27	Not used
28	`GrpErr.Value`
29	`RdyToStart.Value`
30	1 = Input parameter `CtrlChnST` is interconnected
31	1 = Input parameter `Ctrl2ChnST` is interconnected

Status word allocation for `Status4` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via `EventTsIn`
1	Effective signal 2 of the message block connected via `EventTsIn`
2	Effective signal 3 of the message block connected via `EventTsIn`
3	Effective signal 4 of the message block connected via `EventTsIn`
4	Effective signal 5 of the message block connected via `EventTsIn`
5	Effective signal 6 of the message block connected via `EventTsIn`
6	Effective signal 7 of the message block connected via `EventTsIn`
7	Effective signal 8 of the message block connected via `EventTsIn`
8	Delay of the `DQ_AH_Tol` message
9	Delay of the `DQ_AL_Tol` message

Status bit	Parameter
10	Delay of the `PV_AH_Lim` message
11	Delay of the `PV_AL_Lim` message
12	Delay of the `PV_AH2_Lim` message
13	Delay of the `PV_AL2_Lim` message
14	Delay of the `CR_AH_Lim` message
15	Collection of message delays
16 - 22	Not used
23	Hidden bypass signal in Permit
24	Hidden bypass signal in interlock
25	Hidden bypass signal in Protect
26	`Feature2` bit 2: Separate bypass signal
27 - 31	Not used

Status word allocation for `Status5` parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via `EventTsIn`
8 - 31	Not used

See also

DoseL functions (Page 997)

DoseL messaging (Page 1014)

DoseL modes (Page 995)

DoseL error handling (Page 1012)

DoseL block diagram (Page 1031)

6.1.2 DoseL modes

DoseL operating modes

The block can be operated using the following modes:

- Local mode (Page 79)
- Automatic mode (Page 75)
- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) section.

Dosing actions you can control in "local mode":

- "Start" (positive edge `StartLocal`)
- "Cancel" (`CancelLocal` = 1)
- "Pause" (`PauseLocal` = 1)
- "Continue" (`ContLocal` = 1)

If you operate the block in "local mode", control will only take place via "local" signals (input parameters `StartLocal` = 1, `CancelLocal` = 1, `PauseLocal` = 1 and `ContLocal` = 1).

Note

Unlike the general description, at the `LocalSetting` parameter only the values 0, 1 and 3 can be set, i.e. tracking in "local" mode is not possible with DoseL.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) section.

Dosing actions you can control in "automatic mode":

- "Start" (positive edge `StartAut`)
- "Cancel" (`CancelAut` = 1)
- "Pause" (`PauseAut` = 1)
- "Continue" (`ContAut` = 1)

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

Dosing actions you can control in "manual mode":

- "Start" (`StartMan` = 1)
- "Cancel" (`CancelMan` = 1)
- "Pause" (`PauseMan` = 1)
- "Continue" (`ContMan` = 1)

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of DoseL (Page 991)
DoseL functions (Page 997)
DoseL error handling (Page 1012)
DoseL messaging (Page 1014)
DoseL I/Os (Page 1017)
DoseL block diagram (Page 1031)

6.1.3 DoseL functions

Functions of DoseL

The functions for this block are listed below.

Status diagram

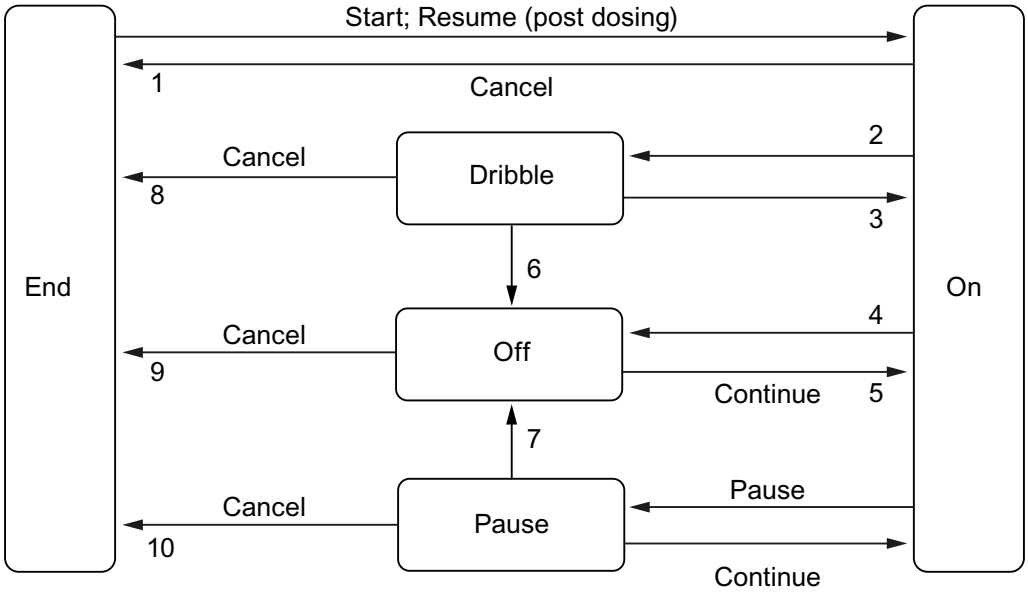
The block provides the following states:

- "End"
  If an underdosage or overdosage is identified, the status display shows the state "Ack Dos End" in the standard view.
- "On"
- "Dribbling"
- "Off"
- "Pause"

The following commands can initiate a state change:

- "Start"
- "Cancel"

- "Pause"
- "Continue"



Dosing can only be started if the dosing setpoint is greater than the current dosing quantity (`DQ_Out`) or if `Feature` Bit 6 is set and no interlock is pending.

However, the following status changes are also performed automatically:

Number in graphic (top)	Function
1	When the dosing quantity is reached, dosing terminates (`DQ_Out` ≥ DQ_SP).
2	If the dosing quantity reaches the dribbling quantity (relative to dosing setpoint `DQ_Out` ≥ `DQ_SP` – `DribbOut`) the dribbling status becomes active.
3	Automatic post dosing in automatic mode via the `Feature` Bit 12, if an underdosage was identified after the dribbling time (`RelaxTime`).
4	• Flow alarm (see input parameter `Feature` Bit 11) or • Interlock. • The dosing process of the DoseL block is set to the "Off" state with a process value status "Bad, device-related" or "Bad, process-related". The response can be set using Feature bit parameter assignment.
5	• The flow alarm is acknowledged (also provided via the "Continue" command if Feature Bit 9 = 1) or • The interlock is acknowledged (also provided via the "Continue" command if Feature Bit 9 = 1). • "Continue" command after underdosing.
6	Underdosage identified after the dribbling time (`RelaxTime`). Overdosage identified after the dribbling time (only with `Feature2` bit 24 = 1).
7	Interlock
8	Dribbling time (`RelaxTime`) has expired and dosing quantity is above the low tolerance limit (`DQ_Out` ≥ `DQ_SP_Tol` - `DQ_AL_Tol`)

Number in graphic (top)	Function
9	The underdosage is acknowledged or the dosing quantity has been reached (DQ_Out ≥ DQ_SP) via creep flow, for example
10	Dosing quantity has been reached (DQ_Out ≥ DQ_SP) via creep flow, for example

Control outputs

The coarse flow (Ctrl) or fine flow (Ctrl2) control outputs are issued in the "On" status.

The coarse flow control output is active if:

- DQ_Out < DQ_SP – DQ2_SP

The fine flow control output is active if:

- DQ_Out ≥ DQ_SP – DQ2_SP
- Where DQ_Out: Dosing quantity actual value
- DQ_SP: Dosing quantity setpoint
- DQ2_SP: fine dosing quantity setpoint ("calculated")

Output signal as a pulse signal or static signal

This block provides the standard function Output signal as a static signal or pulse signal (Page 51).

In addition to the static control outputs Ctrl, Ctrl2, the block also has pulse outputs P_Ctrl, P_Ctrl2, which are dependent on the static control outputs.

Determining the dosing quantity when using flow dosing

You can activate flow dosing using Feature Bit 5 = 0 (Specifying the dosing type (Page 145).)

When using flow dosing, the dosing quantity is determined using the following equation:

$$DQ\_Out = DQ\_Out + \frac{SampleTime}{TI} \cdot \frac{(PV\_Out_{alt} + PV\_Out)}{2}$$

Where:

- DQ_Out: Dosing quantity actual value
- SampleTime: Sampling time [s]
- TI: dimensionless conversion factor for the time basis of the measured value acquisition in the time basis 1 s
- PV_Out_{alt}: Last value PV_OUT

The output parameter PV_OUT is determined using an average calculation:

$$PV\_Out := \frac{PV\_Out_{(t)} + PV\_Out_{(t-1)} + ... + PV\_Out_{(t-(n-1))}}{NumSample_{(N)}}$$

Where:

- `PV_Out(t)` = `PV` · `Gain`

The average calculation is only active if `NumSample` > 1 and `NumSample` ≤ 16 and serves to smooth systematically pulsating measuring signals.

Note

Integration of the flow for the dosing quantity

The integration of the flow for the dosing quantity is implemented using the trapezoid rule. Compared to the summation of rectangles rule, in which the values are simply added up, the procedural error in the case of the trapezoid rule is smaller for the determination of the numerical integral value.

The dosing quantity is determined in the "On" and "Dribbling" state. In the "End", "Off" and "Pause" states, the flow (creep flow) is determined depending on the `Feature` Bit 13 (Creep rate is always detected in the dosing quantity (Page 167)) and the value `CR_AH_Lim`. Creep flow monitoring is disabled with `CR_AH_En` = 0 .

Determining the dosing quantity when dosing using scales

You can activate scale dosing via the `Feature` Bit 5 = 1 (Specifying the dosing type (Page 145).)

The dosing quantity is determined for a scale dosing in the "Start" state following a positive edge at the input parameter `StandStill`. The `StandStill` input parameter is a feedback signal of the scale.

The process of determining the dosing quantity stops after the dosing quantity is reset in the "End" state.

If the signal is no longer available, you need to configure `StandStill` with 1 permanently; the dosing quantity will then be determined right at the start of the dosing.

When weighing the fill volume (`MeterType` = 0), the dosing quantity is determined using the following equations:

$$DQ\_Out = PV\_Out - DQ\_Tare$$

When weighing the removal volume (`MeterType` = 1):

$$DQ\_Out = DQ\_Tare - PV\_Out$$

with `PV_Out` = `PV` · `Gain`

In the "Start" state, the `DQ_Tare` tare memory is set with `PV_Out` with the first positive edge of `StandStill`.

If you have permanently configured `StandStill` with 1, the tare memory is set right at the start of dosing.

Note

The `DQ_Out` output parameter is the dosing quantity and not the actual scale value. In the "End" state (= following resetting of the dosing quantity), nothing further is displayed, since the dosing procedure is complete. When dosing starts, the actual scale value is transferred to the tare memory, and therefore the dosing quantity always remains 0 after the start of dosing. Therefore, the actual scale value is not displayed in the "End" state.

Calculation of the flow rate for dosing by scale

You can activate the calculation of the flow rate via the `Feature` Bit 7 = 1

(Activating calculation of the flow rate for dosing by scale (Page 143)).

The flow rate for scale dosing is determined by the following equations:

$$PV\_Out = (avPV(t) - avPV(t-1)) \cdot \frac{GainEff}{SampleTime}$$

with

$$avPV(t) = \frac{PV(t) + PV(t-1) + ... + PV(t-(N+1))}{NumSample(N)}$$

where `avPV(t)` = internal variable for the mean dosing quantity

Dribbling

The "Dribbling" status is entered automatically in accordance with

- `DQ_Out` ≥ `DQ_SP` – `DribbOut`

Where:

- `DQ_Out`: Dosing quantity actual value
- `DQ_SP`: Dosing quantity setpoint
- `DribbOut`: Dribbling quantity

The dribbling quantity is specified with the input `DribbIn`.

`DribCor` = 1 can be used to automatically determine the dribbling quantity from previous dosing actions:

- `DribbOut` = `DribbOut` - (`DQ_SP` - `DQ_Out`) · `DCF` / 100

`DCF` represents the weighting factor of the last dosing action as a percentage and cannot be set to below 0 or above 100. `DribbOut` is calculated at the end of dosing or the first time an underdosage occurs, and is limited to `DribbMax`. Post dosing is not taken into account.

The "Dribbling" status is active for the period `RelaxTime`. `DribbOut` = 0 deactivates the "Dribbling" status.

When the "Enable configuration of the dribbling quantity (Page 158)" function is enabled (Feature bit 27 = 1) and automatic determination of the dribbling quantity is disabled (DribCor = 0), the quantity can also be configured during dosing, but not in the "Dribbling" state.

Overdosing/underdosing

Once the "Dribbling" state has expired, the dosing quantity is checked for overdosages or underdosages. In the "End" status, no check is performed for over- or underdosages.

An overdosage has occurred if:

- `DQ_Out` > `DQ_SP_Tol` + `DQ_AH_Tol`

An underdosage has occurred if:

- `DQ_Out` < `DQ_SP_Tol` - `DQ_AL_Tol`
- Where `DQ_Out`: Dosing quantity actual value
- `DQ_SP_Tol`: Dosing quantity setpoint for forming tolerance. This is the same as the dosing quantity setpoint (`DQ_SP`) before and after the check that determines overdosing/ underdosing.
- `DQ_AH_Tol`: High tolerance limit
- `DQ_AL_Tol`: Low tolerance limit

If an underdosage is identified after the "Dribbling" state, the "Off" state is entered. The underdosage can be acknowledged in "Off" state (`U_AckOp` = 1 or positive edge `U_AckLi`) and the dosing action completed, or post dosing can be started with the "Continue" command. The dosing end state display in the standard view shows "Ack Dos End".

With `Feature2` bit 24 = 0:

If an overdosage is identified after the "Dribbling“ state, the dosing "End" state is entered at once. A message for overdosage comes and goes in next cycle.

With `Feature2` bit 24 = 1:

If an overdosage is identified after the "Dribbling“ state, the "Off“ state is entered. The overdosage can be acknowledged in the "Off“ state (`U_AckOp` = 1 or positive edge `U_AckLi`) and the dosing action is completed.

Generally, dosages which enter the "Off" state with overdosage, have to be acknowledged.

Post dosing

If an underdosage is detected after dribbling, post dosing can be performed using the "Continue" command. In automatic mode, `Feature` Bit 12 can be used to start post dosing automatically. Post dosing is active for the period `P_DoseTime`. The block then enters the "Dribbling" state or, if that state is deactivated, the "End" state. If the conditions for the "Dribbling" or "End" states are met within the period `P_DoseTime`, the block switches to these states immediately.

Post dosing can also be carried out in the "End" state by increasing the dosing quantity setpoint and issuing the "Continue" command. Once the dosing quantity has been reset, post dosing can no longer be carried out by means of increasing the setpoint.

If the dribbling correction quantity is smaller than the dribbling quantity, dosing is started for time `P_DoseTime`. If no parameter assignment has been made for `P_DoseTime`, the coarse flow (`Ctrl`) or fine flow (`Ctrl2`) control outputs are set for one cycle.

If no parameter assignment has been made for the dribbling quantity, or the dribbling correction quantity is larger than the dribbling quantity, the dosing procedure is continued without taking time `P_DoseTime` into account.

Information on post dosing via setpoint increase:

- After the setpoint has been increased, the tolerance limits are only updated when the "Continue" command is issued. This ensures the overdosage/underdosage display remains consistent.
- Post dosing can only be carried out after a dosing procedure has been aborted if the setpoint for the dosing quantity has been increased and an actual-value quantity is present.
- In "Automatic" and "Local" modes, the "Continue" command must be issued on a positive edge to prevent unwanted dosing procedures from taking place after a setpoint increase.

With `Feature2` bit 24 = 0:

If an overdosage is identified during or after a post dosing, the dosage enters in the "End" state.

With `Feature2` bit 24 = 1:

If an overdosage is identified after a post dosing or a drain, the dosage continues to be in the "Off" state. Only with acknowledge overdosage, the block enters to the "End" state with the status display "Ack Dos End" in the standard view.

Resetting the dosing quantity

The dosing quantity can only be reset in the "End" state using `RstDQ_Op` = 1 or positive edge `RstDQ_Li`.

The state display "Ack Dos End" is also resetted to "End".

`Feature` Bit 6 can be used to reset the dosing quantity automatically when dosing starts.

External/internal setpoint specification

This block provides the standard function Setpoint specification - internal/external (Page 127).

The block always requires the setpoint for the dosing quantity (dosing setpoint). All I/Os for the dosing setpoint start with `DQ_...`. The dosing setpoint is comprised of the coarse and fine setpoints for the coarse/fine flow control. All I/Os for the fine setpoint start with `DQ2_...`. The coarse setpoint is generated internally from the dosing and fine setpoints and is displayed at the output with `DQ1_SP`. If no fine setpoint is available, the coarse setpoint is equal to the dosing setpoint.

Flow setpoints can be specified for the coarse/fine flow setpoint specification. All I/Os for the coarse flow start with `SP_...`, and all I/Os for the fine flow with `SP2_...`. The setpoint for coarse or fine flow is displayed at the output `SP` in the "On" status, depending on the coarse/fine flow control.

Setpoint limitation

The setpoints are limited in the block using shared parameters:

- Coarse metering volume: `DQ_HiLim`, `DQ_LoLim`
- Fine metering volume: `DQ2_HiLim`, `DQ2_LoLim`
- Coarse flow rate: `SP_HiLim`, `SP_LoLim`
- Fine flow rate: `SP2_HiLim`, `SP2_LoLim`

If the setpoint limit is violated, external setpoints are limited to the limit you specified. If there are internal setpoints, the last valid value is output.

Bumpless switchover from external to internal setpoint

The parameter `SP_TrkExt` = 1 is used so that the internal setpoint tracks the external setpoint to achieve a bumpless switchover from the external to the internal setpoint. This allows unwanted jumps at the output parameter to be avoided.

Limit monitoring of the process value

This block provides the standard function Limit monitoring of the process value (Page 86). The calculated flow `PV_Out` is monitored in terms of the limit pairs `PV_AH_Lim` / `PV_AL_Lim` (coarse flow) or `PV_AH2_Lim` / `PV_AL2_Lim` (fine flow), depending on the coarse/fine flow control. Monitoring is only active in the "On" status.

The calculated flow `PV_Out` is monitored in terms of the limit pairs `PV_AH_Lim` / `PV_AL_Lim` (coarse flow) or `PV_AH2_Lim` / `PV_AL2_Lim` (fine flow), depending on the coarse/fine flow control. Monitoring is only active in the "On" status.

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Forcing operating modes

This block provides the standard function Forcing operating modes (Page 41). The inputs `StartForce`, `CancelForce`, `PauseForce` and `ContForce` force the block into the states "On", "End" or "Pause".

The "Pause" status can only be forced if the block is in the "On" status.

The "On" state can only be forced from the states

- "Off"
- "Pause"
- "End" if the quantity actual valve is less than the quantity setpoint.

Simulating signals

This block provides the standard function Simulating signals (Page 58).

The flow is simulated with `SimPV`. This input is not active in scale mode, even when the flow is formed by changing quantities (`Feature` Bit 7 = 1)

You can simulate the following values:

- Flow dosing: The flow (`PV`) is specified in the setpoint view with the input (`SimPV`, `SimPV_Li`).
- Dosing using scales: The dosing quantity (`DQ_Out`) is specified in the standard view with the input (`SimDQ`, `SimDQ_Li`).

Interlocks

This block provides the following interlocks:

- Activation enable
- Interlock without reset ("Interlock")
- Interlock with reset ("Protection")

Refer to the section Interlocks (Page 99).

Dosing can only be started if no interlock is present. The interlock function is not active in the "Dribbling" status; nor is the activation enable active in the "On" status. An interlock in the "On" or "Pause" states will cause the block to enter the "Off" status.

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Resetting the block in case of interlocks or errors

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See DoseL error handling (Page 1012)

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `FaultExt`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `PV_Out.ST`
- `DQ_Out.ST`
- `DQ_SP.ST`
- `SP.ST`
- `Standstill.ST`
- `LocalLi.ST`
- `StartLocal.ST`
- `CancelLocal.ST`
- `PauseLocal.ST`
- `ContLocal.ST`
- `Ctrl1ChnST`
- `Ctrl2ChnST`
- `StartAut.ST` (only if `Feature2.Bit10` = 1)
- `CancelAut.ST` (only if `Feature2.Bit10` = 1)
- `PauseAut.ST` (only if `Feature2.Bit10` = 1)
- `ContAut.ST` (only if `Feature2.Bit10` = 1)
- `SP_Ext.ST` (only if `Feature2.Bit10` = 1)

The following signal status is formed by:

Signal status	Used status
DQ_Out.ST	Formed by the signal status PV_Out.ST. The last DQ_Out.St signal status is also included for flow dosing and the DQ_Tare.ST signal status is included for scale dosing. The worst signal status is applied. The signal status is reset with reset of the dosing quantity.
DQ_SP.ST	With an external setpoint DQ_Ext.ST (for DQ_HiLim.ST or DQ_LoLim.ST limit), otherwise good status
SP.ST	With an external setpoint SP_Ext.ST (coarse flow) / SP2_Ext.ST (fine flow) (with SP_HiLim.ST / SP_LoLim.ST or SP2_HiLim.ST / SP2_LoLim.ST limit), otherwise good status

Considering bad quality of automatic commands or external values

If the Feature2 bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position in the "Automatic" mode:

- StartAut.ST
- CancelAut.ST
- PauseAut.ST
- ContAut.ST
- SP_Ext.ST

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Alarm delays with two time values per limit pair

This block includes the standard function alarm delay for Two time values per limit pair (Page 197).

Display and operator input area for process values and setpoints

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
4	Setting switch or button mode (Page 166)
5	Specifying the dosing type (Page 145)
6	Resetting the dosing quantity when dosing starts (Page 161)
7	Activating calculation of the flow rate for dosing by scale (Page 143)
8	Use an internal or external setpoint for the absolute fine dosing quantity (Page 152)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Stopping dosing at a flow alarm (Page 136)
12	Automatic post dosing for underdosing in automatic mode (Page 142)
13	Creep rate is always detected in the dosing quantity (Page 167)
15	Flow setpoints in percent (Page 145)
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)
18	Activating error state for external process control error CSF (Page 150)
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)
22	Update acknowledgment and error status of the message call (Page 159)
23	Specifying the influence of the signal status on the dosing process (Page 146)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
27	Enable configuration of the dribbling quantity (Page 158)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

Comments on table:

Feature Bit 4:

- Button mode (`Feature` Bit 4 = 0): The automatic commands in automatic mode are pulse signals, in other words `StartAut`, `CancelAut`, `PauseAut`, `ContAut` can be reset to 0 after switchover to the selected status. In "manual" and "local" modes, however, the automatic commands are static signals, the block state is tracked in the absence of automatic commands.
- Switch mode (`Feature` Bit 4 = 1): The conditions are selected via the inputs `StartAut` (0: Cancel, 1: Start) and `PauseAut` (0: Continue, 1: Pause) with static signals.

Feature Bit 23:

Note

When there is a bad signal status and `Feature` Bit 23 = 1 , the flow is no longer recorded in the dosing quantity in any state. The configure creep rate alarm is output.

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
5	Evaluation of the signal status of the interlock signals (Page 141)
8	Forcing operating modes in the "Local" mode (Page 183)
10	Considering bad quality of automatic commands or external values (Page 185)
24	With acknowledge overdosage (Page 178)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	1 = Operator can switch to "local mode"
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can start dosing
5	1 = Operator can pause dosing
6	1 = Operator can continue dosing
7	1 = Operator can cancel dosing
8	1 = Operator can reset interlock and flow alarms
9	1 = Operator can reset dosing quantity
10	1 = Operator can acknowledge overdosage and underdosage

Bit	Function
11	1 = Operator can select external setpoints
12	1 = Operator can select internal setpoints
13	1 = Operator can enter dosing setpoint
14	1 = Operator can enter dosing setpoint factor for fine dosing
15	1 = Operator can enter flow setpoint for coarse flow
16	1 = Operator can enter flow setpoint factor for fine flow
17	1 = Operator can activate external/internal bumpless switchover
18	1 = Operator can enable simulation function
19	1 = Operator can activate the Release for maintenance function
20	Not used
21	1 = Operator can enter dribbling quantity
22	1 = Operator can enter maximum dribbling quantity
23	1 = Operator can activate/deactivate automatic generation of dribbling quantity
24	1 = Operator can enter weighting factor for generation of dribbling quantity
25	1 = Operator can change the simulation value SimPV
26	1 = Operator can change the simulation value SimDQ
27 - 31	Not used

The block has the following operator permissions for the `OS1Perm` parameter:

Bit	Function
0	1 = Operator can change the limit for the high alarm for `PV_OUT` (coarse flow)
1	1 = Operator can change the limit for the low alarm for `PV_OUT` (coarse flow)
2	1 = Operator can change the limit for the hysteresis `PV_OUT` (coarse flow)
3	1 = Operator can change the limit for the high alarm for `PV_OUT` (fine flow)
4	1 = Operator can change the limit for the low alarm for `PV_OUT` (fine flow)
5	1 = Operator can change the limit for the hysteresis `PV_OUT` (fine flow)
6	1 = Operator can change the operation high limit of the dosing setpoint
7	1 = Operator can change the operation low limit of the dosing setpoint
8	1 = Operator can change operation high limit of the dosing setpoint factor for fine dosing
9	1 = Operator can change operation low limit of the dosing setpoint factor for fine dosing
10	1 = Operator can change the tolerance value for overdosage
11	1 = Operator can change the tolerance value for overdosage
12	1 = Operator can change operation high limit of the flow setpoint for coarse flow
13	1 = Operator can change operation low limit of the flow setpoint for coarse flow
14	1 = Operator can change operation high limit of the flow setpoint for fine flow
15	1 = Operator can change operation low limit of the flow setpoint for fine flow
16	1 = Operator can change the dribbling time
17	1 = Operator can change the post dosing
18	1 = Operator can change the limit for the high alarm for `PV_OUT` (creep flow)
19	1 = Operator can change the limit for the hysteresis for `PV_OUT` (creep flow)
20	1 = Operator can activate / deactivate messages via `DQ_AH_MsgEn`
21	1 = Operator can activate / deactivate messages via `DQ_AL_MsgEn`

Bit	Function
22	1 = Operator can activate / deactivate messages via `PV_AH_MsgEn`
23	1 = Operator can activate / deactivate messages via `PV_AL_MsgEn`
24	1 = Operator can activate / deactivate messages via `PV_AH2_MsgEn`
25	1 = Operator can activate / deactivate messages via `PV_AL2_MsgEn`
26	1 = Operator can activate / deactivate messages via `CR_AH_MsgEn`
27 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of DoseL (Page 991)

DoseL messaging (Page 1014)

DoseL I/Os (Page 1017)

DoseL modes (Page 995)

DoseL block diagram (Page 1031)

EventTs functions (Page 1634)

6.1.4 DoseL error handling

Error handling of DoseL

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
11	Configuration error `TI` ≤ 0

12	Configuration error:
	`PV_AH_Lim` < `PV_AL_Lim`
	`PV_AH2_Lim` ≤ `PV_AL2_Lim`
	`DQ_HiLim.Value` ≤ `DQ_LoLim.Value`
	`DQ2_HiLim.Value` ≤ `DQ2_LoLim.Value`
	`SP_HiLim.Value` ≤ `SP_LoLim.Value`
	`SP2_HiLim.Value` ≤ `SP2_LoLim.Value`

30	`PV` is not a valid number
31	`PV_Out` is not a valid number
32	`DQ_Out` is not a valid number
41	The value for the `LocalSetting` I/O is not within the approved limit of 0, 1 or 3.
42	`LocalSetting` = 0 and `LocalLi` = 1
48	`SP_LiOp` = 1 and `SP_IntLi` = 1 and `SP_ExtLi` = 1
51	`AutModLi` = 1 and `ManModLi` = 1 and `Feature` Bit Setting switch or button mode (Page 166) = 0 or two I/Os are set at the same time: • in "local mode" `StartLocal` (pos. edge), `CancelLocal`, `PauseLocal`, `ContLocal`when forcing states: `StartForce`, `CancelForce`, `PauseForce`, `ContForce`in "automatic mode" with pushbutton operation: `StartAut` (pos. edge), `CancelAut`, `PauseAut`, `ContAut` • in "manual mode": `StartMan`, `CancelMan`, `PauseMan`, `ContMan`

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

Invalid input signal

These errors can be output by the block, see the section Error handling (Page 119).

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signal	Control reaction with Feature2 bit 3 =1
Local: `Localsetting` = 1 or `Localsetting` = 3	`StartLocal` = 1 (pos. edge) and `CancelLocal` = 1	Dosing canceled
	`ContLocal` = 1 and `PauseLocal` = 1	Dosing paused
Local: `Localsetting` = 1 or `Localsetting` = 3 and forcing	`StartForce` = 1 and `CancelForce` = 1	Dosing canceled
	`ContForce` = 1 and `PauseForce` = 1	Dosing paused
Forcing and no "local mode"	`StartForce` = 1 and `CancelForce` = 1	Dosing canceled
	`ContForce` = 1 and `PauseForce` = 1	Dosing paused
"Automatic mode" and no forcing	Pushbutton operation (`Feature` bit 4 = 0): `StartAut` =1 (pos. edge) and `CancelAut` =1	Dosing canceled
	Pushbutton operation (Feature bit 4 = 0): `ContAut` = 1 and `PauseAut` = 1	Dosing paused
	Pushbutton operation (Feature bit 4 = 0): `StartAut` = 1 (pos. edge) and `PauseAut` = 1	Dosing paused
"Manual mode" and no forcing	`StartMan` = 1 and `CancelMan` = 1	Dosing canceled
	`ContMan` = 1 and `PauseMan` = 1	Dosing paused

See also

Description of DoseL (Page 991)

DoseL modes (Page 995)

DoseL functions (Page 997)

DoseL messaging (Page 1014)

DoseL I/Os (Page 1017)

DoseL block diagram (Page 1031)

6.1.5 DoseL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages
- Instance-specific messages

Process control fault

The following control system fault messages can be output:

Message instance	Message identifier	Message class	Event
`MsgEvId2`	SIG 1	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a control system fault is triggered (`MsgEvId02`, SIG 1).

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	Alarm - high	$$BlockComment$$ Overdosage
	SIG 2	Alarm - low	$$BlockComment$$ Underdosage
	SIG 3	Alarm - high	$$BlockComment$$ PV - High alarm limit for coarse flow violated
	SIG4	Alarm - low	$$BlockComment$$ PV - Low alarm limit for coarse flow violated
	SIG 5	Alarm - high	$$BlockComment$$ PV - High alarm limit for coarse flow violated
	SIG 6	Alarm - low	$$BlockComment$$ PV - Low alarm limit for fine flow violated
	SIG 7	Alarm - high	$$BlockComment$$ PV – Creep flow too great
	SIG 8	Process message - with acknowledgment	$$BlockComment$$ Dosing canceled

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You can use up to three instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 2	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 3	AS process control message - fault	$$BlockComment$$ External message 2
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 3

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`DQ_OUT`
5	`PV_OUT`
6	`ExtVal06`
7	`ExtVa107`
8	`ExtVa108`
9	Reserved
10	Reserved

The associated values 6 ... 8 are allocated to the parameters `ExtVa106` ... `ExtVa108` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

Associated values for message instance `MsgEvId2`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`ExtVa204`
5	`ExtVa205`
6	`ExtVa206`
7	`ExtVa207`
8	`ExtVa208`
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters `ExtVa204` ... `ExtVa208` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of DoseL (Page 991)

DoseL functions (Page 997)

DoseL I/Os (Page 1017)

DoseL modes (Page 995)

DoseL error handling (Page 1012)

DoseL block diagram (Page 1031)

Time stamp (Page 201)

6.1.6 DoseL I/Os

I/Os of DoseL

Input parameters

Parameter	Description	Type	Default
AutModLi*	1= "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 0)	BOOL	0
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Bypassing interlock in "local mode" and in "simulation"	BOOL	0
CancelAut*	1 = Select Cancel in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CancelForce	1 = Force cancel	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CancelLocal	1 = Select Cancel in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CancelMan*	1 = Select Cancel in "manual mode"	BOOL	0
ContAut*	1 = Select Continue in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ContForce	1 = Force continue	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ContLocal	1 = Select Continue in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ContMan*	1 = Select Continue in "manual mode"	BOOL	0
CR_A_DC*	Creep flow: Delay time for incoming alarms [s]	REAL	3.0

Parameter	Description	Type	Default
CR_A_DG*	Creep flow: Delay time for outgoing alarms [s]	REAL	3.0
CR_AH_En	Creep flow: 1 = Enable high alarm	BOOL	1
CR_AH_Lim	Creep flow: Limit high alarm	REAL	0.0
CR_AH_MsgEn	Creep flow: 1 = Enable high alarm message	BOOL	1
CR_Hyst*	Creep flow: Hysteresis for alarm limit	REAL	0.0
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CtrlChnST	Signal status of output channel Ctrl Should be connected to an output channel block.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Ctrl2ChnST	Signal status of output channel Ctrl2 Should be connected to an output channel block.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
DCF	Dribbling correction [%]	REAL	25.0
DosCancelMsgEn	1 = Cancel dosing message	BOOL	1
DQ_A_DC*	Overdosage/underdosage: Delay time for incoming alarms [s]	REAL	0.0
DQ_A_DG*	Overdosage/underdosage: Delay time for outgoing alarms [s]	REAL	0.0
DQ_AH_En	Overdosage: 1 = Enable high alarm	BOOL	1
DQ_AH_MsgEn	Overdosage: 1 = Enable high alarm message	BOOL	1
DQ_AH_Tol	Overdosage: Limit high alarm (relative to dosing setpoint)	REAL	0.0
DQ_AL_En	Underdosage: 1 = Enable low alarm	BOOL	1
DQ_AL_MsgEn	Underdosage: 1 = Enable low alarm message	BOOL	1
DQ_AL_Tol*	Underdosage: Limit low alarm (relative to dosing setpoint)	REAL	0.0
DQ_Ext	Dosing quantity: External setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DQ_HiLim	Dosing quantity: High setpoint limitation	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
DQ_Int*	Dosing quantity: Internal setpoint	REAL	0.0

Parameter	Description	Type	Default
DQ_LoLim	Dosing quantity: Low setpoint limitation	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DQ_OpScale	Dosing quantity: Limit for scale in bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
DQ_Unit	Unit of measure for dosing quantity	INT	1088
DQ2_Ext	Dosing quantity: External setpoint factor for fine dosing	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DQ2_HiLim*	Dosing quantity: High limitation of setpoint factor for fine dosing	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
DQ2_Int*	Dosing quantity: Internal setpoint factor for fine dosing	REAL	0.0
DQ2_LoLim*	Dosing quantity: Low limitation of setpoint factor for fine dosing	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DribbIn	Dribbling quantity	REAL	0.0
DribbMax	Maximum value of automatic determination of dribbling quantity	REAL	100.0
DribbCor	1 = Automatic determination of dribbling quantity	BOOL	0
EN	1 = Called block will be processed	BOOL	1
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
ExtVa204	Associated value 4 for messages (MsgEvID2)	ANY	
ExtVa205	Associated value 5 for messages (MsgEvID2)	ANY	
ExtVa206	Associated value 6 for messages (MsgEvID2)	ANY	
ExtVa207	Associated value 7 for messages (MsgEvID2)	ANY	
ExtVa208	Associated value 8 for messages (MsgEvID2)	ANY	
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Feature	I/O for additional functions (Page 997)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Feature2	I/O for additional functions (Page 997)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Gain	Gain factor	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
Intlock	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is active	BOOL	1
LocalLi	1 = Activate "local mode" via plant signals	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalOp*	1 = "Local mode" via operator	BOOL	0

Parameter	Description	Type	Default
`LocalSetting`	Properties for the Local mode (Page 79)	INT	0
`ManModLi*`	1 = "Manual mode" via interconnection or SFC (controlled by `ModLiOp` = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ManModOp*`	1 = "Manual mode" via OS operator (controlled by `ModLiOp` = 0)	BOOL	1
`MeterType`	Dosing using scales: 0=Up 1=Down	BOOL	0
`ModLiOp`	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`MS_RelOp*`	1= Release for maintenance via OS operator	BOOL	0
`MsgEvId1`	Message number (assigned automatically)	DWORD	16#00000000
`MsgEvId2`	Message number (assigned automatically)	DWORD	16#00000000
`MsgLock`	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`NumSample`	Number of values for average calculation	INT	0
`U_AckLi`	1 = Acknowledgement of overdosage or underdosage via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`U_AckOp*`	1 = Acknowledgement of overdosage or underdosage via operator	BOOL	0
`Occupied`	1 = In use by a batch	BOOL	0
`OosLi`	1 = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`OosOp*`	1 = "Out of service", via OS operator	BOOL	0
`OpSt_In`	Input parameter for local operator authorization, connected with the `Out` output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
`OS_Perm`	I/O for operator permissions (Page 997)	STRUCT • Bit 0: BOOL • Bit 25: BOOL • Bitt 26: BOOL • Bit 31: BOOL	- • 1 • 1 • 1 • 1

Parameter	Description	Type	Default
OS1Perm	I/O for operator permissions (Page 997)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
P_DoseTime*	Duration of post dosing [s]	REAL	0.0
PauseAut*	1 = Select Pause in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PauseForce	1 = Force pause	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PauseLocal	1 = Select Pause in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PauseMan*	1 = Select Pause in "manual mode"	BOOL	0
Perm_En	1 = Activation enable (enable, Permit parameter) is active	BOOL	1
Permit	1 = Enable for opening / closing from neutral position 0 = Valve activation not enabled on OS	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Prot_En	1 = Protective interlock (protection, Protect parameter) is active	BOOL	1
Protect	0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block 1 = Protective interlocking not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
PulseWidth*	Pulse width of control signal [s]	REAL	3.0
PV	Process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_A_DC*	Delay time for incoming PV alarms [s] when using flow dosing (coarse dosing phase)	REAL	3.0
PV_A_DG*	Delay time for outgoing PV alarms [s] when using flow dosing (coarse dosing phase)	REAL	3.0
PV_A2_DC*	Delay time for incoming PV alarms [s] when using flow dosing (fine dosing phase)	REAL	3.0
PV_A2_DG*	Delay time for outgoing PV alarms [s] when using flow dosing (fine dosing phase)	REAL	3.0
PV_AH_En	1 = Enable PV alarm (high) limit when using flow dosing (coarse dosing phase)	BOOL	1

Parameter	Description	Type	Default
PV_AH_Lim	Limit PV alarm (high) when using flow dosing (coarse dosing phase)	REAL	100.0
PV_AH_MsgEn	1 = Enable PV alarm (high) message when using flow dosing (coarse dosing phase)	BOOL	1
PV_AH2_En	1 = Enable PV alarm (high) when using flow dosing (fine dosing phase)	BOOL	1
PV_AH2_Lim	Limit PV alarm (high) when using flow dosing (fine dosing phase)	REAL	100.0
PV_AH2_MsgEn	1 = Enable PV alarm (high) message when using flow dosing (fine dosing phase)	BOOL	1
PV_AL_En	1 = Enable PV alarm limit (low) when using flow dosing (coarse dosing phase)	BOOL	1
PV_AL_Lim	Limit PV alarm (low) when using flow dosing (coarse dosing phase)	REAL	0.0
PV_AL_MsgEn	1 = Enable PV alarm (low) message when using flow dosing (coarse dosing phase)	BOOL	1
PV_AL2_En	1 = Enable PV alarm (low) when using flow dosing (fine dosing phase)	BOOL	1
PV_AL2_Lim	Limit PV alarm (low) when using flow dosing (fine dosing phase)	REAL	0.0
PV_AL2_MsgEn	1 = Enable PV alarm (low) message when using flow dosing (fine dosing phase)	BOOL	1
PV_Hyst*	Hysteresis for PV alarm limits when using flow dosing (coarse dosing phase)	REAL	1.0
PV_Hyst2	Hysteresis for PV alarm limits when using flow dosing (fine dosing phase)	REAL	1.0
PV_OpScale	Limit for scale in PV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
PV_Unit	Unit of measure for process value	INT	1349
RelaxTime*	Duration of dribbling phase [s]	REAL	3.0
RstDQ_Li	1 = Reset dosing quantity via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstDQ_Op*	1 = Reset dosing quantity via operator	BOOL	0
RstLi*	1 = Reset interlock/flow monitoring via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset interlock/flow monitoring via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3

Parameter	Description	Type	Default
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
SimDQ*	Dosing quantity: Value used in scale mode for SimOn = 1	REAL	0.0
SimDQ_Li	Linkable simulation value DQ	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimPV*	Process value used in flow mode for SimOn = 1	REAL	0.0
SimPV_Li	Process value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_Ext	External flow setpoint - coarse dosing	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtLi	1 = Select external setpoints via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOp*	1 = Select external setpoints via operator	BOOL	0
SP_HiLim	High limit for flow setpoint - coarse dosing	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP_Int*	Internal flow setpoint - coarse dosing	REAL	0.0
SP_IntLi	1 = Select internal setpoints via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_IntOp*	1 = Select internal setpoints via operator	BOOL	0

Parameter	Description	Type	Default
SP_LiOp	Select setpoint source (internal/external): 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_LoLim	Low limit for flow setpoint - coarse dosing	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_TrkExt	1 = Bumpless switchover from external to internal setpoint active	BOOL	1
SP2_Ext	External flow setpoint - fine dosing	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP2_HiLim	High limit for flow setpoint - fine dosing	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
SP2_Int*	Internal flow setpoint - fine dosing	REAL	0.0
SP2_LoLim	Low limit for flow setpoint - fine dosing	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
StandStill	1 = Scales at a standstill, dosing device infeedback	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
StartAut*	1 = Select Start in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StartForce	1 = Forced start	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StartLocal	1 = Select Start in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StartMan*	1 = Select Start in "manual mode"	BOOL	0
StepNo	Batch step number	DWORD	16#00000000
TI	Integral time [s]	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
U_AckLi	1 = Acknowledgment of underdosage via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
U_AckOp*	1 = Acknowledgment of underdosage via operator	BOOL	0

Parameter	Description	Type	Default
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UA2unit	Unit of measure for analog auxiliary value 1	INT	0
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80
CR_AH_Act	Creep flow: 1 = High alarm active. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl	Control output for coarse dosing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl2	Control output for fine dosing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DosEnd	1 = End of dosing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DosOff	1 = Dosing off	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
DosOn	1 = Dosing on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DosPause	1 = Dosing pause	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DosRelax	1 = Dosing dribbling	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DosStart	1 = Dosing started	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DQ_AH_Act	1 = Overdosing You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DQ_AL_Act	1 = Underdosing You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DQ_ExHiAct	Dosing quantity: 1 = High limit for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DQ_ExLoAct	Dosing quantity: 1 = Low limit for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DQ_ExtOut	Dosing quantity: External setpoint, corresponds to input parameter DQ_Ext	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DQ_Out	Dosing quantity	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DQ_SP	Dosing quantity setpoint used by block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DQ_SP_Tol	Setpoint for determining tolerance limits	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
DQ_Tare	Tare memory when dosing using scales	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DQ1_SP	Coarse dosing quantity setpoint used by block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DQ2_ExHiAct	Dosing quantity: 1 = High limit for external setpoint factor for fine dosing has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DQ2_ExLoAct	Dosing quantity: 1 = Low limit for external setpoint factor for fine dosing has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DQ2_ExtOut	Dosing quantity: External setpoint factor for fine dosing [%], corresponds to input parameter DQ2_Ext	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DQ2_SP	Fine dosing quantity setpoint used by block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
DribbOut	Dribbling quantity	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see DoseL error handling (Page 1012)	INT	-1
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalAct	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock (Intlock, Permit or Protect) is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgAckn2	Message acknowledgement status 2 (output ACK_STATE of second ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgErr2	Alarm error 2 (output ERROR of second ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
MsgStat2	Message status 2 (output STATUS of second ALARM_8P)	WORD	16#0000
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Displays of OS_Perm	DWORD	16#FFFFFFFF
OS1PermLog	Display of OS1Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS1PermOut	Display of OS1Perm	DWORD	16#FFFFFFFF
P_Ctrl	1 = Pulse output for coarse dosing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Ctrl2	1 = Pulse output for fine dosing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AH_Act	1 = PV high alarm enabled (coarse dosing phase). You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
PV_AH2_Act	1 = PV high alarm enabled (fine dosing phase). You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AL_Act	1 = PV low alarm enabled (coarse dosing phase). You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_AL2_Act	1 = PV low alarm enabled (fine dosing phase). You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	Output for process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RdyToReset	1 = Ready for reset via RstLi input or commands in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP	Active flow setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExHiAct	Flow for coarse dosing: 1 = High limit for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExLoAct	Flow for coarse dosing: 1 = Low limit for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtAct	1 = External setpoint is active 0 = Internal setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOut	Flow for coarse dosing: External setpoint, corresponds to input parameter SP_Ext	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
SP1	Coarse dosing flow setpoint used by block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP2	Fine dosing flow setpoint used by block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP2_ExHiAct	Flow for fine dosing: 1 = High limit for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP2_ExLoAct	Flow for fine dosing: 1 = Low limit for external setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP2_ExtOut	Flow for fine dosing: External setpoint, corresponds to input parameter SP2_Ext	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 991)	DWORD	16#00000000
Status2	Status word 2 (Page 991)	DWORD	16#00000000
Status3	Status word 3 (Page 991)	DWORD	16#00000000
Status4	Status word 4 (Page 991)	DWORD	16#00000000

See also

DoseL messaging (Page 1014)

DoseL modes (Page 995)

DoseL block diagram (Page 1031)

Error handling (Page 119)

6.1.7 DoseL block diagram

DoseL block diagram

A block diagram is not provided for this block.

See also

Description of DoseL (Page 991)

DoseL modes (Page 995)

DoseL functions (Page 997)

DoseL error handling (Page 1012)

DoseL messaging (Page 1014)

DoseL I/Os (Page 1017)

6.1.8 Operator control and monitoring

6.1.8.1 DoseL views

Views of the DoseL block

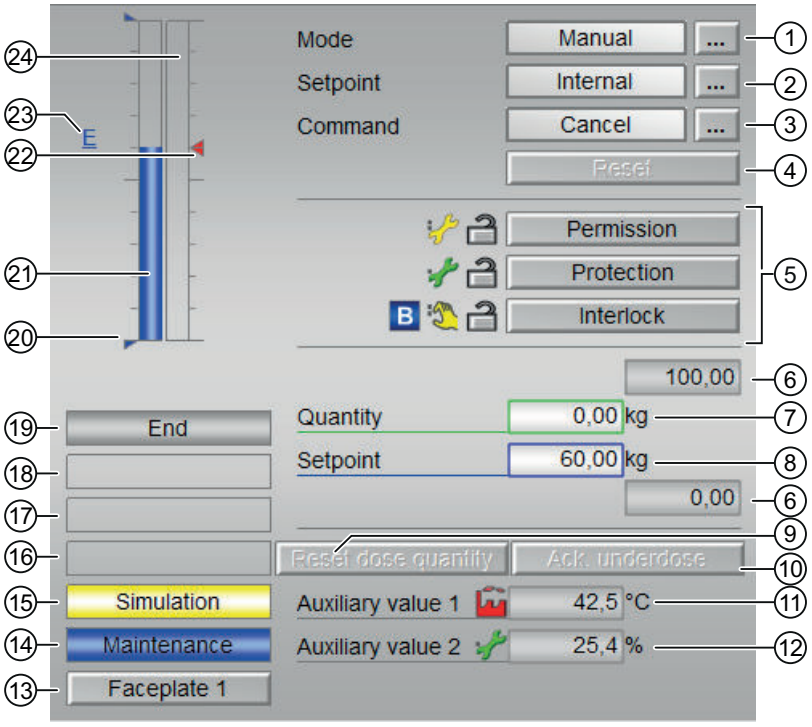
The block DoseL provides the following views:

- DoseL standard view (Page 1033)
- Alarm view (Page 296)
- DoseL limit view (Page 1037)
- Trend view (Page 299)
- DoseL parameter view (Page 1039)
- DoseL flow setpoint view (Page 1041)
- DoseL quantity setpoint view (Page 1044)
- DoseL preview (Page 1046)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for DoseL (Page 1048)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

6.1.8.2 DoseL standard view

DoseL standard view



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual mode (Page 75)
- Automatic mode (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

Refer to chapter Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Displaying and changing the setpoint

The following states can be shown and changed here:

- "Internal"
- "External"

You can find additional information on this in section Manual and automatic mode for motors, valves and dosers (Page 75).

(3) Displaying and changing: Start, continue, pause and cancel

This area shows you the default operating state for the doser. The following states can be shown and changed here:

- "Start"
- "Continue"
- "Pause"
- "Cancel"

You can find additional information on this in section Switching operating states and operating modes (Page 251).

(4) Resetting the block

Click "Reset" for interlocks or errors. You can find additional information on this in section Resetting the block in case of interlocks or errors (Page 43).

(5) Operating range for the interlock functions of the block

You can use this button to control the interlock functions of the block. You can find additional information on this in section Interlocks (Page 99).

- Bypass information (see Forming the group status for interlock information (Page 104)):

(6) High and low scale range for the setpoint

This area is already set and cannot be changed.

(7) Displaying and changing the quantity

You can find additional information on this in section Changing values (Page 253).

(8) Displaying and changing the setpoint

You can find additional information on this in section Changing values (Page 253).

(9) Button: Reset dose quantity

The dosing quantity can only be reset in the "End" or "Ack Dos End“ state.

With resetting dose quantity also the display "Ack Dos End" is resetted to "End".

(10) Button: Acknowledge overdosing and underdosing

Acknowledgment of overdosing and underdosing can only be made in the "Off" state.

Additional with `Feature2` bit 24 = 1:

- an acknowledgement of overdosing is possible.
- generally, overdosing or underdosing has to be acknowledged.

(11) and (12) Display of auxiliary values

You can use this area to display two auxiliary values that have been configured in the Engineering System (ES). You can find additional information on this in section Opening additional faceplates (Page 203).

(13) Button for switching to the standard view of any faceplate

Use this button for the standard view of a block configured in the Engineering System (ES). The visibility of this button depends on the configuration in the engineering system (ES).

You can find additional information on this in section Opening additional faceplates (Page 203).

(14) and (15) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"
- "Maintenance"

You can find additional information on this in section Simulating signals (Page 58).

(16) Display area for block states

This area provides additional information on the operating state of the block:

- "Invalid signal"
- "Changeover error"
- "Flow"

(17) Display area for block states

This area provides additional information on the operating state of the block:

- "Forced start"
- "Force continue"
- "Force pause"
- "Forced stop"
- "Request 0/1": A reset to "automatic mode" is expected.

(18) Display area for block states

This area provides additional information on the operating state of the block:

- "Underdosed"
- "Overdosed"

(19) Display area for block states

This area provides additional information on the operating state of the block:

- "Coarse dosing"
- "Fine dosing"
- "Relax"
- "Pause"
- "Off"
- "End"
  If an underdosage or overdosage is identified, the status display shows the state "Ack Dos End" in the standard view.
- "Taring"

(20) Limit display for the setpoint

These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the ES.

(21) Bar graph for the setpoint

This area shows the current setpoint in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(22) Limit display

These colored triangles indicate the configured limits in the respective bar graph:

- Red: Alarm
- Yellow: Warning
- Blue: Tolerance

(23) Display of external setpoint

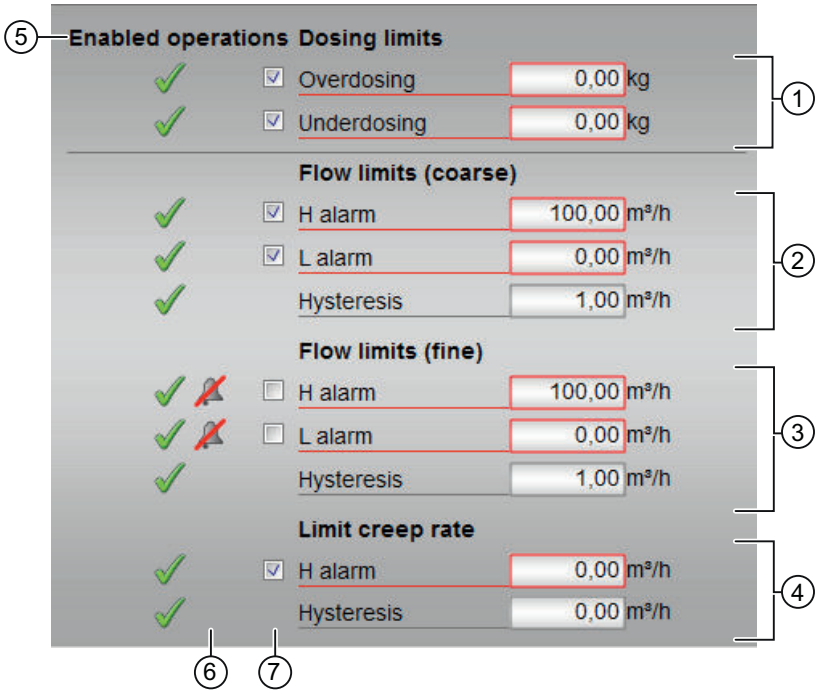
This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

(24) Bar graph for the quantity

This area shows you the current quantity in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

6.1.8.3 DoseL limit view

Limit view of DoseL



Note

In scale mode with `Feature` Bit 7 = 0, the displays **(2)**, **(3)** and **(4)** are hidden.

(1) Displaying and changing the dosing limits

You can change the dosing limits in this area:

- "Overdosing"
- "Underdosing"

You can find additional information on this in the section Changing values (Page 253).

(2) Displaying and changing the flow limits (coarse)

You can change the flow limits (coarse) in this area:

- "H alarm"
- "L alarm"
- "Hysteresis"

You can find additional information on this in the section Changing values (Page 253).

(3) Displaying and changing the flow limits (fine)

You can change the flow limits (fine) in this area:

- "H alarm"
- "L alarm"
- "Hysteresis"

You can find additional information on this in the section Changing values (Page 253).

(4) Displaying and changing the limit for the creep flow

You can change the limit for the creep rate in this area:

- "H alarm"
- "Hysteresis"

You can find additional information on this in the section Changing values (Page 253).

(5) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

(6) "Message suppression/delay"

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

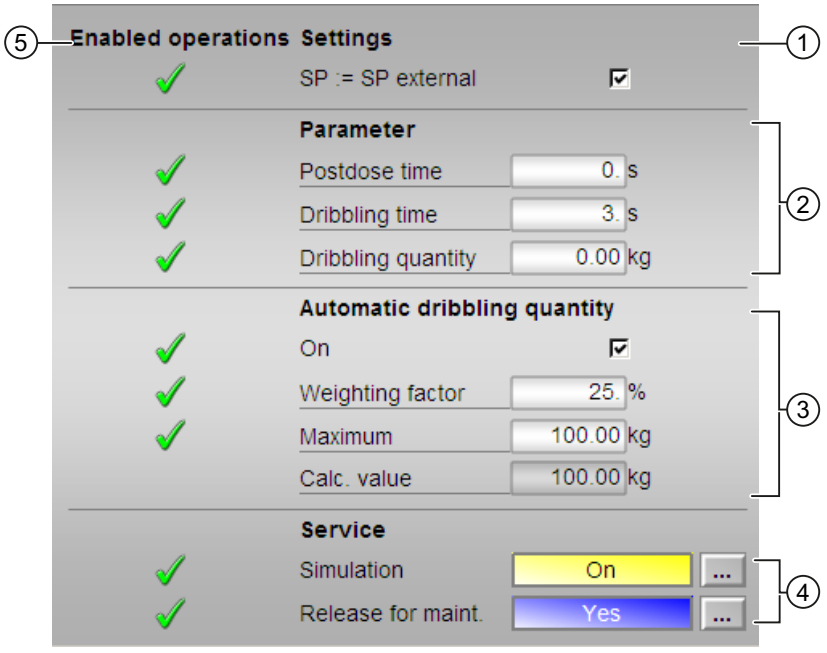
Alarm delays are also displayed in this position; for more on this see section Area of application of the alarm delays (Page 195).

(7) Suppress messages

You can enable / disable messages by setting the check mark.

6.1.8.4 DoseL parameter view

Parameter view of DoseL



(1) Settings

You can select the following functions in this area:

- "SP:=SP external": ☑ Bumpless switchover of setpoint from external to internal. The internal setpoint is tracked to the external one.

(2) Parameter

In this area, you change parameters and therefore influence the doser. Refer also to the Changing values (Page 253) section for more on this.

You can influence the following parameters:

- "Relax time"
- "Dribble time"
- "Dribble value"

(3) Automatic dribbling quantity

In this area, you change the automatic dribbling quantity parameters, which affects the doser. Refer also to the Changing values (Page 253) section for more on this.

You can change the parameters when the "On" check box is selected ☑.

You can influence the following parameters:

- "Weighting factor"
- "Maximum"
- "Calc. value"

(4) Service

You can select the following functions in this area:

- "Simulation"
- "Release for maintenance"

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Simulating signals (Page 58)
- Release for maintenance (Page 64)

(5) Enabled operations

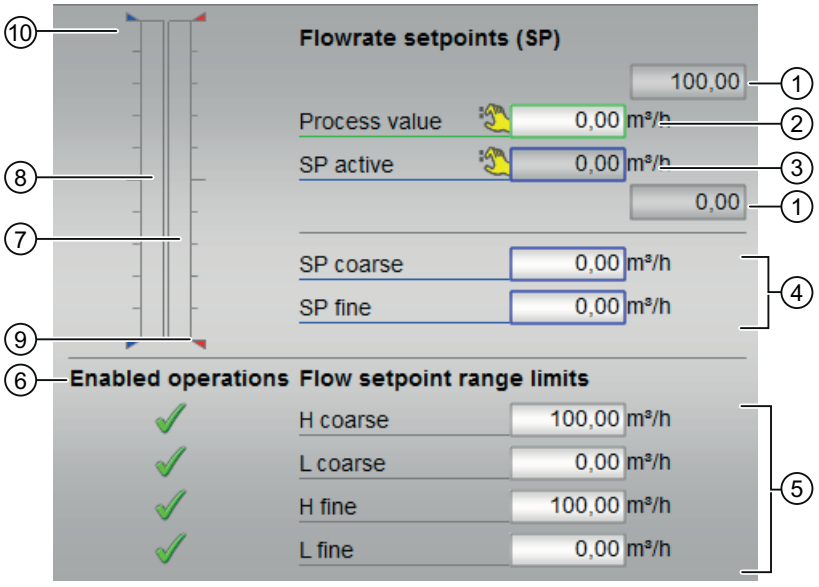
This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

6.1.8.5 DoseL flow setpoint view

Flow setpoint view for DoseL



Note

1.) This view cannot be selected in scale mode with `Feature` Bit 7 = 0 .

2.) In scale mode with `Feature` Bit 7 = 1 the displays **(3)**, **(4)**, **(5)**, **(6)** and **(8)** are hidden.

3.) If `Feature` Bit 15 = 1, the display of this setpoint view changes. For further information, refer to the description below.

(1) High and low scale range for the process value

These values provide information on the display range for the bar graph **(7)** of the process value. The scale range is defined in the engineering system.

(2) Display of the process value including signal status

This area shows the current process value with the corresponding signal status.

(3) Displaying and changing the SP active value

You can find additional information on this in the Changing values (Page 253) section.

(4) Additional setpoint parameters

You can change the following setpoint parameters in this area:

- "SP coarse"
- "SP fine"

You can find additional information on this in the Changing values (Page 253) section.

(5) Displaying and changing the limits

You can change the limits in this area:

- "H coarse"
- "L coarse"
- "H fine"
- "L fine"

You can find additional information on this in the Changing values (Page 253) section.

(6) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

(7) Bar graph for the process value

This area shows the current process value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(8) Bar graph for the SP active setpoint

This area shows the current setpoint "SP active" in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

Note

If `Feature` Bit 15 = 1, there are no **(8)** or **(10)** displays. For further information, refer to the description below.

(9) Limit display

These triangles show limits for the flow limits. The flow limits are set in the limit value view.

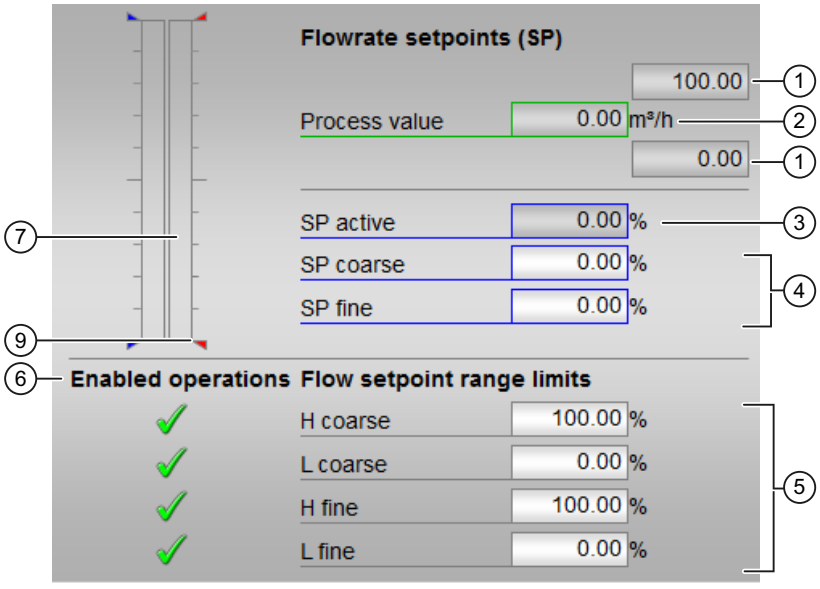
(10) Limit display for the setpoint

These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the Engineering System (ES).

Flow setpoint view for `Feature` Bit 15 = 1

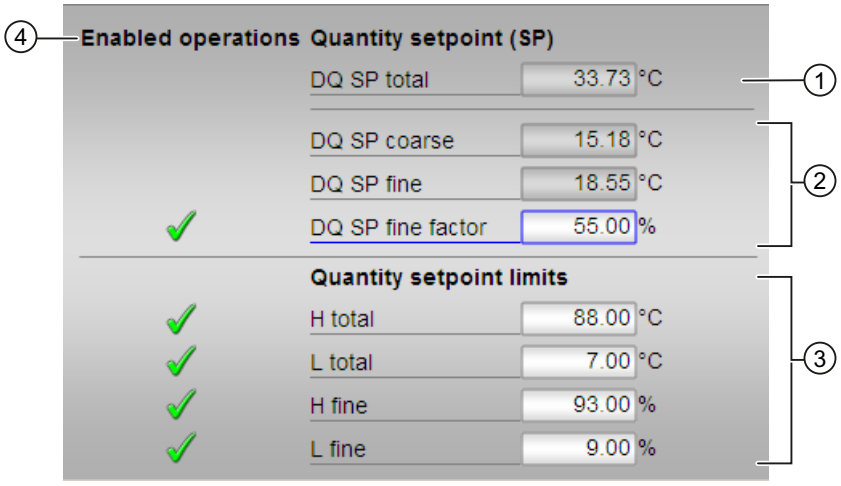
If `Feature` Bit 15 = 1 , the setpoint view changes in the following ways:

- Displays **(8)** and **(10)** are omitted.
- The "SP active" value **(3)** is displayed above the area **(4)**.
- The values in the **(3)**, **(4)** and **(5)** areas contain the unit "%".



6.1.8.6 DoseL quantity setpoint view

Quantity setpoint view for DoseL



(1) Displaying and changing the DQ SP total setpoint

You can change the "DQ SP total" setpoint in this area.

Additional information is available in the section Changing values (Page 253).

(2) Displaying and changing additional setpoints

You can change the limits in this area:

- "DQ SP coarse"
- "DQ SP fine"
- "DQ SP fine factor"

Additional information is available in the section Changing values (Page 253).

Note

Special note for `Feature` Bit 8 = 1

If you set this `Feature` Bit with 1, DQ SP fine is no longer displayed in the faceplate. In addition, the display of the unit for DQ SP fine factor is omitted. The input is now performed in an absolute manner.

(3) Displaying and changing the limits

You can change the limits in this area:

- "H total"
- "L total"

- "H fine"
- "L fine"

Additional information is available in the section Changing values (Page 253).

Note

Special note for `Feature` Bit 8 = 1

If you set this `Feature` Bit to 1, the display of the unit for H fine and L fine is switched to the unit configured at `DQ_Unit` . The input is now performed in an absolute manner.

(4) Enabled operations

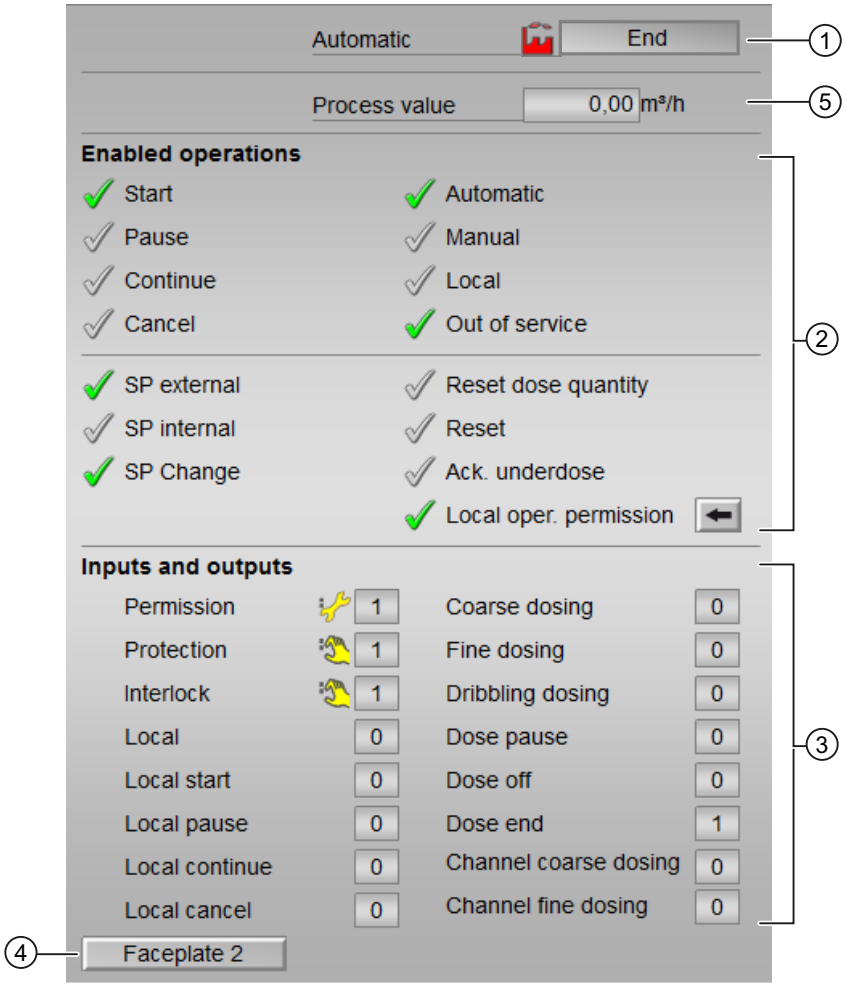
This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: The OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

6.1.8.7 DoseL preview

Preview of DoseL



(1) Automatic preview

This area shows you the block status after it has switched from "manual" to "automatic" mode.

If the block is in "automatic mode", the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `StartAut`
- `CancelAut`
- `PauseAut`
- `ContAut`
- `SP_Ext`

(2) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

The following enabled operations are shown here:

- "Start": You can start the dosing procedure
- "Pause": You can pause the dosing procedure
- "Continue": You can continue the dosing procedure after a pause or an abort.
- "Cancel": You can abort the dosing procedure
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Local": You can switch to "local mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "SP external": You can use the external setpoint
- "SP internal": You can use the internal setpoint
- "Change SP": You can change the setpoint
- "Reset dose quantity": You can reset the dosing quantity
- "Reset": You can reset the block after interlocks or errors
- "Acknowledge underdosing": You can acknowledge underdosing
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248).

(3) Displaying current control signals

This area shows the most important parameters for this block with the current selection:

- "Permission":
  - 0 = No OS release for energizing motor
  - 1 = Enable for "starting"/"stopping" from the neutral position
- "Protection":
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state

- "Interlock":
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Local start": 1 = Block is started in "local mode"
- "Local pause": 1= Dosing paused in "local mode"
- "Local continue": 1 = Dosing is continued in "local mode"
- "Local cancel": 1= Dosing is canceled in "local mode"
- "Coarse dosing": 1 = Coarse dosing is performed
- "Fine dosing": 1 = Fine dosing is performed
- "Relax phase": 1 = Dosing procedure is in the relax phase
- "Dose pause": 1 = Dosing pause
- "Dose off": 1 = No dosing taking place
- "Dose end": 1 = Dosing is stopped
- "Channel Coarse Dosing": Signal from the output channel block for "Coarse dosing"
- "Channel Fine Dosing": Signal from the output channel block for "Fine dosing"

(4) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(5) Process value

This area displays the real process value (`PV`).

6.1.8.8 Block icon for DoseL

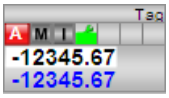
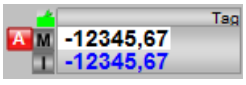
Block symbols for DoseL

A variety of block icons are available with the following functions:

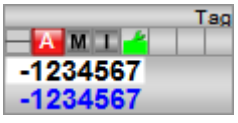
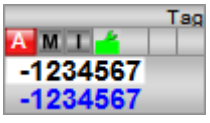
- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults

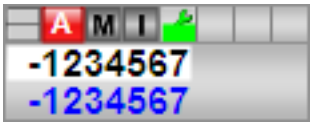
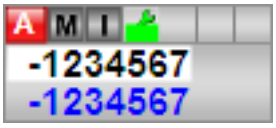
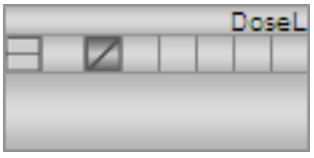
- Operating modes
- Internal and external setpoint specification
- Signal status, release for maintenance
- Track, force, and bypass
- Interlocks
- Memo display
- Process value (black, with and without decimal places)
- Setpoint (blue, with and without decimal places)

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	

Icons	Selection of the block icon in CFC	Special features
	5	
	6	
	7	
	8	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

Motor and valve blocks

7

7.1 Comparison of large & small blocks

7.1.1 MotL compared to MotS

Comparison of the MotL and MotS blocks

The following tables are intended to help you decide which block to use.

Memory and runtime savings of the small block compared to the large block

You save the following resources for each instance:

- Memory space: ~ 40%
- Runtime: ~ 25%

Block operating modes

	MotL	MotS
Local mode (Page 79)	X	X
Manual mode (Page 75)	X	X
Automatic mode (Page 75)	X	X
Out of service (Page 71)	X	X

Functions of the blocks

	MotL	MotS
Opening additional faceplates (Page 203)	X	X
Operator control permissions (Page 248)	X	X
Limit monitoring of an additional analog value (Page 91)	X	
Limit monitoring with hysteresis (Page 97)	X	
Suppressing messages using the MsgLock parameter (Page 201)	X	X
Interlocks (Page 99)	X	X

	MotL	MotS
Motor protection function (Page 99)	X	X
Disabling interlocks (Page 103)	X	X
Rapid stop for motors (Page 106)	X	
Resetting the block in case of interlocks or errors (Page 43)	X	X
Outputting group errors (Page 122)	X	X
Outputting a signal for start readiness (Page 53)	X	X
Restart lock after switching off the motor (Page 1064)	X	
Forming the group status for interlock information (Page 104)	X	X
Forming and outputting the signal status for technologic blocks (Page 108)	X	X
Forcing operating modes (Page 41)	X	
Monitoring the feedbacks (Page 97)	X	X
Release for maintenance (Page 64)	X	X
Specifying warning times for control functions at motors and valves (Page 51)	X	
Simulating signals (Page 58)	X	X
Selecting a unit of measure (Page 207)	X	
Neutral position for motors, valves and controllers (Page 48)	X	X
Output signal as a static signal or pulse signal (Page 51)	X	X
Generating instance-specific messages (Page 200)	X	X
Displaying auxiliary values (Page 207)	X	
Time stamp	X	
SIMATIC BATCH functionality (Page 67)	X	
Labeling of buttons and text (Page 205)	X	X
Displaying current control signals from output channel block	X	X
Display active monitoring time	X	X

	MotL	MotS
Control priority in the event of an invalid input command (Page 174)	X	X
Separate monitoring time for stopping the motor (Page 168)	X	X

Configurable functions using the `Feature` parameter

Bit number	`Feature` bit function	MotL	MotS
0	Set startup characteristics (Page 137)	X	X
1	Reaction to the out of service mode (Page 176)	X	X
2	Resetting the commands for changing the mode (Page 160)	X	X
3	Enabling resetting of commands for the control settings (Page 160)	X	X
4	Setting switch or button mode (Page 166)	X	
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)	X	X
10	Exiting local mode (Page 176)	X	X
11	Activating the run time of feedback signals (Page 152)	X	X
13	Separate monitoring time for stopping the motor (Page 168)	X	
14	Enabling rapid stop via faceplate (Page 167)	X	
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)	X	
19	Reset even with locked state (Page 164)	X	X
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)	X	
22	Update acknowledgment and error status of the message call (Page 159)	X	
24	Enabling local operator authorization (Page 157)	X	X
25	Suppression of all messages (Page 173)	X	X
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)	X	
27	Interlock display with LocalSetting 2 or 4 (Page 177)	X	X

Bit number	`Feature` bit function	MotL	MotS
28	Disabling operating points (Page 144)	X	
29	Signaling limit violation (Page 169)	X	
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)	X	X
31	Activating reset of protection / error in manual mode (Page 164)	X	X

Configurable functions using the `Feature2` parameter

Bit number	`Feature2` bit function	MotL	MotS
3	Control priority in the event of an invalid input command (Page 174)	X	X
5	Evaluation of the signal status of the interlock signals (Page 141)	X	X

7.1.2 VlvL compared to VlvS

Comparison of the VlvL and VlvS blocks

The following tables are intended to help you decide which block to use.

Memory and runtime savings of the small block compared to the large block

You save the following resources for each instance:

- Memory space: ~ 25%
- Runtime: ~ 20%

Modes

	VlvL	VlvS
Local mode (Page 79)	X	X
Manual mode (Page 75)	X	X
Automatic mode (Page 75)	X	X
Out of service (Page 71)	X	X

Functions

	VlvL	VlvS
Opening additional faceplates (Page 203)	X	X
Operator control permissions (Page 248)	X	X
Interlocks (Page 99)	X	X
Disabling interlocks (Page 103)	X	X
Resetting the block in case of interlocks or errors (Page 43)	X	X
Outputting group errors (Page 122)	X	X
Outputting a signal for start readiness (Page 53)	X	X
Forming the group status for interlock information (Page 104)	X	X
Forming and outputting the signal status for technologic blocks (Page 108)	X	X
Forcing operating modes (Page 41)	X	
Monitoring the feedbacks (Page 97)	X	X
Disabling feedback for valves (Page 99)	X	
Suppressing messages using the MsgLock parameter (Page 201)	X	X
Release for maintenance (Page 64)	X	X
Specifying warning times for control functions at motors and valves (Page 51)	X	
Simulating signals (Page 58)	X	X
Selecting a unit of measure (Page 207)	X	
Neutral position for motors, valves and controllers (Page 48)	X	X
Generating instance-specific messages (Page 200)	X	X
Displaying auxiliary values (Page 207)	X	
SIMATIC BATCH functionality (Page 67)	X	X
Output signal as a static signal or pulse signal (Page 51)	X	
Time stamp (Page 1634)	X	

	VlvL	VlvS
Labeling of buttons and text (Page 205)	X	X
Displaying current control signals from output channel block	X	X

Configurable functions using the `Feature` parameter

Bit number	`Feature` bit function	VlvL	VlvS
0	Set startup characteristics (Page 137)	X	X
1	Reaction to the out of service mode (Page 176)	X	X
2	Resetting the commands for changing the mode (Page 160)	X	X
3	Enabling resetting of commands for the control settings (Page 160)	X	X
4	Setting switch or button mode (Page 166)	X	
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)	X	X
10	Exiting local mode (Page 176)	X	X
11	Activating the run time of feedback signals (Page 152)	X	X
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)	X	
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)	X	
22	Update acknowledgment and error status of the message call (Page 159)	X	
24	Enabling local operator authorization (Page 157)	X	X
25	Suppression of all messages (Page 173)	X	X
27	Interlock display with LocalSetting 2 or 4 (Page 177)	X	X
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)	X	X
31	Activating reset of protection / error in manual mode (Page 164)	X	X

Configurable functions using the `Feature2` parameter

Bit number	`Feature2` bit function	VlvL	VlvS
3	Control priority in the event of an invalid input command (Page 174)	X	X
5	Evaluation of the signal status of the interlock signals (Page 141)	X	X

7.1.3 ShrdResL compared to ShrdResS

Comparison of the ShrdResL and ShrdResS blocks

The following tables are intended to help you decide which block to use.

Memory and runtime savings of the small block compared to the large block

You save the following resources for each instance:

- Memory space: ~ 50%
- Runtime: ~ 25%

Parameters for channel

	ShrdResL	ShrdResS
Control commands for channel	12	6
External setpoints for channel	8	2
Additional strings for channel	2	-

Functions

	ShrdResL	ShrdResS
Opening additional faceplates (Page 203)	x	x
Operator control permissions (Page 248)	x	
Readiness signal (Page 1250)	x	x
Channel management (Page 1272)		x
Channel management with parameterized priority parameter and strategy mode (Page 1250)	x	
Allocate/enable channel (Page 1250)	x	x
Enable/disable channel (Page 1250)	x	x
Channel prioritization (Page 1250)	x	x
Cascading (Page 1272)		x
SIMATIC BATCH functionality (Page 67)	x	x

Configurable functions using the `Feature` parameter

Bit number	`Feature` bit function	ShrdResL	ShrdResS
0	Set startup characteristics (Page 137)	x	x

7.2 MotL - Motor (Large)

7.2.1 Description of MotL

Object name (type + number) and family

Type + number: FB 1850

Family: Drives

Area of application for MotL

The block is used for the following applications:

- Control of motors with one control signal

Note

This block is also available as a small block. A comparison of the MotL and MotS blocks is available in the section: MotL compared to MotS (Page 1051)

How it works

The block is used to control motors. Various inputs are available for controlling the motor.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section MotL I/Os (Page 1074).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`

Status bit	Parameter
4	`OosLi.Value`
5	0 = `ManAct.Value` 1 = `AutoAct.Value`
6	`LocalAct.Value`
7	0: Open padlock in the block icon 1: Closed padlock in the block icon
8	`Start.Value`
9	`Stop.Value`
10	`MonDynStopErr.Value`
11	`MonStaErr.Value`
12	`MonDynErr.Value`
13	`BypProt`
14	Invalid signal status
15	Mode switchover error
16	1 = `Intlock` is active
17	1 = `Permit` is active
18	1 = `Protect` is active
19	Trip.Value
20	Display „Forced start“
21	Display „Forced stop“
22	Not used
23	"Interlock" button is enabled
24	Reset request in automatic preview
25	`WarnAct.Value` or `Idle Time` is active
26	Bypass information from previous function block
27	Automatic preview for "starting"
28	Automatic preview for "stopping"
29	External error generated by FaultExt or external control system fault CSF with set Feature bit 18 Activating error state for external process control error CSF (Page 150)
30	"Permission" button is enabled
31	"Protection" button is enabled

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	`AV_AH_Act.Value`
2	`AV_WH_Act.Value`
3	`AV_TH_Act.Value`
4	`AV_TL_Act.Value`
5	`AV_WL_Act.Value`

Status bit	Parameter
6	AV_AL_Act.Value
7	AV_AH_En
8	AV_WH_En
9	AV_TH_En
10	AV_TL_En
11	AV_WL_En
12	AV_AL_En
13	AV_AH_MsgEn
14	AV_WH_MsgEn
15	AV_TH_MsgEn
16	AV_TL_MsgEn
17	AV_WL_MsgEn
18	AV_AL_MsgEn
19	1 = No impact of input signals on "local mode" with LocalSetting = 2 and LocalSetting = 4
20	Motor is stopped
21	Motor is stopping
22	Motor is starting
23	Motor is running
24	Error in motor
25	1 = Input parameter StartChnST is interconnected
26 - 29	Not used
30	Display for interlocks in block icon
31	MS_RelOp

Status word allocation for Status3 parameter

Status bit	Parameter
0	Delay of the AV_AH_Lim message
1	Delay of the AV_WH_Lim message
2	Delay of the AV_TH_Lim message
3	Delay of the AV_TL_Lim message
4	Delay of the AV_WL_Lim message
5	Delay of the AV_AL_Lim message
6	Collection of message delays
7 - 10	Not used
11	Hidden bypass signal in Permit
12	Hidden bypass signal in interlock
13	Hidden bypass signal in Protect
14	Feature2 bit 2: Separate bypass signal
15 - 17	Current monitoring time is visible
18	SimLiOp.Value

Status bit	Parameter
19	1 = Enable for rapid stop (`Feature` Bit Enabling rapid stop via faceplate (Page 167))
20 - 22	Not used
23	Command for rapid stop
24	Output command for starting the motor
25	Output command for stopping the motor
26	Show automatic preview in the standard view
27	Not used
28	`GrpErr.Value`
29	`RdyToStart.Value`
30	Auxiliary value 1 visible
31	Auxiliary value 2 visible

Status word allocation for `Status4` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via `EventTsIn`
1	Effective signal 2 of the message block connected via `EventTsIn`
2	Effective signal 3 of the message block connected via `EventTsIn`
3	Effective signal 4 of the message block connected via `EventTsIn`
4	Effective signal 5 of the message block connected via `EventTsIn`
5	Effective signal 6 of the message block connected via `EventTsIn`
6	Effective signal 7 of the message block connected via `EventTsIn`
7	Effective signal 8 of the message block connected via `EventTsIn`
8	`AV` not connected
9	Motor protection display (`Trip.Status` ≠ 16#FF)
10	1 = Input parameter `FbkRun` is connected
11 - 15	Not used
16 - 23	Effective signal 9..16 of the message block connected via `EventTsIn`
24 - 30	Not used
31	Separate monitoring of shutdown of the motor (Feature bit 13)

Status word allocation for `Status5` parameter

Status bit	Parameter
0 - 15	Effective signal 1...16 of the message block connected via `EventTs2In`
16 - 31	Not used

See also

MotL functions (Page 1064)

MotL messaging (Page 1072)

MotL block diagram (Page 1082)

MotL error handling (Page 1071)

MotL modes (Page 1063)

7.2.2 MotL modes

MotL operating modes

The block can be operated using the following modes:

- Local mode (Page 79)
- Automatic mode (Page 75)
- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) section.

Motor actions you can control in local mode:

- "Start" (`StartLocal` = 1)
- "Stop" (`StopLocal` = 1)

A motor operated in "local mode" is controlled either by "local" signals or by the feedback signals (e.g the `FbkStart` = 1 input parameter). Configuration takes place via the input parameter `LocalSetting`.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) section.

Motor actions you can control in auto mode:

- "Start" (`StartAut` = 1)
- "Stop" (`StopAut` = 1)

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

Motor actions you can control in manual mode:

- "Start" (`StartMan` = 1)
- "Stop" (`StopMan` = 1)

"Out of service"

You can find general information about the "Out of service" mode in the Out of service (Page 71) section.

See also

MotL block diagram (Page 1082)

MotL I/Os (Page 1074)

MotL messaging (Page 1072)

MotL error handling (Page 1071)

MotL functions (Page 1064)

Description of MotL (Page 1059)

7.2.3 MotL functions

Functions of MotL

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	1 = Operator can switch to "local mode"
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can stop the motor
5	1 = Operator can start the motor
6	Not used
7	1 = Operator can reset the motor

Bit	Function
8	1 = Operator can define or change the monitoring time for startup
9	1 = Operator can define the monitoring time for the status
10	1 = Operator can enable the monitoring time function (Bit 8 - 9)
11	1 = Operator can activate the Simulation function
12	1 = Operator can activate the Release for maintenance function
13	1 = Operator can change the high limit (AV) for the alarm
14	1 = Operator can change the high limit (AV) for the warning
15	1 = Operator can change the high limit (AV) for the tolerance
16	1 = Operator can change the limit (AV) for hysteresis
17	1 = Operator can lower the limit (AV) for the alarm
18	1 = Operator can lower the limit (AV) for the warning
19	1 = Operator can lower the limit (AV) for the tolerance
20	1 = Operator can activate / deactivate messages via AV_AH_MsgEn
21	1 = Operator can activate / deactivate messages via AV_WH_MsgEn
22	1 = Operator can activate / deactivate messages via AV_TH_MsgEn
23	1 = Operator can activate / deactivate messages via AV_TL_MsgEn
24	1 = Operator can activate / deactivate messages via AV_WL_MsgEn
25	1 = Operator can activate / deactivate messages via AV_AL_MsgEn
26	1 = Operator can change the simulation value SimAV
27- 29	Not used
30	1 = Operator can define the monitoring time for stopping
31	Not used

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

Limit monitoring of an additional analog value

This block provides the standard function Limit monitoring of an additional analog value (Page 91).

Limit monitoring with hysteresis

This block provides the standard function Limit monitoring with hysteresis (Page 97).

Suppressing messages using the MsgLock parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Interlocks

This block provides the following interlocks:

- Activation enable
- Interlock without reset ("Interlock")
- Interlock with reset ("Protection")

Refer to the section Interlocks (Page 99) as well as Influence of the signal status on the interlock (Page 103).

Motor protection function

This block provides the standard function Motor protection function (Page 99).

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Rapid stop for motors

This block provides the standard function Rapid stop for motors (Page 106).

Resetting the block in case of interlocks or errors

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See MotL error handling (Page 1071)

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `Trip`
- `MonDynErr`
- `MonStaErr`
- `FaultExt`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

Restart lock after switching off the motor

After switching off or stopping the motor, it can only be restarted after the time set with the `IdleTime` input parameter.

When the "Stop" command is given, the motor goes immediately into "Stop" mode, and `IdleTime`starts after the feedback (`FbkRun` = 0) is given. The motor cannot be started again until the `IdleTime` has expired.

The `IdleTime` parameter can be set independently of the `MonTiDynamic` parameter.

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `LocalLi.ST`
- `StartLocal.ST`
- `StopLocal.ST`
- `Trip.ST`
- `FbkRunOut.ST`
- `AV_Out.ST`
- `StartChn.ST`
- `StartAut.ST` (only if `Feature2.Bit10` = 1)
- `StopAut.ST` (only if `Feature2.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position (motor stop) in the "Automatic" mode:

- `StartAut.ST`
- `StopAut.ST`

Forcing operating modes

This block provides the standard function Forcing operating modes (Page 41).

The following states can be enforced:

- "Start" (`StartForce`)
- "Stop" (`StopForce`)

Feedback monitoring

This block provides the standard function Monitoring the feedbacks (Page 97).

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Specify warning times for control functions

This block provides the standard function Specifying warning times for control functions at motors and valves (Page 51).

You can generate warning signals when, for example, motors are started. Warning signals can be generated in the following modes:

- Manual mode (Page 75) (`WarnTiMan` input parameter)
- Automatic mode (Page 75) (`WarnTiAut` input parameter)

You specify the warning times in seconds using the input parameters `WarnTiMan` and `WarnTiAut`. If, for example, a motor is started, then this is displayed at the output parameter with `WarnAct` = 1. The motor then starts after the set warning time has expired and `WarnAct` then goes back to 0.

A corresponding warning is not output if the warning times (`WarnTiMan` or `WarnTiAut`) are specified with a smaller value than the `SampleTime` parameter.

Simulating signals

This block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Additional value (`SimAV`, `SimAV_Li`)

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Neutral position

This block provides the standard function Neutral position for motors, valves and controllers (Page 48).

Output signal as a pulse signal or static signal

This block provides the standard function Output signal as a static signal or pulse signal (Page 51).

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
4	Setting switch or button mode (Page 166)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
14	Enabling rapid stop via faceplate (Page 167)
13	Separate monitoring time for stopping the motor (Page 168)
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)
18	Activating error state for external process control error CSF (Page 150)
19	Reset even with locked state (Page 164)
20	Disable calculation of impulse control in LocalSetting 2 and 4 (Page 185)
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
27	Interlock display with LocalSetting 2 or 4 (Page 177)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

In pushbutton operation (Bit 4 = 0) the automatic commands in "automatic" mode are latching, in other words `StartAut`, `StopAut` can be reset to 0 after changing the control. In "manual" and "local" modes, however, the automatic commands are not saved and in the absence of automatic commands the automatic control is tracked.

In switching mode (Bit 4 = 1), the control is selected with the static signal `StartAut`. If `StartAut` input is not set the motor is stopped. Control via `StopAut` is not needed. If the

"Activate command reset for control" function (Bit 3 = 1) is also activated, the `StartAut` input is reset to the neutral position after evaluation in the block.

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
4	Setting switch or button mode for local commands (Page 180)
5	Evaluation of the signal status of the interlock signals (Page 141)
8	Forcing operating modes in the "Local" mode (Page 183)
10	Considering bad quality of automatic commands or external values (Page 185)

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` or `EventTs2In` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205).

Instance-specific text can be configured for the following parameters:

- `StartMan`
- `StopMan`
- `RapidStp`

See also

MotL messaging (Page 1072)

MotL I/Os (Page 1074)

MotL block diagram (Page 1082)

MotL modes (Page 1063)

Description of MotL (Page 1059)

7.2.4 MotL error handling

Error handling of MotL

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Control system fault (CSF)
- Invalid input signals

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.
41	The value for the `LocalSetting` I/O is not within the approved limit of 0 to 4.
42	`LocalSetting` = 0 or `LocalSetting` = 3 or `LocalSetting` = 4 and `LocalLi` = 1
51	`StartLocal` = 1 and `StopLocal` = 1 `StartAut` = 1 and `StopAut` = 1 `AutModLi` = 1 and `ManModLi` = 1 `StartForce` = 1 and `StopForce` = 1
52	`LocalAct` = 1 and `LocalSetting` = 2 or 4 and `SimOn` = 1

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

Invalid input signals

This error can be output by the block, see the section Error handling (Page 119).

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signal	Control reaction with Feature2 bit 3 =1
Local: `Localsetting` = 1 or `Localsetting` = 3	Pushbutton operation for local mode (`Feature2` bit 4 = 0): `StartLocal` = 1 and `StopLocal` = 1	Motor is stopped
Local: `Localsetting` = 1 or `Localsetting` = 3 and forcing	`StartForce` = 1 and `StopForce` = 1	
Forcing and no "local mode"	`StartForce` = 1 and `StopForce` = 1	
"Automatic mode" and no forcing	Pushbutton operation (`Feature` bit 4 = 0): `StartAut` = 1 and `StopAut` = 1	
"Manual mode" and no forcing	`StopMan` = 1 and `StartMan` = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

MotL block diagram (Page 1082)

MotL I/Os (Page 1074)

MotL messaging (Page 1072)

MotL functions (Page 1064)

MotL modes (Page 1063)

Description of MotL (Page 1059)

7.2.5 MotL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	AS process control message - fault	$$BlockComment$$ Motor feedback error
	SIG 2	AS process control message - fault	$$BlockComment$$ Motor protection triggered
	SIG 3	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter CSF. If it changes to CSF = 1, a process control fault is triggered (MsgEvId1, SIG 3).

Instance-specific messages

You can use up to three instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 4	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 5	AS process control message - fault	$$BlockComment$$ External message 2
	SIG 6	AS process control message - fault	$$BlockComment$$ External message 3

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance **MsgEvId1**

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	ExtVa104
5	ExtVa105
6	ExtVa106
7	ExtVa107
8	ExtVa108

Associated value	Block parameters
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters `ExtVa104` ... `ExtVa108` , and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of MotL (Page 1059)

MotL functions (Page 1064)

MotL I/Os (Page 1074)

MotL block diagram (Page 1082)

MotL error handling (Page 1071)

MotL modes (Page 1063)

7.2.6 MotL I/Os

I/Os of MotL

Input parameters

Parameter	Description	Type	Default
`AutModLi*`	1 = "Automatic mode" via interconnection or SFC (controlled by `ModLiOp` = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`AutModOp*`	1 = "Automatic mode" via operator (controlled by `ModLiOp` = 0)	BOOL	0
`AV`	Input additional analog value, to be connected to `AV_Tech` of the AV block	ANY	
`AV_AH_Lim`	Limit high alarm	REAL	95.0
`AV_AL_Lim`	Limit low alarm	REAL	5.0
`AV_Hyst`	Hysteresis for alarm, warning and tolerance limits	REAL	1.0
`AV_TH_Lim`	Limit high tolerance	REAL	85.0
`AV_TL_Lim`	Limit low tolerance	REAL	15.0
`AV_WH_Lim`	Limit high warning	REAL	90.0
`AV_WL_Lim`	Limit low warning	REAL	10.0
`BatchEn`	1 = Enable allocation	BOOL	0
`BatchID`	Batch ID	DWORD	16#00000000
`BatchName`	Batch name	S7-String	

Parameter	Description	Type	Default
BypProt	1 = Bypassing interlock in "local mode" and in "simulation"	BOOL	0
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EN	1 = Called block will be processed	BOOL	1
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs,Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
EventTs2In	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTs2In input parameter serves to interconnect the EventTsOut output parameter of the EventTs,Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	

Parameter	Description	Type	Default
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkRun	Feedback for starting is present: 1 = Start 0 = Stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Feature	I/O for additional functions (Page 1064) Separate monitoring time for stopping the motor	STRUCT • Bit 0: BOOL • ... • Bit 13: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0 • 0 • 0
Feature2	I/O for additional functions (Page 1064)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
IdleTime*	Wait time for restart in [s]	REAL	5.0
Intlock	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is active	BOOL	1
LocalLi	1 = Activate "local mode" via plant signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalOp*	1 = "Local mode" via operator	BOOL	0
LocalSetting	Properties for the Local mode (Page 79)	INT	0
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
ModLiOp	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Monitor	1 = Feedback monitoring	BOOL	1
MonTiDynamic*	Monitoring time for feedback errors or feedback start error after successful operation in [s]	REAL	3.0

Parameter	Description	Type	Default
MonTiDyStop*	Monitoring time for feedback stop errors after successful operation in [s]	REAL	3.0
MonTiStatic*	Monitoring time for feedback errors without operation in [s]	REAL	3.0
MS_RelOp*	1= Release for maintenance via OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = In use by a batch	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 1064) 1 = Operator can set or change the monitoring time for "Control: Start" 1 = Operator can set or change the monitoring time for "Control: Stop"	STRUCT • Bit 0: BOOL • Bit 8: BOOL • Bit 20: BOOL • Bit 30: BOOL	- • 1 • 1 • 1 • 1
Permit	1 = OS activation enable for motor 0 = No OS release for energizing motor	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Perm_En	1 = Activation enable (enable, Permit parameter) is active	BOOL	1
Protect	0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block 1 = Protective interlocking not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Prot_En	1 = Protective interlock (protection, Protect parameter) is active	BOOL	1
PulseWidth*	Pulse width of control signal [s]	REAL	3.0
RapidStp*	Rapid stop for the motor: 0 = Motor On 1 = Motor Off	BOOL	0

Parameter	Description	Type	Default
RstLi*	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SimAV*	Additional value used for SimOn = 1	REAL	0.0
SimAV_Li	Additional analog value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0 • 16#80
SimLiOp	Activation/deactivation of the simulation 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
StartAut*	1 = Starting the motor in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StartChnST	Signal status of output channel Start Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
StartForce	1 = Force motor start	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StartLocal	1 = Starting the motor in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StartMan*	1 = Starting the motor in "manual mode"	BOOL	0
StepNo	Batch step number	DWORD	16#00000000
StopAut*	1 = Stopping the motor in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
StopForce	1 = Force motor stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopLocal	1 = Stopping the motor in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopMan*	1 = Stopping the motor in "manual mode"	BOOL	0
Trip	1 = Motor is in "good" state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA2unit	Unit of measure for analog auxiliary value 2	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
WarnTiAut	Prewarning of motor start in "automatic mode" in [s]	REAL	0.0
WarnTiMan	Prewarning of motor start in "manual mode" in [s]	REAL	0.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80
AV_OpScale	Limit for scale in AV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

Parameter	Description	Type	Default
AV_Out	Output additional analog value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
AV_Unit	Unit of measure for additional analog value	INT	0
CurrMon	Current monitoring time [s]	DINT	0
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see MotL error handling (Page 1071)	INT	-1
FbkRunOut	Feedback for starting is present: 1 = Start 0 = Stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalAct	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock (Intlock, Permit, Protect) or Trip is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MonDynErr	1 = Feedback error or feedback start error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonDynStopErr	1 = Feedback stop error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonStaErr	1 = Feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0

Parameter	Description	Type	Default
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Stop	0 = Pulse signal for stopping the motor	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
P_Start	1 = Pulse signal for starting the motor	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
R_StpAct	1 = Rapid stop of the motor is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Run	1 = Motor is running	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Start	1 = Control of motor: started	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Status1	Status word 1 (Page 1059)	DWORD	16#00000000
Status2	Status word 2 (Page 1059)	DWORD	16#00000000
Status3	Status word 3 (Page 1059)	DWORD	16#00000000
Status4	Status word 4 (Page 1059)	DWORD	16#00000000

Parameter	Description	Type	Default
`Stop`	1 = Motor stopped	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`WarnAct`	1 = Prewarning for motor start active (parameters `WarnTiAut` and `WarnTiMan`)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

MotL modes (Page 1063)

MotL block diagram (Page 1082)

MotL messaging (Page 1072)

7.2.7 MotL block diagram

MotL block diagram

A block diagram is not provided for this block.

See also

MotL I/Os (Page 1074)

MotL messaging (Page 1072)

MotL error handling (Page 1071)

MotL functions (Page 1064)

MotL modes (Page 1063)

Description of MotL (Page 1059)

7.2.8 Operator control and monitoring

7.2.8.1 MotL views

Views of the MotL block

The MotL block provides the following views:

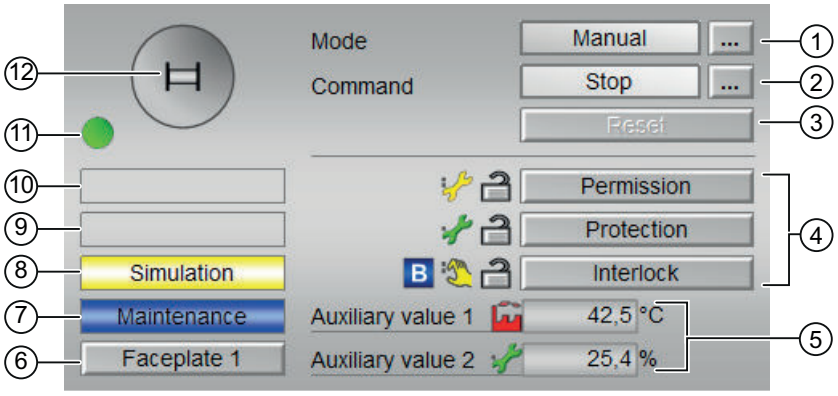
- MotL standard view (Page 1083)
- Alarm view (Page 296)

- Limit view of motors (Page 288)
- Trend view (Page 299)
- Parameter view for motors and valves (Page 280)
- MotL preview (Page 1086)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MotL (Page 1089)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.2.8.2 MotL standard view

MotLstandard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Starting and stopping the motor

This area shows you the default operating state for the motor. The following states can be shown and executed here:

- "Start"
- "Stop"
- "Rapid stop"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(3) Resetting the block

Click "Reset" for interlocks or errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(4) Operator control and display area for interlock functions of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the section Interlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

(5) Display of auxiliary values

This display is only visible when the corresponding block input is connected.

You can use this area to display two auxiliary values that have been configured in the engineering system (ES). You can find additional information on this in the section Displaying auxiliary values (Page 207).

(6) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(7) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

(8) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"
- "Delay"

You will find more detailed information on this in the sections Simulating signals (Page 58) and Display of delay times (Page 250).

(9) Display area for block states

This area provides additional information on the operating state of the block (from high to low according to priority):

- "Motor protection"
- "External error"
- "Status error"
- "Control error"
- "Invalid signal"
- "Changeover error"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97) , Error handling (Page 119) (section "Invalid input signals" and "Mode switchover error") and Motor protection function (Page 99).

(10) Display area for block states

This area provides additional information on the operating state of the block:

- "Forced start"
- "Forced stop"
- "Request 0/1": A reset to "automatic mode" is expected.

You can find additional information on this in the section Forcing operating modes (Page 41).

(11) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the motor would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

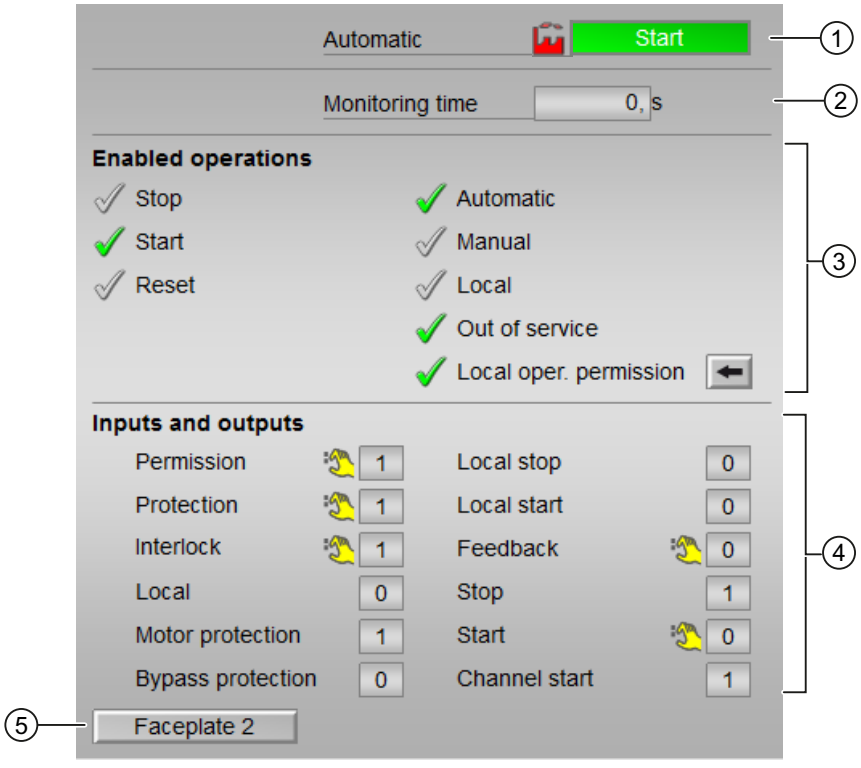
(12) Status display of the motor

The current status of the motor is graphically displayed here.

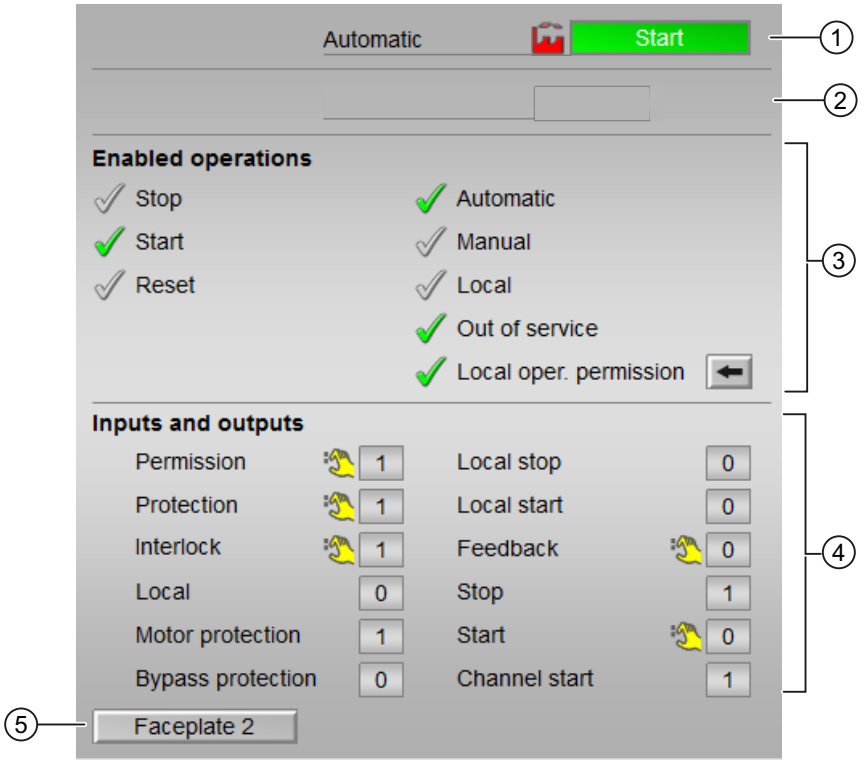
You can find more information about this in section Block icon for MotL (Page 1089)

7.2.8.3 MotL preview

Preview of MotL



Display of the current monitoring time is visible.



Display of the current monitoring time is not visible.

(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `StartAut`
- `StopAut`

(2) Monitoring time

The current "Monitoring time" is displayed in this area.

(3) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Stop": You can stop the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Start": You can start the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Reset": You can reset the motor after interlocks or errors.
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Local": You can switch to "local mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248).
- "Monitoring time": Display of the current monitoring time.

(4) Displaying current control signals

This area shows the most important parameters for this block with the current selection:

- "Permission":
  This display is only visible when the corresponding block input is connected.
  - 0 = No OS release for energizing motor
  - 1 = Enable for "starting"/"stopping" from the neutral position
- "Protection":
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state
- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state

- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Motor protection": 1 = Motor is in "good" state
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"
- "Local stop": 1 = Stopping the motor in "local mode"
- "Local start": 1 = Starting the motor in "local mode"
- "Feedback →": 1 = Motor has started and is running
- "Stop": 1 = Stop motor
- "Start": 1 = Start motor
- "Channel Start": Signal from the output channel block "Start"

(5) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

7.2.8.4 Block icon for MotL

Block symbols for MotL

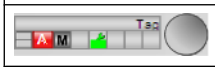
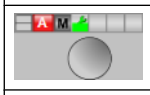
A variety of block symbols are available with the following functions:

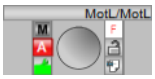
- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes
- Signal status, release for maintenance
- Displays for bypassing interlocks
- Interlocks
- Memo display
- Motor state display

The block symbols from template @TemplateAPLV8.PDL:

Symbols	Selection of the block symbol in CFC	Special features
	1	
	2	
	3	
	4	
	5	
	6	

The block symbols from template @TemplateAPLV7.PDL:

Symbols	Selection of the block symbol in CFC	Special features
	1	Block symbol in the full display
	2	
	3	
	4	
	5	
	6	
	7	
	8	

Symbols	Selection of the block symbol in CFC	Special features
	9	
	10	
	-	Block symbol in "Out of service" mode (example with type 1 block symbol)

Additional information on the block symbol and the control options in the block symbol is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Motor state display

The following motor states are shown here:

Symbol	Meaning
	Motor started (motor symbol changes)
	The motor is running
	Motor stopped (motor symbol changes)
	Motor idle
	Error at motor (monitoring error, motor protection)
	Motor out of service

7.3 MotS - Motor (Small)

7.3.1 Description of MotS

Object name (type + number)

Type + number: FB 1910

Family: Drives

Area of application for MotS

The block is used for the following applications:

- Controlling motors

Note

This block is also available as a large block. A comparison of the MotL and MotS blocks is available in the section: MotL compared to MotS (Page 1051)

How it works

The block is used to control motors. Various inputs are available for controlling the motor.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation

For a description of the individual parameters, see the section MotS I/Os (Page 1103).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`

Status bit	Parameter
4	`OosLi.Value`
5	0 = `ManAct.Value` 1 = `AutoAct.Value`
6	`LocalAct.Value`
7	`LockAct.Value`
8	`Start.Value`
9	Motor is stopped
10	Not used
11	`MonStaErr.Value`
12	`MonDynErr.Value`
13	`BypProt`
14	Invalid signal status
15	Not used
16	1 = `Intlock` is active
17 - 18	Not used
19	`Trip`
20 - 22	Not used
23	"Interlock" button is enabled
24	Reset request in automatic preview
25	Not used
26	Bypass information from previous function block
27	Automatic preview for "starting"
28	Automatic preview for "stopping"
29	External error generated by FaultExt or external control system fault CSF with set Feature bit 18 Activating error state for external process control error CSF (Page 150)
30 - 31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1 - 18	Not used
19	1 = No impact of input signals on "local mode" when `LocalSetting` = 2
20	Motor is stopped
21	Motor is stopping
22	Motor is starting
23	Motor is running
24	Error in motor
25	1 = Input parameter `StartChnST` is interconnected
26 - 29	Not used
30	Display for interlocks in block icon
31	`MS_RelOp`

Status word allocation for `Status3` parameter

Status bit	Parameter
0 - 11	Not used
12	Hidden bypass signal in interlock
13	Not used
14	`Feature2` bit 2: Separate bypass signal
15 - 23	Not used
24	Output command for "starting" the motor
25	Output command for "stopping" the motor
26	Show automatic preview in the standard view
27	Not used
28	`GrpErr.Value`
29	Enable for "starting" the motor
30	Not used
31	Not used

Status word allocation for `Status4` parameter

Status bit	Parameter
0 - 8	Not used
9	`Trip` not connected
10	Do not connect `FbkOutRun`
11 - 31	Not used

See also

MotS messaging (Page 1101)
MotS block diagram (Page 1108)
MotS functions (Page 1096)
MotS error handling (Page 1100)
MotS modes (Page 1094)

7.3.2 MotS modes

MotS operating modes

The block can be operated using the following modes:

- Local mode (Page 79)
- Automatic mode (Page 75)

- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) section.

Note

"Local mode" for the block MotS

In contrast to the "Large" blocks, it is only possible to perform settings in this block `LocalSetting` with 0 or 2. A "local operation" is accordingly possible only via the internal tracking of the feedback value.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) section.

Motor actions you can control in auto mode:

- "Start" (`StartAut` = 1)
- "Stop" (`StopAut` = 1)

Note

Information about the "Small" block

This "Small" block works with pushbutton operation. The automatic commands are therefore latching, in other words, `OpenAut`, `CloseAut` can be reset to 0 after the control is changed. In "manual" and "local" modes, however, the automatic commands are not saved and in the absence of automatic commands the automatic control is tracked.

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

Motor actions you can control in "manual mode":

- "Start" (`StartMan` = 1)
- "Stop" (`StopMan` = 1)

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of MotS (Page 1092)

MotS functions (Page 1096)

MotS error handling (Page 1100)

MotS messaging (Page 1101)

MotS I/Os (Page 1103)

MotS block diagram (Page 1108)

7.3.3 MotS functions

Functions of MotS

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can stop the motor
5	1 = Operator can start the motor
6	Not used
7	1 = Operator can reset the motor
8	1 = Operator can define the monitoring time for startup
9	Not used
10	1 = Operator can activate the monitoring time function (Bit 8)
12	1 = Operator can activate the Release for maintenance function
13 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Button labels

This block provides the standard function Labeling of buttons and text (Page 205).

Instance-specific text can be configured for the following parameters:

- `StartMan`
- `StopMan`

Interlocks

This block provides the following interlocks:

- Interlock without reset (interlock)

Refer to the section Interlocks (Page 99) as well as Influence of the signal status on the interlock (Page 103).

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Motor protection function

This block provides the standard function Motor protection function (Page 99).

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Resetting the block in case of interlocks or errors

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See MotS error handling (Page 1100).

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `Trip`
- `MonDynErr`
- `MonStaErr`
- `FaultExt`

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `LocalLi.ST`
- `Trip.ST`
- `FbkRunOut.ST`
- `StartChn.ST`
- `StartAut.ST` (only if `Feature2.Bit10` = 1)
- `StopAut.ST` (only if `Feature2.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position (motor stop) in the "Automatic" mode:

- `StartAut.ST`
- `StopAut.ST`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

Feedback monitoring

This block provides the standard function Monitoring the feedbacks (Page 97).

Simulating signals

This block provides the standard function Simulating signals (Page 58).

Neutral position

This block provides the standard function Neutral position for motors, valves and controllers (Page 48).

Output signal as a static signal

This block provides the standard function Output signal as a static signal or pulse signal (Page 51).

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
18	Activating error state for external process control error CSF (Page 150)
19	Reset even with locked state (Page 164)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
27	Interlock display with LocalSetting 2 or 4 (Page 177)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)

Bit	Function
5	Evaluation of the signal status of the interlock signals (Page 141)
10	Considering bad quality of automatic commands or external values (Page 185)

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

See also

Description of MotS (Page 1092)

MotS messaging (Page 1101)

MotS I/Os (Page 1103)

MotS block diagram (Page 1108)

MotS modes (Page 1094)

Selecting a unit of measure (Page 207)

7.3.4 MotS error handling

Error handling of MotS

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Control system fault (CSF)
- Invalid input signals

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers.

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.
41	The value for the `LocalSetting` I/O is not 0 or 2

Error number	Meaning of the error number
42	`LocalSetting` = 0 and `LocalLi` = 1
51	`StartAut` = 1 and `StopAut` = 1, `AutModLi` = 1 and `ManModLi` = 1
52	`LocalAct` = 1 and `LocalSetting` = 2 and `SimOn` = 1

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

Invalid input signals

This error can be output by the block, see the section Error handling (Page 119).

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signals	Control reaction with Feature2 bit 3 =1
"Automatic mode"	`StartAut` = 1 and `StopAut` = 1	Motor is stopping
"Manual mode"	`StopMan` = 1 and `StartMan` = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

MotS functions (Page 1096)

MotS messaging (Page 1101)

MotS I/Os (Page 1103)

Description of MotS (Page 1092)

MotS modes (Page 1094)

MotS block diagram (Page 1108)

7.3.5 MotS messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	AS process control message - fault	$$BlockComment$$ Motor feedback error
	SIG 2	AS process control message - fault	$$BlockComment$$ Motor protection triggered
	SIG 3	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter CSF. If it changes to CSF = 1, a process control fault is triggered (MsgEvId1, SIG 3).

Instance-specific messages

You have the option to use two instance-specific messages for this block.

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 4	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 5	AS process control message - fault	$$BlockComment$$ External message 2

Explanation:

$$BlockComment$$: Content of the instance-specific comment.

Associated values for message instance **MsgEvId1**

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	ExtVa104
5	ExtVa105
6 - 10	Reserved

The associated values 4 ... 5 are allocated to the parameters ExtVa104 ... ExtVa105 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of MotS (Page 1092)

MotS functions (Page 1096)

MotS error handling (Page 1100)

MotS block diagram (Page 1108)

MotS I/Os (Page 1103)

MotS modes (Page 1094)

7.3.6 MotS I/Os

I/Os of MotS

Input parameters

Parameter	Description	Type	Default
AutModLi*	1 = "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 0)	BOOL	0
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Bypassing interlock in "local mode" and in "simulation"	BOOL	0
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EN	1 = Called block will be processed	BOOL	1
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	

Parameter	Description	Type	Default
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkRun	Feedback for starting is present: 1 = Start 0 = Stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Feature	I/O for additional functions (Page 1096)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Feature2	I/O for additional functions (Page 1096)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Intlock	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is active	BOOL	1
LocalLi	1 = Activate "local mode" via plant signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalSetting	Properties for the Local mode (Page 79)	INT	0
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
ModLiOp	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Monitor	1 = Feedback monitoring	BOOL	1
MonTiDynamic*	Monitoring time for feedback errors after operation in [s]	REAL	3.0
MS_RelOp*	1= Release for maintenance via OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Occupied	1 = In use by a batch	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 1096)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RstLi*	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SimOn	1 = Simulation on	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
StartAut*	1 = Starting the motor in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StartChnST	Signal status of output channel Start Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
StartMan*	1 = Starting the motor in "manual mode"	BOOL	0
StepNo	Batch step number	DWORD	16#00000000
StopAut*	1 = Stopping the motor in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopMan*	1 = Stopping the motor in "manual mode"	BOOL	0
Trip	1 = Motor is in "good" state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
`AutAct`	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`AutoST`	1 = Worst signal status of automatic commands	BYTE	16#80
`ENO`	1 = Block algorithm completed without errors	BOOL	0
`ErrorNum`	Output of pending error number. For error numbers that can be output by this block, see MotS error handling (Page 1100)	INT	-1
`FbkRunOut`	Feedback for starting is present: 1 = Start 0 = Stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`GrpErr`	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`LocalAct`	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`LockAct`	1 = Interlock or `Trip` is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ManAct`	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
`MonDynErr`	1 = Feedback error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`MonStaErr`	1 = Feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`MS_Release`	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Run	1 = Motor is running	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Start	1 = Control of motor: started	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Status1	Status word 1 (Page 1092)	DWORD	16#00000000
Status2	Status word 2 (Page 1092)	DWORD	16#00000000
Status3	Status word 3 (Page 1092)	DWORD	16#00000000
Status4	Status word 4 (Page 1092)	DWORD	16#00000000
Stop	1 = Motor stopped	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

MotS messaging (Page 1101)

MotS block diagram (Page 1108)

MotS modes (Page 1094)

Error handling (Page 119)

7.3.7 MotS block diagram

MotS block diagram

A block diagram is not provided for this block.

See also

Description of MotS (Page 1092)

MotS functions (Page 1096)

MotS messaging (Page 1101)

MotS I/Os (Page 1103)

MotS modes (Page 1094)

MotS error handling (Page 1100)

7.3.8 Operator control and monitoring

7.3.8.1 MotS views

Views of the MotS block

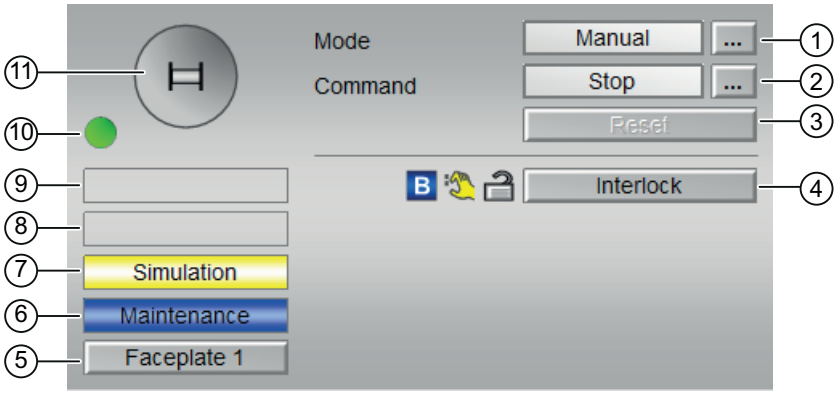
The block MotS provides the following views:

- MotS standard view (Page 1109)
- Alarm view (Page 296)
- Trend view (Page 299)
- Parameter view for motors and valves (Page 280)
- MotS preview (Page 1112)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MotS (Page 1113)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.3.8.2 MotS standard view

MotSstandard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Starting and stopping the motor

This area shows you the default operating state for the motor. The following states can be shown and executed here:

- "Start"
- "Stop"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(3) Resetting the block

Click "Reset" for errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(4) Operator control and display area for the interlock function of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock function of the block. You can find additional information on this in the section Interlocks (Page 99).

The following is displayed in addition to the button:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

(5) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(6) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

(7) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in the section Simulating signals (Page 58).

(8) Display area for block states

This area provides additional information on the operating state of the block (from high to low according to priority):

- "Motor protection"
- "External error"
- "Status error"
- "Control error"
- "Invalid signal"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97) , Error handling (Page 119) (subsection "Invalid input signals") and Motor protection function (Page 99).

(9) Display area for block states

This area provides additional information on the operating state of the block:

- "Request 0/1": A reset to "automatic mode" is expected.

(10) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the motor would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

(11) Status display of the motor

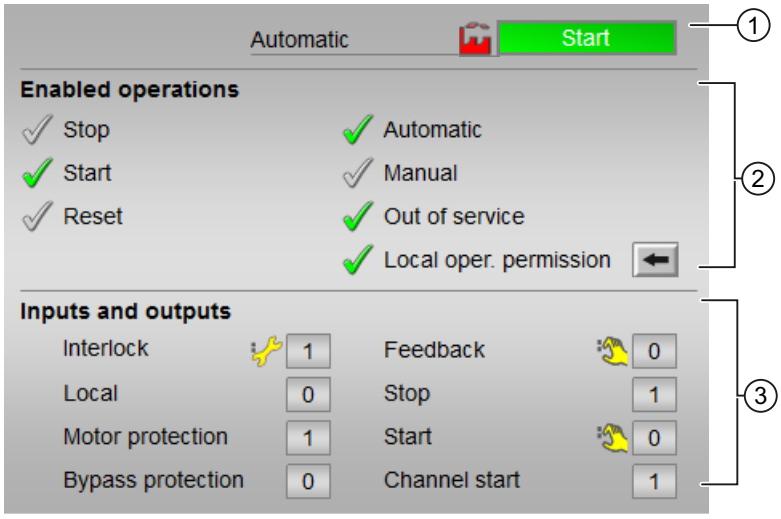
The current status of the motor is graphically displayed here:

- Green: Motor is running
- Gray: Motor idle
- Red: A fault has occurred

You can find additional information on this in the section Block icon for MotS (Page 1113).

7.3.8.3 MotS preview

Preview of MotS



(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `StartAut`
- `StopAut`

(2) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Stop": You can stop the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Start": You can start the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Reset": You can reset the motor after interlocks or errors.
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section sedtion Operator control permissions (Page 248).

(3) Displaying current control signals

This area shows the most important parameters for this block with the current selection:

- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Motor protection": 1 = Motor is in "good" state
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"
- "Feedback →": 1 = Motor has started and is running
- "Stop": 1 = Stop motor
- "Start": 1 = Start motor
- "Channel Start": Signal from the output channel block for "Start"

7.3.8.4 Block icon for MotS

Block symbols for MotS

A variety of block symbols are available with the following functions:

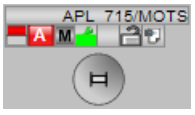
- Process tag type
- Violation of alarm, warning, and tolerance limits as well as control system faults

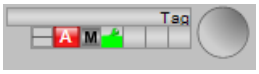
- Operating modes
- Signal status, release for maintenance
- Displays for bypassing interlocks
- Interlocks
- Memo display
- Motor state display

The block symbols from template @TemplateAPLV8.PDL:

Symbols	Selection of the block symbol in CFC	Special features
	1	
	2	
	3	
	4	
	5	
	6	

The block symbols from template @TemplateAPLV7.PDL:

Symbols	Selection of the block symbol in CFC	Special features
	1	Block symbol in the full display
	2	
	3	

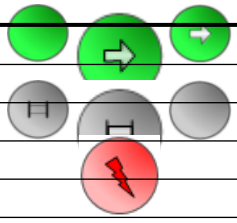
Symbols	Selection of the block symbol in CFC	Special features
	4	
	5	
	6	
	-	Block symbol in "Out of service" mode (example with type 1 block symbol)

Additional information on the block symbol and the control options in the block symbol is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Motor state display

The following motor states are shown here:

Symbol	Meaning
	Motor started (motor symbol changes)
	The motor is running
	Motor stopped (motor symbol changes)
	Motor idle
	Error at motor (monitoring error, motor protection)

7.4 MotRevL - Reversible motor

7.4.1 Description of MotRevL

Object name (type + number) and family

Type + number: FB 1851

Family: Drives

Area of application for MotRevL

The block is used for the following applications:

- Control of reversible motors

How it works

The block is used to control reversible motors. Various inputs are available for controlling the motor.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the MotRevL block, the Advanced Process Library contains a template for process tag types as an example with an application scenario for this block.

Example of process tag types:

- Reversing motor (MotorReversible) (Page 2337)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section MotRevL I/Os (Page 1134).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`

Status bit	Parameter
4	`OosLi.Value`
5	0 = `ManAct.Value` 1 = `AutoAct.Value`
6	`LocalAct.Value`
7	0: Open padlock in the block icon 1: Closed padlock in the block icon
8	`Fwd.Value`
9	Motor is stopped
10	`Rev.Value`
11	`MonStaErr.Value`
12	`MonDynErr.Value`
13	`BypProt`
14	Invalid signal status
15	Mode switchover error
16	1 = `Intlock` is active
17	1 = `Permit` is active
18	1 = `Protect` is active
19	`Trip.Value`
20	Display „Forced forward“
21	Display „Forced stop“
22	Display „Forced reverse“
23	"Interlock" button is enabled
24	Reset request in automatic preview
25	`WarnAct.Value` or `IdleTime` active
26	Bypass information from previous function block
27	Automatic preview for forward mode
28	Automatic preview for "stopping"
29	Automatic preview for reverse mode
30	"Permission" button is enabled
31	"Protection" button is enabled

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	`AV_AH_Act.Value`
2	`AV_WH_Act.Value`
3	`AV_TH_Act.Value`
4	`AV_TL_Act.Value`
5	`AV_WL_Act.Value`
6	`AV_AL_Act.Value`

Status bit	Parameter
7	AV_AH_En
8	AV_WH_En
9	AV_TH_En
10	AV_TL_En
11	AV_WL_En
12	AV_AL_En
13	AV_AH_MsgEn
14	AV_WH_MsgEn
15	AV_TH_MsgEn
16	AV_TL_MsgEn
17	AV_WL_MsgEn
18	AV_AL_MsgEn
19	1 = No impact of input signals on "local mode" with LocalSetting = 2 and LocalSetting = 4
20	Motor is stopped
21	Motor stops in forward mode
22	Motor stops in reverse mode
23	Motor starts in forward mode
24	Motor runs in forward mode
25	Motor starts in reverse mode
26	Motor runs in reverse mode
27	Error when stopping motor
28	Error in forward mode of motor
29	Error in reverse mode of motor
30	Display for interlocks in block icon
31	MS_RelOp

Status word allocation for Status3 parameter

Status bit	Parameter
0	Delay of the AV_AH_Lim message
1	Delay of the AV_WH_Lim message
2	Delay of the AV_TH_Lim message
3	Delay of the AV_TL_Lim message
4	Delay of the AV_WL_Lim message
5	Delay of the AV_AL_Lim message
6	Collection of message delays
7 - 14	Not used
15	Current monitoring time is visible
16	MonDynStopErr.Value
17 - 18	SimLiOp.Value
19	1 = Enable for rapid stop (Feature Bit Enabling rapid stop via faceplate (Page 167))

Status bit	Parameter
20	1 = Input parameter `FwdChnST` is interconnected
21	1 = Input parameter `RevChnST` is interconnected
22	Not used
23	Command for rapid stop
24	Command for starting → the motor
25	Command for starting ← the motor
26	Show automatic preview in the standard view
27	Not used
28	`GrpErr.Value`
29	`RdyToStart.Value`
30	Auxiliary value 1 visible
31	Auxiliary value 2 visible

Status word allocation for `Status4` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via `EventTsIn`
1	Effective signal 2 of the message block connected via `EventTsIn`
2	Effective signal `3` of the message block connected via `EventTsIn`
3	Effective signal 4 of the message block connected via `EventTsIn`
4	Effective signal 5 of the message block connected via `EventTsIn`
5	Effective signal 6 of the message block connected via `EventTsIn`
6	Effective signal 7 of the message block connected via `EventTsIn`
7	Effective signal 8 of the message block connected via `EventTsIn`
8	`AV` not connected
9	Motor protection display (`Trip.Status` ≠ 16#FF)
10	1 = Input parameter `FbkFwd` is connected
11	1 = Input parameter `FbkRev` is connected
12 - 19	Effective signal 1...8 of the message block connected via `EventTs2In`
20 - 21	Not used
22	External error generated by FaultExt or external control system fault from CSF with set `Feature` bit 18 Activating error state for external process control error CSF (Page 150)
23	Hidden bypass signal in Permit
24	Hidden bypass signal in interlock
25	Hidden bypass signal in Protect
26	`Feature2` bit 2: Separate bypass signal
27	`Feature2` bit 16: Separate interlock for each direction or position
28	Hidden bypass signal in Permit Reverse
29	Hidden bypass signal in Interlock Reverse
30	Hidden bypass signal in Protect Reverse
31	Separate monitoring of shutdown of the motor (`Feature` bit 13)

Status word allocation for `Status5` parameter

Status bit	Parameter
0 - 7	Effective signal 9...16 of the message block connected via `EventTsIn`
8 - 15	Effective signal 9...16 of the message block connected via `EventTs2In`
16	The permission button for "Forward" is enabled
17	The permission button for "Reverse" is enabled
18	The interlock button for "Forward" is enabled
19	The Interlock button for "Reverse" is enabled
20	The protection button "Forward" is enabled
21	The protection button "Reverse" is enabled
22	1 = Permission button "Reverse" is active
23	1 = Interlock button "Reverse" is active
24	1 = Protection button "Reverse" is active
25 - 31	Not used

See also

MotRevL functions (Page 1122)
MotRevL messaging (Page 1132)
MotRevL block diagram (Page 1143)
MotRevL error handling (Page 1129)
MotRevL modes (Page 1120)

7.4.2 MotRevL modes

MotRevL operating modes

The block can be operated using the following modes:

- Local mode (Page 79)
- Automatic mode (Page 75)
- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) section.

Motor actions you can control in local mode:

- "Starting in forward" (`FwdLocal` = 1)
- "Starting in reverse" (`RevLocal` = 1)
- "Stopping" (`StopLocal` = 1)

A motor operated in "local mode" is controlled either by "local" signals or by the feedback signals (input parameters `FbkFwd` = 1 and `FbkRev` = 1) . Configuration takes place via the input parameter `LocalSetting`.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) section.

Motor actions you can control in auto mode:

- "Starting in forward" (`FwdAut` = 1)
- "Starting in reverse" (`RevAut` = 1)
- "Stopping" (`StopAut` = 1)

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

Motor actions you can control in "manual mode":

- "Starting in forward" (`FwdMan` = 1)
- "Starting in reverse" (`RevMan` = 1)
- "Stopping" (`StopMan` = 1)

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

MotRevL block diagram (Page 1143)

MotRevL I/Os (Page 1134)

MotRevL messaging (Page 1132)

MotRevL error handling (Page 1129)

MotRevL functions (Page 1122)

Description of MotRevL (Page 1116)

7.4.3 MotRevL functions

Functions of MotRevL

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	1 = Operator can switch to "local mode"
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can stop the motor
5	1 = Operator can start the motor in forward
6	1 = Operator can start the motor in reverse
7	1 = Operator can reset the motor
8	1 = Operator can define or change the monitoring time for startup
9	1 = Operator can define the monitoring time for the status
10	1 = Operator can enable the monitoring time function (Bit 8 - 9)
11	1 = Operator can activate the Simulation function
12	1 = Operator can activate the Release for maintenance function
13	1 = Operator can change the high limit (`AV`) for the alarm
14	1 = Operator can change the high limit (`AV`) for the warning
15	1 = Operator can change the high limit (`AV`) for the tolerance
16	1 = Operator can change the limit (`AV`) for hysteresis
17	1 = Operator can lower the limit (`AV`) for the alarm
18	1 = Operator can lower the limit (`AV`) for the warning
19	1 = Operator can lower the limit (`AV`) for the tolerance
20	1 = Operator can activate / deactivate messages via `AV_AH_MsgEn`
21	1 = Operator can activate / deactivate messages via `AV_WH_MsgEn`
22	1 = Operator can activate / deactivate messages via `AV_TH_MsgEn`
23	1 = Operator can activate / deactivate messages via `AV_TL_MsgEn`
24	1 = Operator can activate / deactivate messages via `AV_WL_MsgEn`
25	1 = Operator can activate / deactivate messages via `AV_AL_MsgEn`
26	1 =Operator can change the simulation value `SimAV`
27 - 29	Not used

Bit	Function
30	1 = Operator can define the monitoring time for stopping
31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Restart lock after changing direction of rotation or switching off the motor

Use the input parameter `IdleTime` to enter a restart lock for changing the direction of rotation or restarting the motor. Use the `Feature` Bit Enabling direct changeover between forward and reverse (Page 145) to define how the change is to take place. When the "Stop" command is given, the motor goes immediately into "Stop" mode, and `IdleTime` starts after the feedback (`FbkFwd` and `FbkRev` = 0) is given. The motor cannot be started again until the `IdleTime` has expired.

The `IdleTime` parameter can be set independently of the `MonTiDynamic` parameter.

Limit monitoring of an additional analog value

This block provides the standard function Limit monitoring of an additional analog value (Page 91).

Limit monitoring with hysteresis

This block provides the standard function Limit monitoring with hysteresis (Page 97). It is performed via the input parameter `AV_Hyst`.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Interlocks

This block provides the following interlocks:

Feature 2 bit 16 = 0:

- Activation enable
- Interlock without reset ("Interlock")
- Interlock with reset ("Protection")

Feature 2 bit 16 = 1:

- Activation enable forward
- Activation enable reverse

- Interlock forward without reset ("Interlock forward")
- Interlock reverse without reset ("Interlock reverse")
- Interlock forward with reset ("Protection interlock forward")
- Interlock reverse with reset ("Protection interlock reverse")

Refer to the section Interlocks (Page 99) as well as Influence of the signal status on the interlock (Page 103).

Motor protection function

This block provides the standard function Motor protection function (Page 99).

Rapid stop for motors

This block provides the standard function Rapid stop for motors (Page 106).

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Resetting the block in case of interlocks or errors

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See MotRevL error handling (Page 1129).

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `Trip`
- `MonDynErr`
- `MonStaErr`
- `FaultExt`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `FbkFwdOut.ST`
- `FbkRevOut.ST`
- `LocalLi.ST`
- `FwdLocal.ST`
- `StopLocal.ST`
- `RevLocal.ST`
- `Trip.ST`
- `AV_Out.ST`
- `FwdChn.ST`
- `RevChn.ST`
- `FwdAut.ST` (only if `Feature2.Bit10` = 1)
- `RevAut.ST` (only if `Feature2.Bit10` = 1)
- `StopAut.ST` (only if `Feature2.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position (motor stop) in the "Automatic" mode:

- `FwdAut.ST`
- `RevAut.ST`
- `StopAut.ST`

Forcing operating modes

This block provides the standard function Forcing operating modes (Page 41).

The following states can be enforced:

- "Start in forward" (`FwdForce`)
- "Start in reverse" (`RevForce`)
- "Stop" (`StopForce`)

Note

The `Feature` Bit Enabling direct changeover between forward and reverse (Page 145) for this block has no function with forced operating states. The motor can always be reversed directly.

Feedback monitoring

This block provides the standard function Monitoring the feedbacks (Page 97).

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Specify warning times for control functions

This block provides the standard function Specifying warning times for control functions at motors and valves (Page 51).

You can generate warning signals when, for example, motors are started. Warning signals can be generated in the following modes:

- Manual mode (Page 75) (`WarnTiMan` input parameter)
- Automatic mode (Page 75) (`WarnTiAut` input parameter)

You specify the warning times in seconds using the input parameters `WarnTiMan` and `WarnTiAut`. If, for example, a motor is started, then this is displayed at the output parameter with `WarnAct` = 1. The motor then starts after the set warning time has expired and `WarnAct` then goes back to 0.

A corresponding warning is not output if the warning times (`WarnTiMan` or `WarnTiAut`) are specified with a smaller value than the `SampleTime` parameter.

Simulating signals

This block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Additional value (`SimAV`, `SimAV_Li`)

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Neutral position

This block provides the standard function Neutral position for motors, valves and controllers (Page 48).

Output signal as a pulse signal or static signal

This block provides the standard function Output signal as a static signal or pulse signal (Page 51).

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
4	Setting switch or button mode (Page 166)
7	Enabling direct changeover between forward and reverse (Page 145)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
13	Separate monitoring time for stopping the motor (Page 168)
14	Enabling rapid stop via faceplate (Page 167)
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)
18	Activating error state for external process control error CSF (Page 150)
19	Reset even with locked state (Page 164)
20	Disable calculation of impulse control in LocalSetting 2 and 4 (Page 185)
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
27	Interlock display with LocalSetting 2 or 4 (Page 177)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

In pushbutton operation (Bit 4 = 0) the automatic commands in "automatic" mode are latching; in other words, `FwdAut`, `RevAut`, `StopAut` can be reset to 0 after changing the control. In "manual" and "local" modes, however, the automatic commands are not saved and in the absence of automatic commands the automatic control is tracked.

In switching mode (Bit 4 = 1), control is selected with the static signals `FwdAut`, `RevAut`. If `FwdAut`, `RevAut` inputs are not set, the motor is stopped. Control via `StopAut` is not needed. If the "Activate command reset for control" function (Bit 3 = 1) is also enabled, the inputs `FwdAut`, `RevAut` are reset to the neutral position after evaluation in the block.

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
4	Setting switch or button mode for local commands (Page 180)
5	Evaluation of the signal status of the interlock signals (Page 141)
8	Forcing operating modes in the "Local" mode (Page 183)
10	Considering bad quality of automatic commands or external values (Page 185)
16	Separate interlock for each direction or position (Page 169)

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` or `EventTs2In` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- `FwdMan`
- `RevMan`

- StopMan
- RapidStp

See also

Description of MotRevL (Page 1116)

MotRevL messaging (Page 1132)

MotRevL I/Os (Page 1134)

MotRevL block diagram (Page 1143)

MotRevL modes (Page 1120)

EventTs functions (Page 1634)

7.4.4 MotRevL error handling

Error handling of MotRevL

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Invalid input signals
- Control system fault (CSF)

Overview of error numbers

The ErrorNum I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
41	The value for the LocalSetting I/O is not within the approved limit of 0 to 4.
42	LocalSetting = 0 or LocalSetting = 3 or LocalSetting = 4 and LocalLi = 1

Error number	Meaning of the error number
51	`FwdLocal` = 1 and `StopLocal` = 1 `RevLocal` = 1 and `StopLocal` = 1 `FwdLocal` = 1 and `RevLocal` = 1 `FwdAut` = 1 and `StopAut` = 1 `RevAut` = 1 and `StopAut` = 1 `FwdAut` = 1 and `RevAut` = 1 `AutModLi` = 1 and `ManModLi` = 1 `FwdForce` = 1 and `StopForce` = 1 `RevForce` = 1 and `StopForce` = 1 `FwdForce` = 1 and `RevForce` = 1
52	`LocalAct` = 1 and `LocalSetting` = 2 or 4 and `SimOn` = 1

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

Invalid input signals

This error can be output by the block, see the section Error handling (Page 119).

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signals	Control reaction with Feature2 bit 3 =1
Local: `Localsetting` = 1 or `Localsetting` = 3	Pushbutton operation for local mode (`Feature2` bit 4 = 0): `FwdLocal` = 1 and `RevLocal` = 1 or `FwdLocal` = 1 and `StopLocal` = 1 or `StopLocal` = 1 and `RevLocal` = 1 Switching mode (`Feature2` bit 4 = 1): `FwdLocal` = 1 and `RevLocal` = 1	Motor is stopped
Local: `Localsetting` = 1 or `Localsetting` = 3 and forcing	`FwdForce` = 1 and `RevForce` = 1 or `FwdForce` = 1 and `StopForce` = 1 or `StopForce` = 1 and `RevForce` = 1	
Forcing and no "local mode"	`FwdForce` = 1 and `RevForce` = 1 or `FwdForce` = 1 and `StopForce` = 1 or `StopForce` = 1 and `RevForce` = 1	
"Automatic mode" and no forcing	Pushbutton operation (`Feature` bit 4 = 0): `FwdAut` = 1 and `RevAut` = 1 or `FwdAut` = 1 and `StopAut` = 1 or `StopAut` = 1 and `RevAut` = 1 Switching mode (`Feature`bit 4 = 1): `FwdAut` = 1 and `RevAut` = 1	
"Manual mode" and no forcing	`FwdMan` = 1 and `RevMan` = 1 or `FwdMan` = 1 and `StopMan` = 1 or `StopMan` = 1 and `RevMan` = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

MotRevL block diagram (Page 1143)

MotRevL I/Os (Page 1134)

MotRevL messaging (Page 1132)

Description of MotRevL (Page 1116)

MotRevL modes (Page 1120)

MotRevL functions (Page 1122)

7.4.5 MotRevL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 1	AS process control message - fault	$$BlockComment$$ Motor feedback error
	SIG 2	AS process control message - fault	$$BlockComment$$ Motor protection triggered
	SIG 3	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a process control fault is triggered (`MsgEvId1`, SIG 3).

Instance-specific messages

You can use up to three instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 4	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 5	AS process control message - fault	$$BlockComment$$ External message 2
	SIG 6	AS process control message - fault	$$BlockComment$$ External message 3

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance **MsgEvId1**

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	ExtVa104
5	ExtVa105
6	ExtVa106
7	ExtVa107
8	ExtVa108
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters ExtVa104 ... ExtVa108 and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of MotRevL (Page 1116)

MotRevL functions (Page 1122)

MotRevL I/Os (Page 1134)

MotRevL block diagram (Page 1143)

MotRevL error handling (Page 1129)

MotRevL modes (Page 1120)

7.4.6 MotRevL I/Os

I/Os of MotRevL

Input parameters

Parameter	Description	Type	Default
AutModLi*	1 = "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 0)	BOOL	0
AV	Input additional analog value, to be connected to AV_Out of the AV block	ANY	
AV_AH_Lim	Limit high alarm	REAL	95.0
AV_AL_Lim	Limit low alarm	REAL	5.0
AV_Hyst	Hysteresis for alarm, warning and tolerance limits	REAL	1.0
AV_TH_Lim	Limit high tolerance	REAL	85.0
AV_TL_Lim	Limit low tolerance	REAL	15.0
AV_WH_Lim	Limit high warning	REAL	90.0
AV_WL_Lim	Limit low warning	REAL	10.0
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Bypassing interlock in "local mode" and in "simulation"	BOOL	0
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EN	1 = Called block will be processed	BOOL	1
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	

Parameter	Description	Type	Default
EventTs2In	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTs2In input parameter serves to interconnect the EventTsOut output parameter of the EventTs,Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkFwd	1 = Feedback for forward mode is present	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkRev	1 = Feedback for reverse mode is present	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF

Parameter	Description	Type	Default
`Feature`	I/O for additional functions (Page 1122) 1 = Separate monitoring time for stopping the motor	STRUCT • Bit 0: BOOL • ... • Bit 13: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0 • 0 • 0
`Feature2`	I/O for additional functions (Page 1122)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`FwdAut*`	1 = Activating forward mode of motor in automatic mode	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`FwdChnST`	Signal status of output channel `Fwd` Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`FwdForce`	1 = Forcing activation of motor forward mode	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`FwdLocal`	1 = Activation of forward motor operation in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`FwdMan*`	1 = Activation of forward motor operation in "manual mode"	BOOL	0
`IdleTime*`	Wait time for change of direction or restart in [s]	REAL	5.0
`Intlock`	0 = Interlock / interlock **Motor Forward** is activated Once the interlock condition has cleared, the block can be operated without reset. 1 = Interlock / interlock **Motor Forward** is deactivated.	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
`Intl_En`	1 = Interlock / **Interlock Motor Forward** (interlock, Intlock parameter) is activated	BOOL	1
`IntlRev`	0 = Interlock Motor Reverse is activated. Once the interlock condition has cleared, the block can be operated without reset. 1 = Interlock Motor Reverse is deactivated.	BOOL	1
`IntlRevEn`	1 = Interlock Motor Reverse (interlock, IntlRev parameter) is activated	BOOL	1
`LocalLi`	1 = Activate "local mode" via plant signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
LocalOp*	1 = "Local mode" via operator	BOOL	0
LocalSetting	Characteristics of Local mode (Page 79)	INT	0
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
ModLiOp	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Monitor	1 = Feedback monitoring	BOOL	1
MonTiDynamic*	Monitoring time for feedback errors or feedback start error after successful operation in [s]	REAL	3.0
MonTiDyStop*	Monitoring time for feedback stop errors after successful operation in [s]	REAL	3.0
MonTiStatic*	Monitoring time for feedback errors without operation in [s]	REAL	3.0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp*	1 = Release for maintenance via OS operator	BOOL	0
Occupied	1 = In use by a batch	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 1122) 1 = Operator can set or change the monitoring time for "Control: Start" 1 = Operator can set or change the monitoring time for "Control: Stop"	STRUCT • Bit 0: BOOL • Bit 8: BOOL • Bit 20: BOOL • Bit 30: BOOL	- • 1 • 1 • 1 • 1

Parameter	Description	Type	Default
Permit	1 = Activation enable / **Activation enable Motor Forward** 0 = No activation enable for motor/**Motor Forward**	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Perm_En	1 = Activation enable / **(activation enable Motor Forward** (enable, Permit parameter)is activated	BOOL	1
PermRev	1 = Activation enable Motor Reverse 0 = No activation enable Motor Reverse	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
PermRevEn	1 = Activation enable Motor Reverse(enable, PermRev parameter) is activated	BOOL	1
Protect	0 = Protective interlock / **protective interlock Motor Forward** is activated. Once the interlock condition has cleared, the block must be reset 1 = Protective interlock / **protective interlock Motor Forward** is deactivated.	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Prot_En	1 = Protective interlock / **protective interlock Motor Forward** (protection, Protect parameter) is activated	BOOL	1
ProtRev	0 = Protective interlock Motor Reverse is activated. Once the interlock condition has cleared, the block must be reset 1 = Protective interlock Motor Reverse is deactivated.	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
ProtRevEN	1 = Protective interlock Motor Reverse (protection, ProtRev parameter) is activated	BOOL	1
PulseWidth*	Pulse width of control signal [s]	REAL	3.0
RapidStp*	Rapid stop for the motor: 0 = Motor On 1 = Motor Off	BOOL	0
RevAut*	1 = Activation of reverse motor operation in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RevChnST	Signal status of output channel Rev Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
RevForce	1 = Force activation of reverse motor operation	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RevLocal	1 = Activation of reverse motor operation in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
RevMan*	1 = Activation of reverse motor operation in "manual mode"	BOOL	0
RstLi*	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SimAV*	Additional value used for SimOn = 1	REAL	0.0
SimAV_Li	Additional analog value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0 • 16#80
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-
StepNo	Batch step number	DWORD	16#00000000
StopAut*	1 = Stop the motor in automatic mode	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopForce	1 = Force motor stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopLocal	1 = Stopping the motor in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopMan*	1 = Stopping the motor in "manual mode"	BOOL	0
Trip	1 = Motor is in "good" state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF

Parameter	Description	Type	Default
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA2unit	Unit of measure for analog auxiliary value 2	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
WarnTiAut*	Prewarning of motor start in automatic mode in [s]	REAL	0.0
WarnTiMan*	Prewarning of motor start in manual mode in [s]	REAL	0.0

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80
AV_OpScale	Limit for scale in AV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
AV_Out	Output additional analog value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
AV_Unit	Unit of measure for additional analog value	INT	0
CurrMon	Current monitoring time [s]	DINT	0
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see MotRevL error handling (Page 1129)	INT	-1
FbkFwdOut	Feedback: 1 = Forward operation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
FbkRevOut	Feedback: 1 = Reverse operation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Fwd	1 = Control of motor forward	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalAct	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock (Intlock, Permit, Protect) or Trip is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MonDynErr	1 = Feedback error or feedback start error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonDynStopErr	1 = Feedback stop error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonStaErr	1 = Feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
P_Fwd	1 = Pulse signal for starting the motor in forward	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Rev	1 = Pulse signal for starting the motor in reverse	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Stop	0 = Pulse signal for stopping the motor	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
R_StpAct	1 = Rapid stop of the motor is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Rev	1 = Control of motor: Reverse	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RunFwd	1 = Motor is running forwards	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RunRev	1 = Motor is running in reverse	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Status1	Status word 1 (Page 1116)	DWORD	16#00000000
Status2	Status word 2 (Page 1116)	DWORD	16#00000000
Status3	Status word 3 (Page 1116)	DWORD	16#00000000
Status4	Status word 4 (Page 1116)	DWORD	16#00000000

Parameter	Description	Type	Default
`Status5`	Status word 5 (Page 1116)	DWORD	16#00000000
`Stop`	1 = Motor stopped	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`WarnAct`	1 = Prewarning for motor start active (parameters `WarnTiAut` and `WarnTiMan`)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

MotRevL messaging (Page 1132)

MotRevL block diagram (Page 1143)

MotRevL modes (Page 1120)

Error handling (Page 119)

7.4.7 MotRevL block diagram

MotRevL block diagram

A block diagram is not provided for this block.

See also

MotRevL I/Os (Page 1134)

MotRevL messaging (Page 1132)

MotRevL error handling (Page 1129)

MotRevL functions (Page 1122)

MotRevL modes (Page 1120)

Description of MotRevL (Page 1116)

7.4.8 Operator control and monitoring

7.4.8.1 MotRevL views

Views of the MotRevL block

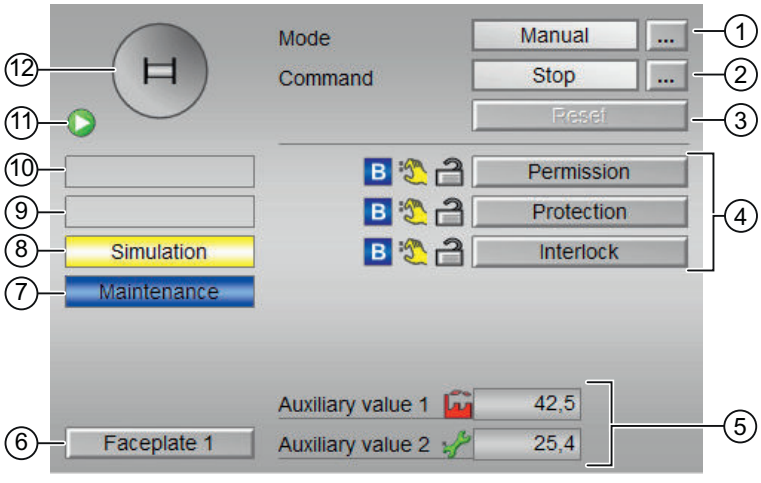
The block MotRevL provides the following views:

- MotRevL standard view (Page 1144)
- Alarm view (Page 296)
- Limit view of motors (Page 288)
- Trend view (Page 299)
- Parameter view for motors and valves (Page 280)
- MotRevL preview (Page 1148)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MotRevL (Page 1153)

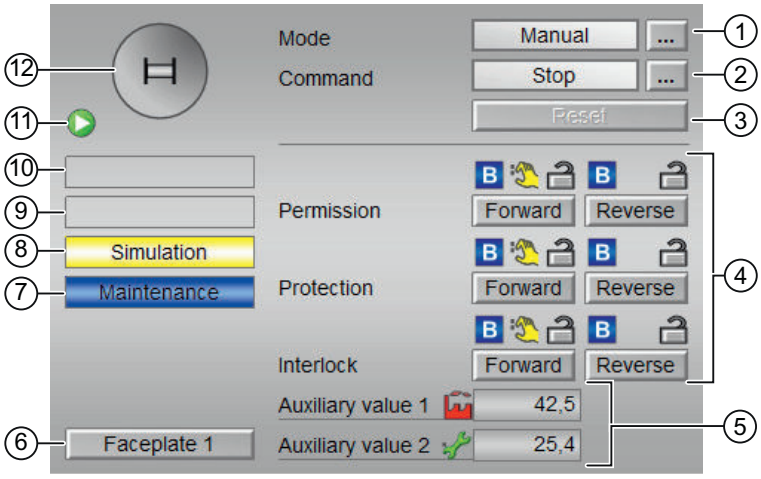
Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.4.8.2 MotRevL standard view

MotRevL standard view



Feature2 bit 16 =0



Feature2 bit 16 =1

(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

See also the section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Starting and stopping the motor

This area shows you the default operating state for the motor. The following states can be shown and executed here:

- "Start→ "
- "Start ←"
- "Stop"
- "Rapid stop"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(3) Resetting the block

Click "Reset" for interlocks or errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(4) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the section Interlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

(5) Display of auxiliary values

This display is only visible when the corresponding block input is connected.

You can use this area to display two auxiliary values that have been configured in the engineering system (ES). You can find additional information on this in the section Displaying auxiliary values (Page 207).

(6) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(7) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

(8) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"
- "Delay"

You will find more detailed information on this in the sections Simulating signals (Page 58) and Display of delay times (Page 250).

(9) Display area for block states

This area provides additional information on the operating state of the block:

- "Motor protection"
- "External error"
- "Status error"
- "Control error"
- "Invalid signal"
- "Changeover error"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97) , Error handling (Page 119) (section "Invalid input signals" and "Mode switchover error") and Motor protection function (Page 99).

(10) Display area for block states

This area provides additional information on the operating state of the block:

- "Forced stop"
- "Forced start →"
- "Forced start ←"
- "Request 0/1": A reset to "automatic mode" is expected.

You can find additional information on this in the section Forcing operating modes (Page 41).

(11) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the motor would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

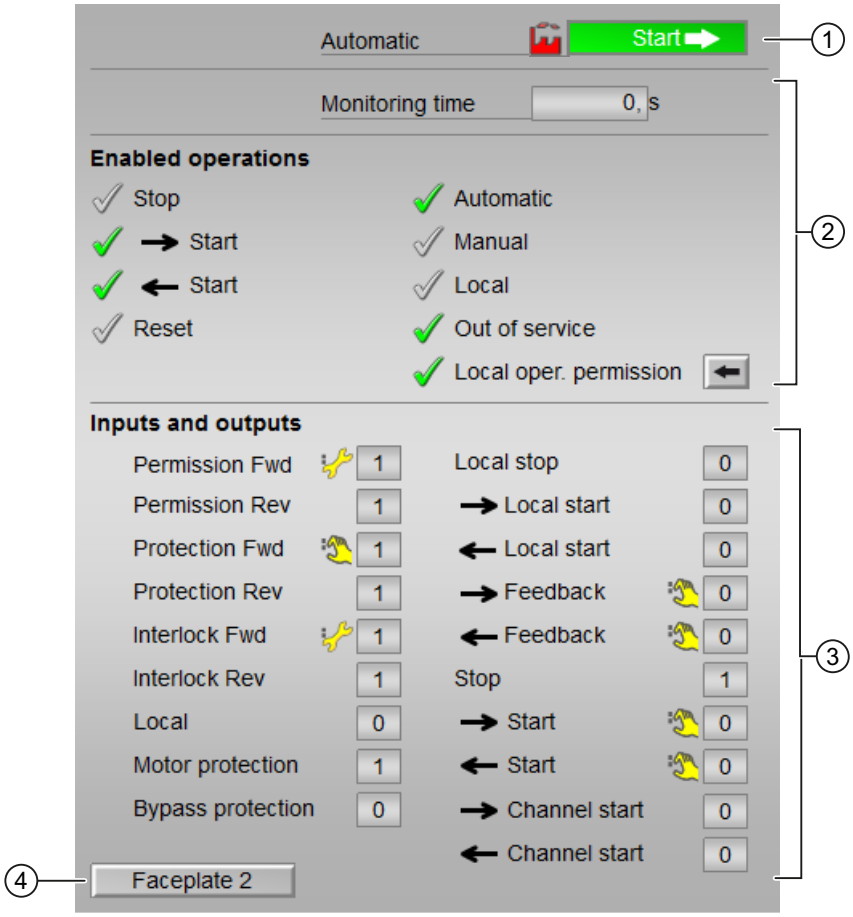
(12) Status display of the motor

The current status of the motor is graphically displayed here.

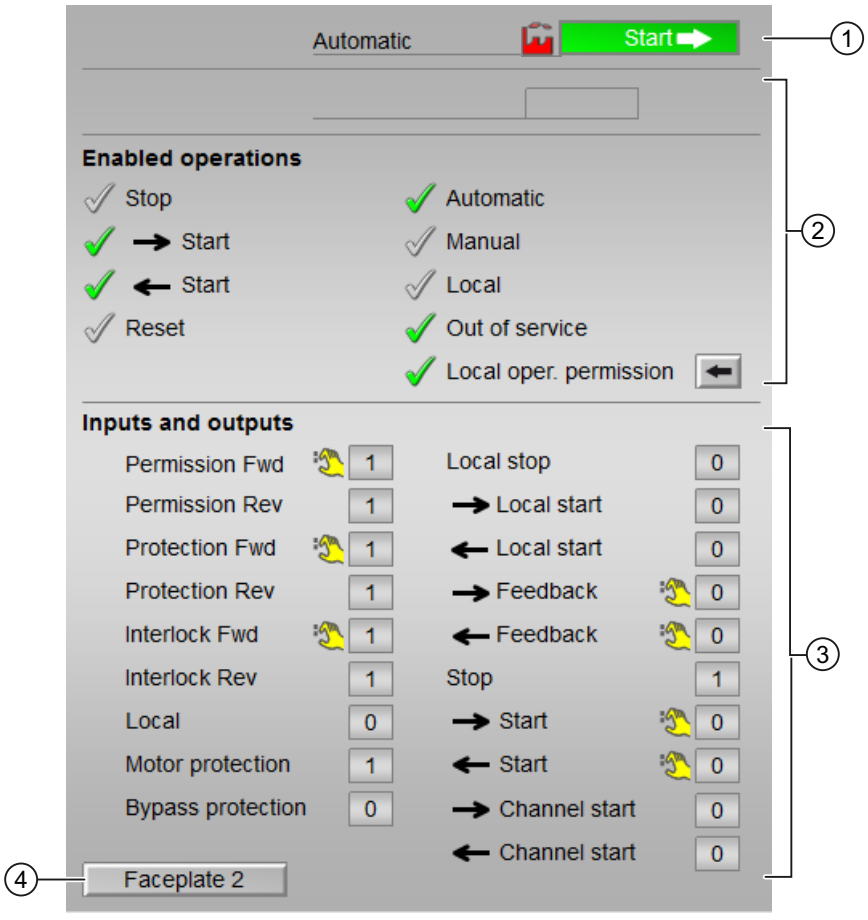
You can find more information about this in section Block icon for MotRevL (Page 1153)

7.4.8.3 MotRevL preview

Preview of MotRevL



Display of the current monitoring time is visible.



Display of the current monitoring time is not visible.

(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `FwdAut`
- `RevAut`
- `StopAut`

(2) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Stop": You can stop the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Start →": You can start the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Start ←": You can start the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Reset": You can reset the motor after interlocks or errors.
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Local": You can switch to "local mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248) .
- "Monitoring time": Display of the current monitoring time.

(3) Displaying current control signals

This area shows the most important parameters for this block with the current selection:

- "Permission":
  This display is only visible when the corresponding block input is connected.
  - 0 = No OS release for energizing motor
  - 1 = Enable for "starting"/"stopping" from the neutral position
- "Enable" (**Feature 2 bit 16 =0**):
  This display is only visible when the corresponding block input is connected.
  - 0 = No OS release for energizing motor
  - 1 = Enable for "Start" from the neutral position

- "Enable Fwd" (**Feature 2 bit 16 =1**):
  This display is only visible when the corresponding block input is connected.
  - 0 = No OS release for energizing motor
  - 1 = Enable for "Start forward" from the neutral position
- "Enable Rev" (**Feature 2 bit 16 =1**):
  This display is only visible when the corresponding block input is connected.
  - 0 = No OS release for energizing motor
  - 1 = Enable for "Start reverse" from the neutral position
- "Protection":
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state
- "Protection" (**Feature 2 bit 16 =0**):
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state
- "Protection Fwd" (**Feature 2 bit 16 =1**):
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state
- "Protection Rev" (**Feature 2 bit 16 =1**):
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state
- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Interlock" (**Feature 2 bit 16 =0**):
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state

- "Interlock Fwd" (**Feature 2 bit 16 =1**):
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Interlock Rev" (**Feature 2 bit 16 =1**):
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Motor protection": 1 = Motor is in "good" state
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"
- "Local stop": 1 = Stopping the motor in "local mode"
- "Local start →": 1 = Starting the motor in "local mode"
- "Local start ←": 1 = Starting the motor in "local mode"
- "Feedback →": 1 = Motor has started and is running
- "Feedback ←": 1 = Motor has started and is running
- "Stop": 1 = Stop motor
- "Start →": 1 = Start motor
- "Start ←": 1 = Start motor
- "Channel Start →": Signal from the output channel block for "Start"
- "Channel Start ←": Signal from the output channel block for "Start"

(4) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

7.4.8.4 Block icon for MotRevL

Block icons for MotRevL

A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes
- Signal status, release for maintenance
- Displays for bypassing interlocks
- Interlocks
- Memo display
- Motor state display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Direction display inverted to symbol 1
	4	Direction display inverted to symbol 2
	5	"M" symbol with small direction display
	6	"M" symbol with small direction display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	9	
	10	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Motor state display

The following motor states are shown here:

Icon	Meaning
	Motor started (motor symbol changes)
	The motor is running
	Motor stopped (motor symbol changes)
	Motor idle
	Error at motor (monitoring error, motor protection)
	Motor out of service

7.5 MotSpdCL - Controllable reversible motor

7.5.1 Description of MotSpdCL

Object name (type + number) and family

Type + number: FB 1854

Family: Drives

Area of application for MotSpdCL

The block is used for the following applications:

- Control of motors with two directions of rotation and different speeds
- Control of a device feed (optionally configurable via feature bit)

How it works

The block is used to control reversible motors for different speeds. Various inputs are available for controlling the motor.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the MotSpdCL block, the Advanced Process Library contains a template for process tag types as an example with an application scenario for this block.

Example of process tag types:

- Reversible motor with controllable speed (MotorSpeedControlled) (Page 2338)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Note

At a warm restart with the `Feature` Bit set to 0, the block is switched to manual mode and the setpoint is set to internal and to 0.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

For a description of the individual parameters, see the section MotSpdCL I/Os (Page 1179).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	0 = `ManAct.Value` 1 = `AutoAct.Value`
6	`LocalAct.Value`
7	0: Open padlock in the block icon 1: Closed padlock in the block icon
8	`Fwd.Value`
9	`SimLiOp.Value`
10	`Rev.Value`
11	`MonStaErr.Value`
12	`MonDynErr.Value`
13	`BypProt`
14	Invalid signal status
15	Mode switchover error
16	1 = `Intlock` is active
17	1 = `Permit` is active
18	1 = `Protect` is active
19	`Trip`
20	Display „Forced forward“
21	Display „Forced stop“
22	Display „Forced reverse“
23	"Interlock" button is enabled
24	Reset request in automatic preview
25	`WarnAct.Value` or `IdleTime` active
26	Bypass information from previous function block
27	Automatic preview for forward mode
28	Automatic preview for "stopping"
29	Automatic preview for reverse mode
30	"Permission" button is enabled
31	"Protection" button is enabled

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	`AV_AH_Act.Value`
2	`AV_WH_Act.Value`
3	`AV_TH_Act.Value`
4	`AV_TL_Act.Value`
5	`AV_WL_Act.Value`
6	`AV_AL_Act.Value`
7	`AV_AH_En`
8	`AV_WH_En`
9	`AV_TH_En`
10	`AV_TL_En`
11	`AV_WL_En`
12	`AV_AL_En`
13	`AV_AH_MsgEn`
14	`AV_WH_MsgEn`
15	`AV_TH_MsgEn`
16	`AV_TL_MsgEn`
17	`AV_WL_MsgEn`
18	`AV_AL_MsgEn`
19	1 = No impact of input signals on "local mode" with `LocalSetting` = 2 and `LocalSetting` = 4
20	Motor is stopped
21	Motor stops in forward mode
22	Motor stops in reverse mode
23	Motor starts in forward mode
24	Motor runs in forward mode
25	Motor starts in reverse mode
26	Motor runs in reverse mode
27	Error when "stopping" the motor
28	Error in forward mode of motor
29	Error in reverse mode of motor
30	`SP_ExtAct.Value`
31	Display for interlocks in block icon

Status word allocation for `Status3` parameter

Status bit	Parameter
0	1 = SP ramp active
1	`RbkWH_Act.Value`
2	Delay of the `AV_AH_Lim` message
3	Delay of the `AV_WH_Lim` message

Status bit	Parameter
4	RbkWL_Act.Value
5	Delay of the AV_TH_Lim message
6	Delay of the AV_TL_Lim message
7	RbkWH_En
8	Delay of the AV_WL_Lim message
9	Delay of the AV_AL_Lim message
10	RbkWL_En
11	Delay of the RbkWH_Lim message
12	Delay of the RbkWL_Lim message
13	Collection of message delays
14	RbkWH_MsgEn
15	Current monitoring time is visible
16	MonDynStopErr.Value
17	RbkWL_MsgEn
18	Motor is stopped
19	1 = Enable for rapid stop (Feature Bit Enabling rapid stop via faceplate (Page 167))
20	SP_RmpModTime
21	SP_RmpOn
22	SP_UpRaAct, SP_DnRaAct limits enabled for gradient mode (SP_RateOn = 1)
23	Command for "rapid stop" of the motor
24	Command for "starting" → the motor
25	Command for "starting" ← the motor
26	Show automatic preview in the standard view
27	RdyToStart.Value
28	GrpErr.Value
29	MS_RelOp
30	Auxiliary value 1 visible
31	Auxiliary value 2 visible

Status word allocation for `Status4` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via EventTsIn
1	Effective signal 2 of the message block connected via EventTsIn
2	Effective signal 3 of the message block connected via EventTsIn
3	Effective signal 4 of the message block connected via EventTsIn
4	Effective signal 5 of the message block connected via EventTsIn
5	Effective signal 6 of the message block connected via EventTsIn
6	Effective signal 7 of the message block connected via EventTsIn
7	Effective signal 8 of the message block connected via EventTsIn
8	AV not connected

Status bit	Parameter
9	Motor protection display (`Trip.Status` ≠ 16#FF)
10	1 = Input parameter `FbkFwd` is connected
11	1 = Input parameter `FbkRev` is connected
12	1 = Setpoint difference high limit violated (`ER_AH_Act.Value`)
13	1 = Setpoint difference low limit violated (`ER_AL_Act.Value`)
14	1 = Monitor setpoint difference high limit (`ER_AH_En`)
15	1 = Monitor setpoint difference low limit (`ER_AL_En`)
16	1 = Report setpoint difference high limit violation (`ER_AH_MsgEn`)
17	1 = Report setpoint difference low limit violation (`ER_AL_MsgEn`)
18	1 = Monitoring of setpoint difference "`SP - Rbk`" activated
19	`DvFdAct.Value` Device feed ON
20	NOT `DvFdAct.Value` Device feed OFF
21	`Feature` Bit 15 Frequency converter with separate device feed (Page 150)
22	External error generated by FaultExt or external control system fault from CSF with set feature bit 18 Activating error state for external process control error CSF (Page 150)
23	Hidden bypass signal in Permit
24	Hidden bypass signal in interlock
25	Hidden bypass signal in Protect
26	`Feature2` bit 2: Separate bypass signal
27	`Feature` bit 16: Setpoint specification with separate display area and custom unit (Page 170)
28	1 = Input parameter `FwdChnST` is interconnected
29	1 = Input parameter `RevChnST` is interconnected
30	1 = Input parameter `SP_OutChnS` is interconnected
31	Separate monitoring of shutdown of the motor (`Feature` bit 13)

Status word allocation for `Status5` parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via `EventTsIn`
8	Delay of the `ER_AH_Lim` message
9	Delay of the `ER_AL_Lim` message
10 - 15	Not used
16 - 31	Effective signal 1...16 of the message block connected via `EventTs2In`

See also

MotSpdCL block diagram (Page 1191)

MotSpdCL messaging (Page 1177)

MotSpdCL error handling (Page 1174)

MotSpdCL functions (Page 1162)

MotSpdCL modes (Page 1161)

7.5.2 MotSpdCL modes

MotSpdCL operating modes

The block can be operated using the following modes:

- Local mode (Page 79)
- Automatic mode (Page 75)
- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) .

Motor actions you can control in local mode:

- "Starting in forward" (`FwdLocal` = 1)
- "Starting in reverse" (`RevLocal` = 1)
- "Stopping" (`StopLocal` = 1)

A motor operated in "local mode" is controlled either by "local" signals or by the feedback signals (input parameters `FbkFwd` = 1and `FbkRev` = 1). Configuration takes place via the input parameter `LocalSetting`.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) .

Motor actions you can control in auto mode:

- "Starting in forward" (`FwdAut` = 1)
- "Starting in reverse" (`RevAut` = 1)
- "Stopping" (`StopAut` = 1)

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

Motor actions you can control in "manual mode":

- "Starting in forward" (`FwdMan` = 1)
- "Starting in reverse" (`RevMan` = 1)
- "Stopping" (`StopMan` = 1)

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

Note

In case of external setpoint value specification, the block switches to internal setpoint value specification.

See also

MotSpdCL block diagram (Page 1191)

MotSpdCL I/Os (Page 1179)

MotSpdCL messaging (Page 1177)

MotSpdCL error handling (Page 1174)

MotSpdCL functions (Page 1162)

Description of MotSpdCL (Page 1156)

7.5.3 MotSpdCL functions

Functions of MotSpdCL

The functions for this block are listed below.

Alarm delays with two time values per limit pair

This block has the standard alarm delay function for Two time values per limit pair (Page 197) limit monitoring of the feedback.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	1 = Operator can switch to "local mode"
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can stop the motor
5	1 = Operator can start the motor in forward
6	1 = Operator can start the motor in reverse
7	1 = Operator can reset the motor
8	1 = Operator can define or change the monitoring time for startup
9	1 = Operator can define the monitoring time for the status
10	1 = Operator can activate the monitoring time
11	1 = Operator can activate the Simulation function
12	1 = Operator can activate the Release for maintenance function
13	1 = Operator can change the limit (`AV`) for high alarm
14	1 = Operator can change the limit (`AV`) for high warning
15	1 = Operator can change the limit (`AV`) for high tolerance
16	1 = Operator can change the limit (`AV`) for hysteresis
17	1 = Operator can change the limit (`AV`) for low alarm
18	1 = Operator can change the limit (`AV`) for low warning
19	1 = Operator can change the limit (`AV`) for low tolerance
20	1 = Operator can enable the bumpless switchover from external to internal `SP_TrkExt`
21	1 = Operator can change the internal setpoint `SP_Int`
22	1 = Operator can switch the setpoint to "external" `SP_ExtOp`
23	1 = Operator can switch the setpoint to "internal" `SP_IntOp`
24	1 = Operator can enable the setpoint's gradient limitation function `SP_RateOn`
25	1 = Operator can change the setpoint's high limit for the ramp `SP_UpRaLim`
26	1 = Operator can change the setpoint's low limit for the ramp `SP_DnRaLim`
27	1 = Operator can enable the setpoint ramp function `SP_RmpOn`
28	1 = Operator can switch between the time value or the value for the ramp `SP_RmpModTime`
29	1 = Operator can change the ramp time `SP_RmpTime`
30	1 = Operator can change the target setpoint `SP_RmpTarget` for the setpoint ramp
31	1 = Operator can enable rapid stop

The block has the following permissions for the `OS1Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can change the limit (`Rbk`) for high warning
2	Not used

Bit	Function
3	1 = Operator can change the limit (Rbk) for hysteresis
4	Not used
5	1 = Operator can change the limit (Rbk) for low warning
6	1 = Operator can change the limit (setpoint difference) ER_AH_Lim for the high alarm
7	1 = Operator can change the hysteresis (setpoint difference) ER_Hyst
8	1 = Operator can change the limit (setpoint difference) ER_AL_Lim for the low alarm
9	1 = Operator can change the simulation value SimRbk
10	1 = Operator can activate the device infeed DvFdOnMan
11	1 = Operator can deactivate the device infeed DvFdOffMan
12	1 =Operator can change the simulation value SimAV
13	1 = Operator can change the derivative gain parameter ER_AH_DFac
14	1 = Operator can change the derivative gain parameter ER_AL_DFac
15 - 19	Not used
20	1 = Operator can activate / deactivate messages via AV_AH_MsgEn
21	1 = Operator can activate / deactivate messages via AV_WH_MsgEn
22	1 = Operator can activate / deactivate messages via AV_TH_MsgEn
23	1 = Operator can activate / deactivate messages via AV_TL_MsgEn
24	1 = Operator can activate / deactivate messages via AV_WL_MsgEn
25	1 = Operator can activate / deactivate messages via AV_AL_MsgEn
26	1 = Operator can activate / deactivate messages via RbkWH_MsgEn
27	1 = Operator can activate / deactivate messages via RbkWL_MsgEn
28	1 = Operator can activate / deactivate messages via ER_AH_MsgEn
29	1 = Operator can activate / deactivate messages via ER_AL_MsgEn
30	1 = Operator can define the monitoring time for stopping
31	Not used

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

Restart lock after changing direction of rotation or switching off the motor

Use the input parameter IdleTime to enter a restart lock for changing the direction of rotation or restarting the motor. Use the Feature Bit Enabling direct changeover between forward and reverse (Page 145) to define how the change is to take place. When the "Stop" command is given, the motor goes immediately into Stop mode and IdleTime starts after the feedback (FbkFwd and FbkRev = 0) is given. The motor cannot be started again until the IdleTime has expired.

The IdleTime parameter can be set independently of the MonTiDynamic parameter.

Limit monitoring of an additional analog value

This block provides the standard function Limit monitoring of an additional analog value (Page 91).

Limit monitoring of the feedback

The block provides the standard function Limit monitoring of the feedback (Page 94). This limit monitoring is only active when the motor has started.

Limit monitoring with hysteresis

This block provides the standard function Limit monitoring with hysteresis (Page 97). It is performed via the input parameter `AV_Hyst`.

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

External/internal setpoint specification

The block provides the standard function Setpoint specification - internal/external (Page 127).

Setpoint adaption in motor stop and starting

With `Feature2.Bit7` = 1; Define the setpoint after stop and start of the motor (Page 183), the setpoint can be set to a stop value and after start of the motor, the setpoint can be set to a minimal value (`SP_LoLim`).

Setpoint limitation

Limit the setpoint using the parameters:

- `SP_HiLim` (top)
- `SP_LoLim` (bottom)

Limit violations are displayed at the `SP_HiAct` and `SP_LoAct` output parameters with a 1.

With `Feature2.Bit7` = 1; Define the setpoint after stop and start of the motor (Page 183), the lower limit value of the setpoint `SP_LoLim` will be limited to `SP_Off` and in case of violation, `SP_LoLim` will be written back to `SP_Off`.

Gradient limit of the setpoint

The block provides the standard function Gradient limit of the setpoint (Page 124).

Using setpoint ramp

The block provides the standard function Using setpoint ramp (Page 123).

Gear reduction or setpoint adaptation with factor

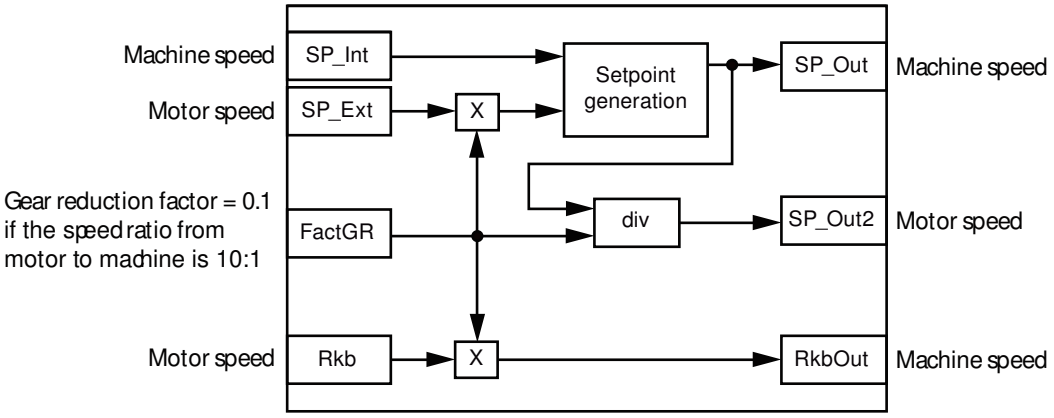
With the parameter `FactGR`, a factor can be entered for gear reduction. It affects the outputs `SP_Out`, `SP_Out2`, and `RbkOut`:

- `SP_Out` = `FactGR` * `SP_Ext` if the setpoint is external (`SP_ExtAct` = 1)
- `SP_Out2` parameter:
  - If `FactGR` <> 0.0 then `SP_Out2` = `SP_Out` / `FactGR`
  - If `FactGR` = 0.0 then `SP_Out2` = `SP_Out`

As a result, the readback value will also be adjusted:

`RbkOut` = `FactGR` * `Rbk`

Below is an example for the gear reduction from motor speed to machine speed in the ratio 10:1.



The factor `FactGR` can also be used for adapting the range in percentage.

Formation of the setpoint difference

The block always forms the setpoint difference:

`ER.Value` = SP`_Out` - `RbkOut.Value`.

Limit monitoring of the setpoint difference

The block provides the standard function Limit monitoring of setpoint, manipulated variable and control deviation (Page 95).

To transmit the messages, you need to enable `Feature` bit 5 Alarm setpoint difference (Page 170).

When the function activated, the message configuration should be adapted as follows:

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 7	Alarm - high	$$BlockComment$$ `ER` - high alarm limit `SP` - `Rbk` violated
	SIG 8	Alarm - low	$$BlockComment$$ `ER` - low alarm limit `SP` - `Rbk` violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Interlocks

This block provides the following interlocks:

- Activation enable
- Interlock without reset ("Interlock")
- Interlock with reset ("Protection")

Refer to the section Interlocks (Page 99) as well as Influence of the signal status on the interlock (Page 103).

Motor protection function

This block provides the standard function Motor protection function (Page 99).

Rapid stop for motors

This block provides the standard function Rapid stop for motors (Page 106).

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Resetting the block in case of interlocks or errors

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See MotSpdCL error handling (Page 1174).

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `Trip`
- `MonDynErr`
- `MonStaErr`
- `FaultExt`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `FbkFwdOut.ST`
- `FbkRevOut.ST`
- `LocalLi.ST`
- `FwdLocal.ST`
- `StopLocal.ST`
- `RevLocal.ST`
- `Trip.ST`
- `AV_Out.ST`
- `RbkOut.ST`
- `SP_Out.ST`
- `FwdChn.ST`
- `RevChn.ST`
- `SP_OutChn.ST`
- `FwdAut.ST` (only if `Feature2.Bit10` = 1)
- `RevAut.ST` (only if `Feature2.Bit10` = 1)

- StopAut.ST (only if Feature2.Bit10 = 1)
- SP_Ext.ST (only if Feature2.Bit10 = 1)

The signal status of the SP_Out output parameter is always equivalent to the signal status of input parameter SP_Ext or SP_Int, depending on how the setpoint is specified. If the internal setpoint SP_Int is used, the signal status is always output as 16#80.

In case of Feature.Bit10 = 1, SP_Ext.ST influences ST_Worst independent of setpoint specification.

Considering bad quality of automatic commands or external values

If the Feature2 bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position (motor stop) in the "Automatic" mode:

- FwdAut.ST
- RevAut.ST
- StopAut.ST
- SP_Ext.ST

Forcing operating modes

This block provides the standard function Forcing operating modes (Page 41).

The following states can be enforced:

- "Start in forward" (FwdForce)
- "Start in reverse" (RevForce)
- "Stop" (StopForce)

Note

The Feature bit Enabling direct changeover between forward and reverse (Page 145) for this block has no function with forced operating states. The motor can always be reversed directly.

Feedback monitoring

This block provides the standard function Monitoring the feedbacks (Page 97).

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Specify warning times for control functions

This block provides the standard function Specifying warning times for control functions at motors and valves (Page 51).

You can generate warning signals when, for example, motors are started. Warning signals can be generated in the following modes:

- Manual mode (Page 75) (`WarnTiMan` input parameter)Automatic mode (Page 75) (`WarnTiAut` input parameter)

You specify the warning times in seconds using the input parameters `WarnTiMan` and `WarnTiAut`. If, for example, a motor is started, then this is displayed at the output parameter with `WarnAct` = `1`. The motor then starts after the set warning time has expired and `WarnAct` then goes back to 0 .

A corresponding warning is not output if the warning times (`WarnTiMan` or `WarnTiAut`) are specified with a smaller value than the `SampleTime` parameter.

Simulating signals

This block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Additional value (`SimAV`, `SimAV_Li`)
- Position feedback (`SimRbk`, `SimRbkLi`)

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Specifying the display and operator input area for process values and setpoints during operation

This block features the standard function Display and control fields for process values and setpoints (Page 203).

Neutral position

This block provides the standard function Neutral position for motors, valves and controllers (Page 48).

Output signal as a pulse signal or static signal

This block provides the standard function Output signal as a static signal or pulse signal (Page 51).

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Delay alarm for control deviation at setpoint step changes

The block provides the standard function Delay alarm for control deviation at setpoint step changes (Page 186).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
4	Setting switch or button mode (Page 166)
5	Alarm setpoint difference (Page 170)
6	Ramp rate calculation (Page 178)
7	Enabling direct changeover between forward and reverse (Page 145)
8	Inverter enable (Page 186)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
12	Gradient limitation with time duration (Page 181)
13	Separate monitoring time for stopping the motor (Page 168)
14	Enabling rapid stop via faceplate (Page 167)
15	Frequency converter with separate device feed (Page 150)
16	Setpoint specification with separate display area and custom unit (Page 170)
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)
18	Activating error state for external process control error CSF (Page 150)
19	Reset even with locked state (Page 164)
20	Disable calculation of impulse control in LocalSetting 2 and 4 (Page 185)
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
27	Interlock display with LocalSetting 2 or 4 (Page 177)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

In pushbutton operation (Bit 4 = 0) , the automatic commands in "automatic" mode are latching; in other words, `FwdAut`, `RevAut`, `StopAut` can be reset to 0 after changing the control. In "manual" and "local" modes, however, the automatic commands are not saved and in the absence of automatic commands the automatic control is tracked.

In switching mode (Bit 4 = 1), control is selected with the static signals `FwdAut`, `RevAut`. If `FwdAut`, `RevAut` inputs are not set, the motor is stopped. Control via `StopAut` is not needed. If the "Activate command reset for control" function (Bit 3 = 1) is also enabled, the inputs `FwdAut`, `RevAut` are reset to the neutral position after evaluation in the block.

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
4	Setting switch or button mode for local commands (Page 180)
5	Evaluation of the signal status of the interlock signals (Page 141)
6	Operator can change the setpoint via faceplate also in the "Local" mode (Page 182)
7	Define the setpoint after stop and start of the motor (Page 183)
8	Forcing operating modes in the "Local" mode (Page 183)
10	Considering bad quality of automatic commands or external values (Page 185)

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` or `EventTs2In` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- `FwdMan`
- `RevMan`
- `StopMan`
- `RapidStp`

Controlling a device infeed or an inverter enable

The function is activated via `Feature` bit 15 Frequency converter with separate device feed (Page 150) and specified via `Feature` bit 8 Inverter enable (Page 186).

When the function is activated, the motor can be started in two steps:

- Switch on device infeed or inverter enable first, then start the motor

When the function is activated, the motor can also be started directly:

- The device infeed or the inverter enable is also activated when starting in forward or reverse. The frequency converter itself ensures the correct order.

The motor can now be stopped and restarted without having to switch off the device infeed or the inverter enable.

The activation of the device infeed or the inverter enable by starting the motor can be performed in the automatic, manual and local operating modes with local setting 1 or 3, or by forcing.

With local setting 2 or 4, the controls are tracked based on feedback signals regardless of the status of the device infeed or the inverter enable.

With motors without device infeed or inverter enable (`Feature` bit 15 = 0 default), the `DvFdAct.Value` control is always zero.

Separate activation/deactivation of the device infeed or the inverter enable is performed in manual mode using `DvFdOnMan` and `DvFdOffMan` via the standard view of the faceplate. In automatic mode, activation/deactivation is performed via the input signals `DvFdOnAut` or `DvFdOffAut`. The `Feature` bits 3 (activate reset of commands for control) and 4 (set switch or pushbutton operation) are also based on the automatic inputs `DvFdOnAut` and `DvFdOffAut`.

The following table shows the effects on MotSpdCL block when parameterizing device infeed and inverter enable:

MotSpdCL specification	Effect on MotSpdCL block when parameterizing device infeed **`Feature` bit 8 = 0**	Effect on MotSpdCL block when parameterizing inverter enable **`Feature` bit 8 = 1**
Motor start - dependent (`Fwd` = 1 or `Rev` = 1)	Device infeed is activated (`DvFdAct` = 1)	Inverter enable remains unchanged
Motor stop (`Fwd` = 0 and `Rev` = 0)	Device infeed remains unchanged	Inverter enable remains unchanged
Activate device infeed or inverter enable (`DvFdAct` = 1)	Motor control remains unchanged	Motor control remains unchanged
Deactivate device infeed or inverter enable (`DvFdAct` = 0)	Motor control is reset (`Fwd` = 0, `Rev` = 0)	Motor control remains unchanged
Monitoring the feedback signals (`FbkFwd`, `FbkRev`)	Feedback signals are monitored for motor controls (`Fwd`, `Rev`)	Feedback signals are monitored for motor controls and the inverter enable: (`Fwd` and `DvFdAct`), (`Rev` and `DvFdAct`)
Monitoring fault, external fault, motor protection, or interlock	Motor control is reset (`Fwd` = 0, `Rev` = 0) Device infeed remains unchanged	Motor control is reset (`Fwd` = 0, `Rev` = 0) Inverter enable remains unchanged
Limit value monitoring of the feedback Limit value monitoring of the setpoint difference	Only active when: `Fwd` = 1 or `Rev` = 1	Only active if the inverter enable is also activated: `Fwd` = 1 or `Rev` = 1; And `DvFdAct` = 1

Commands for starting the motor have higher priority than commands for a device infeed or the inverter enable.

⚠ DANGER

Separate switching on/off of the device infeed

With switch mode (`Feature` Bit 4 = 1) and deactivated control reset (`Feature` Bit 3 = 0), the `DvFdOnAut` input is also set for controlling the device infeed or the inverter enable when starting in forward or in reverse in automatic mode.

See also

MotSpdCL block diagram (Page 1191)

MotSpdCL I/Os (Page 1179)

MotSpdCL messaging (Page 1177)

MotSpdCL modes (Page 1161)

Description of MotSpdCL (Page 1156)

EventTs functions (Page 1634)

7.5.4 MotSpdCL error handling

Error handling of MotSpdCL

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Control system fault (CSF)
- Invalid input signals

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
41	The value for the `LocalSetting` I/O is not within the approved limit of 0 to 4.
42	`LocalSetting` = 0 or `LocalSetting` = 3or `LocalSetting` = 4 and `LocalLi` = 1

Error number	Meaning of the error number
51	FwdLocal = 1 and StopLocal = 1 RevLocal = 1 and StopLocal = 1 FwdLocal = 1 and RevLocal = 1 FwdAut = 1 and StopAut = 1 RevAut = 1 and StopAut = 1 FwdAut = 1 and RevAut = 1 AutModLi = 1 and ManModLi = 1 FwdForce = 1 and StopForce = 1 RevForce = 1 and StopForce = 1 FwdForce = 1 and RevForce = 1 SP_LiOp = 1 and SP_IntLi = 1 and SP_ExtLi = 1 DvFdOnMan and DvFdOffMan and feature bit 15 DvFdOnAut and DvFdOffAut and feature bit 15
52	LocalAct = 1 and LocalSetting = 2 or 4 and SimOn = 1

Note

The relationship of the pushbutton/switch mode feature bit 4 is not taken into account

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

Invalid input signals

This error can be output by the block, see the section Error handling (Page 119).

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signals	Control reaction with Feature2 bit 3 =1
Local: `Localsetting` = 1 or `Localsetting` = 3	Pushbutton operation for local mode (`Feature2` bit 4 = 0): `FwdLocal` = 1 and `RevLocal` = 1 or `FwdLocal` = 1 and `StopLocal` = 1 or `StopLocal` = 1 and `RevLocal` = 1 Switching mode (`Feature2` bit 4 = 1): `FwdLocal` = 1 and `RevLocal` = 1	Motor is stopped
Local: `Localsetting` = 1 or `Localsetting` = 3 and forcing	`FwdForce` = 1 and `RevForce` = 1 or `FwdForce` = 1 and `StopForce` = 1 or `StopForce` = 1 and `RevForce` = 1	
Forcing and no "local mode"	`FwdForce` = 1 and `RevForce` = 1 or `FwdForce` = 1 and `StopForce` = 1 or `StopForce` = 1 and `RevForce` = 1	
"Automatic mode" and no forcing	Pushbutton operation (`Feature` bit 4 = 0): `FwdAut` = 1 and `RevAut` = 1 or `FwdAut` = 1 and `StopAut` = 1 or `StopAut` = 1 and `RevAut` = 1 Switching mode (`Feature`bit 4 = 1): `FwdAut` = 1 and `RevAut` = 1	
"Manual mode" and no forcing	`FwdMan` = 1 and `RevMan` = 1 or `FwdMan` = 1 and `StopMan` = 1 or `StopMan` = 1 and `RevMan` = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

MotSpdCL block diagram (Page 1191)

MotSpdCL I/Os (Page 1179)

MotSpdCL messaging (Page 1177)

MotSpdCL functions (Page 1162)

MotSpdCL modes (Page 1161)

Description of MotSpdCL (Page 1156)

7.5.5 MotSpdCL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	AS process control message - fault	$$BlockComment$$ Motor feedback error
	SIG 2	AS process control message - fault	$$BlockComment$$ Motor protection triggered
	SIG 3	AS process control message - fault	$$BlockComment$$ External error has occurred
	SIG 4	Warning - high	$$BlockComment$$ Rbk - high warning limit violated
	SIG 5	Warning - low	$$BlockComment$$ Rbk - low warning limit violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter CSF. If it changes to CSF = 1,, a process control fault is triggered (MsgEvId1, SIG 3).

Instance-specific messages

You can use up to three instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 6	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 7	Alarm - high	$$BlockComment$$ ER - high alarm limit violated
	SIG 8	Alarm - low	$$BlockComment$$ ER - low alarm limit violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance **MsgEvId1**

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchID
4	ExtVa104
5	ExtVa105
6	ExtVa106
7	ExtVa107
8	ExtVa108
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters ExtVa104 ... ExtVa108 , and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

MotSpdCL block diagram (Page 1191)

MotSpdCL I/Os (Page 1179)

MotSpdCL error handling (Page 1174)

MotSpdCL functions (Page 1162)

MotSpdCL modes (Page 1161)

Description of MotSpdCL (Page 1156)

7.5.6 MotSpdCL I/Os

I/Os of MotSpdCL

Input parameters

Parameter	Description	Type	Default
AutModLi*	1 = "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 0)	BOOL	0
AV	Input additional analog value, to be connected to AV_Tech of the AV block	ANY	
AV_AH_Lim	Limit high alarm	REAL	95.0
AV_AL_Lim	Limit low alarm	REAL	5.0
AV_Hyst	Hysteresis for alarm, warning and tolerance limits	REAL	1.0
AV_TH_Lim	Limit high tolerance	REAL	85.0
AV_TL_Lim	Limit low tolerance	REAL	15.0
AV_WH_Lim	Limit high warning	REAL	90.0
AV_WL_Lim	Limit low warning	REAL	10.0
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Bypassing interlock in "local mode" and in "simulation"	BOOL	0
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DvFdOnAut*	1 = Activation of device infeed in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DvFdOffAut*	1 = Deactivation of device infeed in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DvFdOnMan*	1 = Activation of device infeed in "manual mode"	BOOL	0
DvFdOffMan*	1 = Deactivation of device infeed in "manual mode"	BOOL	0
EN	1 = Called block will be processed	BOOL	1
ER_A_DC*	Reserved	REAL	0.0
ER_A_DG*	Reserved	REAL	0.0

Parameter	Description	Type	Default
ER_AH_En	1 = Activate alarm (high) for setpoint difference monitoring	BOOL	1
ER_AH_DFac*	Delay factor at the positive setpoint step changes for incoming alarms at the error signal monitoring ER_AH_Lim	REAL	0.0
ER_AH_Lim	Alarm limit (high) for setpoint difference monitoring	REAL	100.0
ER_AH_MsgEn	1 = Activate messages for alarm (high) for setpoint difference monitoring Messages are only output if you have also enabled the Feature bit 5 Alarm setpoint difference (Page 170).	BOOL	1
ER_AL_DFac*	Delay factor at the negative setpoint step changes for incoming alarms at the error signal monitoring ER_AL_Lim	REAL	0.0
ER_AL_En	1 = Activate alarm (low) for setpoint difference monitoring	BOOL	1
ER_AL_Lim	Alarm limit (low) for setpoint difference monitoring	REAL	-100.0
ER_AL_MsgEn	1 = Activate messages for alarm (low) for setpoint difference monitoring Messages are only output if you have also enabled the Feature bit 5 Alarm setpoint difference (Page 170).	BOOL	1
ER_Hyst	Alarm hysteresis for error signal	REAL	1.0
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
EventTs2In	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTs2In input parameter serves to interconnect the EventTsOut output parameter of the EventTs,Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	

Parameter	Description	Type	Default
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
FactGR	Gear reducing factor	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkFwd	1 = Feedback for forward mode is present	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkRev	1 = Feedback for reverse mode is present	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Feature	I/O for additional functions (Page 1162) 1 = Separate monitoring time for stopping the motor 1 = Setpoint with own scale and unit of the parameter	STRUCT • Bit 0: BOOL • ... • Bit 13: BOOL • Bit 16: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0 • 0 • 0 • 0
Feature2	I/O for additional functions (Page 1162)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0

Parameter	Description	Type	Default
FwdAut*	1 = Activation of forward motor operation in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FwdChnST	Signal status of output channel for Fwd Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FwdForce	1 = Forcing activation of motor forward mode	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FwdLocal	1 = Activation of forward motor operation in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FwdMan*	1 = Activation of forward motor operation in "manual mode"	BOOL	0
IdleTime*	Wait time for change of direction or restart in [s]	REAL	5.0
Intlock	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is active	BOOL	1
LocalLi	1 = Activate "local mode" via plant signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalOp*	1 = "Local mode" via operator	BOOL	0
LocalSetting	Characteristics of Local mode (Page 79)	INT	0
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
ModLiOp	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Monitor	1 = Feedback monitoring	BOOL	1
MonTiDynamic*	Monitoring time for feedback errors or feedback start error after successful operation in [s]	REAL	3.0
MonTiDyStop*	Monitoring time for feedback stop errors after successful operation in [s]	REAL	3.0
MonTiStatic*	Monitoring time for feedback errors without operation in [s]	REAL	3.0

Parameter	Description	Type	Default
MS_RelOp*	1= Release for maintenance via OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = In use by a batch	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 1162) 1 = Operator can set or change the monitoring time for "Control: Start"	STRUCT • Bit 0: BOOL • Bit8: BOOL • Bit 10: BOOL • Bit 31: BOOL	- • 1 • 1 • 1 • 1
OS1Perm	I/O for operator permissions (Page 1162) 1 = Operator can set or change the monitoring time for "Control: Stop"	STRUCT • Bit 0: BOOL • Bit 9: BOOL • Bit 30: BOOL • Bit 31: BOOL	- • 1 • 1 • 1 • 1
Permit	1 = OS activation enable for motor 0 = No OS release for energizing motor	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Perm_En	1 = Activation enable (enable, Permit parameter) is active	BOOL	1
Protect	0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block 1 = Protective interlocking not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Prot_En	1 = Protective interlock (protection, Protect parameter) is active	BOOL	1
PulseWidth*	Pulse width of control signal [s]	REAL	3.0
RapidStp*	Rapid stop for the motor: 0 = Motor On 1 = Motor Off	BOOL	0

Parameter	Description	Type	Default
Rbk	Position feedback for display on OS	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RbkW_DC*	Delay time for incoming warnings [s]	REAL	0.0
RbkW_DG*	Delay time for outgoing warnings [s]	REAL	0.0
RbkHyst	Hysteresis for warning limits	REAL	1.0
RbkOpScale	Limit for scale in bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
RbkUnit	Unit of measure for additional analog value	INT	0
RbkWH_En	1 = Enable high warning	BOOL	1
RbkWH_Lim	Limit high warning	REAL	90.0
RbkWH_MsgEn	1 = Enable high warning message	BOOL	1
RbkWL_En	1 = Enable low warning	BOOL	1
RbkWL_Lim	Limit low warning	REAL	10.0
RbkWL_MsgEn	1 = Enable low warning message	BOOL	1
RevAut*	1 = Activation of reverse motor operation in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RevChnST	Signal status of output channel for Rev Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
RevForce	1 = Force activation of reverse motor operation	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RevLocal	1 = Activation of reverse motor operation in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RevMan*	1 = Activation of reverse motor operation in "manual mode"	BOOL	0
RstLi*	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	

Parameter	Description	Type	Default
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	
SimAV*	Additional value used for SimOn = 1	REAL	0.0
SimAV_Li	Additional analog value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimRbk*	Position feedback used for SimOn = 1	REAL	0.0
SimRbkLi	Position feedback used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0 • 16#80
SP_DnRaLim	Limit (low) for the gradient of the setpoint [RbkUnit/s]	REAL	100.0
SP_Ext	external setpoint - (to interconnection)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_ExtLi	1 = Select external setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtOp*	1 = Select external setpoint (via operator)	BOOL	0
SP_HiLim	Limit (high) of setpoint	REAL	100.0
SP_LoLim*	Limit (low) of setpoint; if Feature2.Bit7 = 1 then SP_LoLim is limited and written back to SP_Off.	REAL	0.0
SP_Int*	Internal setpoint for operation	REAL	1.0
SP_IntLi	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_IntOp*	1 = Select internal setpoint (via operator)	BOOL	1
SP_LiOp	Select setpoint source (internal/external): 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_Off	Setpoint if the motor is off	REAL	0.0
SP_OpScale	SP bar limit display for OS	ScaVal	

Parameter	Description	Type	Default
SP_OutChnST	Signal status of output channel SP_Out Should be connected to an output channel block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
SP_RateOn*	1 = Activate limitation of setpoint gradients	BOOL	0
SP_RmpModTime	1 = Use time (SP_RmpTime) for setpoint ramp 0 = Use gradient	BOOL	0
SP_RmpOn*	1 = Activate setpoint ramp to target setpoint SP_RmpTarget	BOOL	0
SP_RmpTarget*	Target setpoint for setpoint ramp	REAL	0.0
SP_RmpTime*	Time for setpoint ramp [s] from current SP up to SP_RmpTarget	REAL	0.0
SP_TrkExt	1 = Bumpless switchover from external to internal setpoint active	BOOL	0
SP_Unit	Operator inputs	INT	0
SP_UpRaLim	Gradient limit (high) for the setpoint [RbkUnit/s]	REAL	100.0
StepNo	Batch step number	DWORD	16#00000000
StopAut*	1 = Stopping the motor in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopForce	1 = Force motor stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopLocal	1 = Stopping the motor in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopMan*	1 = Stopping the motor in "manual mode"	BOOL	0
TimeFactor	Time unit: 0 = Seconds 1 = Minutes 2 = Hours	INT	0
Trip	1 = Motor is in "good" state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA1unit	Unit of measure for analog auxiliary value 1	INT	0

Parameter	Description	Type	Default
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA2unit	Unit of measure for analog auxiliary value 2	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
WarnTiAut*	Prewarning of motor start in "automatic mode" in [s]	REAL	0.0
WarnTiMan*	Prewarning of motor start in "manual mode" in [s]	REAL	0.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80
AV_OpScale	Limit for scale in AV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
AV_Out	Output additional analog value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
AV_Unit	Unit of measure for additional analog value	INT	0
DvFdAct	1 = Device infeed enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CurrMon	Current monitoring time [s]	DINT	0
ENO	1 = Block algorithm completed without errors	BOOL	0
ER	Error signal	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ER_A_DCOut	Effective delay time [s] for incoming alarms at the error signal monitoring	REAL	0.0

Parameter	Description	Type	Default
ER_AH_Act	1 = Alarm limit (high) for control deviation violated.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ER_AL_Act	1 = Alarm limit (low) for control deviation violated.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see MotSpdCL error handling (Page 1174)	INT	-1
FbkFwdOut	Feedback: 1 = Forward operation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkRevOut	Feedback: 1 = Reverse operation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Fwd	1 = Control of motor forward	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalAct	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock (Intlock, Permit, Protect) or Trip is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MonDynErr	1 = Feedback error or feedback start error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonDynStopErr	1 = Feedback stop error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonStaErr	1 = Feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
OosAct	1 = Block is "out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS1PermOut	Display of OS1Perm	DWORD	16#FFFFFFFF
OS1PermLog	Display of OS1Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
P_Fwd	1 = Pulse signal for starting the motor in forward	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Rev	1 = Pulse signal for starting the motor in reverse	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Stop	0 = Pulse signal for stopping the motor	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
R_StpAct	1 = Rapid stop of the motor is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkOut	Output of readback value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
RbkWH_Act	1 = Warning (high) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkWL_Act	1 = Warning (low) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Rev	1 = Control of motor: Reverse	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RunFwd	1 = Motor is running forwards	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RunRev	1 = Motor is running in reverse	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_DnRaAct	1 = Negative gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtAct	1 = External setpoint is active 0 = Internal setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_HiAct	1 = Limit (high) for setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_LoAct	1 = Limit (low) for setpoint has been reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_Out	Setpoint used by controller	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
SP_Out2	Additional setpoint, without gear reducing factor	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_RateTarget	Target setpoint for the gradient limitation	REAL	0.0
SP_UpRaAct	1 = Positive gradient limiting of setpoint is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_RemRT	Remaining ramp time of the setpoint	REAL	0.0
Starting	1 = Motor will start	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Status1	Status word 1 (Page 1156)	DWORD	16#00000000
Status2	Status word 2 (Page 1156)	DWORD	16#00000000
Status3	Status word 3 (Page 1156)	DWORD	16#00000000
Status4	Status word 4 (Page 1156)	DWORD	16#00000000
ST_Worst	Worst signal status	BYTE	16#80
Stop	1 = Motor is stopping	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Stopping	1 = Motor will stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
WarnAct	1 = Prewarning for motor start active (parameters WarnTiAut and WarnTiMan)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

MotSpdCL block diagram (Page 1191)

MotSpdCL messaging (Page 1177)

MotSpdCL modes (Page 1161)

Error handling (Page 119)

Setpoint specification with separate display area and custom unit (Page 170)

7.5.7 MotSpdCL block diagram

MotSpdCL block diagram

A block diagram is not provided for this block.

See also

MotSpdCL I/Os (Page 1179)
MotSpdCL messaging (Page 1177)
MotSpdCL error handling (Page 1174)
MotSpdCL functions (Page 1162)
MotSpdCL modes (Page 1161)
Description of MotSpdCL (Page 1156)

7.5.8 Operator control and monitoring

7.5.8.1 MotSpdCL views

Views of the MotSpdCL block

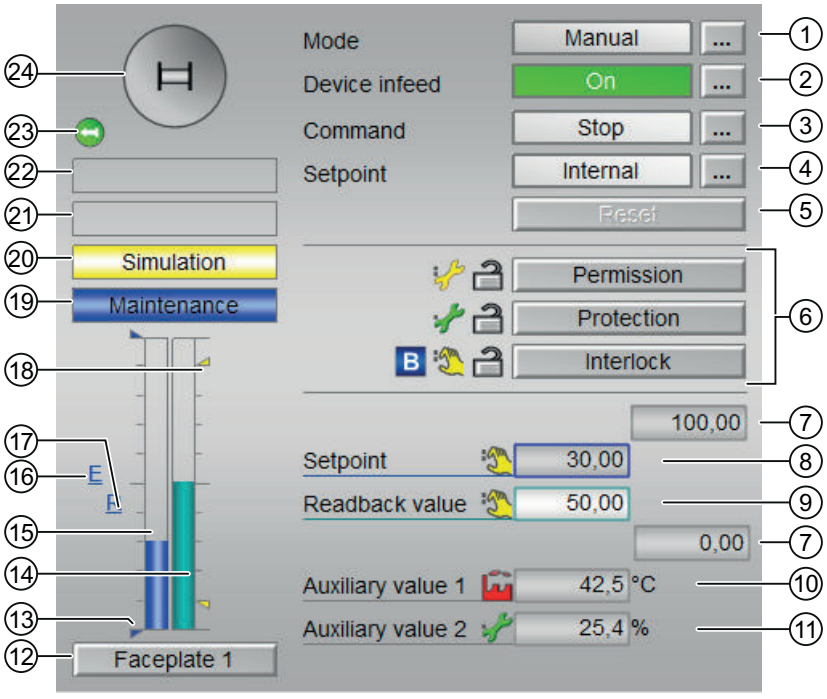
The block MotSpdCL provides the following views:

- MotSpdCL standard view (Page 1193)
- Alarm view (Page 296)
- Limit view of motors (Page 288)
- MotSpdCL limit view for readback values (Page 1202)
- Trend view (Page 299)
- Ramp view (Page 294)
- MotSpdCL parameter view (Page 1204)
- MotSpdCL preview (Page 1199)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MotSpdCL (Page 1207)

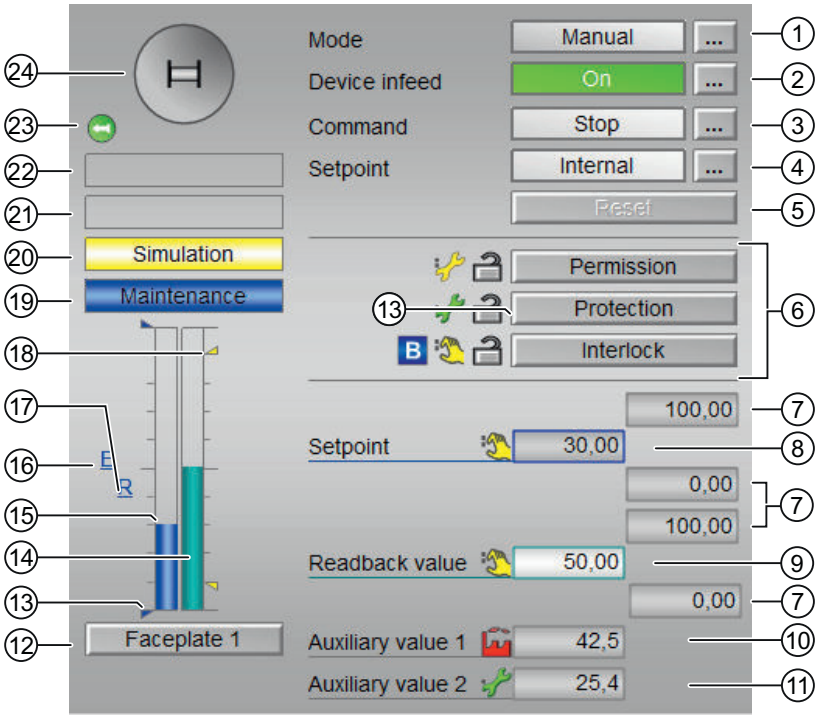
Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.5.8.2 MotSpdCL standard view

MotSpdCL standard view



Feature bit 16 = 0: Setpoint and readback value have the same area of the readback value



Feature bit 16 = 1: Setpoint and readback value have separate areas

(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

See also the section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Separate switching on/off of the device infeed or the inverter enable

This area shows you the default operating state for the device infeed or the inverter enable. The following states can be shown and executed here:

- "On"
- "Off"

If the inverter enable is parameterized (`Feature` bit 15 = 1 and `Feature` bit 8 = 1), the text "Device infeed" is replaced by "Inverter enable".

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find additional information on this in the section Labeling of buttons and text (Page 205).

(3) Starting and stopping the motor

This area shows you the default operating state for the variable motor. The following states can be shown and executed here:

- "Start |→"
- "Start ←|"
- "Stop"
- "Rapid stop"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(4) Switching the setpoint internal/external

This area shows how to specify the setpoint. The setpoint can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the setpoint specification.

You can find additional information on this in the section Setpoint specification - internal/ external (Page 127).

(5) Resetting the block

Click "Reset" for interlocks or errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(6) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the section Interlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

(7) High and low scale range for the setpoint

This area is already set and cannot be changed.

(8) Display and change the setpoint including signal status

This area shows the current setpoint with the corresponding signal status.

Refer to the Changing values (Page 253) section for information on changing the setpoint. The setpoint specification also needs to be set to "Internal" for this block.

(9) Displaying the readback value

This area shows you the current readback value with the corresponding signal status.

(10) and (11) Display of auxiliary values

This display is only visible when the corresponding block input is connected.

You can use this area to display two auxiliary values that have been configured in the engineering system (ES). You can find additional information on this in the section Displaying auxiliary values (Page 207).

(12) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(13) Displaying the limits

These triangles show the `SP_HiLim` and `SP_LoLim` setpoint limits configured in the engineering system (ES).

(14) Bar graph for the readback value

This area shows you the current readback value in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(15) Bar graph for the setpoint

This area shows the current setpoint in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(16) Display of external setpoint

This display [E] is only visible when you have selected "Internal" setpoint specification. It shows the external setpoint that would apply if you were to change the setpoint specification to "external".

(17) Display for the target setpoint of the setpoint ramp

This display [R] shows the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

(18) Limit display

These colored triangles show you the configured limits in the respective bar graph.

(19) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this topic is available in sections Release for maintenance (Page 64) Display area for block states.

(20) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"
- "Delay"

You will find more detailed information on this in the sections Simulating signals (Page 58) and Display of delay times (Page 250).

(21) Display area for block states

This area provides additional information on the operating state of the block:

- "Motor protection"
- "External error"
- "Status error"
- "Control error"
- "Invalid signal"
- "Changeover error"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97) , Error handling (Page 119) (section "Invalid input signals" and "Mode switchover error") and Motor protection function (Page 99).

(22) Display area for block states

This area provides additional information on the operating state of the block:

- "Forced stop"
- "Forced start |→"
- "Forced start ←|"
- "Request 0/1": A reset to "automatic mode" is expected.
- "SP ramp active"

You can find additional information on this in the section Forcing operating modes (Page 41).

(23) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the motor would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

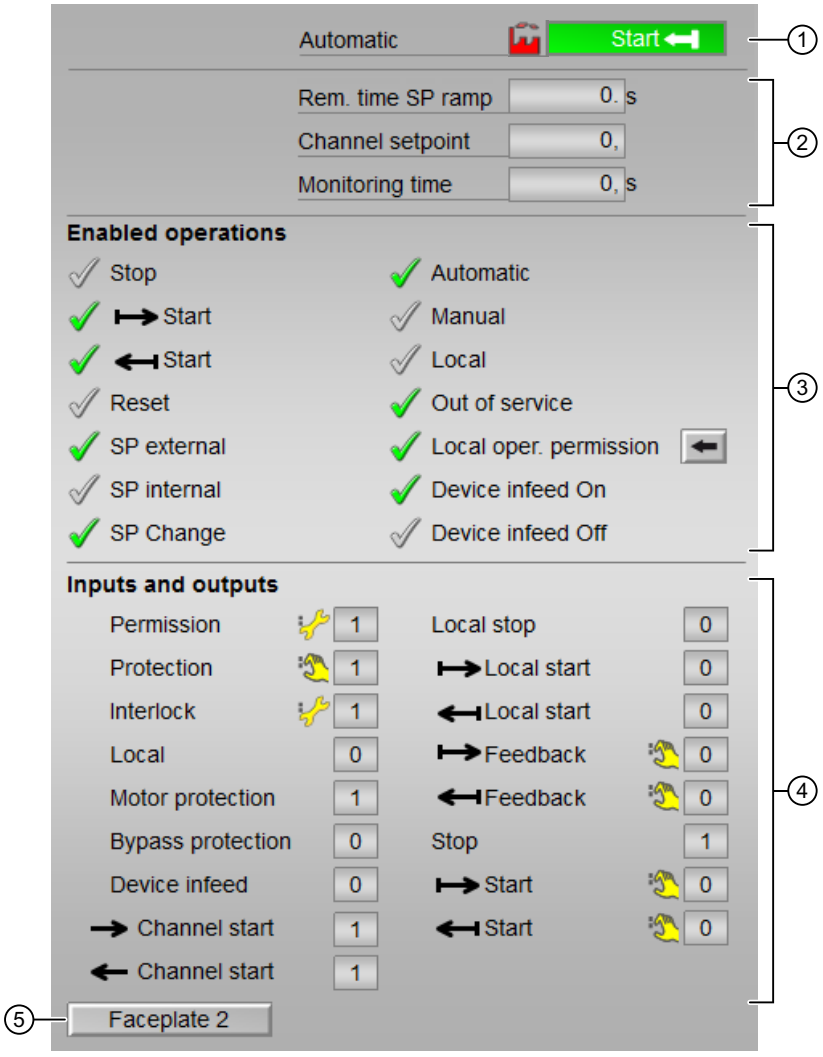
(24) Status display of the motor

The current status of the motor is graphically displayed here.

You can find more information about this in section Block icon for MotSpdCL (Page 1207)

7.5.8.3 MotSpdCL preview

Preview of MotSpdCL



(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `FwdAut`
- `RevAut`
- `StopAut`
- `SP_Ext`

(2) Preview area

This area shows you a preview of the following values:

- "Rem. time SP ramp" : Remaining time to reach the ramp target value.
- "Channel Setpoint": Setpoint from the output channel block.
- "Monitoring time": Display of the current monitoring time.

(3) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Stop": You can stop the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205)
- "Start |→ ": You can start the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205) .
- "Start ←|": You can start the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205) .
- "Reset": You can reset the motor after interlocks or errors.
- "SP external": You can feedforward the external setpoint.
- "SP internal": You can feedforward the internal setpoint.
- "Change SP": You can change the setpoint.
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Local": You can switch to "local mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248) .
- "Device infeed On": You can activate the device infeed
- "Device infeed Off": You can deactivate the device infeed

(4) Displaying current control signals

This area shows the most important parameters for this block with the current selection:

- "Permission":
  This display is only visible when the corresponding block input is connected.
  - 0 = No OS release for energizing motor
  - 1 = Enable for "opening"/"closing" from the neutral position
- "Protection":
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state
- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Motor protection": 1 = Motor is in "good" state
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"
- "Local stop": 1= Block is operated in "local mode"
- "Local start |→": 1= Block is operated in a controlled manner in "local mode"
- "Local start ←|": 1= Block is operated in a controlled manner in "local mode"
- "Feedback |→": 1 = Motor has started and is running
- Feedback←|: 1 = Motor has started and is running
- "Stop": 1 = Stop motor
- "Start | →" 1 = Start motor
- "Start ←|": 1 = Start motor
- "Device infeed": 1 = Enable device infeed
- "Channel Start →":Signal from the output channel block for "Start"
- "Channel Start ←":Signal from the output channel block for "Start"

(5) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

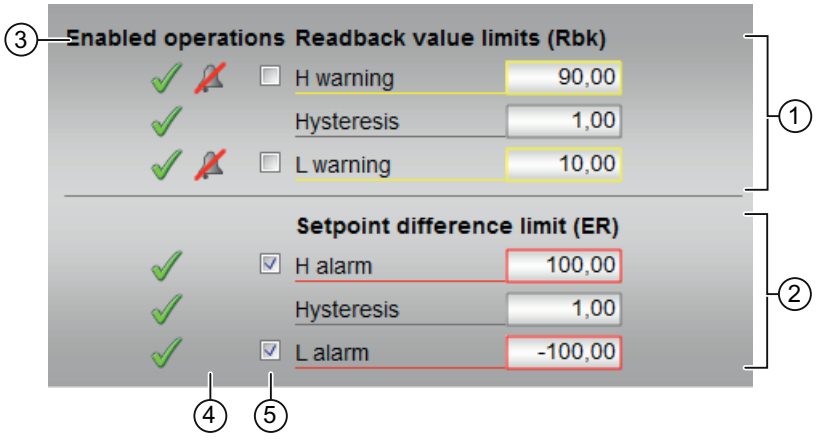
7.5.8.4 MotSpdCL limit view for readback values

Limit view for readback values of MotSpdCL

Several values are set in this view by default:

- Readback value limits

The toolbars of the faceplate and the block icon indicate when the limits are reached or violated.



(1) Displaying and changing the limits for the readback value

In this area, you can enter the limits for the readback value. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H warning": Warning high
- "Hysteresis"
- "L warning": Warning low

(2) Displaying and changing the limits for the setpoint difference

In this area, you can enter the limits for the setpoint difference. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "Hysteresis"
- "L alarm": Alarm low

(3) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

(4) "Message suppression/delay"

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

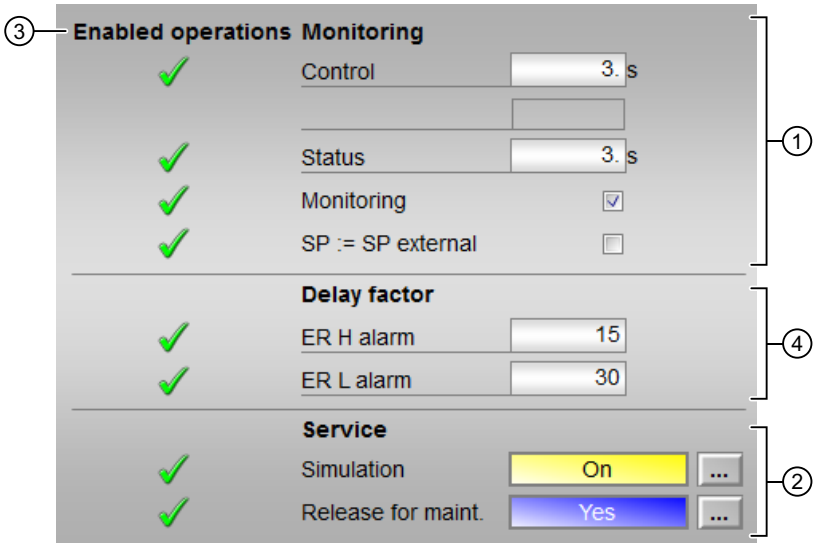
Alarm delays are also displayed in this position; for more on this see section Area of application of the alarm delays (Page 195).

(5) Suppress messages

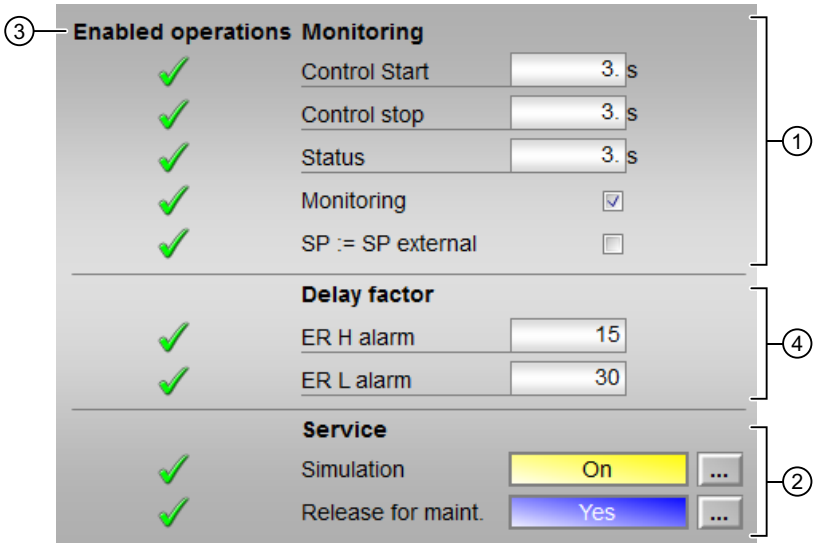
You can enable / disable messages by setting the check mark.

7.5.8.5 MotSpdCL parameter view

Parameter view of MotSpdCL



Parameter view of MotSpdCL with `Feature` bit 13 = 0



Parameter view of MotSpdCL with `Feature` bit 13 = 1

(1) Monitoring

In this area, you change parameters and therefore influence the motor. Refer to the Changing values (Page 253) section for more on this.

You can influence the following parameters:

- Feature bit **13 = 0**
  - "Control": Monitoring time during startup and shutdown of the motor (dynamic)
- Feature bit **13 = 1**
  - "Control stop": Monitoring time during shutdown of the motor (dynamic)
  - "Control start": Monitoring time during startup of the motor (dynamic)

"Status" monitoring

You can monitor time during permanent operation of the motor (static).

Enable monitoring

You can enable monitoring by selecting the check box (☑)

You can find additional information on this in the section Monitoring the feedbacks (Page 97).

Activating bumpless switchover

"SP := SP external": ☑ Bumpless switchover of setpoint from external to internal. The internal setpoint is tracked to the external one.

(2) Service

You can select the following functions in this area:

- Simulation
- Release for maintenance

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Simulating signals (Page 58)
- Release for maintenance (Page 64)

(3) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

(4) Delay factor

In this area, you can change the following parameters:

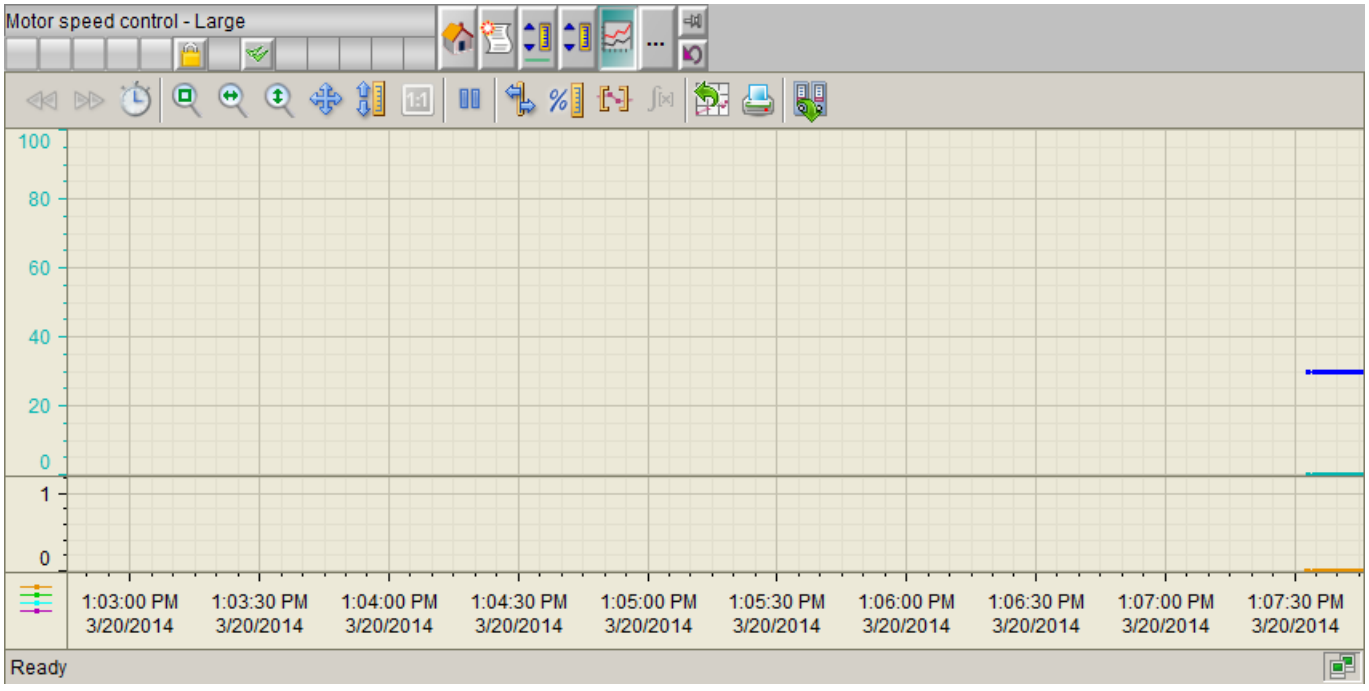
- "ER H alarm": Delay factor at the positive setpoint step changes for incoming alarms at the control deviation monitoring `ER_AH_Lim`.
- "ER L alarm": Delay factor at the negative setpoint step changes for incoming alarms at the control deviation monitoring `ER_AL_Lim`.

7.5.8.6 MotSpdCL trend view

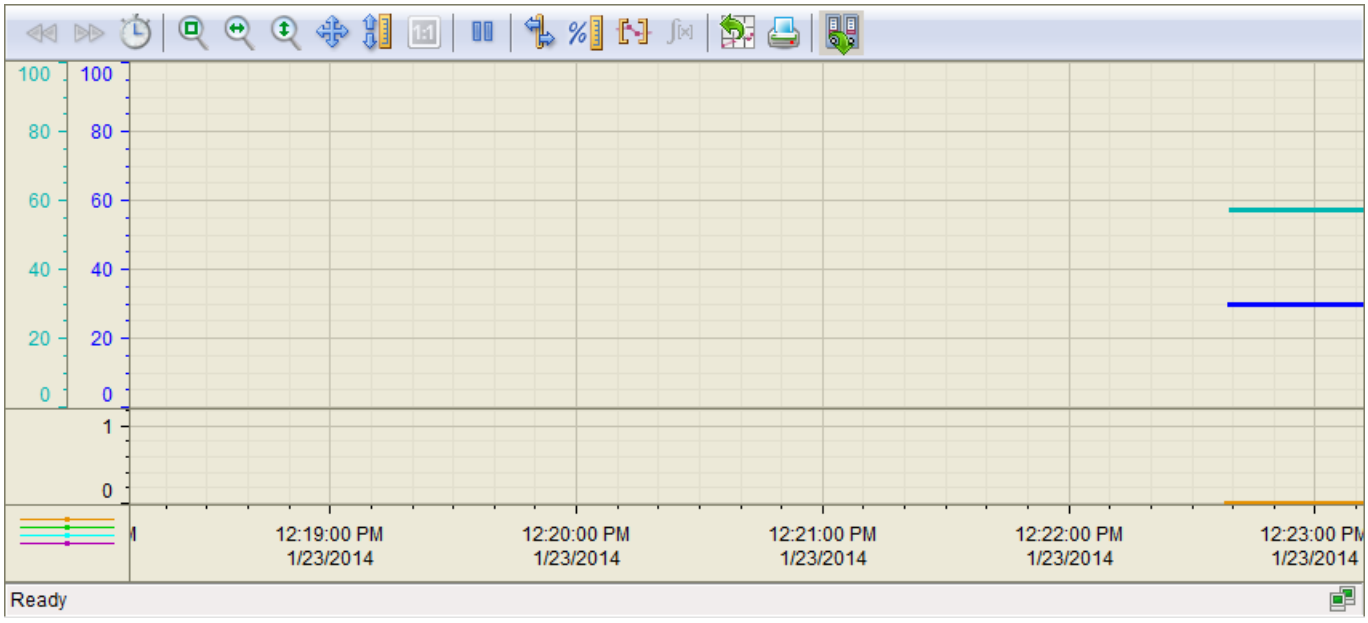
Trend view of MotSpdCL

You can find general information in section Trend view (Page 299).

Depending on feature bit 16 (process value with separate scale range) either one or two value axes are shown in the trend view.



- Process value with separate scale range



- Process value without separate scale range

Note

This function only has an effect when `SP_Out` is configured in "TrendConfiguration1" and `PV_In` is configured in "TrendConfiguration2" in the block icon (default).

7.5.8.7 Block icon for MotSpdCL

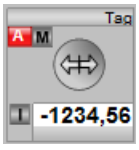
Block icons for MotSpdCL

A variety of block icons are available with the following functions:

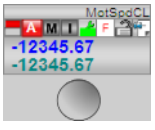
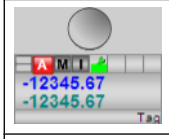
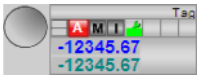
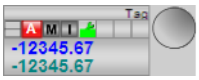
- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes
- Internal and external setpoint specification
- Signal status, release for maintenance
- Displays for bypassing interlocks
- Interlocks
- Memo display
- Motor state display

- Setpoint (blue, with decimal places)
- Feedback value (green, with decimal places)

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Direction display inverted to symbol 1
	4	Direction display inverted to symbol 2
	5	"M" symbol with small direction display
	6	"M" symbol with small direction display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	

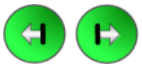
Icons	Selection of the block icon in CFC	Special features
	5	
	6	
	7	
	8	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Motor state display

The following motor states are shown here:

Icon	Meaning
	Motor started (motor symbol changes)
	The motor is running
	Motor stopped (motor symbol changes)
	Motor idle
	Error at motor (monitoring error, Motor protection)
	Motor out of service

7.6 MotSpdL - Two-speed motor

7.6.1 Description of MotSpdL

Object name (type + number) and family

Type + number: FB 1856

Family: Drives

Area of application for MotSpdL

The block is used for the following applications:

- Control of two-speed motors

How it works

The block is used to control motors with two speeds. Various inputs are available for controlling the motor. You can add monitoring of a maximum of two feedbacks produced by the contactor relay and a switching mode with which the speed change can be performed.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB3x). The block is also installed automatically in the startup OB (OB100).

Further addressing is not required.

For the MotSpdL block, the Advanced Process Library contains a template for process tag types as an example with an application scenario for this block.

Example of process tag types:

- Two-speed motor (Motor2Speed) (Page 2336)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section MotSpdL I/Os (Page 1226).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`

Status bit	Parameter
2	SimOn
3	OosAct.Value
4	OosLi.Value
5	0 = ManAct.Value 1 = AutoAct.Value
6	LocalAct.Value
7	0: Open padlock in the block icon 1: Closed padlock in the block icon
8	Spd1.Value
9	Motor is stopped
10	Spd2.Value
11	MonStaErr.Value
12	MonDynErr.Value
13	BypProt
14	Invalid signal status
15	Mode switchover error
16	1 = Intlock is active
17	1 = Permit is active
18	1 = Protect is active
19	Trip.Value
20	Display „Forced speed1“
21	Display „Forced stop“
22	Display „Forced speed2“
23	"Interlock" button is enabled
24	Reset request in automatic preview
25	WarnAct:Value, IdleTime or SwOverTi active
26	Bypass information
27	Automatic preview for (Speed1)
28	Automatic preview for (Stop)
29	Automatic preview for (Speed2)
30	"Permission" button is enabled
31	"Protection" button is enabled

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock
1	AV_AH_Act.Value
2	AV_WH_Act.Value
3	AV_TH_Act.Value

Status bit	Parameter
4	AV_TL_Act.Value
5	AV_WL_Act.Value
6	AV_AL_Act.Value
7	AV_AH_En
8	AV_WH_En
9	AV_TH_En
10	AV_TL_En
11	AV_WL_En
12	AV_AL_En
13	AV_AH_MsgEn
14	AV_WH_MsgEn
15	AV_TH_MsgEn
16	AV_TL_MsgEn
17	AV_WL_MsgEn
18	AV_AL_MsgEn
19	1 = No impact of input signals on "local mode" with LocalSetting = 2 and LocalSetting = 4
20	Motor is stopped
21	Motor stops at speed 1
22	Motor stops at speed 2
23	Motor starts at speed 1
24	Motor running at speed 1
25	Motor starts at speed 2
26	Motor running at speed 2
27	Error when stopping motor
28	Error with motor speed 1
29	Error with motor speed 2
30	Display for interlocks in block icon
31	MS_RelOp

Status word allocation for `Status3` parameter

Status bit	Parameter
0	Delay of the AV_AH_Lim message
1	Delay of the AV_WH_Lim message
2	Delay of the AV_TH_Lim message
3	Delay of the AV_TL_Lim message
4	Delay of the AV_WL_Lim message
5	Delay of the AV_AL_Lim message
6	Collection of message delays
7 - 14	Not used
15	Current monitoring time is visible

Status bit	Parameter
16	MonDynStopErr.Value
17	Not used
18	SimLiOp.Value
19	1 = Enable for rapid stop (Feature Bit Enabling rapid stop via faceplate (Page 167))
20	1 = Input parameter Spd1ChnST is interconnected
21	1 = Input parameter Spd2ChnST is interconnected
22	Not used
23	Command for rapid stop
24	Command for starting > the motor
25	Command for starting >> the motor
26	Show automatic preview in the standard view
27	Not used
28	GrpErr.Value
29	RdyToStart.Value
30	Auxiliary value 1 visible
31	Auxiliary value 2 visible

Status word allocation for Status4 parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via EventTsIn
1	Effective signal 2 of the message block connected via EventTsIn
2	Effective signal 3 of the message block connected via EventTsIn
3	Effective signal 4 of the message block connected via EventTsIn
4	Effective signal 5 of the message block connected via EventTsIn
5	Effective signal 6 of the message block connected via EventTsIn
6	Effective signal 7 of the message block connected via EventTsIn
7	Effective signal 8 of the message block connected via EventTsIn
8	AV not connected
9	Motor protection display (Trip.Status ≠ 16#FF)
10	1 = Input parameter FbkSpd1 is connected
11	1 = Input parameter FbkSpd2 is connected
12 - 21	Not used
22	External error generated by FaultExt or external control system fault CSF with set Feature bit 18 Activating error state for external process control error CSF (Page 150)
23	Hidden bypass signal in Permit
24	Hidden bypass signal in interlock
25	Hidden bypass signal in Protect
26	Feature2 bit 2: Separate bypass signal
27-30	Not used
31	Separate monitoring of shutdown of the motor (Feature bit 13)

Status word allocation for `Status5` parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via `EventTsIn`
8 - 31	Not used

See also

MotSpdL functions (Page 1215)
MotSpdL messaging (Page 1225)
MotSpdL block diagram (Page 1235)
MotSpdL error handling (Page 1223)
MotSpdL modes (Page 1214)

7.6.2 MotSpdL modes

MotSpdL modes

The block can be operated using the following modes:

- Local mode (Page 79)
- Automatic mode (Page 75)
- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions. This includes, for example, the parameters for mode changes.

"Local mode"

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) section.

Motor actions you can control in local mode:

- starting with speed 1 (`Spd1Local` = 1)
- starting with speed 2 (`Spd2Local` = 1)
- stopping (`StopLocal` = 1).

A motor operated in "local" mode is controlled either by "local" signals (input parameters `Spd1Local` = 1, `Spd2Local` = 1 and `StopLocal` = 1) or feedback signals (input parameters `FbkSpd1` = 1 and `FbkSpd2` = 1). Configuration takes place via the input parameter `LocalSetting`.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) section.

Motor actions you can control in auto mode:

- starting with speed 1 (`Spd1Aut` = 1)
- starting with speed 2 (`Spd2Aut` = 1)
- stopping (`StopAut` = 1).

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

Motor actions you can control in "manual mode":

- starting with speed 1 (`Spd1Man` = 1)
- starting with speed 2 (`Spd2Man` = 1)
- stopping (`StopMan` = 1).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71)

See also

MotSpdL block diagram (Page 1235)

MotSpdL I/Os (Page 1226)

MotSpdL messaging (Page 1225)

MotSpdL error handling (Page 1223)

MotSpdL functions (Page 1215)

Description of MotSpdL (Page 1210)

7.6.3 MotSpdL functions

Functions of MotSpdL

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	1 = Operator can switch to "local mode"
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can stop the motor
5	1 = Operator can start the motor at speed 1
6	1 = Operator can start the motor at speed 2
7	1 = Operator can reset the motor
8	1 = Operator can define or change the monitoring time for startup
9	1 = Operator can define the monitoring time for the status
10	1 = Operator can activate the monitoring time function (Bit 8 - 9)
11	1 = Operator can activate the Simulation function
12	1 = Operator can activate the Release for maintenance function
13	1 = Operator can change the limit (`AV`) for high alarm
14	1 = Operator can change the limit (`AV`) for high warning
15	1 = Operator can change the limit (`AV`) for high tolerance
16	1 = Operator can change the limit (`AV`) for hysteresis
17	1 = Operator can change the limit (`AV`) for low alarm
18	1 = Operator can change the limit (`AV`) for low warning
19	1 = Operator can change the limit (`AV`) for low tolerance
20	1 = Operator can activate / deactivate messages via `AV_AH_MsgEn`
21	1 = Operator can activate / deactivate messages via `AV_WH_MsgEn`
22	1 = Operator can activate / deactivate messages via `AV_TH_MsgEn`
23	1 = Operator can activate / deactivate messages via `AV_TL_MsgEn`
24	1 = Operator can activate / deactivate messages via `AV_WL_MsgEn`
25	1 = Operator can activate / deactivate messages via `AV_AL_MsgEn`
26	1 = Operator can change the simulation value `SimAV`
27 - 29	Not used
30	1 = Operator can define the monitoring time for stopping
31	1 = Operator can execute the rapid stop

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Restart lock after switching off the motor

After switching off or stopping the motor, it can only be restarted after the time set with the `IdleTime` input parameter. When the "Stop" command is given, the motor goes immediately into "Stop" mode, and `IdleTime` starts after the feedback (`FbkSpd1` and `FbkSpd2` = 0) is given. The motor cannot be started again until the `IdleTime` has expired.

The `IdleTime` parameter can be set independently of the `MonTiDynamic` parameter.

Limit monitoring of an additional analog value

This block provides the standard function Limit monitoring of an additional analog value (Page 91).

Limit monitoring with hysteresis

This block provides the standard function Limit monitoring with hysteresis (Page 97).

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Interlocks

This block provides the following interlocks:

- Activation enable
- Interlock without reset ("Interlock")
- Interlock with reset ("Protection")

Refer to the section Interlocks (Page 99) as well as Influence of the signal status on the interlock (Page 103).

Motor protection function

This block provides the standard function Motor protection function (Page 99).

Rapid stop for motors

This block provides the standard function Rapid stop for motors (Page 106).

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Resetting the block in case of interlocks or errors

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See MotSpdL error handling (Page 1223).

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `Trip`
- `MonDynErr`
- `MonStaErr`
- `FaultExt`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `FbkSpd1Out.ST`
- `FbkSpd2Out.ST`
- `LocalLi.ST`
- `Spd1Local.ST`
- `StopLocal.ST`
- `Spd2Local.ST`
- `Trip.ST`
- `AV_Out.ST`

- `Spd1Chn.ST`
- `Spd2Chn.ST`
- `Spd1Aut.ST` (only if `Feature2.Bit10` = 1)
- `Spd2Aut.ST` (only if `Feature2.Bit10` = 1)
- `StopAut.ST` (only if `Feature2.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position (motor stop) in the "Automatic" mode:

- `Spd1Aut.ST`
- `Spd2Aut.ST`
- `StopAut.ST`

Forcing operating modes

This block provides the standard function Forcing operating modes (Page 41).

The following states can be enforced:

- Speed 1 (`Spd1Force`)
- Speed 2 (`Spd2Force`)
- Stop (`StopForce`)

Feedback monitoring

This block provides the standard function Monitoring the feedbacks (Page 97).

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Specify warning times for control functions

This block provides the standard function Specifying warning times for control functions at motors and valves (Page 51).

You can generate warning signals when, for example, motors are started. Warning signals can be generated in the following modes:

- Manual mode (Page 75) (`WarnTiMan` input parameter)
- Automatic mode (Page 75) (`WarnTiAut` input parameter)

You specify the warning times in seconds using the input parameters `WarnTiMan` and `WarnTiAut`. If, for example, a motor is started, then this is displayed at the output parameter

with `WarnAct` = 1. The motor then starts after the set warning time has expired and `WarnAct` then goes back to 0.

A corresponding warning is not output if the warning times (`WarnTiMan` or `WarnTiAut`) are specified with a smaller value than the `SampleTime` parameter.

Step control mode for the speed change

Use the input parameter `SwiOverTi` and the `Feature` Bit 5 (Specifying switching mode (Page 167)) to specify how the motor is to carry out the speed change.

In so doing use the input parameter `SwiOverTi` to adjust the switching time. You have the following options:

- Switching on and off occurs immediately
- Switching on via speed 1
- Switching off via speed 1

Switching on and off occurs immediately: This setting enables you to switch directly from the "Off" state to speed 2 or from speed 2 (`Spd2`) to "Off".

Switching on via speed 1: The speed change from the "Off" condition to speed 2 (`Spd2`) is accomplished via speed 1 (`Spd1`) and after expiration of the time at the parameter `SwiOverTi`.

Switching off via speed 1: The speed change from speed 2 (`Spd2`) to the "Off" state is accomplished via speed 1 (`Spd1`) and after expiration of the time at parameter `SwiOverTi`.

Step control mode	`SwiOverTi`	`Feature` Bit 5
Switching on and off occurs immediately	= 0.0	0/1
Switching on only via speed 1	> 0.0	0
Switching on and off via speed 1	> 0.0	1

Simulating signals

This block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Additional value (`SimAV`, `SimAV_Li`)

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Neutral position

This block provides the standard function Neutral position for motors, valves and controllers (Page 48).

Output signal as a pulse signal or static signal

This block provides the standard function Output signal as a static signal or pulse signal (Page 51).

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
4	Setting switch or button mode (Page 166)
5	Specifying switching mode (Page 167)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
13	Separate monitoring time for stopping the motor (Page 168)
14	Enabling rapid stop via faceplate (Page 167)
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)
18	Activating error state for external process control error CSF (Page 150)
19	Reset even with locked state (Page 164)
20	Disable calculation of impulse control in LocalSetting 2 and 4 (Page 185)
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
27	Interlock display with LocalSetting 2 or 4 (Page 177)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

In pushbutton operation (Bit 4 = 0) the automatic commands in "automatic" mode are latching, in other words `Spd1Aut`, `Spd2Aut`, `StopAut` can be reset to 0 after changing the control. In "manual" and "local" modes, however, the automatic commands are not saved and in the absence of automatic commands the automatic control is tracked.

In switching mode (Bit 4 = 1), control is selected with the static signals `Spd1Aut`, `Spd2Aut`. If `Spd1Aut`, `Spd2Aut` inputs are not set, the motor is stopped. Control via `StopAut` is not needed. If the "Activate command reset for control" function (Bit 3 = 1) is also enabled, the inputs `Spd1Aut`, `Spd2Aut` are reset to the neutral position after evaluation in the block.

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
4	Setting switch or button mode for local commands (Page 180)
5	Evaluation of the signal status of the interlock signals (Page 141)
8	Forcing operating modes in the "Local" mode (Page 183)
10	Considering bad quality of automatic commands or external values (Page 185)

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- `Spd1Man`
- `Spd2Man`
- `StopMan`
- `RapidStp`

See also

MotSpdL block diagram (Page 1235)

MotSpdL modes (Page 1214)

EventTs functions (Page 1634)

MotSpdL I/Os (Page 1226)

7.6.4 MotSpdL error handling

Error handling of MotSpdL

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Invalid input signals
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
41	The value for the `LocalSetting` I/O is not within the approved limit of 0 to 4.
42	`LocalSetting` = 0 or `LocalSetting` = 3 or `LocalSetting` = 4 and `LocalLi` = 1
51	`Spd1Local` = 1 and `StopLocal` = 1 `Spd2Local` = 1 and `StopLocal` = 1 `Spd1Local` = 1 and `Spd2Local` = 1 `Spd1Aut` = 1 and `StopAut` = 1 `Spd2Aut` = 1 and `StopAut` = 1 `Spd1Aut` = 1 and `Spd2Aut` = 1 `AutModLi` = 1 and `ManModLi` = 1 `Spd1Force` = 1 and `StopForce` = 1 `Spd2Force` = 1 and `StopForce` = 1 `Spd1Force` = 1 and `Spd2Force` = 1
52	`LocalAct` = 1 and `LocalSetting` = 2 or 4 and `SimOn` = 1

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

Invalid input signals

This error can be output by the block, see the section Error handling (Page 119).

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signals	Control reaction with Feature2 bit 3 =1
Local: `Localsetting` = 1 or `Localsetting` = 3	Pushbutton operation for local mode (`Feature2` bit 4 = 0): `Spd1Local` = 1 and `Spd2Local` = 1 or `Spd1Local` = 1 and `StopLocal` = 1 or `StopLocal` = 1 and `Spd2Local` = 1 Switching mode (`Feature2` bit 4 = 1): `Spd1Local` = 1 and `Spd2Local` = 1	Motor is stopped
Local: `Localsetting` = 1 or `Localsetting` = 3 and forcing	`Spd1Force` = 1 and `Spd2Force` = 1 or `Spd1Force` = 1 and `StopForce` = 1 or `StopForce` = 1 and `Spd2Force` = 1	
Forcing and no "local mode"	`Spd1Force` = 1 and `Spd2Force` = 1 or `Spd1Force` = 1 and `StopForce` = 1 or `StopForce` = 1 and `Spd2Force` = 1	
"Automatic mode" and no forcing	Pushbutton operation (`Feature` bit 4 = 0): `Spd1Aut` = 1 and `Spd2Aut` = 1 or `Spd1Aut` = 1 and `StopAut` = 1 or `StopAut` = 1 and `Spd2Aut` = 1 Switching mode (`Feature`bit 4 = 1): `Spd1Aut` = 1 and `Spd2Aut` = 1	
"Manual mode" and no forcing	`Spd1Man` = 1 and `Spd2Man` = 1 or `Spd1Man` = 1 and `StopMan` = 1 or `StopMan` = 1 and `Spd2Man` = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

MotSpdL block diagram (Page 1235)

MotSpdL I/Os (Page 1226)

MotSpdL functions (Page 1215)

MotSpdL modes (Page 1214)
Description of MotSpdL (Page 1210)
MotSpdL messaging (Page 1225)

7.6.5 MotSpdL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 1	AS process control message - fault	$$BlockComment$$ Motor feedback error
	SIG 2	AS process control message - fault	$$BlockComment$$ Motor protection triggered
	SIG 3	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a process control fault is triggered (`MsgEvId1`, SIG 3).

Instance-specific messages

You can use up to three instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 4	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 5	AS process control message - fault	$$BlockComment$$ External message 2
	SIG 6	AS process control message - fault	$$BlockComment$$ External message 3

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`ExtVa104`
5	`ExtVa105`
6	`ExtVa106`
7	`ExtVa107`
8	`ExtVa108`
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters `ExtVa104` ... `ExtVa108` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

MotSpdL block diagram (Page 1235)

MotSpdL modes (Page 1214)

MotSpdL error handling (Page 1223)

7.6.6 MotSpdL I/Os

I/Os of MotSpdL

Input parameters

Table 7-1

Parameter	Description	Type	Default
`AutModLi*`	1 = "Automatic mode" via interconnection or SFC (controlled by `ModLiOp` = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`AutModOp*`	1 = "Automatic mode" via operator (controlled by `ModLiOp` = 0)	BOOL	0
`AV`	Input additional analog value, to be connected to `AV_Tech` of the AV block	ANY	
`AV_AH_Lim`	Limit high alarm	REAL	95.0
`AV_AL_Lim`	Limit low alarm	REAL	5.0

Parameter	Description	Type	Default
AV_Hyst	Hysteresis for alarm, warning and tolerance limits	REAL	1.0
AV_TH_Lim	Limit high tolerance	REAL	85.0
AV_TL_Lim	Limit low tolerance	REAL	15.0
AV_WH_Lim	Limit high warning	REAL	90.0
AV_WL_Lim	Limit low warning	REAL	10.0
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Bypassing interlock in "local mode" and in "simulation"	BOOL	0
CSF	1 = External error (control system faultError handling (Page 119))	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EN	1 = Called block will be processed	BOOL	1
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	

Parameter	Description	Type	Default
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkSpd1	1 = Feedback for speed 1 is present	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkSpd2	1 = Feedback for speed 2 is present	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Feature	I/O for additional functions (Page 1215) 1 = Separate monitoring time for stopping the motor	STRUCT • Bit 0: BOOL • ... • Bit 13: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0 • 0 • 0
Feature2	I/O for additional functions (Page 1215)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
IdleTime*	Wait time for restart of motor in [s]	REAL	5.0
Intlock	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is active	BOOL	1
LocalLi	1 = Activate "local mode" via plant signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalOp*	1 = "Local mode" via operator	BOOL	0
LocalSetting	Properties for the Local mode (Page 79)	INT	0
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
ModLiOp	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Monitor	1 = Feedback monitoring	BOOL	1

Parameter	Description	Type	Default
MonTiDynamic*	Monitoring time for feedback errors or feedback start error after successful operation in [s]	REAL	3.0
MonTiDyStop*	Monitoring time for feedback stop errors after successful operation in [s]	REAL	3.0
MonTiStatic*	Monitoring time for feedback errors without operation in [s]	REAL	3.0
MS_RelOp	1= Release for maintenance via OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp*	1 = Release for maintenance via OS operator	BOOL	0
Occupied	1 = In use by a batch	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 1215) 1 = Operator can set or change the monitoring time for "Control: Start" 1 = Operator can set the monitoring time for "Control:Stop"	STRUCT • Bit 0: BOOL • Bit 8: BOOL • Bit 20: BOOL • Bit 30: BOOL	- • 1 • 1 • 1 • 1
Permit	1 = OS activation enable for motor 0 = No OS release for energizing motor	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Perm_En	1 = Activation enable (enable, Permit parameter) is active	BOOL	1
Protect	0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block 1 = Protective interlocking not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Prot_En	1 = Protective interlock (protection, Protect parameter) is active	BOOL	1
PulseWidth*	Pulse width of control signal [s]	REAL	3.0

Parameter	Description	Type	Default
RapidStp*	Rapid stop for the motor: 0 = Motor On 1 = Motor Off	BOOL	0
Spd1Aut*	1 = Activation of motor speed 1 in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Spd2Aut*	1 = Activation of motor speed 2 in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Spd1ChnST	Signal status of output channel for Fwd Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Spd2ChnST	Signal status of output channel for Rev Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Spd1Force	1 = Force activation of motor speed 1 in "manual mode" or in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Spd2Force	1 = Force activation of motor speed 2 in "manual mode" or in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Spd1Local	1 = Activation of motor speed 1 in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Spd2Local	1 = Activation of motor speed 2 in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Spd1Man*	1 = Activation of motor speed 1 in "manual mode"	BOOL	0
Spd2Man*	1 = Activation of motor speed 2 in "manual mode"	BOOL	0
RstLi*	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SimAV*	Additional value used for SimOn = 1	REAL	0.0

Parameter	Description	Type	Default
SimAV_Li	Additional analog value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0 • 16#80
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	
StepNo	Batch step number	DWORD	16#00000000
StopAut*	1 = Stopping the motor in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopForce	1 = Force motor stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopLocal	1 = Stopping the motor in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopMan*	1 = Stopping the motor in "manual mode"	BOOL	0
SwOverTi*	Time for speed change	REAL	0.0
Trip	1 = Motor is in "good" state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA1unit	Unit of measure for analog auxiliary value 1	INT	0

Parameter	Description	Type	Default
UA2unit	Unit of measure for analog auxiliary value 2	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
WarnTiAut*	Prewarning of motor start in "automatic mode" in [s]	REAL	0.0
WarnTiMan*	Prewarning of motor start in "manual mode" in [s]	REAL	0.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80
AV_OpScale	Limit for scale in AV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
AV_Out	Output additional analog value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
AV_Unit	Unit of measure for additional analog value	INT	0
CurrMon	Current monitoring time [s]	DINT	0
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see MotSpdL error handling (Page 1223)	INT	-1
FbkSpd1Out	Feedback: 1 = Speed 1 active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkSpd2Out	Feedback: 1 = Speed 2 active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
LocalAct	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock (Intlock, Permit, Protect) or Trip is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MonDynErr	1 = Feedback error or feedback start error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonDynStopErr	1 = Feedback stop error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonStaErr	1 = Feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Spd1	1 = Pulse signal for starting the motor with speed 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
P_Spd2	1 = Pulse signal for starting the motor with speed 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Stop	0 = Pulse signal for stopping the motor	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
R_StpAct	1 = Rapid stop of the motor is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RunSpd1	1 = Motor running at speed 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RunSpd2	1 = Motor running at speed 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Spd1	1 = Control of motor: Speed 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Spd2	1 = Control of motor: Speed 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Starting	1 = Motor will start	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Status1	Status word 1 (Page 1210)	DWORD	16#00000000
Status2	Status word 2 (Page 1210)	DWORD	16#00000000
Status3	Status word 3 (Page 1210)	DWORD	16#00000000
Status4	Status word 4 (Page 1210)	DWORD	16#00000000
Stop	1 = Motor is stopping	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
`Stoping`	1 = Motor will stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`WarnAct`	1 = Prewarning for motor start active (parameters `WarnTiAut` and `WarnTiMan`)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

MotSpdL messaging (Page 1225)

MotSpdL block diagram (Page 1235)

MotSpdL modes (Page 1214)

Error handling (Page 119)

7.6.7 MotSpdL block diagram

MotSpdL block diagram

A block diagram is not provided for this block.

See also

MotSpdL I/Os (Page 1226)

MotSpdL messaging (Page 1225)

MotSpdL error handling (Page 1223)

MotSpdL functions (Page 1215)

MotSpdL modes (Page 1214)

Description of MotSpdL (Page 1210)

7.6.8 Operator control and monitoring

7.6.8.1 MotSpdL views

Views of the MotSpdL block

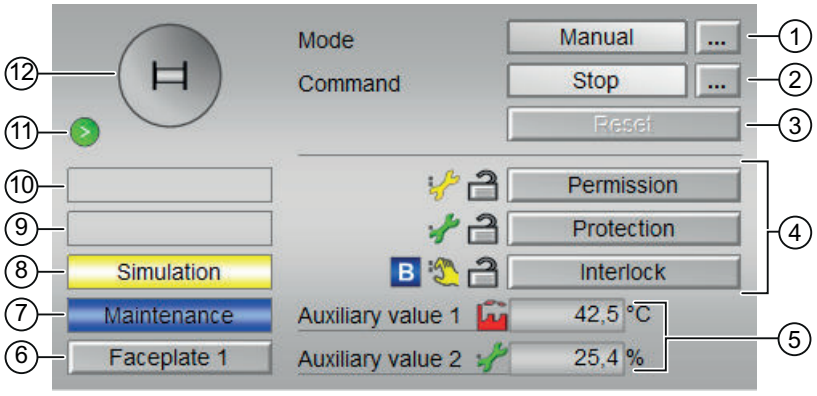
The block MotSpdL provides the following views:

- MotSpdL standard view (Page 1236)
- Alarm view (Page 296)
- Limit view of motors (Page 288)
- Trend view (Page 299)
- Parameter view for motors and valves (Page 280)
- MotSpdL preview (Page 1240)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for MotSpdL (Page 1243)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.6.8.2 MotSpdL standard view

MotSpdL standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Starting and stopping the motor

This area shows you the default operating state for the motor. The following states can be shown and executed here:

- "Start >"
- "Start >>"
- "Stop"
- "Rapid stop"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(3) Resetting the block

Click "Reset" for interlocks or errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(4) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the section Interlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

(5) Display of auxiliary values

This display is only visible when the corresponding block input is connected.

You can use this area to display two auxiliary values that have been configured in the engineering system (ES). You can find additional information on this in the section Displaying auxiliary values (Page 207).

(6) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(7) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

(8) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"
- "Delay"

You will find more detailed information on this in the sections Simulating signals (Page 58) and Display of delay times (Page 250).

(9) Display area for block states

This area provides additional information on the operating state of the block:

- "Motor protection"
- "External error"
- "Status error"
- "Control error"
- "Invalid signal"
- "Changeover error"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97) , Error handling (Page 119) (section "Invalid input signals" and "Mode switchover error") and Motor protection function (Page 99).

(10) Display area for block states

This area provides additional information on the operating state of the block:

- "Forced stop"
- "Forced start >"
- "Forced start >>"
- "Request 0/1": A reset to "automatic mode" is expected.

You can find additional information on this in the section Forcing operating modes (Page 41).

(11) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the motor would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

(12) Status display of the motor

The current status of the motor is graphically displayed here.

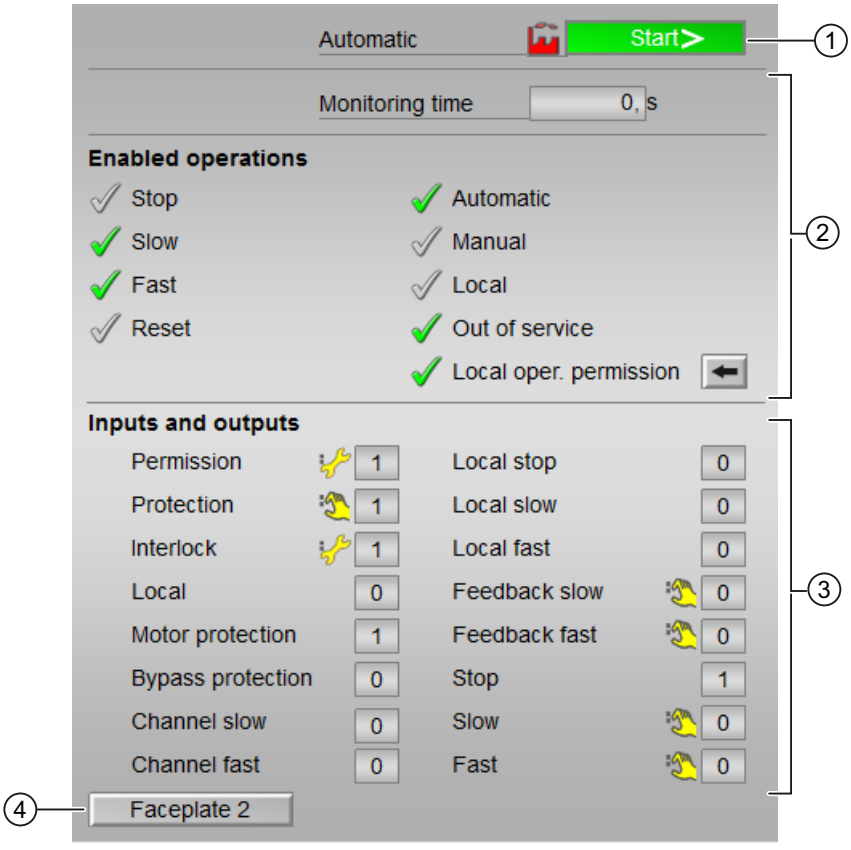
You can find more information about this in section Block icon for MotSpdL (Page 1243)

See also

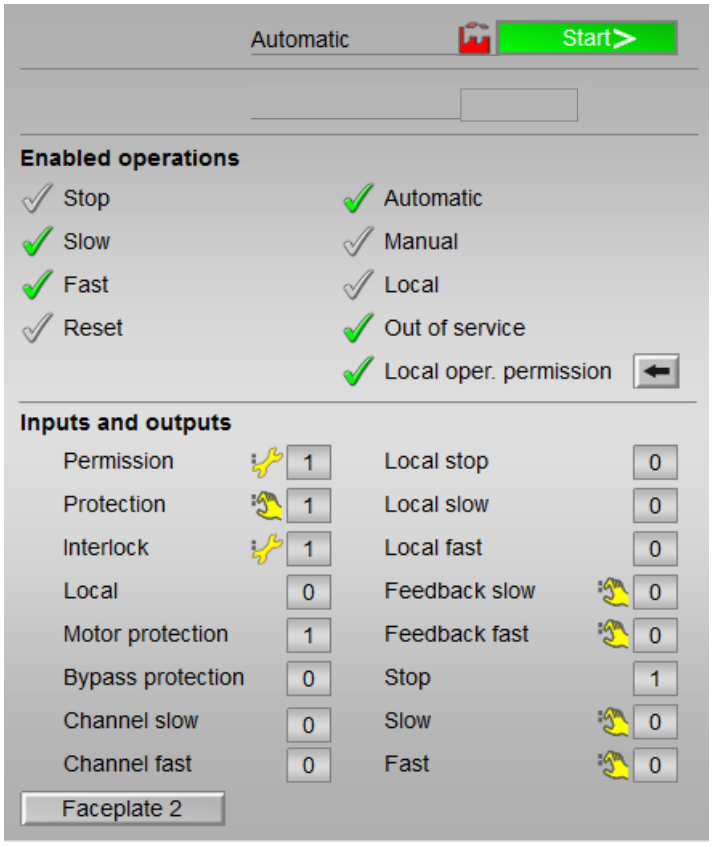
Functions of the blocks (Page 41)

7.6.8.3 MotSpdL preview

Preview of MotSpdL



Display of the current monitoring time is visible.



Display of the current monitoring time is not visible.

(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `Spd1Aut`
- `Spd2Aut`
- `StopAut`

(2) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Stop": You can stop the motor.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Slow": You can start the motor in the "slow" state.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Fast": You can start the motor in the "fast" state.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Reset": You can reset the motor after interlocks or errors.
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Local": You can switch to "local mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248) .
- "Monitoring time": Display of the current monitoring time.

(3) Displaying current control signals

This area shows the most important parameters for this block with the current selection:

- "Permission":
  This display is only visible when the corresponding block input is connected.
  - 0 = No OS release for energizing motor
  - 1 = Enable for "starting"/"stopping" from the neutral position
- "Protection":
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state

- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Motor protection": 1 = Motor is in "good" state
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"
- "Local stop": 1 = Stopping the motor in "local mode"
- "Local slow": 1 = Starting the motor in "local mode", slow
- "Local fast": 1 = Starting the motor in "local mode", fast
- "Feedback slow": 1 = Motor has started and is running slow
- "Feedback fast": 1 = Motor has started and is running fast
- "Stop": 1 = Stop motor
- "Slow": 1 = Motor is running slow
- "Fast": 1 = Motor is running fast
- "Channel Slow": Signal from the output channel block for "Slow"
- "Channel Fast": Signal from the output channel block for "Fast"

(4) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

7.6.8.4 Block icon for MotSpdL

Properties of the MotSpdLblock icon

A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes

- Signal status, release for maintenance
- Displays for bypassing interlocks
- Interlocks
- Memo display
- Motor state display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	
	4	
	5	"M" symbol with small speed display
	6	"M" symbol with small speed display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	5	

Icons	Selection of the block icon in CFC	Special features
	6	
	7	
	8	
	9	
	10	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Motor state display

The following motor states are shown here:

Symbol	Meaning
	Motor started (motor symbol changes)
	The motor is running
	Motor stopped (motor symbol changes)
	Motor idle
	Error at motor (monitoring error, motor protection)
	Motor out of service

7.7 ShrdResL - Multiplexer for shared resources (Large)

7.7.1 Description of ShrdResL

Object name (type + number) and family

Type + number: FB 1917

Family: Drives

Area of application for ShrdResL

The block is used for the following application:

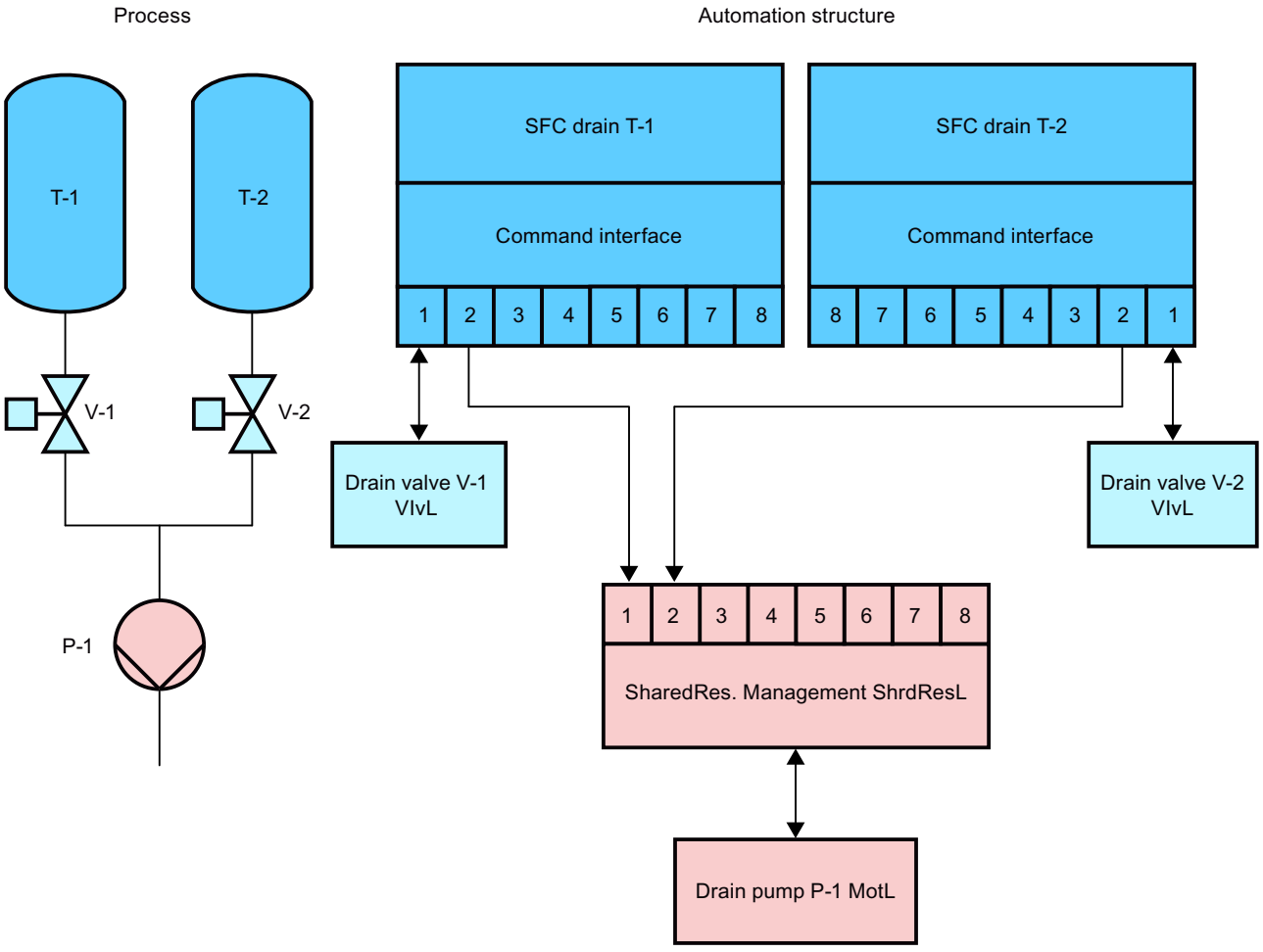
- Organize individual requests for downstream block from upstream applications.

How it works

The block coordinates the access from up to four CFC or SFC step sequencers to a technological block of the families "Drives" or "Dose".These blocks include:

- Motors
- Valves
- Dosers

The block has four channels, each with a standardized command interface.



In case of multiple allocation requests, the channel to be allocated depends on the priority of each channel and a strategy mode which coordinates the sequence of the allocated channels.

As soon as a channel is allocated, its command interface is aligned in accordance with the command interface at the output.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status1` parameter

You can find a description for each parameter in the section ShrdResL I/Os (Page 1257).

Status bit	Parameter
0	`ReadyIn`, Readiness signal of the interconnected technological block
1	`ChnEn_1`
2	`ChnEn_2`
3	`ChnEn_3`
4	`ChnEn_4`
5	`Ready`, Channel allocation is possible
6	`ActChnNo` = 1, 1 = Channel 1 is enabled
7	`ActChnNo` = 2, 1 = Channel 2 is enabled
8	`ActChnNo` = 3, 1 = Channel 3 is enabled
9	`ActChnNo` = 4, 1 = Channel 4 is enabled
10	`MultiOcc`, 1 = Request for more than one channel
11	Not used
12	`BatchEn`
13	`Occupied`
14 - 16	Not used
17	`NxtChnNo` = 1, 1 = Channel 1 is the next allocated channel
18	`NxtChnNo` = 2, 1 = Channel 2 is the next allocated channel
19	`NxtChnNo` = 3, 1 = Channel 3 is the next allocated channel
20	`NxtChnNo` = 4, 1 = Channel 4 is the next allocated channel
21	`BaEn_1`
22	`BaEn_2`
23	`BaEn_3`
24	`BaEn_4`
25	`Occ_1`
26	`Occ_2`
27	`Occ_3`
28	`Occ_4`
29 - 31	Not used

Status word allocation for `Status2` parameter

You can find a description for each parameter in the section ShrdResL I/Os (Page 1257).

Status bit	Parameter
0	Not used
1	`ChnEn_5`
2	`ChnEn_6`
3	`ChnEn_7`
4	`ChnEn_8`
5	Not used

Status bit	Parameter
6	`ActChnNo` = 5, 1 = Channel 5 is enabled
7	`ActChnNo` = 6, 1 = Channel 6 is enabled
8	`ActChnNo` = 7, 1 = Channel 7 is enabled
9	`ActChnNo` = 8, 1 = Channel 8 is enabled
10 - 16	Not used
17	`NxtChnNo` = 5, 1 = Channel 5 is the next allocated channel
18	`NxtChnNo` = 6, 1 = Channel 6 is the next allocated channel
19	`NxtChnNo` = 7, 1 = Channel 7 is the next allocated channel
20	`NxtChnNo` = 8, 1 = Channel 8 is the next allocated channel
21	`BaEn_5`
22	`BaEn_6`
23	`BaEn_7`
24	`BaEn_8`
25	`Occ_5`
26	`Occ_6`
27	`Occ_7`
28	`Occ_8`
29 - 31	Not used

See also

ShrdResL modes (Page 1249)
ShrdResL functions (Page 1250)
ShrdResL error handling (Page 1256)
ShrdResL messaging (Page 1256)
ShrdResL block diagram (Page 1262)

7.7.2 ShrdResL modes

ShrdResL operating modes

This block does not have any modes.

See also

Description of ShrdResL (Page 1246)
ShrdResL functions (Page 1250)
ShrdResL error handling (Page 1256)
ShrdResL messaging (Page 1256)

ShrdResL I/Os (Page 1257)
ShrdResL block diagram (Page 1262)

7.7.3 ShrdResL functions

Functions of ShrdResL

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0 - 5	Not used
6	1 = Operator can change the priority of channel 1
7	1 = Operator can change the priority of channel 2
8	1 = Operator can change the priority of channel 3
9	1 = Operator can change the priority of channel 4
10	1 = Operator can change the priority of channel 5
11	1 = Operator can change the priority of channel 6
12	1 = Operator can change the priority of channel 7
13	1 = Operator can change the priority of channel 8
14 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bit:

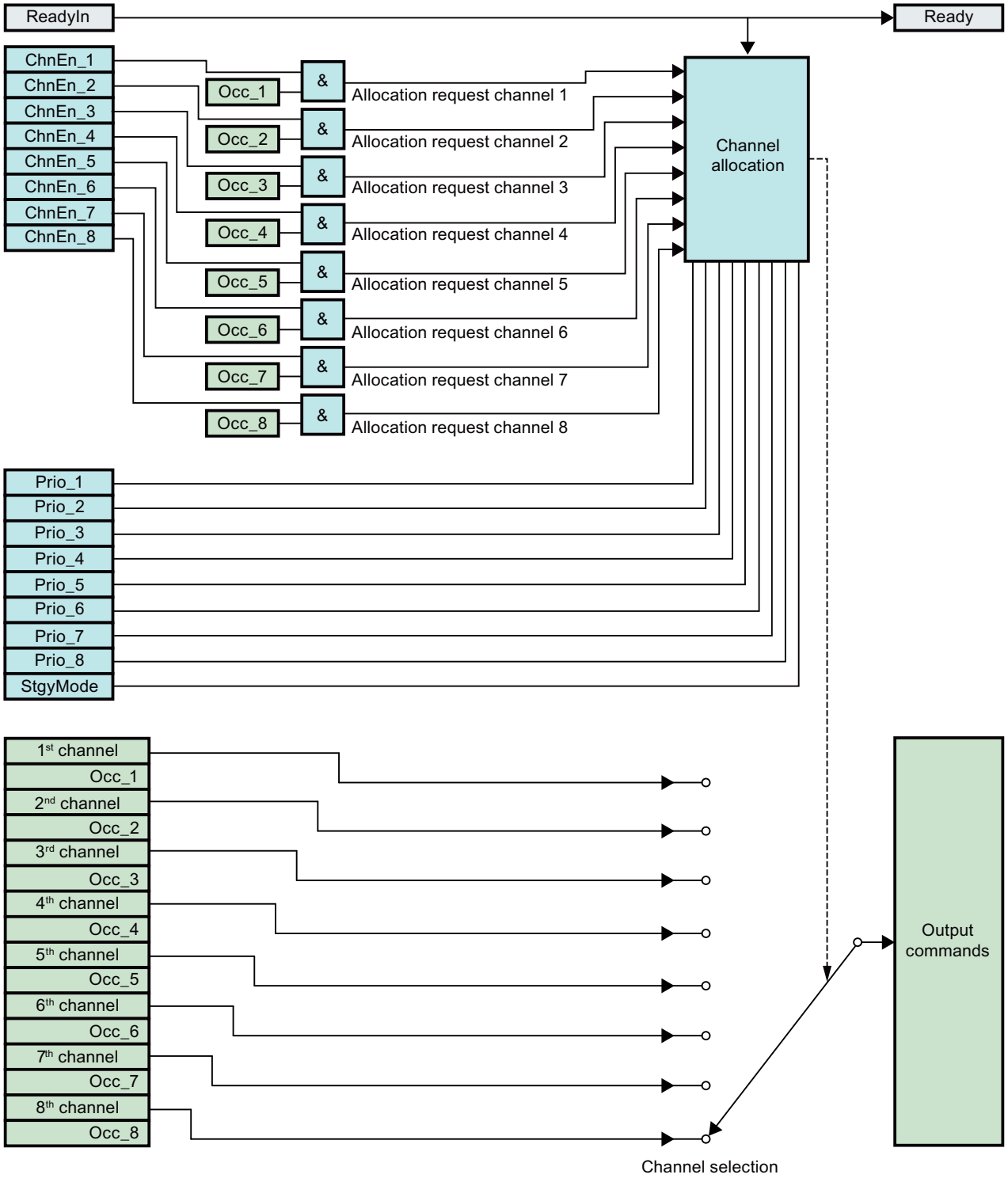
Bit	Function
0	Set startup characteristics (Page 137)

Readiness signal

In order to assign one of the channels, the ready signal must be `ReadyIn` = 1 and at least one channel enable must be `ChnEn_x` = 1 (x = 1 to 8). The ready signal is output at the `Ready` output.

Channel management

Overview of the channel management:



Allocate/enable channel

A channel can only be allocated if the readiness signal (ReadyIn = 1) and the respective channel enabling is available for the allocation ChnEn_x = 1.

With ChnEn_x = 0, the channel x is disabled and cannot be allocated.

The allocation request of a channel by the upstream application is accomplished via the input Occ_x. In case of multiple requests, the channel allocation function stores the requests in a buffer and calculates the allocated channel number with the help of the channel priorities Prio_x (x = 1 to 8) and the strategy mode StgyMode.

Each channel has its own priority. It can be parameterized at the inputs Prio_x (x = 1 to 8).

The strategy mode defines the order of storing the requests in the buffer.

Strategy 1	Strategy 2	Parameter value of StgyMode
First In First Out	Order	0
First In First Out	Maximum Priority; in case equal request Order is used	1
First In First Out	Minimum Priority; in case equal request Order is used	2
Last In First Out	Order	3
Last In First Out	Maximum Priority; in case equal request Order is used	4
Last In First Out	Minimum Priority; in case equal request Order is used	5
Maximum Priority	Order	6
Maximum Priority	First In First Out	7
Maximum Priority	Last In First Out	8
Minimum Priority	Order	9
Minimum Priority	First In First Out	10
Minimum Priority	Last In First Out	11

The allocation request will be evaluated with **Strategy 1**. If after the first evaluation, there are equal requests, the **Strategy 2** will be evaluated. This happens in the following cases:

- Two or more requests are in the same cycle and **Strategy 1** is “First In First Out” or “Last In First Out”.
- Two or more requests have the same priority and **Strategy 1** is “Maximum Priority” or “Minimum Priority”.

The parameterization “Order” in **Strategy 2** means that the priority of the request depends on the channel number:

- Channel 1 has the highest priority
- Channel 8 has the lowest priority

If **Strategy 2** is “Maximum Priority” or “Minimum Priority” and if after the evaluation, there are equal requests, the channel allocation is calculated with “Order”.

The applied commands are aligned in accordance with the output command interface.

The number of the allocated channel is displayed at the output ActChnNo (INT format). If no channel is allocated, the output is 0.

Enable/disable channel

If the allocated channel disabling via `Occ_x` = 0 or via `ChnEn_x` = 0, the next channel in the allocation request buffer is automatically switched to the command outputs. If there is no other allocation request (every `Occ_x` = 0), the commands of the output interface are resetted to the default values.

Channel prioritization

Channel 1 has the highest priority, channel 4 has the lowest. If the input `Occ_x` is set at multiple enabled channels, the channel with the highest priority is allocated and the output `MultiOcc` = 1 is set.

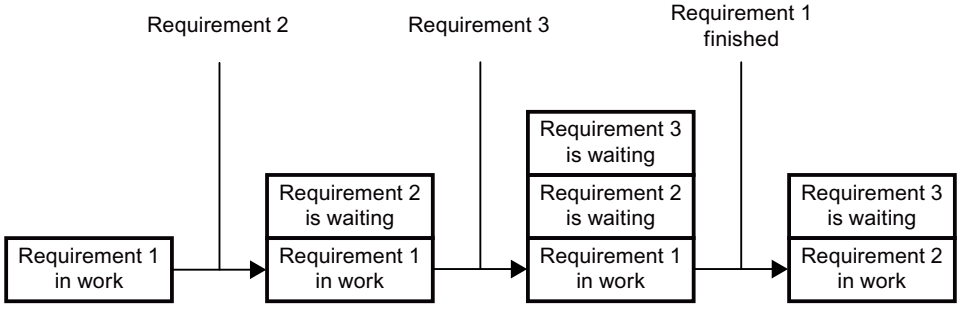
SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Example for "First In First Out" Strategy mode (`StgyMode` = 0, 1, or 2)

The requirement which is issued for the longest duration will be released first.

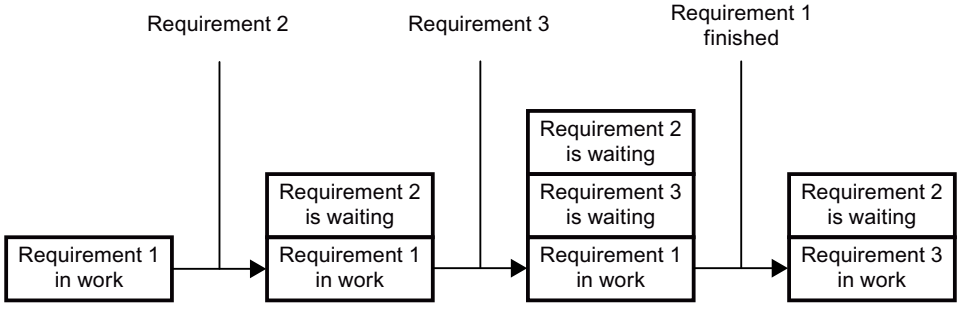
The requirements that are coming in the same cycle will be evaluated after **Strategy 2**.



Example for "Last In First Out" Strategy mode (`StgyMode` = 3, 4, or 5)

The requirement which is issued for the shortest duration will be released first.

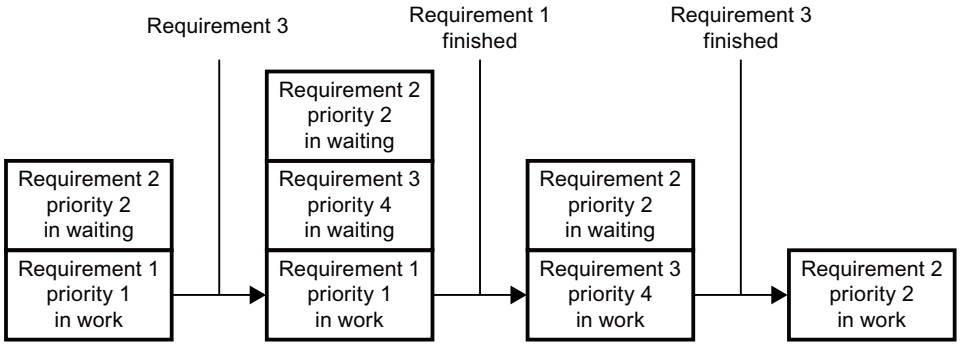
The requirements that are coming in the same cycle will be evaluated after **Strategy 2**.



Example for "Maximum Priority" Strategy mode (`StgyMode` = 6, 7, or 8)

The requirement with the highest priority will be released first.

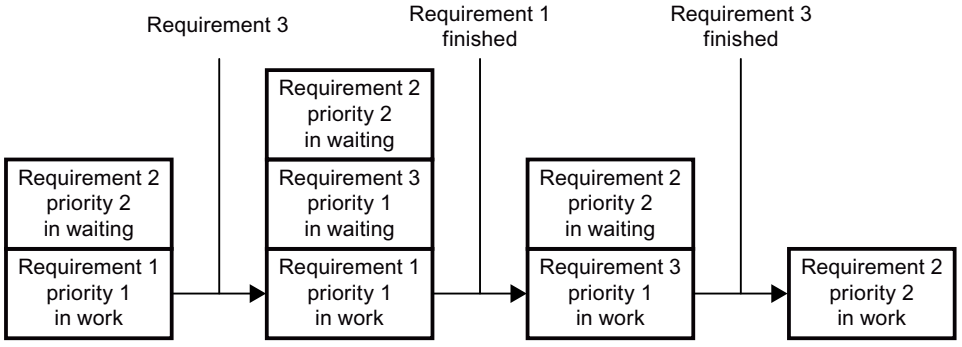
The requirements that are coming in the same cycle will be evaluated after **Strategy 2**.



Example for "Minimum Priority" Strategy mode (`StgyMode` = 9, 10, or 11)

The requirement with the lowest priority will be released first.

The requirements that are coming in the same cycle will be evaluated after **Strategy 2**.



See also

Description of ShrdResL (Page 1246)

ShrdResL modes (Page 1249)

ShrdResL error handling (Page 1256)

ShrdResL messaging (Page 1256)

ShrdResL I/Os (Page 1257)

ShrdResL block diagram (Page 1262)

7.7.4 ShrdResL error handling

ShrdResL error handling

Please refer to the section Error handling (Page 119) in the basic instructions.

The following error can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed.
0	There is no error.
49	Wrong strategy mode number: `StgyMode` < 0 or `StgyMode` > 11

See also

Description of ShrdResL (Page 1246)

ShrdResL modes (Page 1249)

ShrdResL functions (Page 1250)

ShrdResL messaging (Page 1256)

ShrdResL I/Os (Page 1257)

ShrdResL block diagram (Page 1262)

7.7.5 ShrdResL messaging

Messaging

This block does not offer messaging.

See also

Description of ShrdResL (Page 1246)

ShrdResL modes (Page 1249)

ShrdResL functions (Page 1250)

ShrdResL error handling (Page 1256)

ShrdResL I/Os (Page 1257)
ShrdResL block diagram (Page 1262)

7.7.6 ShrdResL I/Os

I/Os of ShrdResL

Input parameters

Parameter	Description	Type	Default
AutMod_1*	1 = "Automatic mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
...	...	...	...
AutMod_8*	1 = "Automatic mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BaEn_1	1 = Enable allocation for channel 1	BOOL	0
...	...	...	...
BaEn_8	1 = Enable allocation for channel 8	BOOL	0
BaID_1	Batch number for channel 1	DWORD	16#00000000
...	...	...	...
BaID_8	Batch number for channel 8	DWORD	16#00000000
BaName_1	Batch designation for channel 1	S7-String	
...	...	...	...
BaName_8	Batch designation for channel 8	S7-String	
ChnEn_1	1 = Channel 1 is enabled and can be allocated 0 = Channel 1 is blocked and cannot be allocated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
...	...	...	...
ChnEn_8*	1 = Channel 8 is enabled and can be allocated 0 = Channel 8 is blocked and cannot be allocated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
ChnCmd_1	Channel input 1 (reserved)	STRUCT	
...	...	...	...
ChnCmd_8	Channel input 8 (reserved)	STRUCT	
Ctrl01_1*	Control command 1 for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
...	...	...	...
Ctrl12_8*	Control command 12 for channel 8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CtrlW1_1*	Available user control word 1 for channel 1	DWORD	16#00000000
...	...	...	...
CtrlW1_8*	Available user control word 1 for channel 8	DWORD	16#00000000
CtrlW2_1*	Available user control word 2 for channel 1	DWORD	16#00000000
...	...	...	...
CtrlW2_8*	Available user control word 2 for channel 8	DWORD	16#00000000
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1250)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
ManMod_1*	1 = "Manual mode" via interconnection for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
...	...	...	...
ManMod_8*	1 = "Manual mode" via interconnection for channel 8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ModLi_1*	1 = Control via interconnection or SFC for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
...	...	...	...
ModLi_8*	1 = Control via interconnection or SFC for channel 8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occ_1*	1 = Occupied It is also used as a control signal for the channel allocation for channel 1	BOOL	0
...	..0	...	...
Occ_8*	1 = Occupied It is also used as a control signal for the channel allocation for channel 8	BOOL	0
Prio_1	Priority for channel 1	INT	1
...	...	...	...
Prio_8	Priority for channel 8	INT	8
ReadyIn	Readiness signal: 1 = Signal for channel enabling active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
RstLi_1*	1 = Resetting via interconnection for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
...	...	...	...
RstLi_8*	1 = Resetting via interconnection for channel 8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SelFpRes	Selection for faceplate resource	ANY	
SP_Ex_1	1 = Select external setpoint (via interconnection) for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
...	...	...	...
SP_Ex_8	1 = Select external setpoint (via interconnection) for channel 8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_In_1*	1 = Select internal setpoint (via interconnection) for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
...	...	...	...
SP_In_8*	1 = Select internal setpoint (via interconnection) for channel 8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP1Ext_1*	External setpoint 1 for channel 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
...	...	...	...
SP8Ext_8*	External setpoint 8 for channel 8	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
StepNo_1	Step number for channel 1	DWORD	16#00000000
...	...	...	...
StepNo_8	Step number for channel 8	DWORD	16#00000000
StgyMode	Strategy mode (0 to 11)	INT	0
Strng1_1	Additional string 1 for channel 1	S7-String	
Strng2_1	Additional string 2 for channel 1	S7-String	
...	...	...	...
Strng1_8	Additional string 1 for channel 8	S7-String	
Strng2_8	Additional string 2 for channel 8	S7-String	
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000

Parameter	Description	Type	Default
OS_Perm	I/O for operator permissions (Page 1250)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
UserStatus	Freely assignable bits for use in PCS7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ActChnNo	Displays the allocated channel	INT	0
AutMod	1= "Automatic mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
Ctrl01	Control command 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
...	...	...	...
Ctrl12	Control command 12	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CtrlW1	Available user control word 1	DWORD	16#00000000
CtrlW2	Available user control word 2	DWORD	16#00000000
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of the current error number. For error numbers that can be outputted by this block, see ShrdResL error handling (Page 1256)	INT	-1
NxtChnNo	Next occupied channel number	INT	0
ManMod	1 = "Manual mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ModLi	1 = Control via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MultiOcc	1 = More than one channel request is present	BOOL	0
Occupied	1 = Occupied Is also used as a control signal for the channel allocation.	BOOL	0

Parameter	Description	Type	Default
`Ready`	1 = Active readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`RstLi`	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SP_ExtLi`	1 = Select external setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SP_IntLi`	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SP1Ext`	External setpoint 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
...	...	...	...
`SP8Ext`	External setpoint 8	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Status1`	Status word 1 (Page 1246)	DWORD	16#00000000
`Status2`	Status word 2 (Page 1246)	DWORD	16#00000000
`StepNo`	Step number	DWORD	16#00000000
`Strng1`	Additional string 1	S7-String	
`Strng2`	Additional string 2	S7-String	
`OpSt_Out`	Value of the `OpSt_In` input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by `Feature` bit 24	DWORD	16#00000000
`OS_PermLog`	Display of `OS_Perm` with settings changed by the block algorithm	DWORD	16#FFFFFFFF
`OS_PermOut`	Display of `OS_Perm`	DWORD	16#FFFFFFFF

See also

ShrdResL modes (Page 1249)

ShrdResL messaging (Page 1256)

ShrdResL block diagram (Page 1262)

7.7.7 ShrdResL block diagram

ShrdResL block diagram

A block diagram is not provided for this block.

See also

Description of ShrdResL (Page 1246)

ShrdResL modes (Page 1249)

ShrdResL functions (Page 1250)

ShrdResL error handling (Page 1256)

ShrdResL messaging (Page 1256)

ShrdResL I/Os (Page 1257)

7.7.8 Operator control and monitoring

7.7.8.1 ShrdResL views

Views of the ShrdResL block

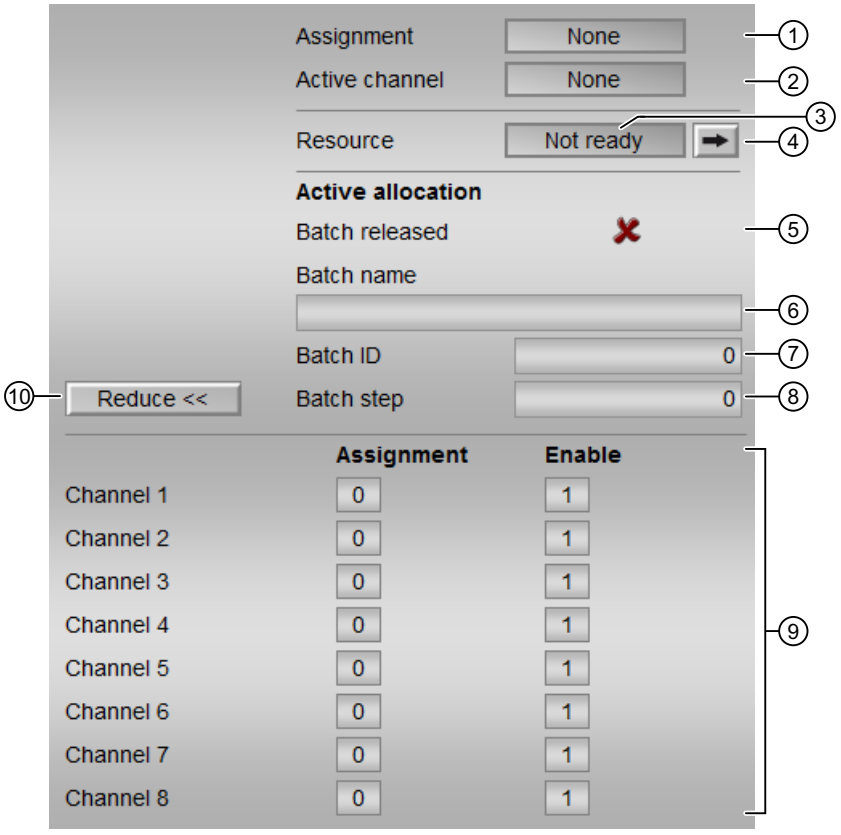
The block ShrdResL provides the following views:

- ShrdResL standard view (Page 1263)
- ShrdResL general preview (Page 1265)
- ShrdResL preview (Page 1265)
- ShrdResL parameter view (Page 1266)
- Memo view (Page 298)
- Block icon for ShrdResL (Page 1267)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.7.8.2 ShrdResL standard view

ShrdResL standard view



(1) Allocation

The allocation status is displayed in this area:

- "None": Display for all `Occ_1`...`Occ_4` = 0 and `Occupied` = 0
- "Requested": Display for one of `Occ_1`...`Occ_4` = 1 and `Occupied` = 0
- "Active": Display for `Occupied` = 1

(2) Active channel

The channel number of the active channel is displayed in this area.

If the text is configured for this command (Text 1 in the object properties), it is displayed as additional text and button label for command selection. Additional information is available in the section Labeling of buttons and text (Page 205).

(3) Resource

The status of the enable signal is displayed in this area:

- "Idle": Display for `ReadyIn` = 1
- "Not ready": Display for `ReadyIn` = 0

(4) Jump key to standard view of any faceplate

This display is only visible when the corresponding block input is interconnected.

Use this navigation button to jump to the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find more information on this in the section Opening additional faceplates (Page 203).

(5) Release batch

This area displays if the block is released for operation via SIMATIC BATCH (`BatchEn` = 1).

(6) Batch name

This area displays the name of the batch that is currently running (`Batchname`).

(7) Batch ID

This area displays the identification number of the batch that is currently running (`BatchID`).

(8) Batch step

This area displays the step number of the batch that is currently running (`StepNo`).

(9) Display channel 1 to 8

This area is only visible when the **(10)** "Expand" button is pressed.

The "Allocation" and "Release" status of the channels 1 to 8 are displayed in this area.

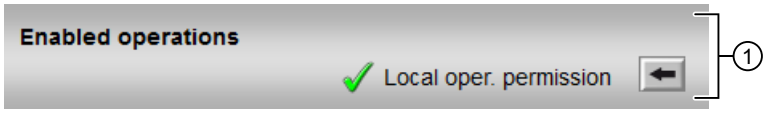
The next channel for allocation, which is in waiting position, is marked with a yellow exclamation mark.

(11) Expand/Collapse

This button enables or disables the display areas **(9)**. The label of the button changes accordingly.

7.7.8.3 ShrdResL general preview

ShrdResL general preview



(1) Enabled operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

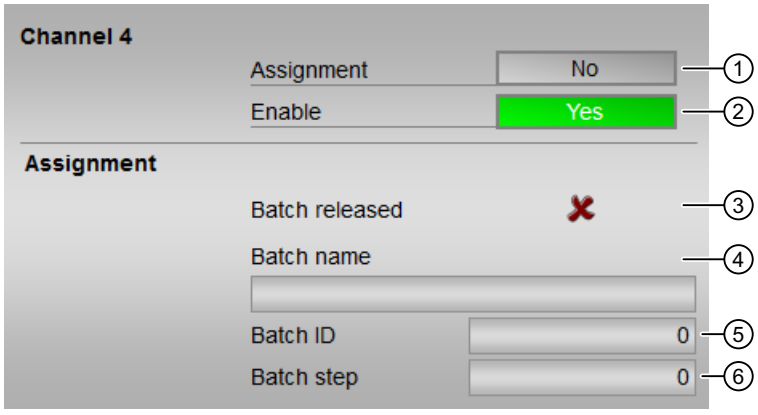
- **Green check mark**: The OS operator can control this parameter.
- **Gray check mark**: The OS operator cannot control this parameter at this time due to the process.
- **Red cross**: The OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm`).

The following enabled operation is shown here:

- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248).

7.7.8.4 ShrdResL preview

ShrdResL preview



Each of the eight channels has its own preview. The previews of the individual channels are identical. The preview shown in this graphic is of channel 4.

(1) Allocation

The allocation status of the channel is displayed in this area.

- "No": Display for `Occ_1` = 0
- "Requested": Display for `Occ_1` = 1and `ActChnNo` = 0
- "Active": Display for `ActChnNo` = 1

(2) Enable

The status of the enable is displayed in this area.

- "Yes": Display for `ChnEn` = 1
- "No": Display for `ChnEn` = 0

(3) Release batch

This area displays if the block is released for operation via SIMATIC BATCH (`BatchEn` = 1).

(4) Batch name

This area displays the name of the batch that is currently running (`Batchname`).

(5) Batch ID

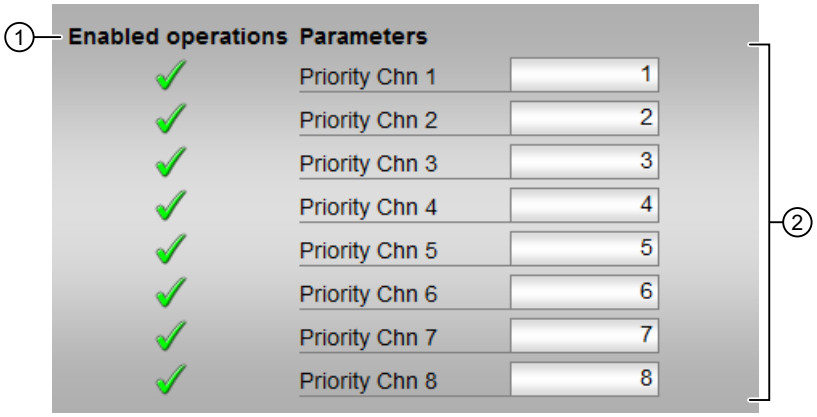
This area displays the identification number of the batch that is currently running (`BatchID`).

(6) Batch step

This area displays the step number of the batch that is currently running (`StepNo`).

7.7.8.5 ShrdResL parameter view

Parameter view of ShrdResL



(1) Enabled operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for operation enable:

- **Green check mark**: The OS operator can control this parameter.
- **Gray check mark**: The OS operator cannot control this parameter at this time due to the process.
- **Red cross**: The OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm`).

(2) Parameters

In this area, you can change the parameters and thereby influence the behavior of ShrdResL. You can find more information on this in the section Changing values (Page 253).

You can influence the following parameters:

- "Priority Chn 1": Priority for channel 1
- "Priority Chn 2": Priority for channel 2
- "Priority Chn 3": Priority for channel 3
- "Priority Chn 4": Priority for channel 4
- "Priority Chn 5": Priority for channel 5
- "Priority Chn 6": Priority for channel 6
- "Priority Chn 7": Priority for channel 7
- "Priority Chn 8": Priority for channel 8

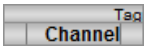
7.7.8.6 Block icon for ShrdResL

Block icons for ShrdResL

A variety of block icons are available with the following functions:

- Display active channel
- Process tag type (2 only)
- Memo display (2 only)
- Fixed text (language dependent, 2 only)

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

7.8 ShrdResS - Multiplexer for shared resources (Small)

7.8.1 Description for ShrdResS

Object name (type + number) and family

Type + number: FB 1914

Family: Drives

Area of application for ShrdResS

The block is used for the following applications:

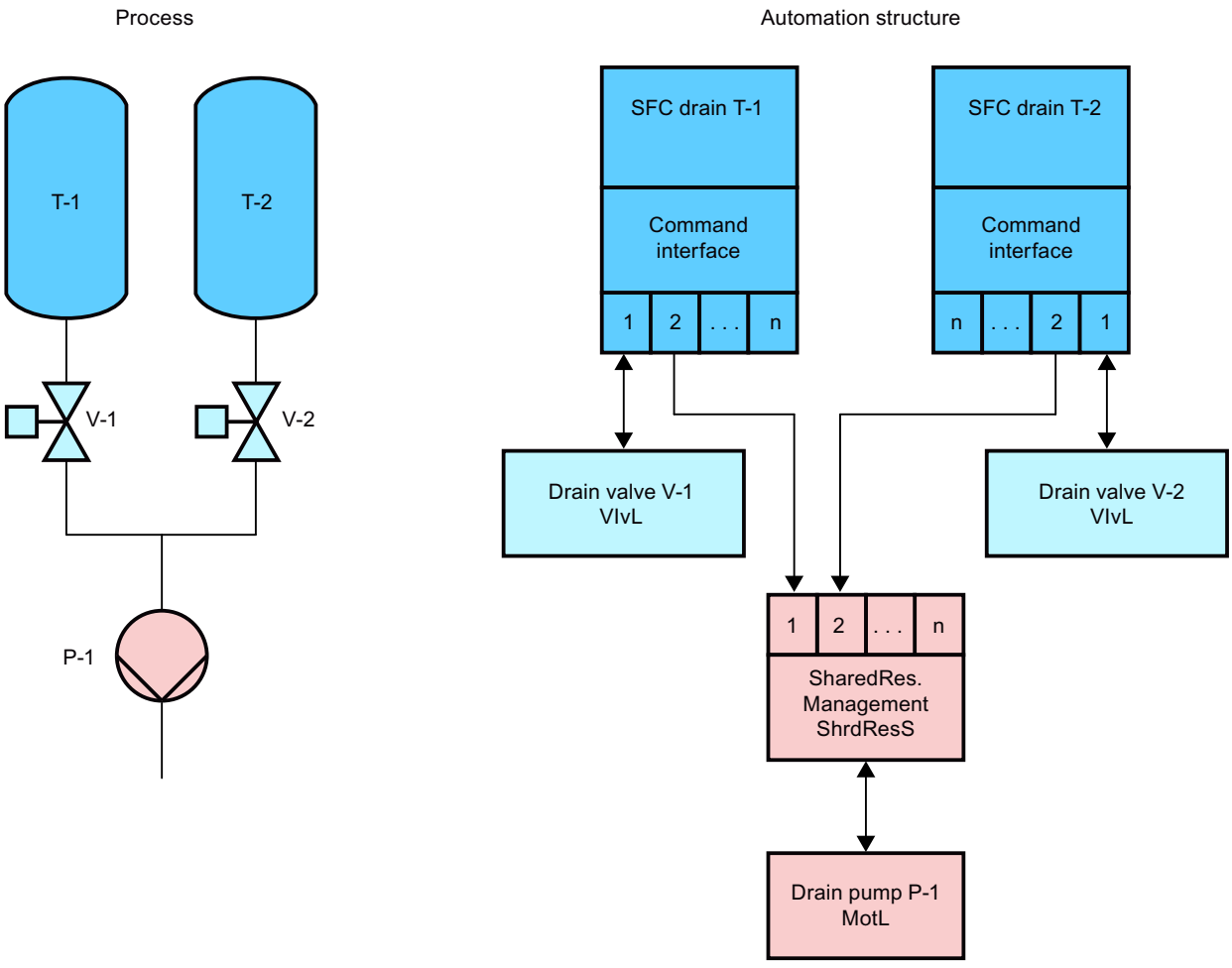
- Organize individual requests for downstream block from upstream applications

How it works

The block coordinates the access from up to four CFC or SFC step sequencers to a technologic block of the families "Drives" or "Dose". These blocks include:

- Motors
- Valves
- Dosers

The block has four channels, each with a standardized command interface.



The block can be cascaded via the fourth channel so that accesses via more than four upstream applications are possible. Use the cascade by interconnecting the output parameter `CasOut` of the first block to the input parameter `CasIn` of the second block. The output interface of the upstream block is then used as channel 4 for the second block.

As soon as a channel is allocated, its command interface is aligned 1-to-1 with the command interface at the output.

The channel with the lower number always has priority with shared allocation. Cascaded blocks are always allocated to channel 4 of the upstream block; they are then in 4th place.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status` parameter

You can find a description for each parameter in section ShrdResS I/Os (Page 1276).

Status bit	Parameter
0	`ReadyIn`, readiness signal of the interconnected technologic block
1	`ChnEn_1`
2	`ChnEn_2`
3	`ChnEn_3`
4	`ChnEn_4`
5	`Ready`, channel allocation is possible
6	`ActChnNo` = 1, 1 = Channel 1 enabled
7	`ActChnNo` = 2, 1 = Channel 2 enabled
8	`ActChnNo` = 3, 1 = Channel 3 enabled
9	`ActChnNo` = 4, 1 = Channel 4 enabled
10	`MultiOcc`, 1 = Request for more than one channel
11	`Cascaded`, 1 = 4. Channel is cascaded
12	`BatchEn`
13	`Occupied`
14 - 20	Not used
21	`BaEn_1`
22	`BaEn_2`
23	`BaEn_3`
24	`BaEn_4`
25	`Occ_1`
26	`Occ_2`
27	`Occ_3`
28	`Occ_4`
29 - 31	Not used

See also

ShrdResS modes (Page 1271)
ShrdResS functions (Page 1272)
ShrdResS error handling (Page 1275)
ShrdResS messaging (Page 1275)
ShrdResS block diagram (Page 1284)

7.8.2 ShrdResS modes

ShrdResS operating modes

This block does not have any modes.

See also

Description for ShrdResS (Page 1269)
ShrdResS functions (Page 1272)
ShrdResS error handling (Page 1275)
ShrdResS messaging (Page 1275)
ShrdResS I/Os (Page 1276)
ShrdResS block diagram (Page 1284)

7.8.3 ShrdResS functions

Functions of ShrdResS

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

If the block is cascaded, the standard view of the ShrdResS block interconnected to `CasIn` can be opened in addition to the freely configured faceplate.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

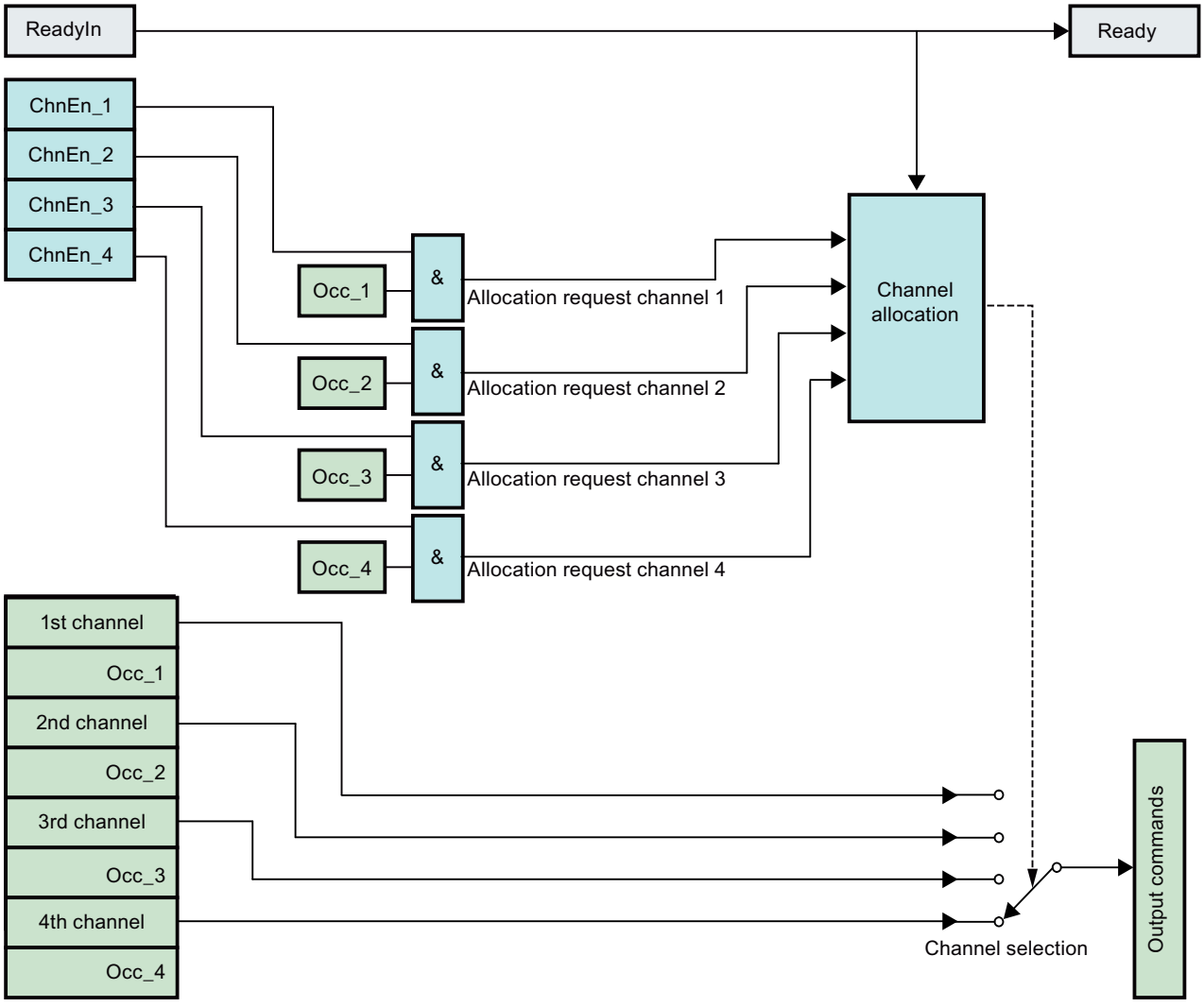
Bit	Function
0	Set startup characteristics (Page 137)

Readiness signal

In order to assign one of the channels, the ready signal must be `ReadyIn` = 1 and at least one channel enable must be `ChnEn_x = 1` (x = 1...4) . The ready signal is output at the `Ready` output.

Channel management

Overview of the channel management



Allocate/enable channel

A channel can only be allocated if the readiness signal (`ReadyIn` = 1) and the respective channel enabling is available for the allocation `ChnEn_x` = 1.

With `ChnEn_x` = 0 the channel x is disabled and cannot be allocated.

The allocation of a channel by the upstream application is accomplished via the input `Occ_x`. As long as this input is 1, the channel is allocated and enabled. The applied commands are aligned 1-to-1 with the output command interface.

The number of the allocated channel is displayed at the output `ActChnNo` (INT format). If no channel is allocated, the output is 0.

Enable/disable channel

If no channel enabling via `Occ_x` = 0 occurs or if the channel is disabled via `ChnEn_x` = 0, the next highest priority channel is automatically switched to the command outputs. If there is no other allocation request (every `Occ_x` = 0), the commands of the output interface are reset to the default values.

Channel prioritization

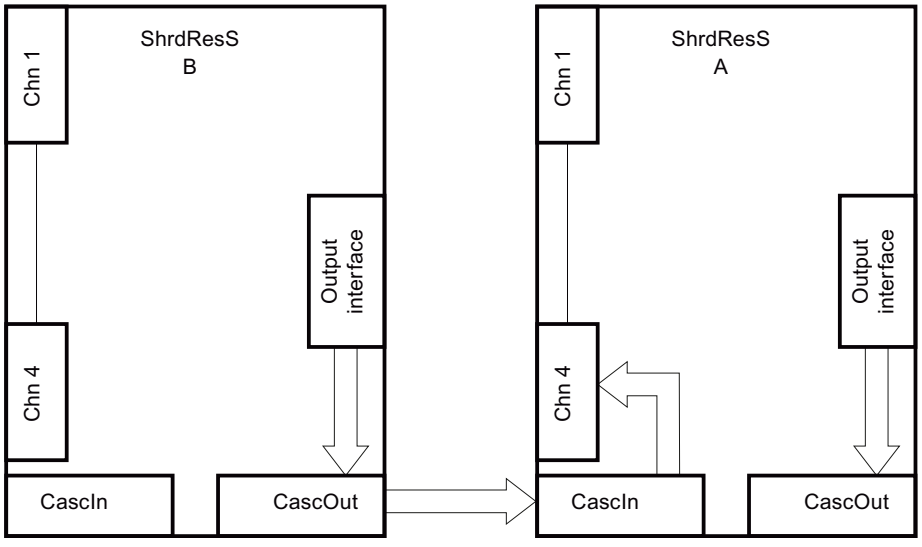
Channel 1 has the highest priority, channel 4 has the lowest. If the input `Occ_x` is set at multiple enabled channels, the channel with the highest priority is allocated and the output `MultiOcc` = 1 is set.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Cascading

For cascading, the output `CasOut` of a ShrdResS block B must be connected with the input `CasIn` of the downstream ShrdResS block A. Therefore, the fourth channel of the downstream block ShrdResS A is allocated by the ShrdResS B block connected to `CasIn`. The command interface of the fourth channel is tracked in this case to the output interface of the connected ShrdResS B block.



Any values applied via interconnection at the 4th channel of block A are not taken into account in the block code during cascading.

See also

Description for ShrdResS (Page 1269)

ShrdResS modes (Page 1271)

7.8.4 ShrdResS error handling

ShrdResS error handling

Please refer to the section Error handling (Page 119) in the basic instructions.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block will not be processed
0	There is no error.
40	Interconnection error on `CasIn`, incorrect block type interconnected

See also

7.8.5 ShrdResS messaging

Messaging

This block does not offer messaging.

See also

Description for ShrdResS (Page 1269)
ShrdResS modes (Page 1271)
ShrdResS functions (Page 1272)
ShrdResS error handling (Page 1275)
ShrdResS I/Os (Page 1276)
ShrdResS block diagram (Page 1284)

7.8.6 ShrdResS I/Os

I/Os of ShrdResS

Input parameters

Parameter	Description	Type	Default
AutMod_1*	1= "Automatic mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutMod_2*	1= "Automatic mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutMod_3*	1= "Automatic mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutMod_4*	1= "Automatic mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BaEn_1	1 = Enable allocation for channel 1	BOOL	0
BaEn_2	1 = Enable allocation for channel 2	BOOL	0
BaEn_3	1 = Enable allocation for channel 3	BOOL	0
BaEn_4	1 = Enable allocation for channel 4	BOOL	0
BaID_1	Batch number for channel 1	DWORD	16#00000000
BaID_2	Batch number for channel 2	DWORD	16#00000000
BaID_3	Batch number for channel 3	DWORD	16#00000000
BaID_4	Batch number for channel 4	DWORD	16#00000000
BaName_1	Batch designation for channel 1	S7-String	
BaName_2	Batch designation for channel 2	S7-String	
BaName_3	Batch designation for channel 3	S7-String	

Parameter	Description	Type	Default
BaName_4	Batch designation for channel 4	S7-String	
CasIn	Input for cascade, to be interconnected with the output parameter CasOut of the preceding ShrdResS block	ANY	
ChnEn_1	1 = Channel 1 is enabled and can be allocated 0 = Channel 1 is blocked and cannot be allocated.	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
ChnEn_2	1 = Channel 2 is enabled and can be allocated 0 = Channel 2 is blocked and cannot be allocated.	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
ChnEn_3	1 = Channel 3 is enabled and can be allocated 0 = Channel 3 is blocked and cannot be allocated.	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
ChnEn_4*	1 = Channel 4 is enabled and can be allocated 0 = Channel 4 is blocked and cannot be allocated.	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
ChnCmd_1	Channel 1: Command interface (reserved)	STRUCT	
ChnCmd_2	Channel 2: Command interface (reserved)	STRUCT	
ChnCmd_3	Channel 3: Command interface (reserved)	STRUCT	
ChnCmd_4	Channel 4: Command interface (reserved)	STRUCT	
Ctrl01_1*	1. Control command for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl02_1*	2. Control command for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl03_1*	3. Control command for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl04_1*	4. Control command for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl05_1*	5. Control command for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Ctrl06_1*	6. Control command for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl01_2*	1. Control command for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl02_2*	2. Control command for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl03_2*	3. Control command for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl04_2*	4. Control command for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl05_2*	5. Control command for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl06_2*	6. Control command for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl01_3*	1. Control command for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl02_3*	2. Control command for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl03_3*	3. Control command for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl04_3*	4. Control command for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl05_3*	5. Control command for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl06_3*	6. Control command for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Ctrl01_4*	1. Control command for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl02_4*	2. Control command for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl03_4*	3. Control command for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl04_4*	4. Control command for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl05_4*	5. Control command for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl06_4*	6. Control command for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CtrlW1_1*	1. Available user control word for channel 1	DWORD	16#00000000
CtrlW1_2*	1. Available user control word for channel 2	DWORD	16#00000000
CtrlW1_3*	1. Available user control word for channel 3	DWORD	16#00000000
CtrlW1_4*	1. Available user control word for channel 4	DWORD	16#00000000
CtrlW2_1*	2. Available user control word for channel 1	DWORD	16#00000000
CtrlW2_2*	2. Available user control word for channel 2	DWORD	16#00000000
CtrlW2_3*	2. Available user control word for channel 3	DWORD	16#00000000
CtrlW2_4*	2. Available user control word for channel 4	DWORD	16#00000000
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1272)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
ManMod_1*	1 = "Manual mode" via interconnection for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
ManMod_2*	1 = "Manual mode" via interconnection for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManMod_3*	1 = "Manual mode" via interconnection for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManMod_4*	1 = "Manual mode" via interconnection for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ModLi_1*	1 = Control via interconnection or SFC for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ModLi_2*	1 = Control via interconnection or SFC for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ModLi_3*	1 = Control via interconnection or SFC for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ModLi_4*	1 = Control via interconnection or SFC for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occ_1	1 = Occupied Is also used as a control signal for the channel allocation for channel 1	BOOL	0
Occ_2	1 = Occupied Is also used as a control signal for the channel allocation for channel 2	BOOL	0
Occ_3	1 = Occupied Is also used as a control signal for the channel allocation for channel 3	BOOL	0
Occ_4*	1 = Occupied Is also used as a control signal for the channel allocation for channel 4	BOOL	0
ReadyIn	Readiness signal: 1 = Signal for channel enabling active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstLi_1*	1 = Resetting via interconnection for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstLi_2*	1 = Resetting via interconnection for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
RstLi_3*	1 = Resetting via interconnection for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstLi_4*	1 = Resetting via interconnection for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SelFpRes	Selection for faceplate resource	ANY	
SP_Ex_1	1 = Select external setpoint (via interconnection) for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_Ex_2	1 = Select external setpoint (via interconnection) for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_Ex_3	1 = Select external setpoint (via interconnection) for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_Ex_4	1 = Select external setpoint (via interconnection) for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_In_1*	1 = Select internal setpoint (via interconnection) for channel 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_In_2*	1 = Select internal setpoint (via interconnection) for channel 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_In_3*	1 = Select internal setpoint (via interconnection) for channel 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_In_4*	1 = Select internal setpoint (via interconnection) for channel 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP1Ext_1*	1. External setpoint for channel 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP2Ext_1*	2. External setpoint for channel 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP1Ext_2*	1. External setpoint for channel 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
SP2Ext_2*	2. External setpoint for channel 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP1Ext_3*	1. External setpoint for channel 3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP2Ext_3*	2. External setpoint for channel 3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP1Ext_4*	1. External setpoint for channel 4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP2Ext_4*	2. External setpoint for channel 4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
StepNo_1	Step number for channel 1	DWORD	16#00000000
StepNo_2	Step number for channel 2	DWORD	16#00000000
StepNo_3	Step number for channel 3	DWORD	16#00000000
StepNo_4	Step number for channel 4	DWORD	16#00000000

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ActChnNo	Displaying the allocated channel	INT	0
AutMod	1= "Automatic mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
Cascaded	1 = Block is cascaded	BOOL	0
CasOut	1 = Output parameter for cascade formation, to be interconnected with the input parameter CasIn of the following ShrdResS block	DWORD	16#00000000
Ctrl01	1. Control command	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Ctrl02	2. Control command	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl03	3. Control command	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl04	4. Control command	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl05	5. Control command	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl06	6. Control command	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CtrlW1	1. Available user control word	DWORD	16#00000000
CtrlW2	2. Available user control word	DWORD	16#00000000
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see ShrdResS error handling (Page 1275)	INT	-1
ManMod	1 = "Manual mode" via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ModLi	1 = Control via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MultiOcc	1 = More than one channel request is present	BOOL	0
Occupied	1 = Occupied Is also used as a control signal for the channel allocation	BOOL	0
Ready	1 = Active readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstLi	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP_ExtLi	1 = Select external setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
SP_IntLi	1 = Select internal setpoint (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP1Ext	1. External setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP2Ext	2. External setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Status1	Status word 1 (Page 1269)	DWORD	16#00000000
StepNo	Step number	DWORD	16#00000000

See also

ShrdResS modes (Page 1271)

ShrdResS messaging (Page 1275)

ShrdResS block diagram (Page 1284)

7.8.7 ShrdResS block diagram

ShrdResS block diagram

A block diagram is not provided for this block.

See also

Description for ShrdResS (Page 1269)

ShrdResS modes (Page 1271)

ShrdResS functions (Page 1272)

ShrdResS error handling (Page 1275)

ShrdResS messaging (Page 1275)

ShrdResS I/Os (Page 1276)

7.8.8 Operator control and monitoring

7.8.8.1 ShrdResS views

Views of the ShrdResS block

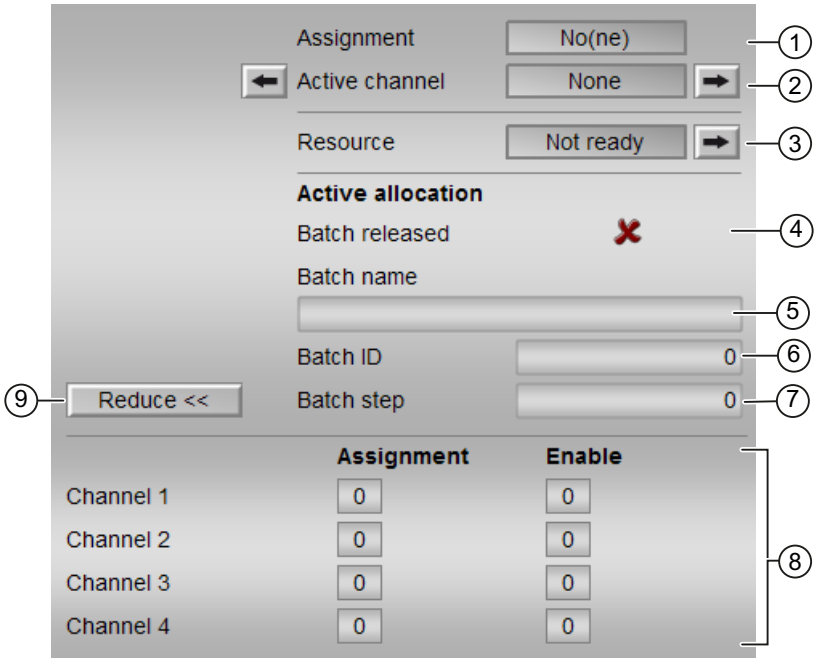
The block ShrdResS provides the following views:

- ShrdResS standard view (Page 1285)
- ShrdResS preview (Page 1287)
- Memo view (Page 298)
- Block icon for ShrdResS (Page 1288)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.8.8.2 ShrdResS standard view

ShrdResS standard view



(1) Allocation

The allocation status is displayed in this area.

- "None": Display for `Status1.Bit 10 = 0` and `Status1.Bit 13 = 0`
- "Requested": Display for `Status1.Bit 10 = 1` and `Status1.Bit 13 = 0`
- "Active": Display for `Status1.Bit 13 = 1`

(2) Active channel

The channel number of the active channel is displayed in this area.

If text is configured for this command (Text 1 in the object properties), it is displayed as additional text and button label for command selection. Additional information is available in the section Labeling of buttons and text (Page 205).

Use the ← button to switch to the standard view of the connected `CasIn` block.

Use the → button to switch to the standard view of the connected `CasOut` block.

(3) Resource

The status of the general enable signal is displayed in this area.

- "Idle": Display for `ReadyIn = 1`
- "Not ready": Display for `ReadyIn = 0`

Use the → button to switch to the standard view of the connected `SelFpRes` block.

(4) Release batch:

This area shows you if the block is released for operation via SIMATIC BATCH (`BatchEn = 1`).

(5) Batch name

This area shows the name of the batch that is currently running (`Batchname`).

(6) Batch ID

This area shows the identification number of the batch that is currently running (`BatchID`).

(7) Batch step

This area shows you the step number of the batch that is currently running (`StepNo`).

(8) Display channel 1-4

This area is only visible when the **(9)** "Expand" button is pressed.

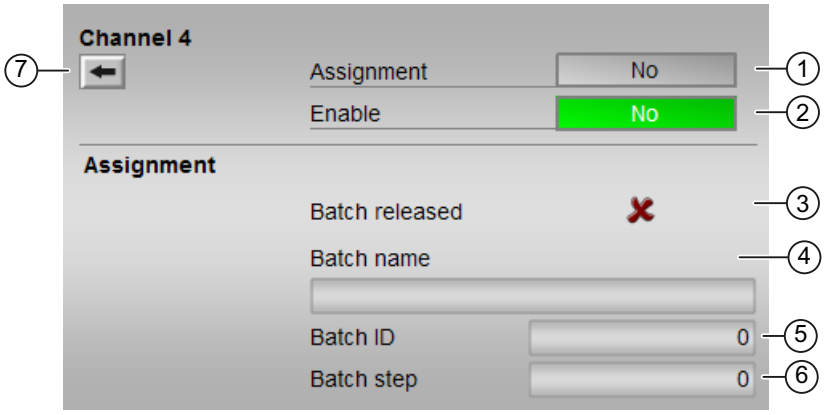
The "Allocation" und "Release" status of channels 1-4 are displayed in this area.

(9) Expand / Collapse

This button enables or disables the display area **(8)**. The label of the button changes accordingly.

7.8.8.3 ShrdResS preview

ShrdResS preview



Each of the four channels has its own preview. The previews of the individual channels are identical except for number **(7)** . The preview described here is based on channel 4.

(1) Allocation

The allocation status of the channel is displayed in this area.

- "No": Display for `Occ_1 = 0`
- "Requested": Display for `Occ_1 = 1` and `ActChnNo = 0`
- "Active": Display for `ActChnNo = 1`

(2) Enable

The status of the enable is displayed in this area.

- "Yes": Display for `ChnEn = 1`
- "No": Display for `ChnEn = 0`

(3) Release batch:

This area shows you if the block is released for operation via SIMATIC BATCH (`BatchEn = 1`).

(4) Batch name

This area shows the name of the batch that is currently running (`Batchname`).

(5) Batch ID

This area shows the identification number of the batch that is currently running (`BatchID`).

(6) Batch step

This area shows you the step number of the batch that is currently running (`StepNo`).

(7) Button ←

Use the ← button to switch to the standard view of the cascaded ShrdResS block. This button is only available for channel 4.

7.8.8.4 Block icon for ShrdResS

Block icons for ShrdResS

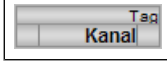
A variety of block icons are available with the following functions:

- Display active channel
- Process tag type (2 only)
- Memo display (2 only)
- Fixed text (language dependent, 2 only)

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

7.9 Vlv2WayL - Two-way valve

7.9.1 Description of Vlv2WayL

Object name (type + number) and family

Type + number: FB 1897

Family: Drives

Area of application for Vlv2WayL

The block is used for the following applications:

- Control of multi-way valves with up to three switching positions. One of these positions is the neutral position (de-energized position)
- Control of three individual valves (valve network) to implement a 2-way valve circuit with neutral position (de-energized position)

How it works

The multi-way valve (or valve network) is controlled via position 0 (neutral position), position 1 (way 1), or position 2 (way 2). Various inputs are available for controlling the positions.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the Vlv2WayL block, the Advanced Process Library contains a template for process tag types as an example with an application scenario for this block.

Example of process tag types:

- Two-way valve (Valve2Way) (Page 2342)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section Vlv2WayL I/Os (Page 1306).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutoAct.Value`
6	`LocalAct.Value`
7	0: Open padlock in the block icon 1: Closed padlock in the block icon
8	"Open"/"Closed" command for `V0`
9	"Open"/"Closed" command for `V1`
10	"Open"/"Closed" command for `V2`
11	Feedback error without control change
12	Feedback error due to control change
13	`BypProt` active
14	Invalid signal status
15	Mode switchover error
16	1 = `Intlock` is active
17	1 = `Permit` is active
18	1 = `Protect` is active
19	Display „Force Pos0“
20	Display „Force Pos1“
21	Display „Force Pos2“
22	Feedback for Pos0 OK
23	Feedback for Pos1 OK
24	Feedback for Pos2 OK
25	Feedback for current position OK
26	Automatic preview Pos0
27	Automatic preview Pos1
28	Automatic preview Pos2
29	`SafeV0`
30	`SafeV1`
31	`SafeV2`

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	Forcing active

Status bit	Parameter
2	Display for interlocks in block icon
3	`WarnAct.Value`
4	External error generated by FaultExt or external control system fault from CSF with set Feature bit 18 Activating error state for external process control error CSF (Page 150)
5	For the status display error in the valve
6	Current monitoring time `P0` is visible
7	Current monitoring time `V0` is visible
8	Current monitoring time `V1` is visible
9	Current monitoring time `V2` is visible
10	Current monitoring time `P1` is visible
11	Current monitoring time `P2` is visible
12	1 = Input parameter `FbkP1` is connected and Feature bit 12 = 1
13	1 = Input parameter `FbkP2` is connected and Feature bit 12 = 1
14	1 = Input parameter `FbkP0` is connected
15	1 = Input parameter `FbkV0` is connected and Feature bit 12 = 0
16	1 = Input parameter `FbkV1` is connected and Feature bit 12 = 0
17	1 = Input parameter `FbkV2` is connected and Feature bit 12 = 0
18	Reset request in automatic
19	1 = No impact of input signals on "local mode" with `LocalSetting` = 2 and `LocalSetting` = 4
20	`CtrlV0.Value`
21	`CtrlV1.Value`
22	`CtrlV2.Value`
23	`FbkV0Out.Value`
24	`FbkV1Out.Value`
25	`FbkV2Out.Value`
26	`FbkP0Out.Value`
27	Feedback `V0` (`FbkV0`), for OS display only
28	Feedback `V1` (`FbkV1`), for OS display only
29	Feedback `V2` (`FbkV2`), for OS display only
30	Bypass information from previous function block
31	`MS_RelOp`

Status word allocation for `Status3` parameter

Status bit	Parameter
0	"Interlock" button is enabled
1	"Permission" button is enabled
2	"Protection" button is enabled
3	`Pos0Out`
4	Travel in position 0
5	Monitoring error in position 0

Status bit	Parameter
6	`Pos1Out`
7	Travel in position 1
8	Monitoring error in position 1
9	`Pos2Out`
10	Travel in position 2
11	Monitoring error in position 2
12	Preview position 0 control `CtrlV0`
13	Preview position 0 control `CtrlV1`
14	Preview position 0 control `CtrlV2`
15	Preview position 1 control `CtrlV0`
16	Preview position 1 control `CtrlV1`
17	Preview position 1 control `CtrlV2`
18	Preview position 2 control `CtrlV0`
19	Preview position 2 control `CtrlV1`
20	Preview position 2 control `CtrlV2`
21	Preview for automatic control `CtrlV0`
22	Preview for automatic control `CtrlV1`
23	Preview for automatic control `CtrlV2`
24	`UserAna1` interconnected
25	`UserAna2` interconnected
26	Show automatic preview in the standard view
27	Not used
28	`GrpErr.Value`
29	`RdyToStart.Value`
30	1 = with position feedback messages (Feature bit 12)
31	Not used

Status word allocation for `Status4` parameter

Status bit	Parameter
0	Effective signal `1` of the message block connected via `EventTsIn`
1	Effective signal 2 of the message block connected via `EventTsIn`
2	Effective signal 3 of the message block connected via `EventTsIn`
3	Effective signal 4 of the message block connected via `EventTsIn`
4	Effective signal 5 of the message block connected via `EventTsIn`
5	Effective signal 6 of the message block connected via `EventTsIn`
6	Effective signal 7 of the message block connected via `EventTsIn`
7	Effective signal 8 of the message block connected via `EventTsIn`
8 - 15	Effective signal 9..16 of the message block connected via `EventTsIn`
16 - 22	Not used
23	Hidden bypass signal in Permit
24	Hidden bypass signal in interlock

Status bit	Parameter
25	Hidden bypass signal in Protect
26	`Feature2` bit 2: Separate bypass signal
27	1 = Input parameter `CtrlV0ChnST` is interconnected
28	1 = Input parameter `CtrlV1ChnST` is interconnected
29	1 = Input parameter `CtrlV2ChnST` is interconnected
30 - 31	Not used

See also

Vlv2WayL functions (Page 1295)

Vlv2WayL messaging (Page 1305)

Vlv2WayL block diagram (Page 1317)

Vlv2WayL error handling (Page 1303)

Vlv2WayL modes (Page 1294)

Resetting the block in case of interlocks or errors (Page 43)

7.9.2 Vlv2WayL modes

Vlv2WayL operating modes

The block can be operated using the following modes:

- Local mode (Page 79)
- Automatic mode (Page 75)
- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) section.

Valve actions you can control in "local mode":

- Travel to neutral position (`Pos0Local` = 1)
- Moving to position 1 (`Pos1Local` = 1)
- Moving to position 2 (`Pos2Local` = 1).

A block operated in "local mode" is controlled either by "local" signals (input parameters `Pos0Local` = 1, `Pos1Local` = 1 and `Pos2Local` = 1) or by feedback signals (input

parameters `FdbV0`, `FdbV1`, `FdbV2` and `FdbP0`; if no position can be assigned, the last valid position is accepted). Configuration takes place via the input parameter `LocalSetting`.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) section.

Valve actions you can control in "automatic mode":

- Travel to neutral position (`Pos0Aut` = 1)
- Moving to position 1 (`Pos1Aut` = 1)
- Moving to position 2 (`Pos2Aut` = 1).

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

Valve actions you can control in "manual mode":

- Travel to neutral position (`Pos0Man` = 1)
- Moving to position 1 (`Pos1Man` = 1)
- Moving to position 2 (`Pos2Man` = 1).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Vlv2WayL block diagram (Page 1317)

Vlv2WayL I/Os (Page 1306)

Vlv2WayL messaging (Page 1305)

Vlv2WayL error handling (Page 1303)

Vlv2WayL functions (Page 1295)

Description of Vlv2WayL (Page 1290)

7.9.3 Vlv2WayL functions

Functions of Vlv2WayL

The functions for this block are listed below.

Defining valve positions for individual valves

The control outputs for position 1 and position 2 can be selected individually with `DefPos1` and `DefPos2`:

Route 1 or route 2	Control outputs		
`DefPos1` or `DefPos2`	**Valve `V0` (`CtrlV0`)**	**Valve `V1` (`CtrlV1`)**	**Valve `V2` (`CtrlV2`)**
0	closed	closed	closed
1	closed	closed	open
2	closed	open	closed
3	closed	open	open
4	open	closed	closed
5	open	closed	open
6	open	open	closed
7	open	open	open

Position 0 is the neutral position (de-energized state) and cannot be configured. In position 0 all control outputs are de-energized (`CtrlVx` = 0).

Output signal as a pulse signal or static signal

This block provides the standard function Output signal as a static signal or pulse signal (Page 51). In addition to the static control outputs `CtrlV0`, `CtrlV1`, `CtrlV2`, `Pos0Out`, `Pos1Out`, `Pos2Out`, the block also has pulse outputs `P_CtrlV0`, `P_CtrlV1`, `P_CtrlV2`, `P_CtrlP0`, `P_CtrlP1`, `P_CtrlP2` which are output depending on the static control outputs.

Neutral position

This block provides the standard function Neutral position for motors, valves and controllers (Page 48). The neutral position (de-energized state) is set individually using parameters `SafeV0`, `SafeV1`, `SafeV2` for each valve (`CtrlV0`, `CtrlV1`, `CtrlV2`):

- `SafeVx` = 0 means that at `CtrlVx` = 0 the valve drive closes and at `CtrlVx` = 1 it opens (de-energized state is "closed")
- `SafeVx` = 1 means that at `CtrlVx` = 0 the valve drive opens and at `CtrlVx` = 1 it closes(de-energized state is "open")

Specify warning times for control functions

This block provides the standard function Specifying warning times for control functions at motors and valves (Page 51). The warning signal is output before the valve moves into position 1 or position 2 . No warning signal is output for position 0 (neutral position).

You can generate warning signals when, for example, valves open. Warning signals can be generated in the following modes:

- Manual and automatic mode for motors, valves and dosers (Page 75) (Input parameter `WarnTiMan`)
- Manual and automatic mode for motors, valves and dosers (Page 75) (Input parameter `WarnTiAut`)

You specify the warning times in seconds using the input parameters `WarnTiMan` and `WarnTiAut`. If, for example, a valve opens then, this is displayed at the output parameter with `WarnAct` = 1. The valve then opens after the set warning time has expired and `WarnAct` then returns to 0.

A corresponding warning is not output if the warning times (`WarnTiMan` or `WarnTiAut`) are specified with a smaller value than the `SampleTime` parameter.

Feedback monitoring

This block provides the standard function Monitoring the feedbacks (Page 97).

The monitoring of the feedback signal is dependent on Feature bit 12 "Position feedback signals are active (Page 168)".

Valve feedback signals are active (Feature bit 12 = 0):

Startup characteristic monitoring is individually set up for every output signal `CtrlV0`, `CtrlV1`, `CtrlV2` via parameters `MonTiV0Dynamic`, `MonTiV1Dynamic` and `MonTiV2Dynamic` and is set for position 0 via `MonTiP0Dynamic`. The parameter `MonTiStatic` monitors compliance with the position.

The position feedback signals (`FbkP1`, `FbkP2`) are not used to monitor the control outputs.

Note

The monitoring function does not take into consideration the neutral positions (`SafeV0`, `SafeV1`, `SafeV2`), which means the feedback messages `FbkV0`, `FbkV1`, `FbkV2` must correspond to the `CtrlV0`, `CtrlV1`, `CtrlV2` controls (e.g., `CtrlV0` = 1 means the feedback `FbkV0` is monitored for "1").

`FbkP0` must not occur for positions 1 or 2; at position 0, `FbkV0`, `FbkV1` and `FbkV2` must not occur.

If there are several feedback messages for position 0 (e.g. with a valve network), these must be combined using an upstream `AND` block at `FbkP0`.

Position feedback signals are active (Feature bit 12 = 1):

Startup characteristic monitoring is individually set up for the position feedback signals `Pos1Out` and `Pos2Out` via the parameters `MonTiP1Dynamic` and `MonTiVP2Dynamic` and for position 0 via `MonTiP0Dynamic`. The `MonTiStatic` parameter monitors compliance with the position.

The position feedback signals `FbkP0`, `FbkP1` and `FbkP2` are used to monitor the output signals `Pos0Out`, `Pos1Out` and `Pos2Out`.

Note

The separate valve feedback signals(`FbkV0`, `FbkV1`, `FbkV2`) are not used to monitor the control outputs.

Disabling feedback

This block provides the standard function Disabling feedback for valves (Page 99). Feedback monitoring can be deactivated separately for each feedback with `NoFbkV0, NoFbkV1, NoFbkV2, NoFbkP0, NoFbkP1, NoFbkP2`.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Forcing operating modes

This block provides the standard function Forcing operating modes (Page 41). The inputs `Pos0Force`, `Pos1Force`, `Pos2Force` force the block into position 0, position 1 or position 2.

Simulating signals

This block provides the standard function Simulating signals (Page 58)

Interlocks

This block provides the following interlocks:

- Activation enable
- Interlock without reset ("Interlock")
- Interlock with reset ("Protection")

Refer to the Interlocks (Page 99) section for more on this.

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Resetting the block in case of interlocks or errors

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See Vlv2WayL error handling (Page 1303)

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `MonDynV0` only with valve feedback signals (Feature bit 12 = 0)
- `MonDynV1` only with valve feedback signals (Feature bit 12 = 0)
- `MonDynV2` only with valve feedback signals (Feature bit 12 = 0)
- `MonDynP0`
- `MonDynP1`only with position feedback signals (Feature bit 12 = 1)
- `MonDynP2`only with position feedback signals (Feature bit 12 = 1)
- `MonStaV0`
- `MonStaV1`
- `MonStaV2`
- `MonStaP0`
- `FaultExt`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `FbkV0Out.ST` only with valve feedback signals (Feature bit 12 = 0)
- `FbkV1Out.ST` only with valve feedback signals (Feature bit 12 = 0)
- `FbkV2Out.ST` only with valve feedback signals (Feature bit 12 = 0)
- `FbkP0Out.ST`
- `FbkP1Out.ST`only with position feedback signals (Feature bit 12 = 1)
- `FbkP2Out.ST`only with position feedback signals (Feature bit 12 = 1)
- `LocalLi.ST`
- `Pos0Local.ST`
- `Pos1Local.ST`
- `Pos2Local.ST`
- `V0P0ChnST.ST`

- `V1P1ChnST.ST`
- `V2P2ChnST.ST`
- `Pos0Aut.ST` (only if `Feature2.Bit10` = 1)
- `Pos1Aut.ST` (only if `Feature2.Bit10` = 1)
- `Pos2Aut.ST` (only if `Feature2.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position in the "Automatic" mode:

- `Pos0Aut.ST`
- `Pos1Aut.ST`
- `Pos2Aut.ST`

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	1 = Operator can switch to "local mode"
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can switch to position 0
5	1 = Operator can switch to position 1
6	1 = Operator can switch to position 2

Bit	Function
7	1 = Operator can reset the valve
8	1 = Operator can define the monitoring time for startup
9	1 = Operator can define the monitoring time for the end position
10	1 = Operator can activate the monitoring time function (Bit 8 - 9)
11	Not used
12	1 = Operator can activate the Release for maintenance function
13 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable reactions with the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
4	Setting switch or button mode (Page 166)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
12	Position feedback signals are active (Page 168)
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)
18	Activating error state for external process control error CSF (Page 150)
20	Disable calculation of impulse control in LocalSetting 2 and 4 (Page 185)
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
27	Interlock display with LocalSetting 2 or 4 (Page 177)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

In pushbutton operation (Bit 4 = 0) the automatic commands in "automatic mode" are latching, in other words `Pos0Aut`, `Pos1Aut`, `Pos2Aut` can be reset to 0 after switching to the selected position. In "manual" and "local" modes, however, the automatic commands are not saved and in the absence of automatic commands the position is tracked.

In switching mode (Bit 4 = 1) positions 1 and 2 are selected by static signals via inputs `Pos1Aut` and `Pos2Aut`. If inputs `Pos1Aut` and `Pos2Aut` are not set, the block switches to position 0. Control via `Pos0Aut` is not needed.

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
4	Setting switch or button mode for local commands (Page 180)
5	Evaluation of the signal status of the interlock signals (Page 141)
8	Forcing operating modes in the "Local" mode (Page 183)
10	Considering bad quality of automatic commands or external values (Page 185)

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- `Pos0Man`
- `Pos1Man`
- `Pos2Man`

See also

Description of Vlv2WayL (Page 1290)

Vlv2WayL messaging (Page 1305)

Vlv2WayL I/Os (Page 1306)
Vlv2WayL block diagram (Page 1317)
Vlv2WayL modes (Page 1294)
EventTs functions (Page 1634)

7.9.4 Vlv2WayL error handling

Vlv2WayL error handling

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Control system fault (CSF)
- Invalid input signals

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.
41	The value for the `LocalSetting` I/O is not within the approved limit of 0 to 4.
42	`LocalSetting` = 0 and `LocalLi` = 1
51	For `ModLiOp` = 1: • `AutModLi` = 1 and `ManModLi` = 1 If "local mode" is enabled: • `Pos0Local` = 1 and `Pos1Local` = 1 • `Pos0Local` = 1 and `Pos2Local` = 1 • `Pos1Local` = 1 and `Pos2Local` = 1 If "automatic mode" is enabled: • `Pos0Aut` = 1 and `Pos1Aut` = 1 • `Pos0Aut` = 1 and `Pos2Aut` = 1 • `Pos1Aut` = 1 and `Pos2Aut` = 1 Generally: • `Pos0Force` = 1 and `Pos1Force` = 1 • `Pos0Force` = 1 and `Pos2Force` = 1 • `Pos1Force` = 1 and `Pos2Force` = 1
52	`LocalAct` = 1 and `LocalSetting` = 2 or 4 and `SimOn` = 1

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

Invalid input signals

This error can be output by the block, see the section Error handling (Page 119).

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signals	Control reaction with Feature2 bit 3 =1
Local: `Localsetting` = 1 or `Localsetting` = 3	Pushbutton operation for local mode (`Feature2` bit 4 = 0): `Pos1Local` = 1 and `Pos2Local` = 1 or `Pos1Local` = 1 and `Pos0Local` = 1 or `StopLocal` = 1 and `Pos2Local` = 1 Switching mode (`Feature2` bit 4 = 1): `Pos1Local` = 1 and `Pos2Local` = 1	Valve is set to Pos0.
Local: `Localsetting` = 1 or `Localsetting` = 3 and forcing	`Pos1Force` = 1 and `Pos2Force` = 1 or `Pos1Force` = 1 and `Pos0Force` = 1 or `Pos0Force` = 1 and `Pos2Force` = 1	
Forcing and no "local mode"	`Pos1Force` = 1 and `Pos2Force` = 1 or `Pos1Force` = 1 and `Pos0Force` = 1 or `Pos0Force` = 1 and `Pos2Force` = 1	
"Automatic mode" and no forcing	Pushbutton operation (`Feature` bit 4 = 0): `Pos1Aut` = 1 and `Pos2Aut` = 1 or `Pos1Aut` = 1 and `Pos0Aut` = 1 or `Pos0Aut` = 1 and `Pos2Aut` = 1 Switching mode (`Feature`bit 4 = 1): `Pos1Aut` = 1 and `Pos2Aut` = 1	
"Manual mode" and no forcing	`Pos1Man` = 1 and `Pos0Man` = 1 or `Pos1Man` = 1 and `Pos0Man` = 1 or `Pos0Man` = 1 and `Pos2Man` = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

Vlv2WayL block diagram (Page 1317)

Vlv2WayL I/Os (Page 1306)

Vlv2WayL messaging (Page 1305)

Vlv2WayL functions (Page 1295)
Vlv2WayL modes (Page 1294)
Description of Vlv2WayL (Page 1290)

7.9.5 Vlv2WayL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	AS process control message - fault	$$BlockComment$$ Position 0 feedback error (neutral position)
	SIG 2	AS process control message - fault	$$BlockComment$$ Position 1 or 2 feedback error
	SIG 3	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter CSF. If it changes to CSF = 1, a process control fault is triggered (MsgEvId1, SIG 3).

Instance-specific messages

You can use up to three instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 4	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 5	AS process control message - fault	$$BlockComment$$ External message 2
	SIG 6	AS process control message - fault	$$BlockComment$$ External message 3

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`ExtVa104`
5	`ExtVa105`
6	`ExtVa106`
7	`ExtVa107`
8	`ExtVa108`
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters `ExtVa104` ... `ExtVa108` , and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of Vlv2WayL (Page 1290)

Vlv2WayL functions (Page 1295)

Vlv2WayL I/Os (Page 1306)

Vlv2WayL block diagram (Page 1317)

Vlv2WayL error handling (Page 1303)

Vlv2WayL modes (Page 1294)

7.9.6 Vlv2WayL I/Os

I/Os of Vlv2WayL

Input parameters

Parameter	Description	Type	Default
`AutModLi*`	1= "Automatic mode" via: Interconnection or SFC (controlled via `ModLiOp` = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`AutModOp*`	1 = "Automatic mode" via operator (controlled by `ModLiOp` = 0)	BOOL	0

Parameter	Description	Type	Default
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Bypassing interlock in "local mode" and in "simulation"	BOOL	0
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
DefPos1	Output signal parameter setting for position 1	INT	3
DefPos2	Output signal parameter setting for position 2	INT	6
EN	1 = Called block will be processed	BOOL	1
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	

Parameter	Description	Type	Default
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkP0	1 = Feedback for position 0	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#FF
FbkP1	1 = Feedback for position 1	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#FF
FbkP2	1 = Feedback for position 2	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#FF
FbkV0	1 = Feedback for control output CtrlV0	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#FF
FbkV1	1 = Feedback for control output CtrlV1	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#FF
FbkV2	1 = Feedback for control output CtrlV2	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#FF
Feature	I/O for additional functions (Page 1295)	STRUCT • Bit:0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Feature2	I/O for additional functions (Page 1295)	STRUCT • Bit:0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Intlock	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value:BOOL • ST:BYTE	- • 1 • 16#FF
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is active	BOOL	1
LocalLi	1 = Activate "local mode" via plant signals	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
LocalOp*	1 = "Local mode" via operator	BOOL	0
LocalSetting	Properties for the Local mode (Page 79)	INT	0

Parameter	Description	Type	Default
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
ManModOp*	1 = "Manual mode" via OS operator (controlled by ModLiOp = 0)	BOOL	1
ModLiOp	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Monitor	1 = Feedback monitoring	BOOL	1
MonSafePos	1 = Go to neutral position in the event of monitoring errors	BOOL	1
MonTiP0Dynamic*	Monitoring time for position 0 after operation in [s]	REAL	3.0
MonTiP1Dynamic*	Monitoring time for position 1 after operation in [s]	REAL	3.0
MonTiP2Dynamic*	Monitoring time for position 2 after operation in [s]	REAL	3.0
MonTiV0Dynamic*	Monitoring time for feedback errors FdbV0 after operation in [s]	REAL	3.0
MonTiV1Dynamic*	Monitoring time for feedback errors FdbV1 after operation in [s]	REAL	3.0
MonTiV2Dynamic*	Monitoring time for feedback errors FdbV2 after operation in [s]	REAL	3.0
MonTiStatic*	Monitoring time for feedback errors without operation in [s]	REAL	3.0
MS_RelOp*	1= Release for maintenance via OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NoFbkP0	1 = Feedback for position 0 not present	BOOL	0
NoFbkP1	1 = Feedback for position 1 not present	BOOL	0
NoFbkP2	1 = Feedback for position 2 not present	BOOL	0
NoFbkV0	1 = Feedback for control output CtrlV0 not present	BOOL	0
NoFbkV1	1 = Feedback for control output CtrlV1 not present	BOOL	0
NoFbkV2	1 = Feedback for control output CtrlV2 not present	BOOL	0
Occupied	1 = In use by a batch	BOOL	0

Parameter	Description	Type	Default
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 1295)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
Permit	1 = Enable for opening / closing from neutral position 0 = Valve activation not enabled on OS	STRUCT • Value:BOOL • ST:BYTE	- • 1 • 16#FF
Perm_En	1 = Activation enable (enable, Permit parameter) is active	BOOL	1
Pos0Aut*	1 = Select position 0 in "automatic mode"	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos0Force	1 = Force position 0	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos0Local	1 = Select position 0 in "local mode"	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos0Man*	1 = Select position 0 in "manual mode"	BOOL	0
Pos1Aut*	1 = Select position 1 in "automatic mode"	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos1Force	1 = Force position 1	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos1Local	1 = Select position 1 in "local mode"	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos1Man*	1 = Select position 1 in "manual mode"	BOOL	0
Pos2Aut*	1 = Select position 2 in "automatic mode"	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Pos2Force	1 = Force position 2	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos2Local	1 = Select position 2 in "local mode"	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos2Man*	1 = Select position 2 in "manual mode"	BOOL	0
Protect	0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block 1 = Protective interlocking not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Prot_En	1 = Protective interlock (protection, Protect parameter) is active	BOOL	1
PulseWidth*	Pulse width of control signal [s]	REAL	3.0
RstLi*	1 = Reset via interconnection	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SafeV0	Neutral position for valve V0 (CtrlV0): 1= Open 0 = Closed	BOOL	0
SafeV1	Neutral position for valve V1 (CtrlV1): 1= Open 0 = Closed	BOOL	0
SafeV2	Neutral position for valve V2 (CtrlV2): 1= Open 0 = Closed	BOOL	0
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	

Parameter	Description	Type	Default
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	
StepNo	Batch step number	DWORD	16#00000000
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA2unit	Unit of measure for analog auxiliary value 2	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
V0P0ChnST	Signal status of output channel for CtrlV0 or Pos0Out Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
V1P1ChnST	Signal status of output channel for CtrlV1 or Pos1Out Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
V2P2ChnST	Signal status of output channel for CtrlV2 or Pos2Out Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
WarnTiAut	Prewarning for valve movement into position 1 or position 2 in "automatic mode" in [s]	REAL	0.0
WarnTiMan	Prewarning for valve movement into position 1 or position 2 in "manual mode" in [s]	REAL	0.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80

Parameter	Description	Type	Default
CtrlV0	Control output V0	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
CtrlV1	Control output V1	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
CtrlV2	Control output V2	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
CurrMonP0	Current monitoring time P0 [s]	DINT	0
CurrMonP1	Current monitoring time P1 [s]	DINT	0
CurrMonP2	Current monitoring time P2 [s]	DINT	0
CurrMonV0	Current monitoring time V0 [s]	DINT	0
CurrMonV1	Current monitoring time V1 [s]	DINT	0
CurrMonV2	Current monitoring time V2 [s]	DINT	0
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see Vlv2WayL error handling (Page 1303)	INT	-1
FbkP0Out	Feedback from position 0	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
FbkP1Out	Feedback from position 1	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
FbkP2Out	Feedback from position 2	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
FbkV0Out	Control output feedback CtrlV0	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
FbkV1Out	Control output feedback CtrlV1	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
FbkV2Out	Control output feedback CtrlV2	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
`LocalAct`	1 = "Local mode" enabled	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
`LockAct`	1 = Interlock (`Intlock`, `Permit` or `Protect`) is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ManAct`	1 = "Manual mode" enabled	STRUCT • Value:BOOL • ST:BYTE	- • 1 • 16#80
`MonDynP0`	1 = Feedback error for position 0 due to a control change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
`MonDynP1`	1 = Feedback error for position 1 due to a control change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
`MonDynP2`	1 = Feedback error for position 2 due to a control change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
`MonDynV0`	1 = Feedback error for `FdbV0` due to control change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
`MonDynV1`	1 = Feedback error for `FdbV1` due to control change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
`MonDynV2`	1 = Feedback error for `FdbV2` due to control change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
`MonStaP0`	1 = Feedback error for position 0 due to an unexpected feedback change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
`MonStaP1`	1 = Feedback error for position 1 due to an unexpected feedback change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
`MonStaP2`	1 = Feedback error for position 2 due to an unexpected feedback change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
`MonStaV0`	1 = Feedback error `FdbV0` due to unexpected feedback change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MonStaV1	1 = Feedback error FdbV1 due to unexpected feedback change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
MonStaV2	1 = Feedback error FdbV2 due to unexpected feedback change	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
OosAct	1 = Block is "out of service"	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
P_CtrlP0	1 = Pulse signal for moving the valve to position 0	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
P_CtrlP1	1 = Pulse signal for moving the valve to position 1	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
P_CtrlP2	1 = Pulse signal for moving the valve to position 2	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
P_CtrlV0	1 = Pulse signal for moving the valve to route 0 (V0)	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
P_CtrlV1	1 = Pulse signal for moving the valve to route 1 (V1)	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
P_CtrlV2	1 = Pulse signal for moving the valve to route 2 (V2)	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Pos0	1 = Pos0 is reached	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos1	1 = Pos1 is reached	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos2	1 = Pos2 is reached	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos0Out	1 = Position 0 is active	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos1Out	1 = Position 1 is active	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
Pos2Out	1 = Position 2 is active	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1290)	DWORD	16#00000000
Status2	Status word 2 (Page 1290)	DWORD	16#00000000
Status3	Status word 3 (Page 1290)	DWORD	16#00000000
Status4	Status word 4 (Page 1290)	DWORD	16#00000000
WarnAct	1 = Prewarning for valve movement to position 1 or position 2 enabled (parameter WarnTiAut and WarnTiMan)	STRUCT • Value:BOOL • ST:BYTE	- • 0 • 16#80

See also

Vlv2WayL messaging (Page 1305)
Vlv2WayL block diagram (Page 1317)

Vlv2WayL modes (Page 1294)

Error handling (Page 119)

7.9.7 Vlv2WayL block diagram

Vlv2WayL block diagram

A block diagram is not provided for this block.

See also

Vlv2WayL I/Os (Page 1306)

Vlv2WayL messaging (Page 1305)

Vlv2WayL error handling (Page 1303)

Vlv2WayL functions (Page 1295)

Vlv2WayL modes (Page 1294)

Description of Vlv2WayL (Page 1290)

7.9.8 Operator control and monitoring

7.9.8.1 Vlv2WayL views

Views of the Vlv2WayL block

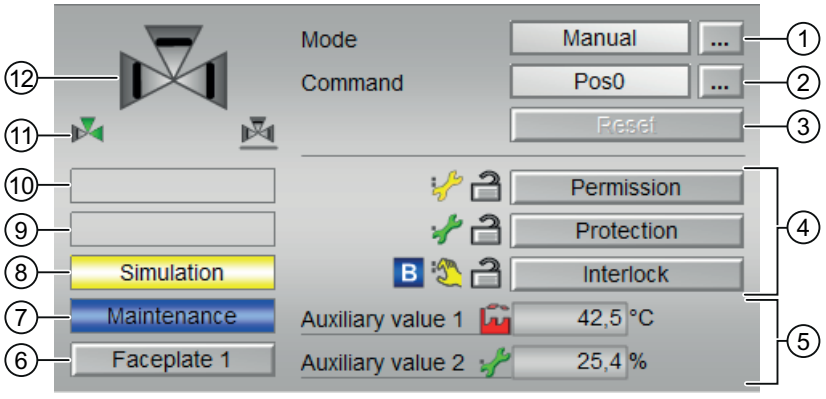
The block Vlv2WayL provides the following views:

- Vlv2WayL standard view (Page 1318)
- Alarm view (Page 296)
- Trend view (Page 299)
- Vlv2WayL parameter view (Page 1321)
- Vlv2WayL preview (Page 1323)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for Vlv2WayL (Page 1328)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.9.8.2 Vlv2WayL standard view

Vlv2WayL standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Selecting the position for 2-way valve

This area shows you the default operating state for the valve. The following states can be shown and executed here:

- "Pos0"
- "Pos1"
- "Pos2"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(3) Resetting the block

Click "Reset" for interlocks or errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(4) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the section Interlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

(5) Display of auxiliary values

This display is only visible when the corresponding block input is connected.

You can use this area to display two auxiliary values that have been configured in the engineering system (ES). You can find additional information on this in the section Displaying auxiliary values (Page 207).

(6) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(7) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

(8) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"
- "Delay"

You will find more detailed information on this in the sections Simulating signals (Page 58) and Display of delay times (Page 250).

(9) Display area for block states

This area provides additional information on the operating state of the block:

- "External error"
- "End position error"
- "Control error"
- "Invalid signal"
- "Changeover error"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97) , Error handling (Page 119) (section "Invalid input signals" and "Mode switchover error") and Motor protection function (Page 99).

(10) Display area for block states

This area provides additional information on the operating state of the block:

- "Force Pos0"
- "Force Pos1"
- "Force Pos2"
- "Request 0/1": A reset to "automatic mode" is expected.

You can find additional information on this in the section Forcing operating modes (Page 41).

(11) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the valve would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

(12) Status display of the valve

You can find additional information on this in the section Block icon for Vlv2WayL (Page 1328).

(13) Neutral position of the valve

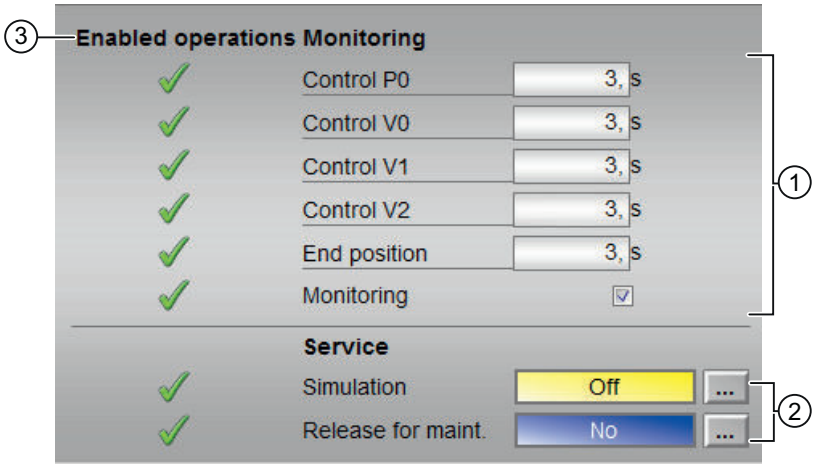
The neutral position of the valve is shown here.

If the neutral position of the valve is "Closed" (SafePos = 0), a gray valve is shown.

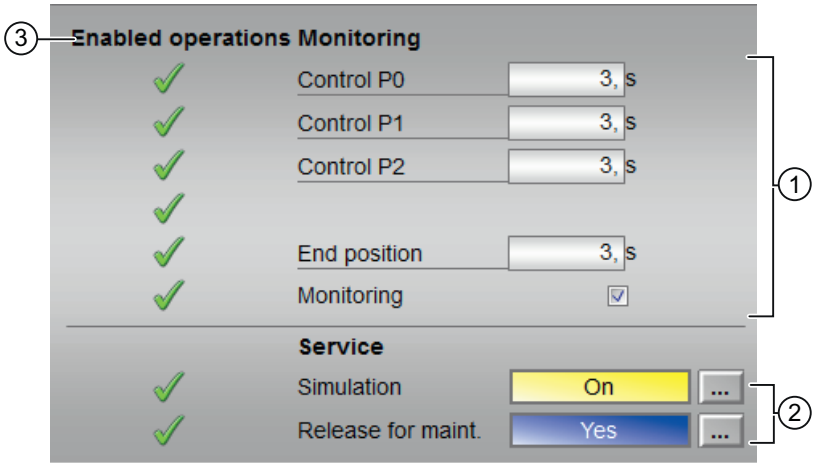
If the neutral position of the valve is "Open" (SafePos = 0), a green valve is shown.

7.9.8.3 Vlv2WayL parameter view

Parameter view of Vlv2WayL



Two-way valve with separate valve feedback signals (Feature bit 12 = 0)



Two-way valve with position feedback signals (Feature bit 12 = 1)

(1) Monitoring

In this area, you change parameters and therefore influence the valve. Refer to the Changing values (Page 253) section for more on this.

You can influence the following parameters:

- "Control P0": Monitoring time while "opening"/"closing" the valve
- "Control P1": Monitoring time while "opening"/"closing" the valve
- "Control P2": Monitoring time while "opening"/"closing" the valve
- "Control V0": Monitoring time while "opening"/"closing" the valve
- "Control V1": Monitoring time while "opening"/"closing" the valve
- "Control V2": Monitoring time while "opening"/"closing" the valve
- "End position": Monitoring time for maintaining the valve position

Enable monitoring

You can enable monitoring by selecting the check box (☑)

You can find additional information on this in the section Monitoring the feedbacks (Page 97).

(2) Service

You can select the following functions in this area:

- "Simulation"
- "Release for maintenance"

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

(3) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

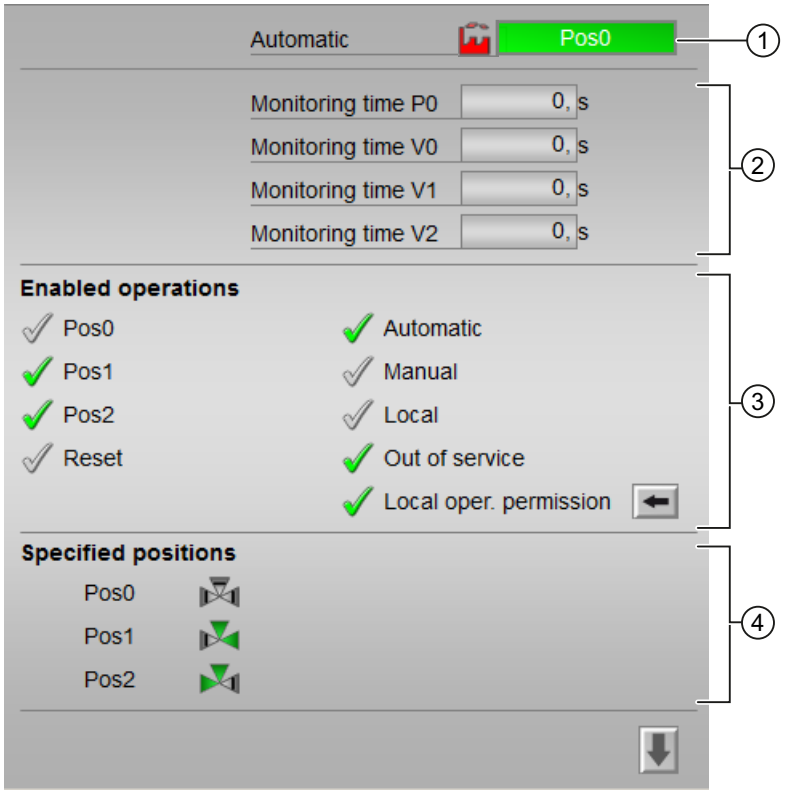
Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

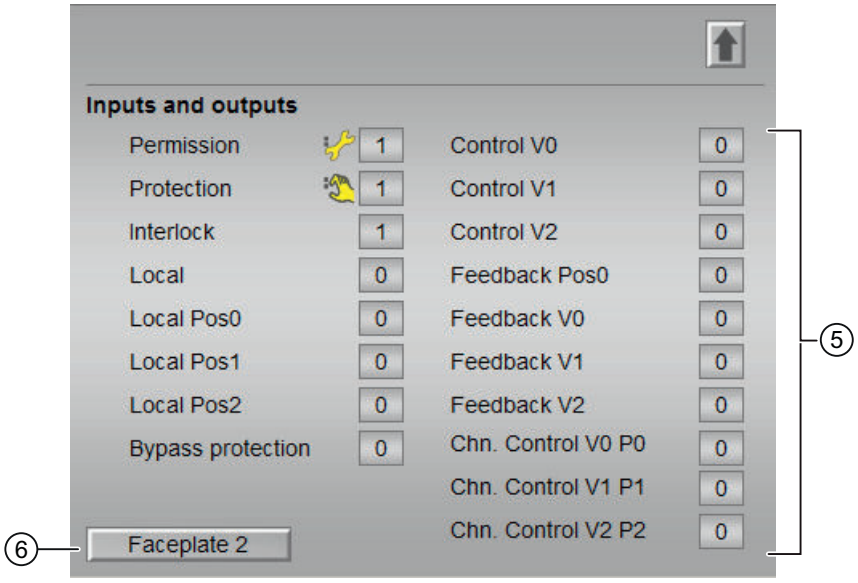
7.9.8.4 Vlv2WayL preview

Preview of Vlv2WayL

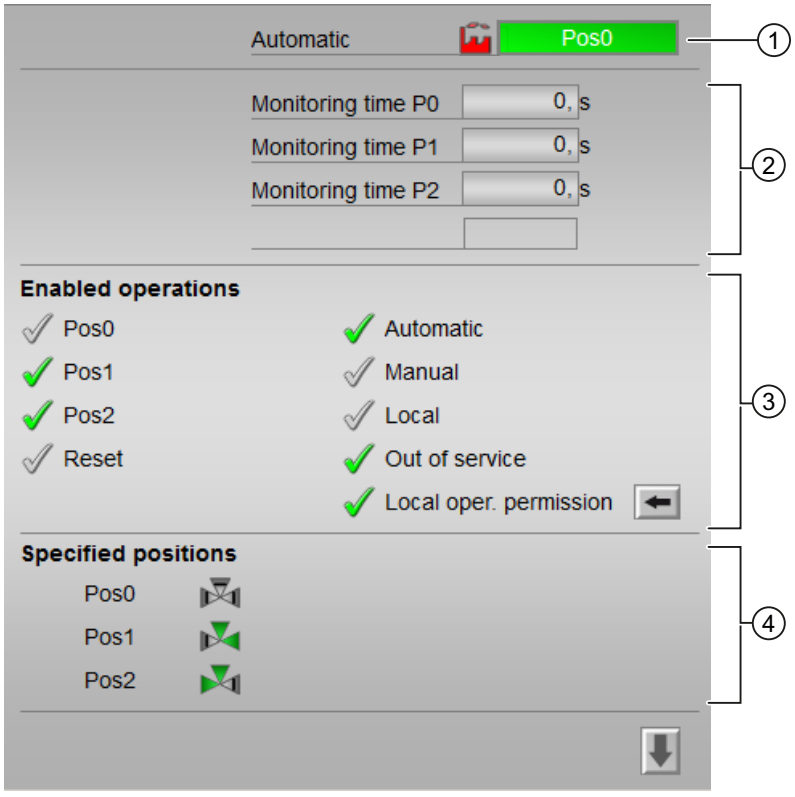
Two-way valve with single value feedback signals (`Feature` bit 12 = 0)



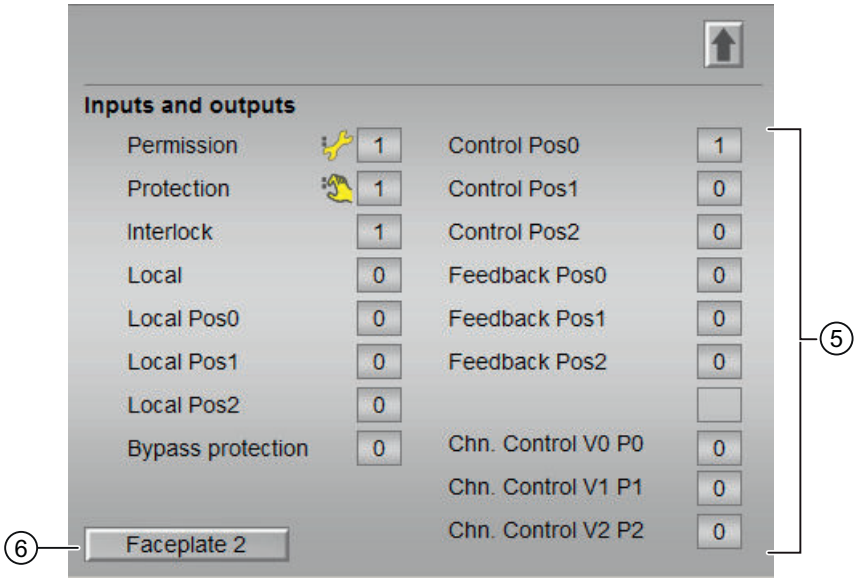
The preview has an upper half and a lower half. You can change between the two halves with the arrow keys.



Two-way valve with position feedback signals (`Feature` bit 12 = 1)



The preview has an upper half and a lower half. You can change between the two halves with the arrow keys.



(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `Pos0Aut`
- `Pos1Aut`
- `Pos2Aut`

(2) Preview area

The following enabled operations are shown here:

- "Monitoring time `P0`": Display of the current monitoring time `P0` (`Feature` bit 12 = 1)
- "Monitoring time `P1`": Display of the current monitoring time `P1` (`Feature` bit 12 = 1)
- "Monitoring time `P2`": Display of the current monitoring time `P2` (`Feature` bit 12 = 1)
- "Monitoring time `V0`": Display of the current monitoring time `V0` (`Feature` bit 12 = 0)
- "Monitoring time `V1`": Display of the current monitoring time `V1` (`Feature` bit 12 = 0)
- "Monitoring time `V2`": Display of the current monitoring time `V2` (`Feature` bit 12 = 0)

(3) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Pos0": You can set the valve to position 0.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Pos1": You can set the valve to position 1.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Pos2": You can set the valve to position 2.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Reset": You can reset the valve if errors occur.
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Local": You can switch to "local mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248).

(4) Specified Position

Preview of the valve positions, as configured in the engineering system (ES).

(5) Displaying current control signals

This area shows the most important parameters for this block with the current selection:

- "Permission":
  This display is only visible when the corresponding block input is connected.
  - 0 = Valve activation not enabled on OS
  - 1 = Enable for "starting"/"stopping" from the neutral position
- "Protection":
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state

- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Local Pos0": 1 = Block was set to position 0 in "local mode"
- "Local Pos1": 1 = Block was set to position 1 in "local mode"
- "Local Pos2": 1 = Block was set to position 2 in "local mode"
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"

Control and feedback signals with `Feature` bit 12 = 0

- "Control Pos0": 1 = Control signal for the position 0
- "Control V0": 1 = Control signal for the valve 0
- "Control V1": 1 = Control signal for the valve 1
- "Control V2": 1 = Control signal for the valve 2
- "Feedback Pos0": 1 = Valve is in position 0
- "Feedback V0": 1 = Feedback if valve 0 was opened
- "Feedback V1": 1 = Feedback if valve 1 was opened
- "Feedback V2": 1 = Feedback if valve 2 was opened

Control and feedback signals with `Feature` bit 12 = 1

- "Control Pos0": 1 = Control signal for the position 0
- "Control Pos1": 1 = Control signal for the position 1
- "Control Pos2": 1 = Control signal for the position 2
- "Feedback Pos0": 1 = Valve is in position 0
- "Feedback Pos1": 1 = Valve is in position 1
- "Feedback Pos2": 1 = Valve is in position 2

Control signals of the output channel block

- "Channel control V0 P0": Control signal for V0 or P0 of the output channel block
- "Channel control V1 P1": Control signal for V1 or P1 of the output channel block
- "Channel control V2 P2": Control signal for V2 or P2 of the output channel block

(6) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

7.9.8.5 Block icon for Vlv2WayL

Properties of the Vlv2WayLblock icon

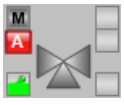
A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as the control system fault
- Operating modes
- Signal status, release for maintenance
- Forcing states
- Displays for bypassing interlocks
- Interlocks
- Memo display
- Valve status display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	
	4	

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	9	
	10	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Valve status display

The following valve states are shown here:

Icon	Meaning
	Valve open
	Error at valve
	Valve is opening

Icon	Meaning
	Valve closed
	Valve is closing

7.10 VlvL - Valve (Large)

7.10.1 Description of VlvL

Object name (type + number) and family

Type + number: FB 1899

Family: Drives

Area of application for VlvL

The block is used for the following applications:

- Controlling a valve in two positions ("open"/"closed") with adjustable neutral position

Note

This block is also available as a small block. A comparison of the VlvL and VlvS blocks is available in the section: VlvL compared to VlvS (Page 1054)

How it works

The valve is opened or closed by a control signal. The signal 0 corresponds to the de-energized state (neutral position) of the valve.

The control is monitored by the "open"/"close" (feedback) signals. Missing feedback can be derived from the control in the block.

Various inputs are available for control purposes. The next sections provide more detailed information on configuration, operating principles, visualization and operation.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the VlvL block, the Advanced Process Library contains a template for process tag types as an example with an application scenario for this block.

Example of process tag types:

- Valve (Valve_Lean) (Page 2341)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section VlvL I/Os (Page 1344).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutoAct.Value`
6	`LocalAct.Value`
7	0: Open padlock in the block icon 1: Closed padlock in the block icon
8	"Open"/"Closed" command (1 = "Open")
9	`FbkOpenOut.Value`
10	`FbkCloseOut.Value`
11	Feedback error without control change
12	Feedback error due to control change
13	`BypProt`
14	Invalid signal status
15	Mode switchover error
16	1 = `Intlock` is active
17	1 = `Permit` is active
18	1 = `Protect` is active
19	Display „Forced open“
20	Display „Forced close“
21	`Force`
22	Automatic preview (1 = "Open")
23	Bumpless switchover to "automatic mode" enabled
24	`SafePos`
25	`UserAna1` interconnected
26	`UserAna2` interconnected
27	`WarnAct.Value`
28	For the Error status display in the Closed valve
29	For the Error status display in the Opened valve
30	External error generated by FaultExt or external control system fault from CSF with set `Feature` bit 18 Activating error state for external process control error CSF (Page 150)
31	1 = Input parameter `CtrlChnST` is interconnected

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	Not used
2	Display for interlocks in block icon
3 - 15	Not used
16	1 = Input parameter `FbkClose` is connected
17	1 = Input parameter `FbkOpen` is connected
18	Reset request in automatic
19	1 = No impact of input signals on "local mode" with `LocalSetting` = 2 and `LocalSetting` = 4
20	1 = Valve open
21	1 = Valve closed
22	1 = Valve opens
23	1 = Valve closes
24 - 29	Not used
30	Bypass information from previous function block
31	`MS_RelOp`

Status word allocation for `Status3` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via `EventTsIn`
1	Effective signal 2 of the message block connected via `EventTsIn`
2	Effective signal 3 of the message block connected via `EventTsIn`
3	Effective signal 4 of the message block connected via `EventTsIn`
4	Effective signal 5 of the message block connected via `EventTsIn`
5	Effective signal 6 of the message block connected via `EventTsIn`
6	Effective signal 7 of the message block connected via `EventTsIn`
7	Effective signal 8 of the message block connected via `EventTsIn`
8	"Interlock" button is enabled
9	"Permission" button is enabled
10	"Protection" button is enabled
11	Hidden bypass signal in Permit
12	Hidden bypass signal in interlock
13	Hidden bypass signal in Protect
14	`Feature2` bit 2: Separate bypass signal
15	Current monitoring time is visible
15 - 25	Not used
26	Show automatic preview in the standard view
27	Not used
28	`GrpErr.Value`

Status bit	Parameter
29	`RdyToStart.Value`
30 - 31	Not used

Status word allocation for `Status4` parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via `EventTsIn`
8 - 31	Not used

See also

VlvL functions (Page 1335)

VlvL messaging (Page 1343)

Overview of the modes (Page 69)

VlvL block diagram (Page 1352)

VlvL error handling (Page 1341)

VlvL modes (Page 1334)

Resetting the block in case of interlocks or errors (Page 43)

7.10.2 VlvL modes

VlvL operating modes

The block supports all standard modes:

- Local mode (Page 79)
- Automatic mode (Page 75)
- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) section.

Valve actions you can control in "local mode":

- "Open" (`OpenLocal` = 1)
- "Close" (`CloseLocal` = 1)

A block operated in "local mode" is controlled either by "local" signals or by feedback signals (input parameters `FbkOpen` and `FbkClose`; if no position can be assigned, the last valid position is accepted). Configuration takes place via the input parameter `LocalSetting`.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) section.

Valve actions you can control in "automatic mode":

- "Open" (`OpenAut` = 1)
- "Close" (`CloseAut` = 1)

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

Valve actions you can control in "manual mode":

- "Open" (`OpenMan` = 1)
- "Close" (`CloseMan` = 1)

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of VlvL (Page 1331)

VlvL block diagram (Page 1352)

VlvL I/Os (Page 1344)

VlvL messaging (Page 1343)

VlvL error handling (Page 1341)

VlvL functions (Page 1335)

7.10.3 VlvL functions

Functions of VlvL

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	1 = Operator can switch to "local mode"
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can open the valve
5	1 = Operator can close the valve
6	1 = Operator can reset the valve
7	1 = Operator can define the monitoring time for startup
8	1 = Operator can define the monitoring time for the end position
9	1 = Operator can activate the monitoring time function (Bit 7 - 8)
10	Not used
11	1 = Operator can activate the Release for maintenance function
12 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Interlocks

This block provides the following interlocks:

- Activation enable
- Interlock without reset ("Interlock")
- Interlock with reset ("Protection")

Refer to the Interlocks (Page 99) section for more on this.

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Resetting the block in case of interlocks

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See VlvL error handling (Page 1341)

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `MonDynErr`
- `MonStaErr`
- `FaultExt`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status ST_Worst for the block is formed from the following parameters:

- `FbkOpenOut.ST`
- `FbkCloseOut.ST`
- `LocalLi.ST`
- `OpenLocal.ST`
- `CloseLocal.ST`
- `CtrlChn.ST`
- `OpenAut.ST` (only if `Feature2.Bit10` = 1)
- `CloseAut.ST` (only if `Feature2.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position in the "Automatic" mode:

- `OpenAut.ST`
- `CloseAut.ST`

Forcing operating modes

This block provides the standard function Forcing operating modes (Page 41). Inputs `OpenForce` and `CloseForce` force the block to open or close.

Feedback monitoring

This block provides the standard function Monitoring the feedbacks (Page 97). Startup characteristics are monitored by setting parameter`MonTiDynamic`. The parameter `MonTiStatic` monitors compliance with the position.

Disabling feedback

This block provides the standard function Disabling feedback for valves (Page 99). Feedback monitoring can be deactivated separately for each feedback with `NoFbkOpen` or `NoFbkClose` as required.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Specify warning times for control functions

This block provides the standard function Specifying warning times for control functions at motors and valves (Page 51). The warning signal is output before the valve moves away from the neutral position. No signal is output for movement to the neutral position.

You can generate warning signals when, for example, valves open. Warning signals can be generated in the following modes:

- Manual and automatic mode for motors, valves and dosers (Page 75) (Input parameter `WarnTiMan`)
- Description of VlvL (Page 1331) (Input parameter `WarnTiAut`)

You specify the warning times in seconds using the input parameters `WarnTiMan` and `WarnTiAut`. If, for example, a valve opens then, this is displayed at the output parameter with

`WarnAct` = 1. The valve then opens after the set warning time has expired and `WarnAct` then returns to 0.

A corresponding warning is not output if the warning times (`WarnTiMan` or `WarnTiAut`) are specified with a smaller value than the `SampleTime` parameter.

Simulating signals

This block provides the standard function Simulating signals (Page 58).

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Neutral position

This block provides the standard function Neutral position for motors, valves and controllers (Page 48). The neutral position (de-energized state) is set using the `SafePos` parameter.

- `SafePos` = 0 means that at `Ctrl` = 0 the valve drive closes and at `Ctrl` = 1 it opens (de-energized state is "closed")
- `SafePos` = 1 means that at `Ctrl` = 0 the valve drive opens and at `Ctrl` = 1 it closes (de-energized state is "open")

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
4	Setting switch or button mode (Page 166)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)
18	Activating error state for external process control error CSF (Page 150)
20	Disable calculation of impulse control in LocalSetting 2 and 4 (Page 185)
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)
22	Update acknowledgment and error status of the message call (Page 159)

Bit	Function
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
27	Interlock display with LocalSetting 2 or 4 (Page 177)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

In pushbutton operation (Bit 4 = 0) the automatic commands in "automatic" mode are latching, in other words `OpenAut`, `CloseAut` can be reset to zero after changing the control. In "manual" and "local" modes, however, the automatic commands are not saved and in the absence of automatic commands the automatic control is tracked.

In switching mode (Bit 4 = 1), the control is selected with the static signal `OpenAut`. If input `OpenAut` is not set the valve is closed. Control via `CloseAut` is not needed. If the "Activate command reset for control" function (Bit 3 = 1) is also activated, the `OpenAut` input is reset to the neutral position after evaluation in the block.

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
4	Setting switch or button mode for local commands (Page 180)
5	Evaluation of the signal status of the interlock signals (Page 141)
8	Forcing operating modes in the "Local" mode (Page 183)
10	Considering bad quality of automatic commands or external values (Page 185)

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Output signal as a pulse signal or static signal

This block provides the standard function Output signal as a static signal or pulse signal (Page 51). In addition to the static control output `Out`, the block also has pulse outputs `P_Open`, `P_Close`, which are dependent on the static control output.

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- `OpenMan`
- `CloseMan`

See also

VlvL messaging (Page 1343)

VlvL I/Os (Page 1344)

VlvL modes (Page 1334)

VlvL block diagram (Page 1352)

EventTs functions (Page 1634)

7.10.4 VlvL error handling

Error handling of VlvL

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Invalid input signals
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
41	The value for the `LocalSetting` I/O is not within the approved limit of 0 to 4.
42	`LocalSetting` = 0 or `LocalSetting` = 3 or `LocalSetting` = 4 and `LocalLi` = 1
51	`AutModLi` = 1 and `ManModLi` = 1 `OpenLocal` = 1 and `CloseLocal` = 1 `OpenAut` = 1 and `CloseAut` = 1 `OpenForce` = 1 and `CloseForce` = 1
52	`LocalAct` = 1 and `LocalSetting` = 2 or `4` and `SimOn` = 1

Mode switchover error

This error can be output by the block. Refer to the Error handling (Page 119) section for more on this.

Invalid input signals

This error can be output by the block. Refer to the Error handling (Page 119) section for more on this.

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signals	Control reaction with Feature2 bit 3 =1
Local: `Localsetting` = 1 or `Localsetting` = 3	Pushbutton operation for local mode (`Feature2` bit 4 = 0): `OpenLocal` = 1 and `CloseLocal` = 1	Valve is set to its neutral position.
Local: `Localsetting` = 1 or `Localsetting` = 3 and forcing	`OpenForce` = 1 and `CloseForce` = 1	
Forcing and no "local mode"	`OpenForce` = 1 and `CloseForce` = 1	
"Automatic mode" and no forcing	Pushbutton operation (`Feature` bit 4 = 0): `OpenAut` = 1 and `CloseAut` = 1	
"Manual mode" and no forcing	`OpenMan` = 1 and `CloseMan` = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

Description of VlvL (Page 1331)

VlvL modes (Page 1334)

VlvL block diagram (Page 1352)

VlvL I/Os (Page 1344)

VlvL messaging (Page 1343)

VlvL functions (Page 1335)

7.10.5 VlvL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Instance-specific messages

Process control fault

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 1	AS process control message - fault	$$BlockComment$$ Feedback error
	SIG 2	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a process control fault is triggered (`MsgEvId1`, SIG 2).

Instance-specific messages

You can use up to three instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 3	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 2
	SIG 5	AS process control message - fault	$$BlockComment$$ External message 3

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`ExtVa104`
5	`ExtVa105`
6	`ExtVa106`
7	`ExtVa107`
8	`ExtVa108`
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters `ExtVa104` ... `ExtVa108` and can be used. See the "Process Control System PCS 7 - Engineering System" manual.

See also

VlvL modes (Page 1334)

VlvL block diagram (Page 1352)

VlvL error handling (Page 1341)

7.10.6 VlvL I/Os

I/Os of VlvL

Input parameters

Parameter	Description	Type	Default
AutModLi*	1= "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 1)	BOOL	0
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Bypassing interlock is active in "local mode" and in simulation	BOOL	0
CloseAut*	1 = Select Close valve in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseForce	1 = Force valve closure	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseLocal	1 = Select Close valve in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseMan*	1 = Select Close valve in "manual mode"	BOOL	0
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CtrlChnST	Signal status of output channel Ctrl Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
EN	1 = Called block will be processed	BOOL	1
EventTsIn	For interconnecting data between a technological block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs block. When this interconnection is configured, the messages of the EventTs block are displayed on the OS in the alarm view of the technological block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpen	1 = Valve open feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkClose	1 = Valve closed feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Feature	I/O for additional functions (Page 1335)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Feature2	I/O for additional functions (Page 1335)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Intlock	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is active	BOOL	1
LocalLi	1 = Activate "local mode" via plant signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
LocalOp*	1 = "Local mode" via operator	BOOL	0
LocalSetting	Properties for the Local mode (Page 79)	INT	0
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1= "Manual mode" via: OS operator (controlled via ModLiOp = 0)	BOOL	1
ModLiOp	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Monitor	1 = Feedback monitoring	BOOL	1
MonSafePos	1 = Go to neutral position in the event of monitoring errors	BOOL	1
MonTiDynamic*	Monitoring time after operation in [s]	REAL	3.0
MonTiStatic*	Monitoring time for feedback errors without operation in [s]	REAL	3.0
MS_RelOp*	1= Release for maintenance via OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NoFbkClose	1 = No feedback present for "valve closed"	BOOL	0
NoFbkOpen	1 = No feedback present for "valve open"	BOOL	0
Occupied	1 = In use by a batch	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpenAut*	1 = Select Open valve in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenForce	1 = Force valve opening	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenLocal	1 = Select Open valve in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenMan*	1 = Select Open valve in "manual mode"	BOOL	0

Parameter	Description	Type	Default
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 1335)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
Permit	1 = Enable for opening / closing from neutral position 0 = Valve activation not enabled on OS	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Perm_En	1 = Activation enable (enable, Permit parameter) is active	BOOL	1
Protect	0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block 1 = Protective interlocking not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Prot_En	1 = Protective interlock (protection, Protect parameter) is active	BOOL	1
PulseWidth*	Pulse width of control signal [s]	REAL	3.0
RstLi*	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SafePos	Neutral position for valve: 1= Open 0 = Closed	BOOL	0
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	-

Parameter	Description	Type	Default
StepNo	Batch step number	DWORD	16#00000000
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA2unit	Unit of measure for analog auxiliary value 2	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
WarnTiAut	Prewarning of valve movement from neutral position in "automatic mode" in [s]	REAL	0.0
WarnTiMan	Prewarning of valve movement from neutral position in "manual mode" in [s]	REAL	0.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80
Closed	1 = Valve is closed	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Closing	1 = Valve is closing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl	Control output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CurrMon	Current monitoring time [s]	DINT	0
ENO	1 = Block algorithm completed without errors	BOOL	0

Parameter	Description	Type	Default
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see VlvL error handling (Page 1341)	INT	-1
FbkCloseOut	Valve closed feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpenOut	Valve open feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalAct	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock (Intlock, Permit or Protect) is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MonDynErr	1 = Feedback error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonStaErr	1 = Feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Opened	1 = Valve is open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Opening	1 = Valve is opening	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
P_Close	1 = Pulse signal to close valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Open	1 = Pulse signal to open valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1331)	DWORD	16#00000000
Status2	Status word 2 (Page 1331)	DWORD	16#00000000
Status3	Status word 3 (Page 1331)	DWORD	16#00000000
WarnAct	1 = Prewarning for valve movement away from neutral position active (parameters WarnTiAut and WarnTiMan)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

VlvL messaging (Page 1343)

VlvL modes (Page 1334)

VlvL block diagram (Page 1352)

Error handling (Page 119)

7.10.7 VlvL block diagram

VlvL block diagram

A block diagram is not provided for this block.

See also

Description of VlvL (Page 1331)

VlvL modes (Page 1334)

VlvL error handling (Page 1341)

VlvL messaging (Page 1343)

VlvL I/Os (Page 1344)

VlvL functions (Page 1335)

7.10.8 Operator control and monitoring

7.10.8.1 VlvL views

Views of the VlvL block

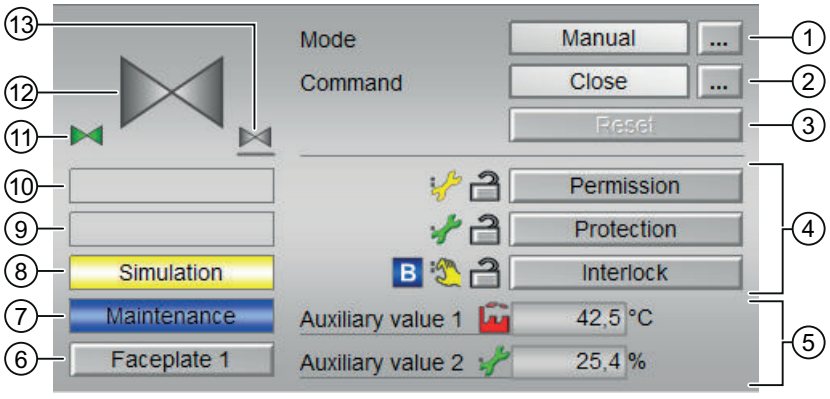
The block VlvL provides the following views:

- VlvL standard view (Page 1353)
- Alarm view (Page 296)
- Trend view (Page 299)
- Parameter view for motors and valves (Page 280)
- VlvL preview (Page 1356)
- Memo view (Page 298)
- Batch view (Page 296)
- VlvL block icon (Page 1359)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.10.8.2 VlvL standard view

VlvL standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Opening and closing the valve

This area shows you the default operating state for the valve. The following states can be shown and executed here:

- "Open"
- "Close"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(3) Resetting the block

Click "Reset" for interlocks or errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(4) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the section Interlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

(5) Display of auxiliary values

This display is only visible when the corresponding block input is connected.

You can use this area to display two auxiliary values that have been configured in the engineering system (ES). You can find additional information on this in the section Displaying auxiliary values (Page 207).

(6) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(7) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

(8) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"
- "Delay"

You will find more detailed information on this in the sections Simulating signals (Page 58) and Display of delay times (Page 250).

(9) Display area for block states

This area provides additional information on the operating state of the block:

- "External error"
- "End position error"
- "Control error"
- "Invalid signal"
- "Changeover error"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97) , Error handling (Page 119) (section "Invalid input signals" and "Mode switchover error") and Motor protection function (Page 99).

(10) Display area for block states

This area provides additional information on the operating state of the block:

- "Forced open"
- "Forced close"
- "Request 0/1": A reset to "automatic mode" is expected.

You can find additional information on this in the section Forcing operating modes (Page 41).

(11) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the valve would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

(12) Status display of the valve

The current status of the valve is graphically displayed here.

You can find more information about this in section VlvL block icon (Page 1359)

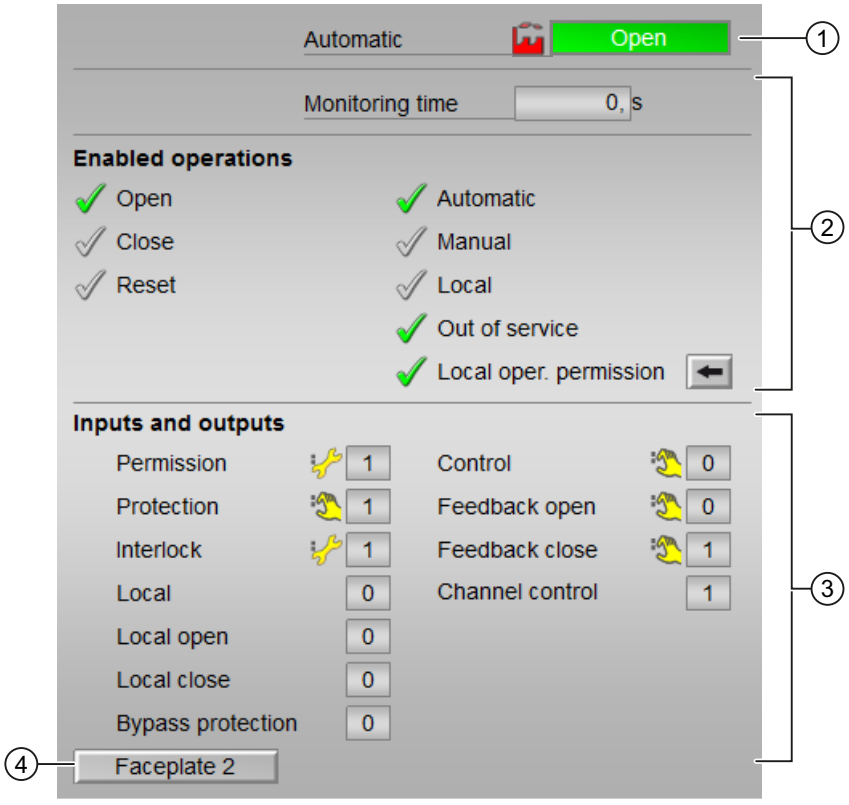
(13) Neutral position of the valve

Display the neutral position for the valve:

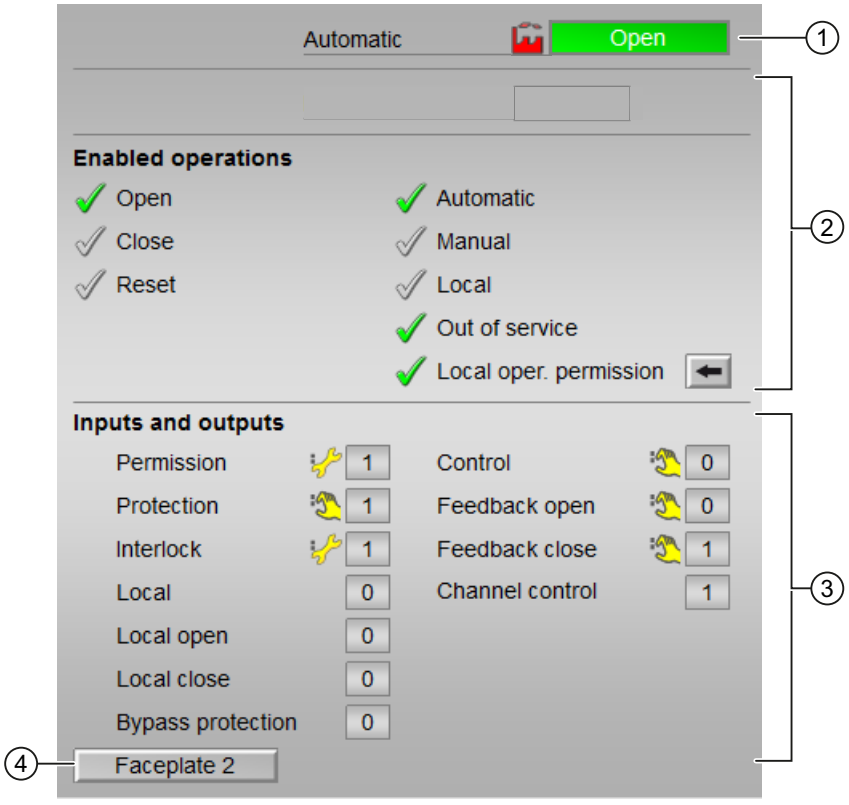
- If the neutral position of the valve is "Closed" (`SafePos` = 0), , a gray valve is shown.
- If the neutral position of the valve is "Open" (`SafePos` = 1), a green valve is shown.

7.10.8.3 VlvL preview

Preview of VlvL



Display of the current monitoring time is visible.



Display of the current monitoring time is not visible.

(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `OpenAut`
- `CloseAut`

(2) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Open": You can open the valve.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Close": You can close the valve.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Reset": You can reset the valve if interlocks or errors occur.
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Local": You can switch to "local mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248) .
- "Monitoring time": Display of the current monitoring time.

(3) Displaying current control signals

This area shows the most important parameters for this block with the current selection:

- "Permission":
  This display is only visible when the corresponding block input is connected.
  - 0 = Valve activation not enabled on OS
  - 1 = Enable for "opening"/"closing" from the neutral position
- "Protection":
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state
- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Local open": 1 = Opening the valve in "local mode"
- "Local close": 1 = Closing the valve in "local mode"
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"

- "Control": Display for valve control:
  - 0 = Valve is closing
  - 1 = Valve is opening
- "Feedback open": 1 = Valve is open
- "Feedback close": 1 = Valve is closed
- "Channel Control": Control signal of the output channel block

(4) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

7.10.8.4 VlvL block icon

Block icons for VlvL

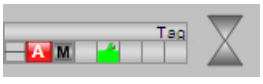
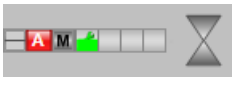
A variety of block icons are available with the following functions:

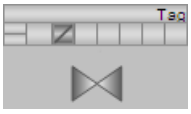
- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes
- Signal status, release for maintenance
- Forcing states
- Displays for bypassing interlocks
- Interlocks
- Memo display
- Valve status display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Non-rotating block icon
	4	Non-rotating block icon

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	9	

Icons	Selection of the block icon in CFC	Special features
	10	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Valve status display

The following valve states are shown here:

Icon	Meaning
	Valve open
	Valve closed
	Error at valve
	Valve is opening
	Valve is closing
	Valve closed
	Valve is closing

7.11 VlvS - Valve (small)

7.11.1 Description of VlvS

Object name (type + number) and family

Type + number: FB 1911

Family: Drives

Area of application for VlvS

The block is used for the following applications:

- Controlling a valve in two positions ("open"/"closed") with adjustable neutral position

Note

This block is also available as a large block. A comparison of the VlvL and VlvS blocks is available in the section: VlvL compared to VlvS (Page 1054)

How it works

The valve is opened or closed by a control signal. The signal 0 corresponds to the de-energized state (neutral position) of the valve.

The control is monitored by the "open"/"close" (feedback) signals. Missing feedback can be derived from the control in the block.

Various inputs are available for control purposes. The next sections provide more detailed information on configuration, operating principles, visualization and operation.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section VlvS I/Os (Page 1373).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutoAct.Value = 1; ManAct.Value = 0`
6	`LocalAct.Value`
7	`LockAct.Value`
8	"Open"/"Closed" command (1 = "Open")
9	`FbkOpenOut.Value`
10	`FbkCloseOut.Value`
11	Feedback error without control change
12	Feedback error due to control change
13	`BypProt`
14	Invalid signal status
15	Not used
16	1 = `Intlock` is active
17 - 21	Not used
22	Automatic preview (1 = "Open")
23	Bumpless switchover to "automatic mode" enabled
24	`SafePos`
25 - 27	Not used
28	For the Error status display in the Closed valve
29	For the Error status display in the Opened valve
30	External error generated by FaultExt or external control system fault CSF with set Feature bit 18 Activating error state for external process control error CSF (Page 150)
31	1 = Input parameter `CtrlChnST` is interconnected

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	Not used
2	Display for interlocks in block icon
3 - 15	Not used
16	1 = Input parameter `FbkClose` is interconnected
17	1 = Input parameter `FbkOpen` is interconnected
18	Reset request in automatic
19	1 = No impact of input signals on "local mode" when `LocalSetting` = 2

Status bit	Parameter
20	1 = Valve open
21	1 = Valve closed
22	1 = Valve opens
23	1 = Valve closes
24 - 29	Not used
30	Bypass information from previous function block
31	`MS_RelOp`

Status word allocation for `Status3` parameter

Status bit	Parameter
0 - 7	Not used
8	"Interlock" button is enabled
9 - 11	Not used
12	Hidden bypass signal in interlock
13	Not used
14	`Feature2` bit 2: Separate bypass signal
15 - 25	Not used
26	Show automatic preview in the standard view
27	Not used
28	`GrpErr.Value`
29	`RdyToStart.Value`
30 - 31	Not used

See also

VlvS modes (Page 1364)
VlvS functions (Page 1366)
VlvS error handling (Page 1370)
VlvS reporting (Page 1371)
VlvS block diagram (Page 1378)

7.11.2 VlvS modes

VlvS operating modes

The block supports all standard modes:

- Local mode (Page 79)
- Automatic mode (Page 75)

- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) section.

Note

"Local mode" for the block VlvS

In contrast to the "Large" blocks, it is only possible to perform settings in this block `LocalSetting` with 0, 2 and 5.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) section.

Valve actions you can control in "automatic mode":

- "Open" (`OpenAut` = 1)
- "Close" (`CloseAut` = 1)

Note

Information about the "Small" block

This "Small" block works with pushbutton operation. The automatic commands are therefore latching, in other words, `OpenAut` and `CloseAut` can be reset to 0 after the control is changed. In "manual" and "local" modes, however, the automatic commands are not saved and in the absence of automatic commands the automatic control is tracked.

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

Valve actions you can control in "manual mode":

- "Open" (`OpenMan` = 1)
- "Close" (`CloseMan` = 1)

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of VlvS (Page 1362)

VlvS functions (Page 1366)

VlvS error handling (Page 1370)

VlvS reporting (Page 1371)

VlvS I/Os (Page 1373)

VlvS block diagram (Page 1378)

7.11.3 VlvS functions

Functions of VlvS

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	1 = Operator can switch to "local mode"
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can open the valve
5	1 = Operator can close the valve
6	1 = Operator can reset the valve
7	1 = Operator can define the monitoring time for startup
8	Not used
9	1 = Operator can activate the monitoring time function (Bit 7)
11	1 = Operator can activate the Release for maintenance function
12 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- `OpenMan`
- `CloseMan`

Interlocks

This block provides the following interlocks:

- Interlock without reset ("Interlock")

Refer to the Interlocks (Page 99) section for more on this.

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Resetting the block in case of interlocks

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See VlvS error handling (Page 1370)

Group error

This block provides the standard function Outputting group errors (Page 122)

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `MonDynErr`

- MonStaErr
- FaultExt

Neutral position

This block provides the standard function Neutral position for motors, valves and controllers (Page 48). The neutral position (de-energized state) is set using the SafePos parameter.

- SafePos = 0 means that at Ctrl = 0 the valve drive closes and at Ctrl = 1 it opens (de-energized state is "closed")
- SafePos = 1 means that at Ctrl = 0 the valve drive opens and at Ctrl = 1 it closes (de-energized state is "open")

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status ST_Worst for the block is formed from the following parameters:

- FbkOpenOut.ST
- FbkCloseOut.ST
- LocalLi.ST
- CtrlChn.ST
- OpenAut.ST (only if Feature2.Bit10 = 1)
- CloseAut.ST (only if Feature2.Bit10 = 1)

Considering bad quality of automatic commands or external values

If the Feature2 bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position in the "Automatic" mode:

- OpenAut.ST
- CloseAut.ST

Feedback monitoring

This block provides the standard function Monitoring the feedbacks (Page 97). Startup characteristics are monitored by setting parameterMonTiDynamic. The parameter MonTiStatic monitors compliance with the position.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Simulating signals

This block provides the standard function Simulating signals (Page 58).

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
18	Activating error state for external process control error CSF (Page 150)
25	Suppression of all messages (Page 173)
27	Interlock display with LocalSetting 2 or 5 (Page 177)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
5	Evaluation of the signal status of the interlock signals (Page 141)
10	Considering bad quality of automatic commands or external values (Page 185)

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of VlvS (Page 1362)

VlvS modes (Page 1364)

VlvS reporting (Page 1371)

VlvS I/Os (Page 1373)

VlvS block diagram (Page 1378)

Disabling feedback for valves (Page 99)

Selecting a unit of measure (Page 207)

Enabling local operator authorization (Page 157)

7.11.4 VlvS error handling

Error handling of VlvS

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Invalid input signals
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
41	The value for the `LocalSetting` I/O is not within the valid limit of 0, 2 or 5
42	`LocalSetting` = 0 and `LocalLi` = 1
51	`AutModLi` = 1 and `ManModLi` = 1 `OpenAut` = 1 and `CloseAut` = 1
52	`LocalAct` = 1 and `LocalSetting` = 2 or 5 and `SimOn` = 1

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

Invalid input signals

This error can be output by the block, see the section Error handling (Page 119).

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signals	Control reaction with Feature2 bit 3 =1
"Automatic mode"	`OpenAut` = 1 and `CloseAut` = 1	Valve is set to its neutral position.
"Manual mode"	`OpenMan` = 1 and `CloseMan` = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

Description of VlvS (Page 1362)

VlvS modes (Page 1364)

VlvS functions (Page 1366)

VlvS reporting (Page 1371)

VlvS I/Os (Page 1373)

VlvS block diagram (Page 1378)

7.11.5 VlvS reporting

Messaging

The following messages can be generated for this block:

- Process control fault
- Instance-specific messages

Process control fault

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 1	AS process control message - fault	$$BlockComment$$ Feedback error
	SIG 2	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter `CSF`. If it changes to `CSF` = 1, a process control fault is triggered (`MsgEvId1`, SIG 2).

Instance-specific messages

You have the option to use two instance-specific messages for this block.

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 3	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 4	AS process control message - fault	$$BlockComment$$ External message 2

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`ExtVa104`
5	`ExtVa105`
6 - 10	Reserved

The associated values 4 ... 5 are allocated to the parameters `ExtVa104` ... `ExtVa105` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of VlvS (Page 1362)

VlvS modes (Page 1364)

7.11.6 VlvS I/Os

I/Os of VlvS

Input parameters

Parameter	Description	Type	Default
`AutModLi*`	1= "Automatic mode" via interconnection or SFC (controlled by `ModLiOp` = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`AutModOp`	1 = "Automatic mode" via operator (controlled by `ModLiOp` = 1)	BOOL	0
`BatchEn`	1 = Enable allocation	BOOL	0
`BatchID`	Batch ID	DWORD	16#00000000
`BatchName`	Batch name	S7-String	
`BypProt`	1 = Bypassing interlock is active in "local mode" and in simulation	BOOL	0
`CloseAut*`	1 = Select Close valve in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`CloseMan*`	1 = Select Close valve in "manual mode"	BOOL	0
`CSF`	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`CtrlChnST`	Signal status of output channel `Ctrl` Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`EN`	1 = Called block will be processed	BOOL	1
`ExtMsg1`	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ExtMsg2`	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ExtVa104`	Associated value 4 for messages (`MsgEvID1`)	ANY	

Parameter	Description	Type	Default
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpen	1 = Valve open feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkClose	1 = Valve closed feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Feature	I/O for additional functions (Page 1366)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Feature2	I/O for additional functions (Page 1366)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Intlock	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is active	BOOL	1
LocalLi	1 = Activate "local mode" via plant signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalSetting	Properties for the Local mode (Page 79)	INT	0
ManModLi*	1 = Manual mode via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1 = Manual mode via: OS operator (controlled via ModLiOp = 0)	BOOL	1
ModLiOp	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Monitor	1 = Feedback monitoring	BOOL	1
MonSafePos	1 = Go to neutral position in the event of monitoring errors	BOOL	1
MonTiDynamic*	Monitoring time after operation in [s]	REAL	3.0

Parameter	Description	Type	Default
MS_RelOp*	1= Release for maintenance via OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = In use by a batch	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpenAut*	1 = Select Open valve in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenMan*	1 = Select Open valve in "manual mode"	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 1366)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RstLi*	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SafePos	Neutral position for valve: 1= Open 0 = Closed	BOOL	0
SimOn	1 = Simulation on	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
StepNo	Batch step number	DWORD	16#00000000
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80
Ctrl	Control output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Closed	1 = Valve is closed	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Closing	1 = Valve is closing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see VlvS error handling (Page 1370)	INT	-1
FbkCloseOut	Valve closed feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpenOut	Valve open feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalAct	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MonDynErr	1 = Feedback error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonStaErr	1 = Feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Opened	1 = Valve is open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Opening	1 = Valve is opening	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1362)	DWORD	16#00000000
Status2	Status word 2 (Page 1362)	DWORD	16#00000000
Status3	Status word 3 (Page 1362)	DWORD	16#00000000

See also

VlvS modes (Page 1364)
VlvS block diagram (Page 1378)
VlvS reporting (Page 1371)
Error handling (Page 119)

7.11.7 VlvS block diagram

VlvS block diagram

A block diagram is not provided for this block.

See also

Description of VlvS (Page 1362)
VlvS modes (Page 1364)
VlvS functions (Page 1366)
VlvS error handling (Page 1370)
VlvS I/Os (Page 1373)
VlvS reporting (Page 1371)

7.11.8 Operator control and monitoring

7.11.8.1 VlvS views

Views of the VlvS block

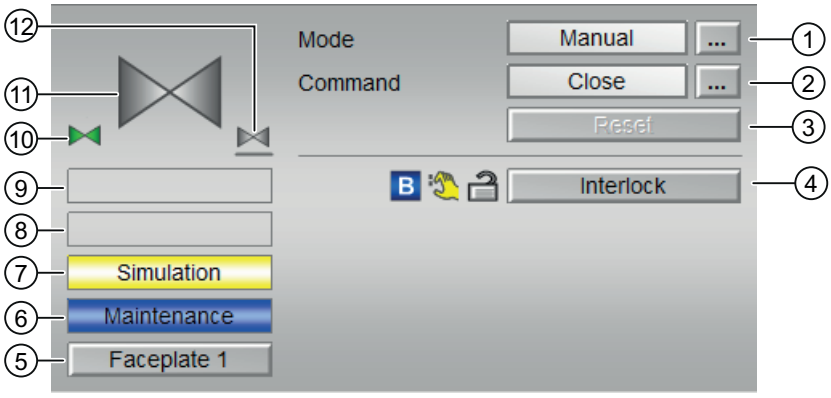
The block VlvS provides the following views:

- VlvS standard view (Page 1379)
- Alarm view (Page 296)
- Trend view (Page 299)
- Parameter view for motors and valves (Page 280)
- VlvS preview (Page 1382)
- Memo view (Page 298)
- Batch view (Page 296)
- VlvS block icon (Page 1384)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.11.8.2 VlvS standard view

VlvS standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Opening and closing the valve

This area shows you the default operating state for the valve. The following states can be shown and executed here:

- "Open"
- "Close"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(3) Resetting the block

Click "Reset" for errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(4) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the section Interlocks (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

(5) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(6) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

(7) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in the section Simulating signals (Page 58).

(8) Display area for block states

This area provides additional information on the operating state of the block:

- "External error"
- "End position error"
- "Control error"
- "Invalid signal"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97) , Error handling (Page 119) (section "Invalid input signals" and "Mode switchover error") and Motor protection function (Page 99).

(9) Display area for block states

This area provides additional information on the operating state of the block:

- "Request 0/1": A reset to "automatic mode" is expected.

You can find additional information on this in the section Forcing operating modes (Page 41).

(10) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the valve would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

(11) Status display of the valve

The current status of the valve is graphically displayed here.

- Green: Valve is open
- Gray: Valve is closed
- Red: Fault at valve

You can find more information about this in section VlvS block icon (Page 1384)

(12) Neutral position of the valve

Display the neutral position for the valve:

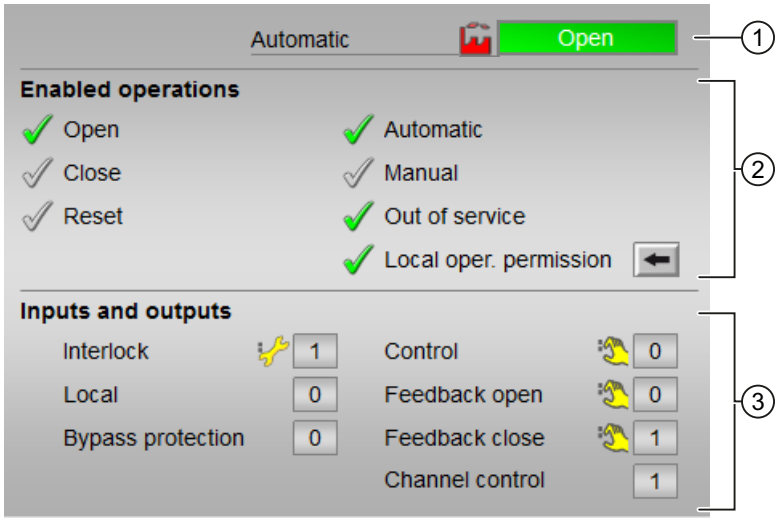
- If the neutral position of the valve is "Closed" (`SafePos` = 0), , a gray valve is shown.
- If the neutral position of the valve is "Open" (`SafePos` = 1), a green valve is shown.

See also

Displaying auxiliary values (Page 207)

7.11.8.3 VlvS preview

Preview of VlvS



(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `OpenAut`
- `CloseAut`

(2) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Open": You can open the valve.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Close": You can close the valve.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Reset": You can reset the valve if interlocks or errors occur.
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Labeling of buttons and text (Page 205) .

(3) Displaying current control signals

This area shows the most important parameters for this block with the current selection:

- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"
- "Control": Display for valve control:
  - 0 = Valve is closing
  - 1 = Valve is opening
- "Feedback open": 1 = Valve is open
- "Feedback close": 1 = Valve is closed
- "Channel Control": Control signal of the output channel block

See also

Operator control permissions (Page 248)

7.11.8.4 VlvS block icon

Block icons for VlvS

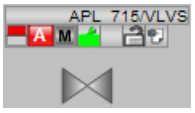
A variety of block icons are available with the following functions:

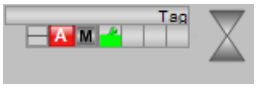
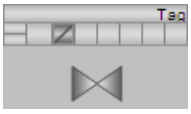
- Process tag type
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes
- Signal status, release for maintenance
- Displays for bypassing interlocks
- Interlocks
- Memo display
- Valve status display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Non-rotating block icon
	4	Non-rotating block icon

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	

Icons	Selection of the block icon in CFC	Special features
	3	
	4	
	5	
	6	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Valve status display

The following valve states are shown here:

Symbol	Meaning
	Valve open
	Valve closed
	Error at valve
	Valve is opening
	Valve is closing
	Valve closed
	Valve is closing

7.12 VlvMotL - Motor valve

7.12.1 Description of VlvMotL

Object name (type + number) and family

Type + number: FB 1900

Family: Drives

Area of application for VlvMotL

The block is used for the following applications:

- Motor valve control

How it works

Various operating modes are available for controlling the motor-driven valve. This functionality allows you to set specific valve states. All changes of modes or states and faults occurring in this context are monitored, visualized in the faceplate and reported to the operator. Operators with suitable permissions can use the block icon and the faceplate to view the current states of the motor-driven valve and to operate it.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the VlvMotL block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Step controller with assigned actuator block and position feedback (StepControlActor) (Page 2308)
- Motor valve (ValveMotor) (Page 2343)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

After a startup without control (`Open`, `Close` = 0)", no monitoring of the feedback signals `FbkOpen` and `FbkClose` takes place during the `V_MonTiStatic` time. Changes to `FbkOpen` and `FbkClose` are applied. This means that the feedback is monitored again, also in stop state.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section VlvMotL I/Os (Page 1406).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutoAct.Value`
6	`LocalAct.Value`
7	0: Open padlock in the block icon 1: Closed padlock in the block icon
8	`Open.Value`
9	Motor is stopped
10	`Close.Value`
11	Torque shutdown enabled (`TorqOpen` or `TorqClose` = 1) When the seal valve function is enabled, the torque shutoff is only active if the `TorqClose` signal is enabled before the valve feedback.
12	`WarnAct.Value` or `IdleTime` active
13	Feedback error without control change
14	Feedback error due to control change
15	`Mode Switch Fail`
16	1 = `Intlock` is active
17	1 = `Permit` is active
18	1 = `Protect` is active
19	`Trip.Value`
20	Display „Forced open“
21	Display „Forced stop“
22	Display „Forced close“
23	"Interlock" button is enabled
24	0 = Display neutral position "Closed" 1 = Display neutral position "Open"
25	1 = Display neutral position "Stop"
26	Bypass information from previous function block
27	Bypass enabled (`BypProt` = 1) and `Local.Act` = 1 or `SimOn` = 1
28	Invalid signal status
29	0 = closed 1 = open
30	"Permission" button is enabled
31	"Protection" button is enabled

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	`AV_AH_Act.Value`
2	`AV_WH_Act.Value`
3	`AV_TH_Act.Value`
4	`AV_TL_Act.Value`
5	`AV_WL_Act.Value`
6	`AV_AL_Act.Value`
7	`AV_AH_En`
8	`AV_WH_En`
9	`AV_TH_En`
10	`AV_TL_En`
11	`AV_WL_En`
12	`AV_AL_En`
13	`AV_AH_MsgEn`
14	`AV_WH_MsgEn`
15	`AV_TH_MsgEn`
16	`AV_TL_MsgEn`
17	`AV_WL_MsgEn`
18	`AV_AL_MsgEn`
19	1 = No impact of input signals on "local mode" with `LocalSetting` = 2 and `LocalSetting` = 4
20	1 = Valve closes
21	1 = Valve closed
22	1 = Valve stopped
23	1 = Valve opens
24	1 = Valve open
25	For the Error status display in the Closed valve
26	For the Error status display in the Opened valve
27	Automatic preview for "opening"
28	Automatic preview for "closing"
29	Automatic preview for "stopping"
30	Display for interlocks in block icon
31	`MS_RelOp`

Status word allocation for `Status3` parameter

Status bit	Parameter
0	`M_MonStaErr.Value`
1	`M_MonDynErr.Value`
2	`V_MonStaErr.Value`
3	`V_MonDynErr.Value`

Status bit	Parameter
4	M_MonStopErr.Value
5-7	Not used
8	Reset request in automatic
9	External error generated by FaultExt or external control system fault from CSF with set Feature bit 18 Activating error state for external process control error CSF (Page 150)
10	Not used
11	Motor protection display (Trip.Status ≠ 16#FF)
12	1 = Input parameter FbkClose is connected
13	1 = Input parameter FbkClosing is connected
14	1 = Input parameter FbkOpen is connected
15	1 = Input parameter FbkOpening is connected
16	1 = Input parameter TorOpen is connected
17	1 = Input parameter TorClose is connected
18	SimLiOp.Value
19	1 = Enable for "rapid stop"(Feature Bit Enabling rapid stop via faceplate (Page 167))
20	1 = Input parameter OpenChnST is interconnected
21	1 = Input parameter CloseChnST is interconnected
22	Not used
23	Command for "rapid stop"
24	"Open"/"Stop" command
25	"Close"/"Stop" command
26	Show automatic preview in the standard view
27	Not used
28	GrpErr.Value
29	RdyToStart.Value
30	Auxiliary value 1 visible
31	Auxiliary value 2 visible

Status word allocation for `Status4` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via EventTsIn
1	Effective signal 2 of the message block connected via EventTsIn
2	Effective signal 3 of the message block connected via EventTsIn
3	Effective signal 4 of the message block connected via EventTsIn
4	Effective signal 5 of the message block connected via EventTsIn
5	Effective signal 6 of the message block connected via EventTsIn
6	Effective signal 7 of the message block connected via EventTsIn
7	Effective signal 8 of the message block connected via EventTsIn
8	AV not connected
9	Delay of the AV_AH_Lim message

Status bit	Parameter
10	Delay of the `AV_WH_Lim` message
11	Delay of the `AV_TH_Lim` message
12	Delay of the `AV_TL_Lim` message
13	Delay of the `AV_WL_Lim` message
14	Delay of the `AV_AL_Lim` message
15	Collection of message delays
16 - 22	Not used
23	Hidden bypass signal in Permit
24	Hidden bypass signal in interlock
25	Hidden bypass signal in Protect
26	`Feature2` bit 2: Separate bypass signal
27 - 28	Not used
29	Current motor monitoring time is visible
30	Current valve monitoring time is visible
31	Separate monitoring of shutdown of the motor (`Feature` bit 13)

Status word allocation for `Status5` parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via `EventTsIn`
8	1 = Valve still closes (`Status2` Bit 20) but the motor is not running
9	1 = Valve still opens (`Status2` Bit 23) but the motor is not running
10 - 15	Not used
16 - 31	Effective signal 1...16 of the message block connected via `EventTs2In`

See also

VlvMotL functions (Page 1392)

VlvMotL messaging (Page 1405)

VlvMotL block diagram (Page 1416)

VlvMotL error handling (Page 1402)

VlvMotL modes (Page 1390)

7.12.2 VlvMotL modes

VlvMotL operating modes

The block supports all standard modes:

- Local mode (Page 79)
- Automatic mode (Page 75)

- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) section.

Motor valve actions you can control in "local mode"

- "Open" (`OpenLocal` = 1)
- "Close" (`CloseLocal` = 1)
- "Stop" (`StopLocal` = 1).

A block operated in "local mode" is controlled either by "local" signals or by feedback signals (input parameters `FbkOpen` and `FbkClose`; if no position can be assigned, the last valid position is accepted). Configuration takes place via the input parameter `LocalSetting`.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) section.

Motor valve actions you can control in "automatic mode":

- "Open" (`OpenAut` = 1)
- "Close" (`CloseAut` = 1)
- "Stop" (`StopAut` = 1)

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

Motor valve actions you can control in "manual mode":

- "Open" (`OpenMan` = 1)
- "Close" (`CloseMan` = 1)
- "Stop" (`StopMan` = 1)

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

VlvMotL block diagram (Page 1416)
VlvMotL I/Os (Page 1406)
VlvMotL error handling (Page 1402)
VlvMotL functions (Page 1392)
VlvMotL messaging (Page 1405)
Description of VlvMotL (Page 1386)

7.12.3 VlvMotL functions

Functions of VlvMotL

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	1 = Operator can switch to "local mode"
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can stop the motor
5	1 = Operator can open the valve
6	1 = Operator can close the valve
7	1 = Operator can reset the valve
8	1 = Operator can define the monitoring time for the valve startup
9	1 = Operator can define the monitoring time for the end position of the valve
10	1 = Operator can enable the monitoring time function of the valve (Bit 8 - 9)
11	1 = Operator can define or change the monitoring time for startup
12	1 = Operator can define the monitoring time for the motor status
13	1 = Operator can enable the monitoring time function of the motor (Bit 8 - 9)
14	1 = Operator can activate the Simulation function
15	1 = Operator can activate the Release for maintenance function
16	1 = Operator can change the limit (`AV`) for high alarm

Bit	Function
17	1 = Operator can change the limit (AV) for high warning
18	1 = Operator can change the limit (AV) for high tolerance
19	1 = Operator can change the limit (AV) for hysteresis
20	1 = Operator can change the limit (AV) for low alarm
21	1 = Operator can change the limit (AV) for low warning
22	1 = Operator can change the limit (AV) for low tolerance
23	1 = Operator can activate / deactivate messages via AV_AH_MsgEn
24	1 = Operator can activate / deactivate messages via AV_WH_MsgEn
25	1 = Operator can activate / deactivate messages via AV_TH_MsgEn
26	1 = Operator can activate / deactivate messages via AV_TL_MsgEn
27	1 = Operator can activate / deactivate messages via AV_WL_MsgEn
28	1 = Operator can activate / deactivate messages via AV_AL_MsgEn
29	1 =Operator can change the simulation value SimAV
30	1 = Operator can define the monitoring time for stopping
31	Not used

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

Restart lock after changing direction of rotation or switching off the motor

Use the input parameter IdleTime to enter a restart lock for changing the direction of rotation or restarting the motor. When the "Stop" command is given, the motor goes immediately into "Stop" mode, and IdleTime starts after the feedback (FbkOpening and FbkClosing = 0) is given. The motor cannot be started again (open or close) until the IdleTime has expired.

Limit monitoring of an additional analog value

This block provides the standard function Limit monitoring of an additional analog value (Page 91).

Limit monitoring with hysteresis

This block provides the standard function Limit monitoring with hysteresis (Page 97). It is performed via the input parameter AV_Hyst.

Suppressing messages using the MsgLock parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Interlocks

This block provides the following interlocks:

- Activation enable
- Interlock without reset ("Interlock")
- Interlock with reset ("Protection")

Refer to the section Interlocks (Page 99) as well as Influence of the signal status on the interlock (Page 103).

Motor protection function

This block provides the standard function Motor protection function (Page 99).

Rapid stop for motors

This block provides the standard function Rapid stop for motors (Page 106).

Torque monitoring

The block provides torque monitoring.

The signals of the torque monitoring switches are interconnected to input parameters `TorqOpen` and `TorqClose` for opening and closing the motor valve.

An active torque shutdown is displayed in the parameter either by the value 0 or by the signal status 16#00 or 16#28.

If the torque shutdown is active, the motor is stopped. You have the option of moving the valve in the opposite direction.

If, for example, the torque shutdown is active when the valve opens, you can still close the valve.

Active torque shutoff appears in the standard view of the display area for block states.

When the "Seal valve" function is enabled via Feature bit 8, the torque shutoff for the closing `TorqClose` is also evaluated (see section Sealing the valve).

Reset of motor control after valve end position has been reached

After a motor control in the direction of one of the end positions, open or closed, the valve is opened or closed accordingly. After the end position is reached, the block is notified of this via the `FbkOpen` or `FbkClose` input. The `Feature` bit Motor stop in end position depends only on the corresponding feedback signal (Page 184) can affect the dependencies of the feedback signals to downscale the motor in the end position.

Note

If you deactivate one of the valve feedback signals via the inputs `NoFbkOpen`, `NoFbkClose`, the corresponding valve feedback signal `FbkOpen`, `FbkClose` for the motor switch off in the end position is no longer evaluated. After the valve monitoring time `V_MonTiDynamic` has expired, the corresponding output of the valve feedback signal `FbkOpenOut`, `FbkCloseOut` is set and the associated motor control `Open`, `Close` is reset again.

In the case of missing valve feedback signals, the corresponding signals of the torque monitoring with the inputs `TorqOpen` or `TorqClose` must be used.

Sealing the valve

The function is activated via `Featurebit` 8 Sealing the valve (Page 175). The seal valve function combines the query of the end position CLOSED via the input parameter `FbkClose` with the limit violation of the configured torque via the input parameter `TorqClose`. This ensures that the valve is completely sealed.

The valve is only considered completely sealed when the feedback for the end position CLOSE has been received (0->1) and the torque cutoff for Closed is enabled. The torque shutoff should not come before the feedback in this case. The `FbkCloseOut` output shows whether the valve is sealed:

```
FbkCloseOut := FbkClose.Value has been received (0->1)
        And "Torque shutoff closed is enabled"
        And "Torque shutoff closed is enabled" did not come before FbkClose (0->1)
```

"Torque shutoff closed is enabled" means that `TorqClose = 0` or the signal status is `16#00` or `16#28`.

When "Torque shutoff closed is enabled" comes before the end position feedback closed, this is displayed in the faceplate in the standard view and the motor stops. Opening the valve is still possible.

Note

The command text for Close in the standard view and preview can be changed to Seal in the CFC at the `CloseMan` parameter in Text 1. See Section Labeling of buttons and text (Page 205)

Disabling interlocks

This block provides the standard function Disabling interlocks (Page 103).

Resetting the block in case of interlocks

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See VlvMotL error handling (Page 1402)

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `Trip`
- `V_MonDynErr`
- `V_MonStaErr`
- `M_MonDynErr`
- `M_MonStaErr`
- `FaultExt`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status for the block is formed from the following parameters:

- `FbkClsgOut.ST`
- `FbkOpngOut.ST`
- `FbkOpenOut.ST`
- `FbkCloseOut.ST`
- `LocalLi.ST`
- `OpenLocal.ST`
- `StopLocal.ST`
- `TorqClose.ST`
- `CloseLocal.ST`

- `Trip.ST`
- `TorqOpen.ST`
- `AV_Out.ST`
- `OpenChn.ST`
- `CloseChn.ST`
- `OpenAut.ST` (only if `Feature2.Bit10` = 1)
- `CloseAut.ST` (only if `Feature2.Bit10` = 1)
- `StopAut.ST` (only if `Feature2.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position in the "Automatic" mode:

- `OpenAut.ST`
- `CloseAut.ST`
- `StopAut.ST`

Forcing operating modes

This block provides the standard function Forcing operating modes (Page 41). Inputs `OpenForce` and `CloseForce` and `StopForce`force the block to open, close or stop.

Feedback monitoring

This block provides the standard function Monitoring the feedbacks (Page 97).

The `FbkOpen` and `FbkClose` feedback is monitored for the valve; the `FbKOpening` and `FbkClosing` feedback is monitored for the motor.

The monitoring of the feedback for the valve will not be active if it was stopped during opening or closing.

Monitoring valve feedback

The monitoring of valve feedback is set using the parameter `V_Monitor`.

Startup characteristics are monitored by setting parameter`V_MonTiDynamic`. The parameter `V_MonTiStatic` monitors compliance with the position.

Feedback errors are displayed at the corresponding parameters `V_MonDynErr` and/or `V_MonStaErr`.

Note

After the motor valve stops in the intermediate position or end position or after a startup without control (`Open`, `Close` = 0)" no monitoring of the feedback signals `FbkOpen` and `FbkClose` takes place during the `V_MonTiStatic` time. Changes to `FbkOpen` and `FbkClose` are applied. This means that the feedback is monitored again, also in stop state.

Note

When the "Seal valve" function is enabled via Feature bit 8, the torque shutoff for closing `TorqClose` is also evaluated (see section Sealing the valve (Page 175)).

Monitoring the motor feedback

The monitoring of motor feedback is set using the parameter `M_Monitor`.

Startup characteristics are monitored by setting parameter`M_MonTiDynamic`. The parameter `M_MonTiStatic` monitors compliance with the position.

Feedback errors are displayed at the corresponding parameters `M_MonDynErr` and/or `M_MonStaErr`.

Disabling feedback

This block provides the standard function Disabling feedback for valves (Page 99). Feedback monitoring can be deactivated separately for each feedback with `NoFbkOpen` or `NoFbkClose` as required.

Tracking the valve position after a motor stop or after a startup

Because of vibration or mass inertia after a motor stop the block tracks the valve position for the time `V_MonTiStatic`. During this time the static feedback monitoring is not active. The monitoring starts after the time `V_MonTiStatic`.

- Motor is stopped in end position open (`FbkOpen` =1, `FbkClose` =0):
  Open and intermediate position will be tracked.
  `FbkOpen` =1, `FbkClose` =0: valve goes to open position
  `FbkOpen` =0, `FbkClose` =0: valve goes to intermediate position
- Motor is stopped in intermediate position (`FbkOpen` =0, `FbkClose` =0):
  Open, Close and intermediate position can be tracked.
  `FbkOpen` =1, `FbkClose` =0: valve goes to open position
  `FbkOpen` =0, `FbkClose` =0: valve goes to intermediate position
  `FbkOpen` =0, `FbkClose` =1: valve goes to close position
- Motor is stopped in end position close (`FbkOpen` =0, `FbkClose` =1):
  Close and intermediate position can be tracked.
  `FbkOpen` =1, `FbkClose` =0: valve goes to open position
  `FbkOpen` =0, `FbkClose` =0: valve goes to intermediate position

Because of preparing the feedback signals after startup the block tracks the valve position for the time `V_MonTiStatic`.

- Open, Close and intermediate position can be tracked.
  `FbkOpen` =1, `FbkClose` =0: valve goes to open position
  `FbkOpen` =0, `FbkClose` =0: valve goes to intermediate position
  `FbkOpen` =0, `FbkClose` =1: valve goes to close position

Note

After resetting the block, in the case of interlocks or errors, the valve position will always be tracked, depending on `FbkOpen` and `FbkClose`.

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Specify warning times for control functions

This block provides the standard function Specifying warning times for control functions at motors and valves (Page 51).

You can generate warning signals when, for example, valves open. Warning signals can be generated in the following modes:

- Manual and automatic mode for motors, valves and dosers (Page 75) (Input parameter `WarnTiMan`)
- Manual and automatic mode for motors, valves and dosers (Page 75) (Input parameter `WarnTiAut`)

You specify the warning times in seconds using the input parameters `WarnTiMan` and `WarnTiAut`. If, for example, a valve opens, this is displayed at the output parameter with `WarnAct` = 1. The valve then opens after the set warning time has expired and `WarnAct` then returns to 0.

A corresponding warning is not output if the warning times (`WarnTiMan` or `WarnTiAut`) are specified with a smaller value than the `SampleTime` parameter.

Note

The warning is activated for each actuation that causes the motor to start, even if this means that the valve is moved to the neutral position

Simulating signals

This block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Additional value (`SimAV`, `SimAV_Li`)

In the case of internal simulation with immediate tracking of feedback, it is possible to simulate a position between the open and closed state (`FbkOpenOut` = `FbkCloseOut` = 0) by means of a stop command.

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Neutral position

This block provides the standard function Neutral position for motors, valves and controllers (Page 48).

Output signal as a pulse signal or static signal

This block provides the standard function Output signal as a static signal or pulse signal (Page 51). In addition to the static control outputs `Open` and `Close`, the block also has pulse outputs `P_Open`, `P_Close`, and `P_Stop`, which are dependent on the static control output.

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
4	Setting switch or button mode (Page 166)
8	Sealing the valve (Page 175)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
12	Motor feedback is not available (Page 156)
13	Separate monitoring time for stopping the motor (Page 168)
14	Enabling rapid stop via faceplate (Page 167)
15	Motor stop in end position depends only on the corresponding feedback signal (Page 184)
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)
18	Activating error state for external process control error CSF (Page 150)
19	Reset even with locked state (Page 164)

Bit	Function
20	Disable calculation of impulse control in LocalSetting 2 and 4 (Page 185)
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
27	Interlock display with LocalSetting 2 or 4 (Page 177)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

In switching mode (Bit 4 = 1), control is selected with the static signals `OpenAut` and `CloseAut`. If the `OpenAut` and `CloseAut` inputs are not set, the motor is stopped. Control via `StopAut` is not needed. If the "Activate command reset for control" function (Bit 3 = 1) is activated, the inputs `OpenAut` and `CloseAut` are reset to 0 after evaluation in the block.

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
4	Setting switch or button mode for local commands (Page 180)
5	Evaluation of the signal status of the interlock signals (Page 141)
8	Forcing operating modes in the "Local" mode (Page 183)
10	Considering bad quality of automatic commands or external values (Page 185)

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` or `EventTs2In` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- `OpenMan`
- `CloseMan`
- `StopMan`
- `RapidStp`

See also

Description of VlvMotL (Page 1386)

VlvMotL messaging (Page 1405)

VlvMotL I/Os (Page 1406)

VlvMotL block diagram (Page 1416)

VlvMotL modes (Page 1390)

EventTs functions (Page 1634)

7.12.4 VlvMotL error handling

Error handling of VlvMotL

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Invalid input signals
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
41	The value for the `LocalSetting` I/O is not within the approved limit of 0 to 4.
42	`LocalSetting` = 0 or `LocalSetting` = 3 or `LocalSetting` = 4 and `LocalLi` = 1

Error number	Meaning of the error number
51	`OpenLocal` = `1` and `StopLocal` = 1 `CloseLocal` = `1` and `StopLocal` = 1 `OpenLocal` = `1` and `CloseLocal` = 1 `OpenAut` = `1` and `StopAut` = 1 `CloseAut` = `1` and `StopAut` = 1 `OpenAut` = `1` and `CloseAut` = 1 `AutModLi` = `1` and `ManModLi` = 1 `OpenForce` = `1` and `StopForce` = 1 `CloseForce` = `1` and `StopForce` = 1 `OpenForce` = `1` and `CloseForce` = 1
52	`LocalAct` = `1` and `LocalSetting` = 2 or 4 and `SimOn` = 1

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

Invalid input signals

This error can be output by the block, see the section Error handling (Page 119).

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signals	Control reaction with Feature2 bit 3 =1
Local: `Localsetting` = 1 or `Localsetting` = 3	Pushbutton operation for local mode (`Feature2` bit 4 = 0): `OpenLocal` = 1 and `CloseLocal` = 1 or `OpenLocal` = 1 and `StopLocal` = 1 or `StopLocal` = 1 and `CloseLocal` = 1 Switching mode (`Feature2` bit 4 = 1): `OpenLocal` = 1 and `CloseLocal` = 1	Motor is started in valve neutral position direction.
Local: `Localsetting` = 1 or `Localsetting` = 3 and forcing	`OpenForce` = 1 and `CloseForce` = 1 or `OpenForce` = 1 and `StopForce` = 1 or `StopForce` = 1 and `CloseForce` = 1	
Forcing and no "local mode"	`OpenForce` = 1 and `CloseForce` = 1 or `OpenForce` = 1 and `StopForce` = 1 or `StopForce` = 1 and `CloseForce` = 1	
"Automatic mode" and no forcing	Pushbutton operation (`Feature` bit 4 = 0): `OpenAut` = 1 and `CloseAut` = 1 or `OpenAut` = 1 and `StopAut` = 1 or `StopAut` = 1 and `CloseAut` = 1 Switching mode (`Feature`bit 4 = 1): `OpenAut` = 1 and `CloseAut` = 1	
"Manual mode" and no forcing	`OpenMan` = 1 and `CloseMan` = 1 or `OpenMan` = 1 and `StopMan` = 1 or `StopMan` = 1 and `CloseMan` = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

VlvMotL block diagram (Page 1416)

VlvMotL I/Os (Page 1406)

VlvMotL functions (Page 1392)

VlvMotL modes (Page 1390)

Description of VlvMotL (Page 1386)

7.12.5 VlvMotL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Instance-specific messages

Process control fault

The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	AS process control message - fault	$$BlockComment$$ Motor feedback error
	SIG 2	AS process control message - fault	$$BlockComment$$ Motor protection triggered
	SIG 3	AS process control message - fault	$$BlockComment$$ Valve feedback error
	SIG 4	AS process control message - fault	$$BlockComment$$ External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter CSF. If it changes to CSF = 1, a process control fault is triggered (MsgEvId1, SIG 4).

Instance-specific messages

You can use up to three instance-specific messages with this block.

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 5	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 6	AS process control message - fault	$$BlockComment$$ External message 2
	SIG 7	AS process control message - fault	$$BlockComment$$ External message 3

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`ExtVa104`
5	`ExtVa105`
6	`ExtVa106`
7	`ExtVa107`
8	`ExtVa108`
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters `ExtVa104` ... `ExtVa108` , and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

VlvMotL block diagram (Page 1416)

VlvMotL modes (Page 1390)

7.12.6 VlvMotL I/Os

I/Os of VlvMotL

Input parameters

Parameter	Description	Type	Default
`AutModLi*`	1= "Automatic mode" via interconnection or SFC (controlled by `ModLiOp` = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`AutModOp*`	1 = "Automatic mode" via operator (controlled by `ModLiOp` = 1)	BOOL	0
`AV`	Input additional analog value, to be connected to `AV_Tech` of the AV block	ANY	
`AV_AH_Lim`	Limit high alarm	REAL	95.0
`AV_AL_Lim`	Limit low alarm	REAL	5.0
`AV_Hyst`	Hysteresis for alarm, warning and tolerance limits	REAL	1.0
`AV_TH_Lim`	Limit high tolerance	REAL	85.0
`AV_TL_Lim`	Limit low tolerance	REAL	15.0
`AV_WH_Lim`	Limit high warning	REAL	90.0

Parameter	Description	Type	Default
AV_WL_Lim	Limit low warning	REAL	10.0
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Bypassing interlock is active in "local mode" and in simulation	BOOL	0
CloseAut*	1 = Select Close valve in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseChnST	Signal status of output channel for Close Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
CloseForce	1 = Force valve closure	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseLocal	1 = Select Close valve in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseMan*	1 = Select Close valve in "manual mode"	BOOL	0
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EN	1 = Called block will be processed	BOOL	1
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16TS block. When this interconnection is configured, the messages of the EventTs, Event16TS block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	

Parameter	Description	Type	Default
EventTs2In	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTs2In input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkClose	1 = Valve closed feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkClosing	1 = Valve closing feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkOpen	1 = Valve open feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF

Parameter	Description	Type	Default
FbkOpening	1 = Valve opening feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Feature	I/O for additional functions (Page 1392) 1 = Separate monitoring time for stopping the motor	STRUCT • Bit 0: BOOL • ... • Bit 13: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0 • 0 • 0
Feature2	I/O for additional functions (Page 1392)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
IdleTime*	Wait time for change of direction or restart in [s]	REAL	5.0
Intlock	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is active	BOOL	1
LocalLi	1 = Activate "local mode" via plant signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalOp*	1 = "Local mode" via operator	BOOL	0
LocalSetting	Properties for the Local mode (Page 79)	INT	0
ManModLi*	1 = Manual mode via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1= "Manual mode" via: OS operator (controlled via ModLiOp = 0)	BOOL	1
ModLiOp	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonSafePos	1 = Go to neutral position in the event of monitoring errors	BOOL	1
MonTiDynamic*	Monitoring time for feedback errors or feedback start error after successful operation in [s]	REAL	3.0
MonTiDyStop*	Monitoring time for feedback stop errors after successful operation in [s]	REAL	3.0
M_Monitor	1 = Motor feedback monitoring	BOOL	1

Parameter	Description	Type	Default
M_MonTiDynamic*	Motor monitoring time after operation in [s]	REAL	3.0
M_MonTiStatic*	Monitoring time for feedback errors without operation in [s]	REAL	3.0
MS_RelOp*	1= Release for maintenance via OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp	1 = Release for maintenance via OS operator	BOOL	0
NoFbkClose	1 = No feedback present for "valve closed"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NoFbkOpen	1 = No feedback present for "valve open"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = In use by a batch	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpenAut*	1 = Select Open valve in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenChnST	Signal status of output channel for Open Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
OpenForce	1 = Force valve opening	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenLocal	1 = Select Open valve in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenMan*	1 = Select Open valve in "manual mode"	BOOL	0
OpSt_In	Input parameter for local operator authorization, to be connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000

Parameter	Description	Type	Default
OS_Perm	I/O for operator permissions (Page 1392) 1 = Operator can set or change the monitoring time for "Control: Start" 1 = Operator can set or change the monitoring time for "Control: Stop"	STRUCT • Bit 0: BOOL • Bit11: BOOL • Bit 20: BOOL • Bit 30: BOOL	- • 1 • 1 • 1 • 1
Permit	1 = Enable for opening / closing from neutral position 0 = Valve activation not enabled on OS	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Perm_En	1 = Activation enable (enable, Permit parameter) is active	BOOL	1
Protect	0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block 1 = Protective interlocking not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Prot_En	1 = Protective interlock (protection, Protect parameter) is active	BOOL	1
PulseWidth*	Pulse width of control signal [s]	REAL	3.0
RapidStp*	Rapid stop for the motor 0 = Motor On 1 = Motor Off	BOOL	0
RstLi*	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SafePos	Neutral position for valve: 0 = Closed 1 = Open 2 = stop	INT	2
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SimAV*	Additional value used for SimOn = 1	REAL	0.0
SimAV_Li	Additional analog value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0 • 16#80
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
SimOnLi	1= Simulation per interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	
StepNo	Batch step number	DWORD	16#00000000
StopAut*	1 = Stopping the motor in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopForce	1 = Force motor stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopLocal	1 = Stopping the motor in "local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopMan*	1 = Stopping the motor in "manual mode"	BOOL	0
TorqOpen	0 = Torque shutdown active when opening 1 = "Good" state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
TorqClose	0 = Torque shutdown active when closing 1 = "Good" state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Trip	1 = Motor is in "good" state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UA2unit	Unit of measure for analog auxiliary value 2	INT	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
V_Monitor	1 = Valve feedback monitoring	BOOL	1

Parameter	Description	Type	Default
V_MonTiDynamic*	Valve monitoring time after operation in [s]	REAL	5.0
V_MonTiStatic*	Monitoring time for valve feedback errors without operation in [s]	REAL	5.0
WarnTiAut*	Prewarning of valve movement from neutral position in "automatic mode" in [s]	REAL	0.0
WarnTiMan*	Prewarning of valve movement from neutral position in "manual mode" in [s]	REAL	0.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80
AV_OpScale	Limit for scale in AV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
AV_Out	Output additional analog value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
AV_Unit	Unit of measure for additional analog value	INT	0
CascaCut	Cascade connection: 1 = Control chain from master controller to secondary valve is interrupted	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Close	Control output 1= Close valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Closed	1 = Valve is closed	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Closing	1 = Valve is closing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CurrMonV	Current valve monitoring time [s]	DINT	0
CurrMonM	Current motor monitoring time [s]	DINT	0

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see VlvMotL error handling (Page 1402)	INT	-1
FbkCloseOut	Valve closed feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkClsgOut	Valve closing feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpenOut	Valve open feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpngOut	Valve opening feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalAct	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock (Intlock, Permit, Protect) or Trip is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MonDynErr	1 = Feedback error or feedback start error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MonDynStopErr	1 = Feedback stop error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
M_MonDynErr	1 = Motor feedback error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
M_MonStaErr	1 = Motor feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Open	Control output: 1 = Open the valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Opened	1 = Valve is open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Opening	1 = Valve is opening	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
P_Close	1 = Pulse signal to close valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Open	1 = Pulse signal to open valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Stop	0 = Pulse signal for stopping the valve	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
R_StpAct	1 = Rapid stop of the motor is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1386)	DWORD	16#00000000
Status2	Status word 2 (Page 1386)	DWORD	16#00000000
Status3	Status word 3 (Page 1386)	DWORD	16#00000000
Status4	Status word 4 (Page 1386)	DWORD	16#00000000
Stop	1 = Motor stopped and valve is in intermediate position	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
V_MonDynErr	1 = Valve feedback error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
V_MonStaErr	1 = Valve feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
WarnAct	1 = Prewarning for valve movement away from neutral position active (parameters WarnTiAut and WarnTiMan)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

VlvMotL messaging (Page 1405)

VlvMotL block diagram (Page 1416)

VlvMotL modes (Page 1390)

Error handling (Page 119)

7.12.7 VlvMotL block diagram

VlvMotL block diagram

A block diagram is not provided for this block.

See also

VlvMotL I/Os (Page 1406)
VlvMotL messaging (Page 1405)
VlvMotL error handling (Page 1402)
VlvMotL functions (Page 1392)
VlvMotL modes (Page 1390)
Description of VlvMotL (Page 1386)

7.12.8 Operator control and monitoring

7.12.8.1 VlvMotL views

Views of the VlvMotL block

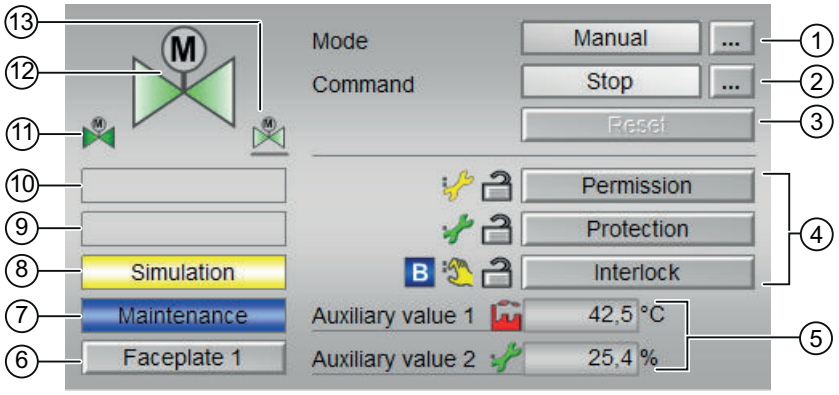
The VlvMotL block provides the following views:

- VlvMotL standard view (Page 1418)
- Limit view of motors (Page 288)
- Alarm view (Page 296)
- Trend view (Page 299)
- VlvMotL parameter view (Page 1421)
- VlvMotL preview (Page 1424)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for VlvMotL (Page 1427)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

7.12.8.2 VlvMotL standard view

VlvMotL standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Open, close and stop the motor valve

This area shows you the default operating state for the motor valve. The following states can be shown and executed here:

- "Open"
- "Close"
- "Stop"
- "Rapid stop"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

(3) Resetting the block

Click "Reset" for interlocks or errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(4) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the section Interlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

(5) Display of auxiliary values

This display is only visible when the corresponding block input is connected.

You can use this area to display two auxiliary values that have been configured in the engineering system. You can find additional information on this in the section Displaying auxiliary values (Page 207).

(6) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(7) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

(8) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"
- "Delay"

You will find more detailed information on this in the sections Simulating signals (Page 58) and Display of delay times (Page 250).

(9) Display area for block states

This area provides additional information on the operating state of the block:

- "Motor protection"
- "External error"
- "Torque active"
- "End position error"
- "Control error"
- "Invalid signal"
- "Changeover error"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97) , Error handling (Page 119) (section "Invalid input signals" and "Mode switchover error") and Motor protection function (Page 99).

(10) Display area for block states

This area provides additional information on the operating state of the block:

- "Forced open"
- "Forced close"
- "Forced stop"
- "Request 0/1": A reset to "automatic mode" is expected.

You can find additional information on this in the section Forcing operating modes (Page 41).

(11) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the valve would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

(12) Status display of the motor valve

The current status of the motor valve is graphically displayed here.

You can find more information about this in section Block icon for VlvMotL (Page 1427)

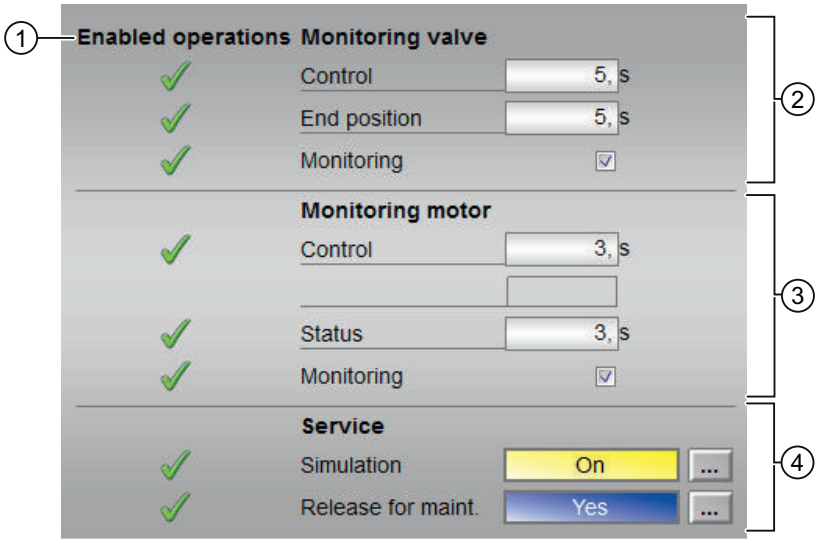
(13) Neutral position of the valve

This representation shows the neutral position for the valve:

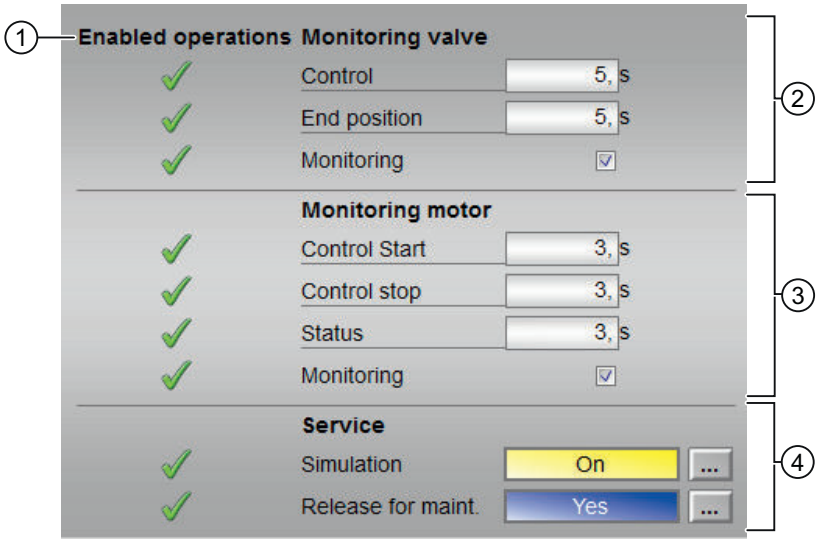
- Green: Neutral position is "Open"
- Gray: Neutral position is "Closed"
- Light green: Neutral position is "Stop"

7.12.8.3 VlvMotL parameter view

Parameter view of VlvMotL



Parameter view for VlvMotL with `Feature` bit 13 = 0



Parameter view for VlvMotL with `Feature` bit 13 = 1

(1) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

(2) Monitoring valve

In this area, you change parameters and therefore influence the valve. You can find additional information on this in the section Changing values (Page 253).

You can influence the following parameters:

- "Control": Monitoring time for the valve run time (dynamic)
- "End position": Monitoring time for maintaining the valve position (static)
- "Monitoring":
  You can enable monitoring by selecting the check box (☑)
  You can find additional information on this in the section Monitoring the feedbacks (Page 97).

(3) Monitoring motor

In this area, you change parameters and therefore influence the motor. You can find additional information on this in the section Changing values (Page 253).

You can influence the following parameters:

Feature bit **13 = 0**

- "Control": Monitoring time during startup and shutdown of the motor (dynamic)
- "Status": Monitoring time during permanent operation of the motor (static)

Feature bit **13 = 1**

- "Control start": Monitoring time during startup of the motor (dynamic)
- "Control stop": Monitoring time during shutdown of the motor (dynamic)
- "Status": Monitoring time during permanent operation of the motor (static)
- "Monitoring":
  You can enable monitoring by selecting the check box (☑)
  You can find additional information on this in the section Monitoring the feedbacks (Page 97).

(4) Service

You can select the following functions in this area:

- "Simulation"
- "Release for maintenance" (with display for a maintenance request)

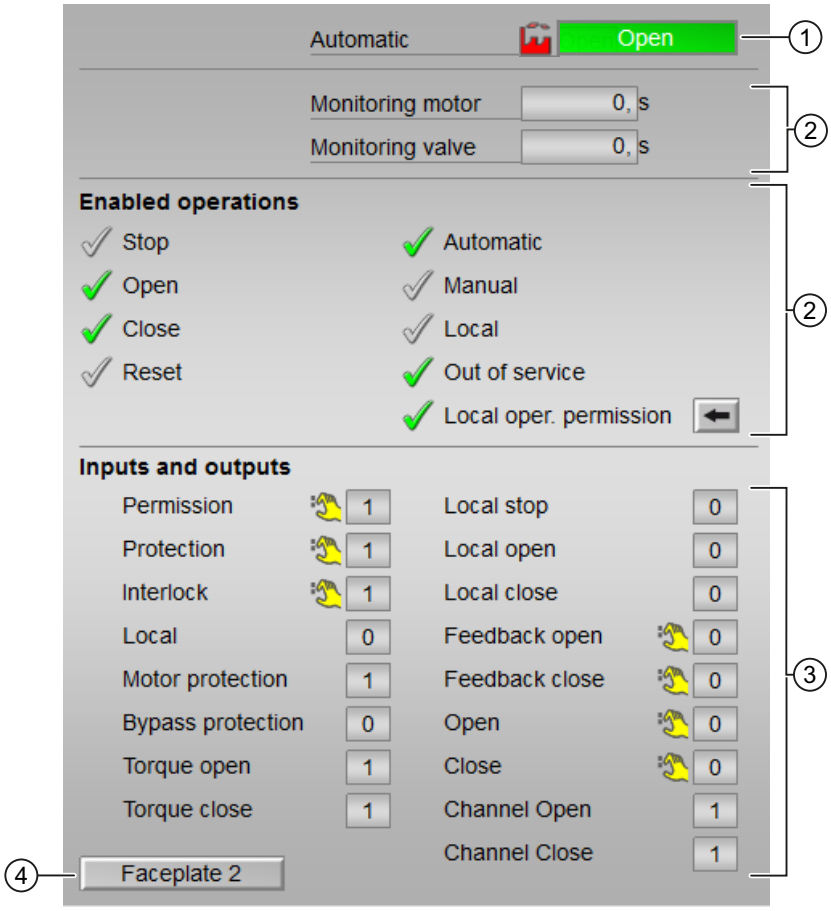
You can find additional information on this in the section Switching operating states and operating modes (Page 251).

You can find information on this area in the section:

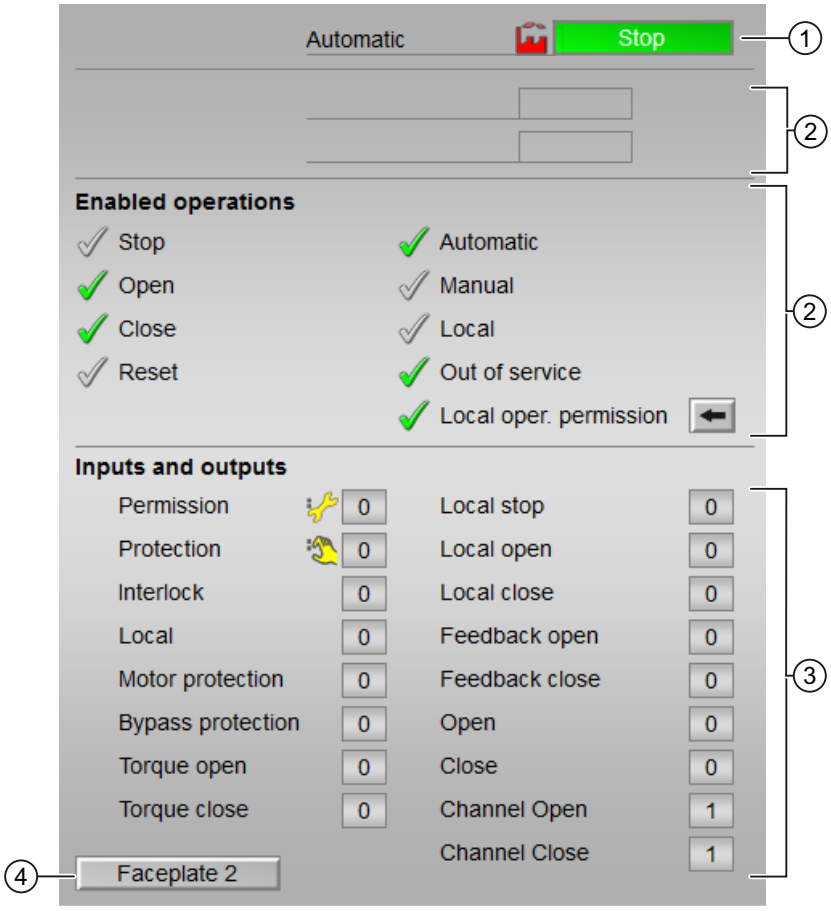
- Simulating signals (Page 58)
- Release for maintenance (Page 64)

7.12.8.4 VlvMotL preview

Preview of VlvMotL



Display of the current monitoring time of the motor/valve is visible.



Display of the current monitoring time of the motor/valve is not visible.

(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `OpenAut`
- `CloseAut`
- `StopAut`

(2) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Stop": You can stop the motor of the valve.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205) .
- "Open": You can open the motor valve.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205) .
- "Close": You can close the motor valve.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205) .
- "Reset": You can reset the motor valve if interlocks or errors occur.
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Local": You can switch to "local mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248) .
- "Monitoring motor": Display of the current monitoring time of the motor.
- "Monitoring valve": Display of the current monitoring time of the valve.

(3) Displaying current control signals

This area shows the most important parameters for this block with the current selection:

- "Permission":
  This display is only visible when the corresponding block input is connected.
  - 0 = Motor valve activation not enabled on OS
  - 1 = Enable for "opening"/"closing" from the neutral position
- "Protection":
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block
  - 1 = "Good" state

- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"
- "Torque opening": 0 = Torque shutdown when opening
- "Torque closing": 0 = Torque shutdown when closing
- "Local stop": 1 = Stopping the motor valve in "local mode"
- "Local open": 1 = Opening the motor valve in "local mode"
- "Local close": 1 = Closing the motor valve in "local mode"
- "Feedback open": 1 = Motor valve is open
- "Feedback close": 1 = Motor valve is closed
- "Open": 1 = Motor valve is opened
- "Close": 1 = Motor valve is closed
- "Channel Open": Signal from the output channel block for "Open"
- "Channel Close": Signal from the output channel block for "Close"

(4) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

7.12.8.5 Block icon for VlvMotL

Block icons for VlvMotL

A variety of block icons are available with the following functions:

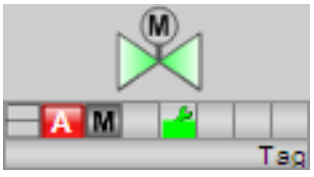
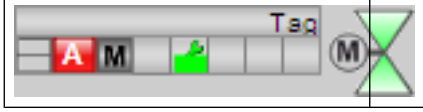
- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes

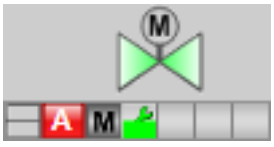
- Signal status, release for maintenance
- Displays for bypassing interlocks
- Interlocks
- Memo display
- Valve status display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Non-rotating block icon
	4	Non-rotating block icon

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	

Icons	Selection of the block icon in CFC	Special features
	5	
	6	
	7	
	8	
	9	
	10	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Valve status display

The following valve states are shown here:

Icon	Meaning
	Valve open
	Valve closed

Icon	Meaning
	Error at valve
	Valve is opening
	Valve is closing
	Valve stop
	Valve closed
	Valve is closing

7.13 VlvAnL - Control valve

7.13.1 Description of VlvAnL

Object name (type + number) and family

Type + number: FB 1896

Family: Drives

Area of application for VlvAnL

The block is used for the following applications:

- Control of an analog control valve and positioner with adjustable neutral position
- Control of an optional auxiliary valve for regulating the auxiliary power of the control valve

How it works

The control valve is brought to a specified position using and analog activation signal. The activation signal can be formed by a ramp function in this case.

The block forms the manipulated variable error from the difference between the activation signal and the acquired position feedback and can monitor it for adherence to high and low limits.

The control valve is monitored for the "Open"/"Closed" position. The block can be connected with a digital limit switch for this purpose. The block can generate the digital position signals itself through the adjustable limits for the "Open"/"Closed" position.

Missing feedback can be derived from the control in the block.

Various inputs are available for control purposes. The next sections provide more detailed information on configuration, operating principles, visualization and operation.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

For the VlvAnL block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Control valve (VlvAnL) (Page 2344)
- Control valve for PA/FF devices (ValveAnalog_Fb) (Page 2345)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

If `Feature` Bit Set startup characteristics (Page 137) = 0, the following applies to the startup characteristics:

- `Feature` Bit 16 = 0 closes the main valve
- With `Feature` Bit 16 = 1 the main valve is moved into the neutral position

Status word allocation for `Status1` parameter

You can find a description for each parameter in section VlvAnL I/Os (Page 1454).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutoAct.Value`
6	`LocalAct.Value`
7	0: Open padlock in the block icon 1: Closed padlock in the block icon
8	Control valve "Open"/"Closed" command (0 = "Closed", 1 ="Open")
9	`FbkOpenOut.Value` control valve
10	`FbkCloseOut.Value` control valve
11	1 = Feedback error control valve without control change
12	1 = Feedback error control valve due to control change
13	`BypProt`
14	1 = Invalid signal status
15	1 = Mode changeover error
16	1 = `Intlock` is active
17	1 = `Permit` is active
18	1 = `Protect` is active
19	Display "Forced open"
20	Display "Forced stop"
21	1 = `Force`
22	Automatic preview of auxiliary valve (1 = "Open")
23	1 = Bumpless switchover to "automatic mode" enabled
24	1 = Actuator active (`PosDiClose` < `MV` <`PosDiOpen`)
25	`UserAna1` interconnected
26	`UserAna2` interconnected
27	Auxiliary valve "Open"/"Closed" command (0 =" Closed", 1 = "Open")

Status bit	Parameter
28	`FbkOpenAuxVOut.Value` (auxiliary valve)
29	`FbkCloseAuxVOut.Value` (auxiliary valve)
30	1 = Feedback error auxiliary valve without control change
31	1 = Feedback error auxiliary valve due to control change

Status word allocation for `Status2` parameter

Status bit	Parameter
0	1 = `MsgLock` message suppression active
1	1 = "Open" and "Close" commands deactivated (command line not visible)
2	1 = Display for interlocks in block icon
3	1 = Neutral position control valve "Open"
4	1 = Neutral position control valve "Closed"
5	1 = Neutral position control valve "Stop"
6	1 = Operator can reset the valve
7	`WarnAct.Value`
8	1 = External manipulated variable active (`MV_ExtAct.Value`)
9	1 = Forced manipulated variable (`MV_Forced`) is output without limit at output `MV`
10	1 = Tracking of manipulated variables `MV`, `MV_TrkOn.Value` = 1 and `MV_ForOn.Value` = 0
11	1 = Manipulated variable greater than limit (low) for manipulated variable `MV` (`MV.Value` > `ManLoLim`)
12	1 = Input parameter `Rbk` is not interconnected (`RbkOut.ST` = 16#FF)
13	1 = Input parameter `FbkAuxVClose` is connected
14	1 = Input parameter `FbkAuxVOpen` is connected
15	1 = Input parameter `FbkClose` is connected
16	1 = Input parameter `FbkOpen` is connected
17	For the Error status display in the Closed valve
18	Reset request in automatic
19	1 = No impact of input signals on "local mode" with `LocalSetting` = 2 and `LocalSetting` = 4
20	1 = Control valve open
21	1 = Control valve closed
22	1 = Control valve opening
23	1 = Control valve closing
24	1 = Control valve in intermediate position ("Stop")
25	1 = Position reached
26	Control of valve, open auxiliary valve
27	Control of valve, close auxiliary valve
28	1 = Control valve is in intermediate position
29	For the Error status display in the Opened valve
30	1 = Bypass information from previous function block
31	`MS_RelOp`

Status word allocation for `Status3` parameter

Status bit	Parameter
0	Effective signal 1 of the message block connected via `EventTsIn`
1	Effective signal 2 of the message block connected via `EventTsIn`
2	Effective signal 3 of the message block connected via `EventTsIn`
3	Effective signal 4 of the message block connected via `EventTsIn`
4	Effective signal 5 of the message block connected via `EventTsIn`
5	Effective signal 6 of the message block connected via `EventTsIn`
6	Effective signal 7 of the message block connected via `EventTsIn`
7	Effective signal 8 of the message block connected via `EventTsIn`
8	1 = "Interlock" button is enabled
9	1 = "Permission" button is enabled
10	1 = "Protection" button is enabled
11	1 = Manipulated variable difference high limit violated (`ER_AH_Act.Value`)
12	1 = Manipulated variable difference low limit violated (`ER_AL_Act.Value`)
13	1 = Monitor manipulated variable difference high limit (`ER_AH_En`)
14	1 = Monitor manipulated variable difference low limit (`ER_AL_En`)
15	1 = Report manipulated variable difference high limit violation (`ER_AH_MsgEn`)
16	1 = Report manipulated variable difference low limit violation (`ER_AL_MsgEn`)
17	1 = Readback value high limit violated (`RbkWH_Act.Value`)
18	1 = Readback value low limit violated (`RbkWL_Act.Value`)
19	1 = Monitor readback value high limit (`RbkWH_En`)
20	1 = Monitor readback value low limit (`RbkWL_En`)
21	1 = Report readback value high limit violation (`RbkWH_MsgEn`)
22	1 = Report readback value low limit violation (`RbkWL_MsgEn`)
23	1 = Automatic preview control valve "Open"
24	1 = Automatic preview control valve "Closed"
25	1 = Automatic preview control valve "Stop"
26	1 = Show automatic preview in the standard view
27	1 = Auxiliary valve present
28	`GrpErr.Value`
29	`RdyToStart.Value`
30	`MV_UpRaAct`, `MV_DnRaAct` limits enabled for gradient mode (`MV_RateOn = 1`)
31	`SimLiOp.Value`

Status word allocation for `Status4` parameter

Status bit	Parameter
0	External error generated by FaultExt or external control system fault CSF with set Feature bit 18 Activating error state for external process control error CSF (Page 150)
1	Delay of the `ER_AH_Lim` message
2	Delay of the `ER_AL_Lim` message

Status bit	Parameter
3	Delay of the RbkWH_Lim message
4	Delay of the RbkWL_Lim message
5	Collection of message delays
6	1 = MV ramp active
7 - 22	Not used
23	Hidden bypass signal in Permit
24	Hidden bypass signal in interlock
25	Hidden bypass signal in Protect
26	Feature2 bit 2: Separate bypass signal
27	1 = Input parameter MV_ChnST is interconnected
28	1 = Input parameter CtrlChnST is interconnected
29	Current monitoring time is visible
30	Current auxiliary valve monitoring time is visible
31	Not used

Status word allocation for Status5 parameter

Status bit	Parameter
0 - 7	Effective signal 9..16 of the message block connected via EventTsIn
8 - 31	Not used

See also

VlvAnL modes (Page 1435)

VlvAnL functions (Page 1437)

VlvAnL error handling (Page 1450)

VlvAnL messaging (Page 1452)

VlvAnL block diagram (Page 1466)

7.13.2 VlvAnL modes

VlvAnL operating modes

The block can be operated using the following modes:

- Local mode (Page 79)
- Automatic mode (Page 75)
- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

The block supports the local modes 2 and 4. Therefore, the control settings for the block are also made based on internal adjustment of the feedback value.

You can find general information on "Local mode", switching modes and Bumpless switchover in the Local mode (Page 79) section.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and Bumpless switchover in the Manual and automatic mode for motors, valves and dosers (Page 75) section.

With auxiliary valve

In "automatic mode", the automatic commands affect the auxiliary valve, which you can

- "Open" (`OpenAut` = 1)
- "Close" (`CloseAut` = 1)

Without auxiliary valve

In "automatic mode", the automatic commands affect the control valve, which you can

- "Open" (`OpenAut` = 1)
- "Close" (`CloseAut` = 1)

Note

If no auxiliary valve is configured, no internal manipulated variable specifications can be made in "automatic mode". Manipulated variable specification is set to external when the mode is switched to "automatic".

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

With auxiliary valve:

In "manual mode", the operator actions affect the auxiliary valve, which you can:

- "Open" (`OpenMan` = 1)
- "Close" (`CloseMan` = 1)

Note

If no auxiliary valve is configured, no external manipulated variable specifications can be made in the manual operating mode. With an internal manipulated value, the switchover from "automatic mode" to "manual mode" is bumpless. With an external manipulated value, the switchover is bumpless only if `MV_TrkExt` = 1 is parameterized.

Without auxiliary valve:

In "manual mode", the operator actions affect the control valve, which you can:

- "Open" (`OpenMan` = 1)
- "Close" (`CloseMan` = 1)

Note

If no auxiliary valve is configured, no external manipulated variable specifications can be made in the manual operating mode. Manipulated variable specification is set to internal when the mode is switched to manual. The switching is bumpless depending on the parameter `MV_TrkExt`.

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of VlvAnL (Page 1431)

VlvAnL functions (Page 1437)

VlvAnL error handling (Page 1450)

VlvAnL messaging (Page 1452)

VlvAnL I/Os (Page 1454)

VlvAnL block diagram (Page 1466)

7.13.3 VlvAnL functions

Functions of VlvAnL

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Interlocks

This block provides the following interlocks:

- Activation enable ("Permission")
- Interlock without reset ("Interlock")
- Interlock with reset ("Protection")

Refer to the section Interlocks (Page 99) as well as Influence of the signal status on the interlock (Page 103).

Resetting the block in case of interlocks or errors

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External error (`FaultExt`), external control system fault (`CSF`)

This block provides the possibility to pass an external error via the `FaultExt` parameter or an external control system fault via the CSF parameter. See VlvAnL error handling (Page 1450)

Group error

This block provides the standard function Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- `CSF`
- `MonDynErr`
- `MonStaErr`
- `MonDynAuxVErr`
- `MonStaAuxVErr`
- `FaultExt`

Outputting a signal for start readiness

This block provides the standard function Outputting a signal for start readiness (Page 53).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

Simulating signals

This block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Position feedback (`SimRbk`, `SimRbkLi`)

Using a manipulated variable ramp

The block provides the standard function Using a manipulated variable ramp (Page 125)

This function is ignored with tracking and forced tracking.

Gradient limiting of the manipulated variable

This block provides the standard function Gradient limiting of the manipulated variable (Page 126)

Note

This function is ignored with tracking, forced tracking, forced operating states and travel to the neutral position.

Tracking and limiting a manipulated variable

You can correct the manipulated variable output to the tracking value `MV_Trk` or the manipulated variable feedback `Rbk` to realize Bumpless switchover. To track the manipulated variable output, you have to set the parameter `MV_TrkOn` = 1.

If the parameter is `MV_TrkRbk` = 0, the manipulated variable output is corrected to the tracking value `MV_Trk`. The manipulated variable output `MV` is limited to the `MV_HiLim` and `MV_LoLim` parameters.

If the parameter is `MV_TrkRbk` = 1, the manipulated variable output is tracked to the position feedback `Rbk`. There are no limits here.

The "Tracking" text is displayed additionally in the standard view of the faceplate.

Tracking has a higher priority than interlock for a control valve.

Forming the group status for interlocks

This block provides the standard function Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The signal status of the output for the manipulated variable feedback `RbkOut` always corresponds to the signal status of the `Rbk` input or when the block is in simulation, the signal status of the 16#60 output.

The signal status of the `FbkCloseOut` and `FbkOpenOut` outputs is fetched and sent from the worst signal status from the `RbkOut` output and the corresponding feedback signal inputs `FbkClose` and `FbkOpen`.

The signal status of the manipulated variable output `MV` always corresponds to the signal status of the input parameter `MV_Ext` or `MV_Int`, depending on how the setpoint is specified. If the internal manipulated variable `MV_Int` is used, the signal status is always output as 16#80.

The signal status of manipulated variable difference `ER` is fetched and sent from the worst signal status of the two outputs `RbkOut` and `MV`.

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `FbkOpenOut.ST`
- `FbkCloseOut.ST`
- `FbkAuxVOpenOut.ST`
- `FbkAuxVCloseOut.ST`
- `RbkOut.ST`
- `LocalLi.ST`
- `MV_Chn.ST`
- `CtrlChn.ST`
- `OpenAut.ST` (only if `Feature2.Bit10` = 1)
- `CloseAut.ST` (only if `Feature2.Bit10` = 1)
- `MV_Ext.ST` (only if `Feature2.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position in the "Automatic" mode:

- `OpenAut.ST`
- `CloseAut.ST`
- `MV_Ext.ST`

Forcing operating modes

This block provides the standard function Forcing operating modes (Page 41).

Note

If the block is operated with a auxiliary valve, the `OpenForce` and `CloseForce` commands also affect the auxiliary valve.

The high range limit (`MV_HiLim`) and the low range limit (`MV_LoLim`) are output at `MV` for `OpenForce` and `CloseForce` respectively.

With `CloseForce`, the auxiliary valve is closed, thereby triggering the neutral position of the control valve regardless of the `MV` output.

A configured ramp limit has no effect.

Feedback monitoring

This block provides the standard function Monitoring the feedbacks (Page 97).

Digital feedback signals for the "Open" and "Closed" positions are formed from the position feedback:

- Feedback for "Open" position: `Rbk` ≥ `PosDiOpen`
- Feedback for "Closed" position: `Rbk` ≤ `PosDiClose`

Specifying control valve positions:

There are the positions:

- Valve closed ("Closed")
- Valve closes
- Valve open ("Open")
- Valve opens
- Valve has reached defined position

Valve closed ("Closed")

When the control valve reaches the "Close" position, the output is `FbkCloseOut.Value` = 1:

- With a binary limit switch for the "Close" position (`NoFbkClose` = 0):
  The valve is considered closed when `FbkCloseOut` is set. `FbkCloseOut` is set when `Rbk` ≤ `PosDiClose` and `FbkClose` = 1.
- Without a binary limit switch for the "Close" position (`NoFbkClose` = 1):
  The valve is considered closed when `FbkCloseOut` is set. `FbkCloseOut` is set when `Rbk` ≤ `PosDiClose`.

Valve closes

If the control valve travels in the direction of the "Close" position, the output is `FbkClsgOut = 1`:

- `FbkClsgOut` is set when `MV.Value < RbkOut.Value` and not when `PosReached.Value = 1 .`

Valve open ("Open")

When the control valve reaches the "Open" position, the output is `FbkOpenOut.Value = 1`:

- With a binary limit switch for the "Open" position (`NoFbkOpen = 0`):
  The valve is considered open when `FbkOpenOut` is set. `FbkOpenOut` is set when `Rbk` ≥ `PosDiOpen` and `FbkOpen = 1`.
- Without a binary limit switch for the "Open" position (`NoFbkOpen = 1`):
  The valve is considered open when `FbkOpenOut` is set. `FbkOpenOut` is set, when `Rbk` ≥ `PosDiOpen`.

Valve opens

If the control valve travels in the direction of the "Open" position, the output is `FbkOpngOut` = 1:

- `FbkOpngOut` is set when `MV.Value > RbkOut.Value` and is not `PosReached.Value = 1`.

Valve has reached defined position

If the control valve is sent to a specified intermediate position (`MV` > `PosDiClose` and `MV` < `PosDiOpen`), the targeted position is reached when the difference `MV.Value - RbkOut.Value` is within the configured tolerance range ± `PosDeadBand` and therefore `ER.Value = 0.0`.

If the control valve is set to the end position "Open" (`MV.Value >= PosDiOpen`), this position is reached when `ER.Value = 0.0` and also `FbkOpenOut.Value = 1`.

If the control valve is set to the end position "Closed" (`MV.Value <= PosDiClose`), this position is reached when `ER.Value = 0.0` and also `FbkCloseOut.Value = 1`.

When the control valve reaches the specified position, the `PosReached` = 1 output is set.

Dynamic monitoring

The monitoring is based on the effective feedback signals `FbkOpenOut` and `FbkCloseOut` and not directly on the feedback inputs `FbkOpen` and `FbkClose`.

A dynamic monitoring error is generated if:

- The "Close" end position is targeted (`MV` ≤ `PosDiClose`) and this position (`FbkCloseOut`) is not reached within the monitoring time `MonTiDynamic`.
- The "Close" end position is targeted (`MV` ≤ `PosDiClose`) and the feedback of the binary limit switch `FbkClose` is already within the control range.
- The "Open" position is targeted (`MV` ≥ `PosDiOpen`) and this position (`FbkOpenOut`) is not reached within the monitoring time `MonTiDynamic`.
- The "Open" end position is targeted (`MV` ≥ `PosDiOpen`) and the feedback for the binary limit switch `FbkOpen` is already within the control range.
- A further intermediate position is controlled and this position is not reached (`PosReached` = 0) within the monitoring time `MonTiDynamic`.

Static monitoring

The monitoring is based on the effective feedback signals `FbkOpenOut` and `FbkCloseOut` and not on the feedback inputs `FbkOpen` and `FbkClose`.

A static monitoring error is generated if:

- The "Close" end position (`FbkCloseOut` = 1) is abandoned without a command having been issued beforehand, and the monitoring time `MonTiStatic` has expired.
- The "Close" end position is reached (`FbkCloseOut` = 1), the feedback of the binary limit switch `FbkClose` is gone, and monitoring time `MonTiStatic` has expired.
- The "Open" end position (`FbkOpenOut` = 1) is abandoned without a command having been issued beforehand and the monitoring time `MonTiStatic` has expired.
- The "Open" end position is reached (`FbkOpenOut` = 1), the feedback of the binary limit switch `FbkOpen` is gone, and monitoring time `MonTiStatic` has expired.
- The valve is in intermediate position and with constant `MV`, the analog feedback `Rbk` has been changed so much that the position is no longer reached (`PosReached` = 0) and the monitoring time `MonTiStatic` has expired.

Suppressing messages using the MsgLock parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Monitoring the feedback for the auxiliary valve

The auxiliary valve provides dynamic and static monitoring of the feedback. No special monitoring time can be configured for overseeing the retention of the end position of the auxiliary valve. If the end position is exited without the corresponding command, a "monitoring runtime error" is reported after expiration of the time configured under "Control".

Monitoring is disabled by default and no connections are displayed.

The following parameters can be used to monitor the auxiliary valve

Parameter	Function
FbkAuxVOpen	1 = Auxiliary valve open feedback signal
FbkAuxVClose	1 = Auxiliary valve closed feedback signal
NoFbkAuxVOpen	1 = No feedback present for Auxiliary valve open (default = 1)
NoFbkAuxVClose	1 = No feedback present for Auxiliary valve closed (default = 1)
MonitorAuxV	1 = Feedback monitoring of the auxiliary valve (default = 0)
MonAuxVTime	Monitoring time after auxiliary valve operation in [s]
MonDynAuxVErr	Dynamic monitoring error pending
MonStaAuxVErr	Static monitoring error pending

Otherwise, the monitoring works like the monitoring function of the control valve.

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Neutral position

This block provides the standard function Neutral position for motors, valves and controllers (Page 48).

The neutral position (de-energized state) for the auxiliary valve is set using the `SafePosAux` parameter.

- `SafePosAux` = 0 means that the auxiliary valve closes with `Ctrl` = 0 and opens with `Ctrl` = 1.
- `SafePosAux` = 1 means that the auxiliary valve opens with `Ctrl` = 0 and closes with `Ctrl` = 1 .

The setting is made with the `SafePos` parameter for the control valve.

- `SafePos` = 0 means that the control valve closes in the de-energized state (`MV` is set to `MV_OpScale.Low`)
- `SafePos` = 1 means that the control valve opens in the de-energized state (`MV` is set to `MV_OpScale.High`)
- `SafePos` = 2 means that the control valve retains its position in the de-energized state. (`MV` remains unchanged)

The control valve is brought to the neutral position when the `FbkAuxVCloseOut` = 1 control valve is closed.

Generating instance-specific messages

This block provides the standard function Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
4	Setting switch or button mode (Page 166)
5	Control via auxiliary valve (Page 171)
6	Disabling opening and closing (Page 157)
7	Ramp rate calculation (Page 178)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
12	Gradient limitation with time duration (Page 181)
15	Neutral position manipulated variable takes effect with "out of service" operating mode (Page 165)
16	Neutral position manipulated variable takes effect at startup (Page 165)
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)
18	Activating error state for external process control error CSF (Page 150)
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)

Bit	Function
27	Interlock display with LocalSetting 2 or 4 (Page 177)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

With auxiliary valve (Feature bit 5 = 1):

In this case the automatic commands `OpenAut` and `CloseAut` control the auxiliary valve

In pushbutton operation (Bit 4 = 0) the automatic commands in automatic mode are latching, in other words `OpenAut`, and `CloseAut`, can be reset to 0 after changing the control. In manual and local modes, however, the automatic commands are not saved and in the absence of automatic commands the automatic control is tracked.

In switching mode (Bit 4 = 1), control is selected with the static signals `OpenAut`. If input `OpenAut` is not set the auxiliary valve is closed. Control via `CloseAut` is not needed. If the "Activate command reset for control settings" function (Bit 3 = 1) is activated, the `OpenAut` input is reset to the neutral position after evaluation in the block.

Without auxiliary valve (Feature bit 5 = 0):

The analog valve is opened or closed in automatic mode with the automatic commands `OpenAut` and `CloseAut`. If `OpenAut` and CloseAut are reset, the output MV is set to `MV_Ext`. For "Activate command reset for control settings" (Bit 3 = 1), `OpenAut` and CloseAut are reset and the control elements are self-locking. This means the output `MV` is only set to `MV_Ext` after a change of `MV_Ext`. Control via switch or button mode (bit 4) does not affect the behavior of `OpenAut` and `CloseAut`.

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
5	Evaluation of the signal status of the interlock signals (Page 141)
10	Considering bad quality of automatic commands or external values (Page 185)

Displaying auxiliary values

This block provides the standard function Displaying auxiliary values (Page 207).

Alarm delays with two time values per limit pair

This block has the standard alarm delay function Two time values per limit pair (Page 197) for limit monitoring of the feedback and limit monitoring of the manipulated variable difference.

The function here relates solely to the limits of the manipulated variable difference.

Digital feedback from the readback value

This block forms a digital position feedback for "Closed" and "Open" from the configured operating points of the position feedback values `PosDiClose` and `PosDiOpen`.

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can switch to "automatic mode"
1	1 = Operator can switch to "manual mode"
2	1 = Operator can switch to "local mode"
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can open the valve
5	1 = Operator can close the valve
6	1 = Operator can reset the valve
7	1 = Operator can define the monitoring time for startup
8	1 = Operator can define the monitoring time for the end position
9	1 = Operator can enable the monitoring the control valve feedback function (Bit 7 - 8)
10	1 = Operator can activate the Simulation function
11	1 = Operator can activate the Release for maintenance function
12	1 = Operator can change the simulation value `SimRbk`
13 - 23	Not used
24	1 = Operator can define the monitoring time of the auxiliary valve for startup
25	Not used
26	1 = Operator can enable the monitoring the auxiliary valve feedback function (Bit 24)
27 - 31	Not used

The block has the following operator permissions for the `OS1Perm` parameter:

Bit	Function
0 - 3	Not used
4	1 = Operator can switch the manipulated variable to "external" `MV_ExtOp`
5	1 = Operator can switch the manipulated variable to "internal" `MV_IntOp`
6	Not used
7	1 = Operator can change the manual manipulated variable `MV_Int`
8 - 9	Not used
10	1 = Operator can change the operation high limit of the manipulated variable `MV_HiLim`
11	1 = Operator can change the operation low limit of the manipulated variable `MV_LoLim`
12	1 = Operator can enable the manipulated variable's gradient limitation function `MV_RateOn`
13	1 = Operator can change the manipulated variable's high limit for the ramp `MV_UpRaLim`
14	1 = Operator can change the manipulated variable's low limit for the ramp `MV_DnRaLim`

Bit	Function
15	1 = Operator can switch between the time value or the value for the ramp (MV_RmpModTime)
16	1 = Operator can change the ramp time MV_RmpTime
17	1 = Operator can change the target manipulated variable MV_RmpTarget for the manipulated variable ramp
18	1 = Operator can enable the manipulated variable ramp function MV_RmpOn
19	Not used
20	1 = Operator can enable the track manipulated variable function in tracking mode MV_TrkRbk
21	1 = Operator can enable the bumpless switchover from external to internal MV_TrkExt
22	1 = Operator can activate / deactivate messages via ER_AH_MsgEn
23	1 = Operator can activate / deactivate messages via ER_AL_MsgEn
24	1 = Operator can activate / deactivate messages via RbkWH_MsgEn
25	1 = Operator can activate / deactivate messages via RbkWL_MsgEn
26	1 = Operator can change the limit (manipulated variable difference) for the high alarm ER_AH_Lim
27	1 = Operator can change the hysteresis (manipulated variable difference) ER_Hyst
28	1 = Operator can change the limit (manipulated variable difference) for the low alarm ER_AL_Lim
29	1 = Operator can change the limit (position feedback) for the high warning RbkWH_Lim
30	1 = Operator can change the hysteresis (position feedback) RbkHyst
31	1 = Operator can change the hysteresis (position feedback) for low warning RbkWL_Lim

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

Generation of manipulated variables

The manipulated variable MV is formed as follows:

MV_ForOn	CloseForce	OpenForce	ManAct	AutAct	LocalAct	MV_TrkOn	MV_TrkRbk	MV_ExtAct	MV =	Limit	State
1	0	0	-	-	-	-	-	-	MV_Forced	none	Forced tracking without limit
-	1	0	-	-	-	-	-	-	MV_OpScale.Low	MV_OpScale.Low	Forced close
-	0	1	-	-	-	-	-	-	MV_OpScale.High	MV_OpScale.High	Forced open
0	0	0	1	0	0	0	-	0	MV_Int	MV_HiLim MV_LoLim	Manual mode with internal manipulated variable

MV_ For On	Close Force	OpenF orce	Man Act	Aut Act	Local Act	MV_ Trk On	MV_ Trk Rbk	MV_ Ext Act	MV =	Limit	State
0	0	0	1	0	0	0	–	1	MV_ExtOut	MV_HiLim MV_LoLim	Manual mode with external manipulated variable and limit
0	0	0	0	1	0	0	–	1	MV_Int	MV_HiLim MV_LoLim	Automatic with external manipulated variable and limit
0	0	0	0	1	0	0	–	1	MV_ExtOut	MV_HiLim MV_LoLim	Automatic with external manipulated variable and limit
0	0	0	–	–	0	1	0	–	MV_Trk	MV_HiLim MV_LoLim	Tracking with limitation
0	0	0	0	–	0	1	1	–	Rbk	none	Tracking to position feedback without limit
0	0	0	0	0	1	–	–	–	Rbk	None	Local mode with tracking to position feedback without limit

"Actuator active" information

- The following applies for `PosReached.Value` = 0:
  With `PosDiClose` < `MV` < `PosDiOpen`, the control valve is detected as active and Bit 24 is set in `Status1`.
- The following applies for `PosReached.Value` = 1:
  `Status1.Bit 24` = 0

This status can be used to indicate a customized symbol in the process image, for example, and is saved in the status word (see Status word section in Description of VlvAnL (Page 1431)).

General function "MV difference"

The manipulated variable difference is sent to the `ER` output and is calculated with the following formula:

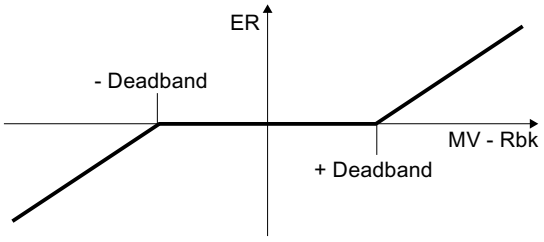
`ER` = `MV` - `Rbk`

If `ER` is within the dead band, `ER` = 0 is set and the manipulated variable is considered reached.

Manipulated variable difference generation and dead band

The manipulated variable difference is formed by the effective manipulated variable `MV` and the position feedback `Rbk`, and sent to the `ER` output. A dead band can be set at the `PosDeadBand` input:

- `PosDeadBand` = 0 Dead band is disabled
- `PosDeadBand` ≠ 0 Dead band is enabled



Limit monitoring of manipulated variable and error signal

The block provides the standard function Limit monitoring of setpoint, manipulated variable and control deviation (Page 95)

Monitoring is disabled in the following situations:

- The auxiliary valve is closed
- The control valve is in the neutral position

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Specifying warning times for control functions at motors and valves

This block provides the standard function Specifying warning times for control functions at motors and valves (Page 51)

The warning time affect the analog manipulated variable `MV`. The output is updated only with the specification of a new manipulated variable after the warning time has expired.

Warning time is ignored in tracking `MV_TrkOn` = 1 and in forced tracking `MV_ForOn`.

Disabling feedback

This block provides the standard function Disabling feedback for valves (Page 99).

This function is available for both the control valve and auxiliary valve. The feedback for the auxiliary valve is disabled by default, connections are not displayed.

The disable setting is made with the `NoFbkOpen` and `NoFbkClose` parameters for the control valve.

The disable setting is made with the `NoFbkAuxVOpen` and `NoFbkAuxVClose` parameters for the auxiliary valve.

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

Button labels

This block provides the standard function Labeling of buttons and text (Page 205)

Instance-specific text can be configured for the following parameters:

- `OpenMan`
- `CloseMan`

See also

EventTs functions (Page 1634)

VlvAnL modes (Page 1435)

VlvAnL messaging (Page 1452)

VlvAnL I/Os (Page 1454)

VlvAnL block diagram (Page 1466)

Disable calculation of impulse control in LocalSetting 2 and 4 (Page 185)

7.13.4 VlvAnL error handling

Error handling of VlvAnL

Refer to section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Invalid input signals
- Control system fault (CSF)

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
41	The value of the `LocalSetting` connection is outside the valid range. Valid values are 0, 2 and 4
42	`LocalSetting` = 0 or `LocalSetting` = 4 and `LocalLi` = 1
51	`AutModLi` = 1 and `ManModLi` = 1 `OpenAut` = 1 and `CloseAut` = 1 `OpenForce` = 1 and `CloseForce` = 1
52	`LocalSetting` = 2 or 4 and `SimOn` = 1

Mode switchover error

This error can be output by the block, see the section Error handling (Page 119).

Invalid input signals

This error can be output by the block, see the section Error handling (Page 119).

For the following invalid input signals, the control output can be kept or switched to the neutral position. This depends on the function control priority for invalid input command (Page 174).

Operating mode	Invalid input signals	Control reaction with Feature2 bit 3 =1
Forcing	`OpenForce` = 1 and `CloseForce` = 1	Valve without auxiliary valve (`Feature` bit 5 = 0): Valve is set to its neutral position. Valve with auxiliary valve (`Feature` bit 5 = 1): Valve is set to its neutral position.
"Automatic mode" and no forcing	Pushbutton operation (`Feature` bit 4 = 0): `OpenAut` = 1 and `CloseAut` = 1	
"Manual mode" and no forcing	`OpenMan` = 1 and `CloseMan` = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. A control system fault is triggered if this signal changes to 1. Refer to the Error handling (Page 119) section for more on this.

See also

VlvAnL messaging (Page 1452)

Description of VlvAnL (Page 1431)

VlvAnL modes (Page 1435)
VlvAnL functions (Page 1437)
VlvAnL I/Os (Page 1454)
VlvAnL block diagram (Page 1466)

7.13.5 VlvAnL messaging

Messaging

The following messages can be generated for this block:

- Process control fault
- Instance-specific messages
- Process messages

Process control fault

- The following control system error messages can be output:

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	AS process control message - fault	$$BlockComment$$ Feedback error
	SIG 6	AS process control message - fault	External error has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to input parameter CSF. If it changes to CSF = 1, a process control fault is triggered (MsgEvId1, SIG 6).

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 2	Alarm - high	$$BlockComment$$ ER - high alarm limit violated
	SIG 3	Alarm - low	$$BlockComment$$ ER - low alarm limit violated
	SIG 4	Warning - high	$$BlockComment$$ Rbk - high warning limit violated
	SIG 5	Warning - low	$$BlockComment$$ Rbk - low warning limit violated

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Instance-specific messages

You have the option to use one or two instance-specific messages for this block.

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 7	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 8	AS process control message - fault	$$BlockComment$$ External message 2

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`ExtVal04`
5	`ExtVal05`
6	`ExtVal06`
7	`ExtVa107`
8	`ExtVa108`
9	Reserved
10	Reserved

The associated values 4 ... 8 are allocated to the parameters `ExtVa104` ... `ExtVa108` and can be used. Additional information is available in the "Process Control System PCS 7 - Engineering System" manual.

See also

Description of VlvAnL (Page 1431)

VlvAnL functions (Page 1437)

VlvAnL I/Os (Page 1454)

VlvAnL modes (Page 1435)

VlvAnL error handling (Page 1450)

VlvAnL block diagram (Page 1466)

7.13.6 VlvAnL I/Os

I/Os of VlvAnL

Input parameter master

Parameter	Description	Type	Default
AutModLi*	1= "Automatic mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled by ModLiOp = 0)	BOOL	0
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypProt	1 = Bypassing interlock is active in "local mode" and in simulation	BOOL	0
CloseAut*	1 = Select Close valve in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseForce	1 = Force valve closure	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseMan*	1 = Select Close valve in "manual mode"	BOOL	0
CSF	1 = External error (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CtrlChnST	Signal status of output channel for Ctrl Should be connected to an output channel block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
EN	1 = Called block will be processed	BOOL	1
ER_A_DC*	Delay for incoming alarms during manipulated variable difference monitoring	REAL	0.0
ER_A_DG*	Delay for outgoing alarms during manipulated variable difference monitoring	REAL	0.0
ER_AH_En	1 = Activate alarm (high) for manipulated variable difference monitoring	BOOL	1
ER_AH_Lim	Alarm limit (high) for manipulated variable difference monitoring	REAL	100.0
ER_AH_MsgEn	1 = Activate messages for alarm (high) for manipulated variable difference monitoring	BOOL	1
ER_AL_Lim	Alarm limit (low) for manipulated variable difference monitoring	REAL	-100.0

Parameter	Description	Type	Default
ER_AL_En	1 = Activate alarm (low) for manipulated variable difference monitoring	BOOL	1
ER_AL_MsgEn	1 = Activate messages for alarm (low) for manipulated variable difference monitoring	BOOL	1
ER_Hyst	Alarm hysteresis for manipulated variable difference monitoring	REAL	1.0
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTsIn input parameter serves to interconnect the EventTsOut output parameter of the EventTs, Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvID1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvID1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvID1)	ANY	
ExtVa107	Associated value 7 for messages (MsgEvID1)	ANY	
ExtVa108	Associated value 8 for messages (MsgEvID1)	ANY	
FaultExt	1 = External error Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkAuxVClose	1 = Auxiliary valve closed feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkAuxVOpen	1 = Auxiliary valve open feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF

Parameter	Description	Type	Default
FbkClose	1 = Valve closed feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkOpen	1 = Valve open feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
Feature	I/O for additional functions (Page 1437)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Feature2	I/O for additional functions (Page 1437)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Intl_En	1 = Interlock without reset (interlock, Intlock parameter) is active	BOOL	1
Intlock	0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared 1 = Interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
LocalLi	1 = Activate "local mode" via plant signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalOp*	1 = "Local mode" via operator	BOOL	0
LocalSetting	Properties for the Local mode (Page 79)	INT	0
ManModLi*	1 = "Manual mode" via interconnection or SFC (controlled by ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManModOp*	1= "Manual mode" via: OS operator (controlled via ModLiOp = 0)	BOOL	1
ModLiOp	Switchover of operating mode between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonAuxVTime*	Monitoring time for feedback monitoring of the auxiliary valve	REAL	3.0
Monitor	1 = Monitoring of control valve feedback	BOOL	1
MonitorAuxV	1 = Monitoring of auxiliary valve feedback	BOOL	1
MonSafePos	1 = Go to neutral position in the event of monitoring errors	BOOL	1
MonTiDynamic*	Monitoring time after operation in [s]	REAL	3.0
MonTiStatic*	Monitoring time for feedback errors without operation in [s]	REAL	3.0

Parameter	Description	Type	Default
MS_RelOp*	1= Release for maintenance via OS operator	BOOL	0
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_ChnST	Signal status of output channel for MV Should be connected to an output channel block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
MV_DnRaLim	Gradient limit (low) for manipulated variable MV_Unit	REAL	100.0
MV_Ext	External manipulated variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_ExtLi	Select external manipulated variable (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_ExtOp*	Select external manipulated variable (via operator)	BOOL	0
MV_Forced	Forced manipulated variable that is not limited and assumes top priority	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_ForOn	1 = Output forced manipulated variable MV_Forced unlimited at output MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_HiLim	Limit (high) for manipulated variable MV	REAL	100.0
MV_Int*	Internal manipulated variable	REAL	0.0
MV_IntLi	Select internal manipulated variable (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_IntOp*	Select internal manipulated variable (via operator)	BOOL	1
MV_LiOp	Select manipulated variable source (internal/external): 1 = Via interconnection 0 = Via operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_LoLim	Limit (low) for manipulated variable MV	REAL	0.0
MV_OpScale	OS display range for manipulated variable MV	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

Parameter	Description	Type	Default
MV_RateOn*	Change of manipulated variable is limited by MV_UpRaLim and MV_DnRaLim	BOOL	0
MV_RmpModTime	1 = Use time (MV_RmpTime) for manipulated variable ramp 0 = Use gradient	BOOL	0
MV_RmpOn*	1 = Activate manipulated variable ramp to target value MV_RmpTarget	BOOL	0
MV_RmpTarget*	Time value for manipulated variable ramp	REAL	0.0
MV_RmpTime*	Time for manipulated variable ramp [s] from current MV up to MV_RmpTarget	REAL	0.0
MV_Trk	Tracking value for the manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_TrkExt	1 = Bumpless switchover from external to internal manipulated variable active	BOOL	0
MV_TrkOn	1 = Tracking of manipulated variable MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_TrkRbk	1 = Bumpless switchover of manipulated variable tracking active (track manipulated variable to position feedback)	BOOL	0
MV_Unit	Unit of measure for manipulated variable	INT	1342
MV_UpRaLim	Gradient limit (high) for manipulated variable MV_Unit	REAL	100.0
NoFbkAuxVClose	1 = No "closed" feedback for auxiliary valve present	BOOL	1
NoFbkAuxVOpen	1 = No "open" feedback for auxiliary valve present	BOOL	1
NoFbkClose	1 = No feedback present for "control valve closed"	BOOL	0
NoFbkOpen	1 = No feedback present for "control valve open"	BOOL	0
Occupied	1 = In use by a batch	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpenAut*	1 = Select Open valve in "automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenForce	1 = Forced open of control valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
OpenMan*	1 = Select Open valve in "manual mode"	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for Operator control permissions (Page 248)	STRUCT • Bit 0: BOOL • Bit 12: BOOL • Bit 31: BOOL	- • 1 • 1 • 1
OS1Perm	I/O for Operator control permissions (Page 248)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
Perm_En	1 = Activation enable (enable, Permit parameter) is active	BOOL	1
Permit	1 = Enable for opening / closing from neutral position 0 = No activation enable for the valve	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
PosDeadBand	Dead band for forming manipulated variable difference	REAL	0.1
PosDiClose	Limit for control valve position "closed"	REAL	5.0
PosDiOpen	Limit for control valve position "open"	REAL	95.0
Prot_En	1 = Protective interlock (protection, Protect parameter) is active	BOOL	1
Protect	0 = Protective interlocking is effective; once the interlocking condition has disappeared, you will have to reset the block 1 = Protective interlocking not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Rbk	Position feedback for display on OS	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RbkHyst	Alarm hysteresis for position feedback	REAL	1.0
RbkW_DC*	Delay time for incoming warnings [s]	REAL	0.0
RbkW_DG*	Delay time for outgoing warnings [s]	REAL	0.0
RbkWH_En	1 = Enable warning (high) for position feedback	BOOL	0
RbkWH_Lim	Limit for position feedback of warning (high)	REAL	90.0
RbkWH_MsgEn	1 = Enable messages for warning (high) for position feedback	BOOL	1
RbkWL_En	1 = Enable warning (low) for position feedback	BOOL	0
RbkWL_Lim	Limit for position feedback of warning (low)	REAL	10.0

Parameter	Description	Type	Default
RbkWL_MsgEn	1 = Enable messages for warning (low) for position feedback	BOOL	1
RstLi*	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SafePos	Neutral position for control valve: 0 = Closed 1 = Open 2 = Stop	INT	0
SafePosAux	Neutral position for auxiliary valve: 1= Open 0 = Closed	BOOL	0
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimRbk*	Position feedback used for SimOn = 1	REAL	0.0
SimRbkLi	Position feedback used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
StepNo	Batch step number	DWORD	16#00000000
TimeFactor	Time unit: 0 = Seconds 1 = Minutes 2 = Hours	INT	0
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UA2unit	Unit of measure for analog auxiliary value 2	INT	0

Parameter	Description	Type	Default
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
WarnTiAut	Prewarning of valve movement in automatic mode in [s]	REAL	0.0
WarnTiMan	Prewarning of valve movement in manual mode in [s]	REAL	0.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode", "Local mode" or "Out of service" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80
AuxClosed	1 = Auxiliary valve is closed	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AuxClsing	1 = Auxiliary valve is closing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AuxOpened	1 = Valve is open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AuxOpning	1 = Auxiliary valve is opening	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CascaCut	Cascade connection: 1 = Control chain from master controller to secondary valve is interrupted	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Closed	1 = Valve is closed	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
`Closing`	1 = Valve is closing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`Ctrl`	Control output for auxiliary valve (depends on `SafePosAuxV`)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`CurrMon`	Current monitoring time [s]	DINT	0
`CurrMonAuxV`	Current auxiliary valve monitoring time [s]	DINT	0
`ENO`	1 = Block algorithm completed without errors	BOOL	0
`ER`	Manipulated variable difference	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`ER_AH_Act`	1 = Alarm limit (high) for manipulated variable difference violated. You can change the reaction for this parameter with `Feature` bit 28 (Disabling operating points (Page 144)) and with `Feature` bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ER_AL_Act`	1 = Alarm limit (low) for manipulated variable difference violated. You can change the reaction for this parameter with `Feature` bit 28 (Disabling operating points (Page 144)) and with `Feature` bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ErrorNum`	Output of pending error number. For error numbers that can be output by this block, see VlvAnL error handling (Page 1450)	INT	-1
`FbkAuxVCloseOut`	1 = Auxiliary valve is closed	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`FbkAuxVOpenOut`	1 = Auxiliary valve open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`FbkCloseOut`	1 = Control valve is open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`FbkClsgOut`	Control valve closing feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
FbkOpenOut	1 = Control valve is open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpngOut	Control valve opening feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalAct	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock (Intlock, Permit or Protect) is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
MonDynAuxVErr	1 = Feedback error auxiliary valve due to output change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonDynErr	1 = Feedback error control valve due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonStaAuxVErr	1 = Feedback error from auxiliary valve due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MonStaErr	1 = Feedback error from control valve due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message acknowledgement status 1 (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr1	Alarm error 1 (output ERROR of first ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (output STATUS of first ALARM_8P)	WORD	16#0000
MV	Manipulated variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
MV_DnRaAct	Positive limit (low) for manipulated variable is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_ExtAct	1 = External manipulated variable active 0 = Internal manipulated variable active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_ExtOut	Output for external manipulated variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_HiAct	1 = Limit (high) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_LoAct	1 = Limit (low) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_RateTarget	Target manipulated variable for the gradient limitation	REAL	0.0
MV_UnitOut	Unit of measure for manipulated variable	INT	0
MV_UpRaAct	Positive limit (high) for manipulated variable is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_RemRT	Remaining ramp time of the manipulated variable	REAL	0.0
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Opened	1 = Valve is open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Opening	1 = Valve is opening	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS1PermLog	Display of OS1Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS1PermOut	Display of OS1Perm	DWORD	16#FFFFFFFF

Parameter	Description	Type	Default
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PosReached	1 = Control valve has reached specified position	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkOut	Position feedback output	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RbkWH_Act	1 = Warning (high) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkWL_Act	1 = Warning (low) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Active start readiness	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1431)	DWORD	16#00000000
Status2	Status word 2 (Page 1431)	DWORD	16#00000000
Status3	Status word 3 (Page 1431)	DWORD	16#00000000
Status4	Status word 3 (Page 1431)	DWORD	16#00000000
WarnAct	1 = Prewarning for control valve movement away from neutral position active (parameters WarnTiAut and WarnTiMan))	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

"local", "automatic", or "manual" mode"

See also

VlvAnL modes (Page 1435)

VlvAnL messaging (Page 1452)

VlvAnL block diagram (Page 1466)

Error handling (Page 119)

7.13.7 VlvAnL block diagram

VlvAnL block diagram

A block diagram is not provided for this block.

See also

Description of VlvAnL (Page 1431)

VlvAnL modes (Page 1435)

VlvAnL functions (Page 1437)

VlvAnL error handling (Page 1450)

VlvAnL messaging (Page 1452)

VlvAnL I/Os (Page 1454)

7.13.8 Operator control and monitoring

7.13.8.1 VlvAnL views

Views of the VlvAnL block

The VlvAnL block provides the following views:

- VlvAnL standard view with auxiliary valve (Page 1467)
- VlvAnL standard view without auxiliary valve (Page 1472)
- VlvAnL limit view (Page 1476)
- Alarm view (Page 296)
- Trend view (Page 299)
- VlvAnL parameter view (Page 1483)
- VlvAnL preview (Page 1479)
- Memo view (Page 298)
- Batch view (Page 296)
- Ramp view (Page 294)
- Block icon for VlvAnL (Page 1485)

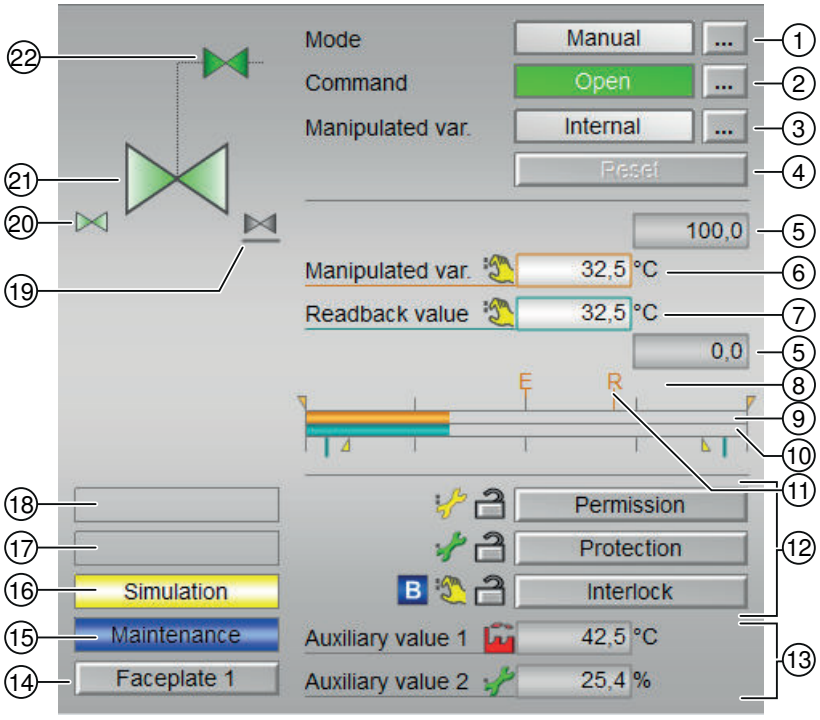
Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

See also

Parameter view for motors and valves (Page 280)

7.13.8.2 VlvAnL standard view with auxiliary valve

Standard view with auxiliary valve of VlvAnL



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Opening, closing and stopping the control valve

This area shows you the default operating state for the valve. The following states can be shown and executed here:

- "Open"
- "Close"
- "Stop" (display only, no operation possible)

"Close"/"open" command relates to the auxiliary valve. If no auxiliary valve is required for controlling the control valve, the "Open" and "Close" commands affect the control valve.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

(3) Displaying and switching the default manipulated variable

This area shows how to specify the manipulated variable. The manipulated variable can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the setpoint specification.

(4) Resetting the block

Click "Reset" for interlocks or errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(5) High and low scale range for the manipulated variable

These values provide information on the display range for the bar graph of the manipulated variable. The scale range is defined in the engineering system.

(6) Displaying and changing the manipulated variable including signal status:

This area shows the current manipulated variable with the corresponding signal status.

The manipulated variable can only be changed by

- Internal manipulated variable specification (`MV_ExtAct` = 0) and
- An opened auxiliary valve (due to the dependency on the standard function Neutral position for motors, valves and controllers (Page 48))

For more information on changing the manipulated variable, refer to section Changing values (Page 253).

(7) Display of the position feedback including signal status

This area shows the current feedback of the manipulated variable with the corresponding signal status.

(8) Display for the target manipulated variable of the manipulated variable ramp

This display [R] shows the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

(9) Bar graph for the manipulated variable

This area shows the current manipulated variable in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(10) Bar graph for position feedback

This area shows the current position feedback in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

The limits for the "open" and "close" positions are shown with 2 green lines.

(11) Display of external manipulated variable

This display [E] is only visible when you have selected "Internal" manipulated variable specification. It shows the external manipulated variable that would apply if you were to change the manipulated variable specification to "External".

(12) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the section Interlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

  If there is a bypass, it is displayed instead of the signal status.

(13) Display of auxiliary values

This display is only visible when the corresponding block input is connected.

You can use this area to display two auxiliary values that have been configured in the engineering system (ES). You can find additional information on this in the section Displaying auxiliary values (Page 207).

(14) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

See also the section Opening additional faceplates (Page 203) for more on this.

(15) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

(16) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in the section Simulating signals (Page 58).

(17) Display area for block states

This area provides additional information on the operating state of the block:

- "External error"
- "End position error"
- "Control error"
- "Invalid signal"
- "Changeover error"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97), Error handling (Page 119) (section "Invalid input signals" and "Mode changeover error").

(18) Display area for block states

This area provides additional information on the operating state of the block:

- "Forced open" (`OpenForce`)
- "Forced close" (`CloseForce`)
- "Forced tracking" (`MV_ForOn`)
- "Tracking" (`MV_TrkOn`)
- "Request 0/1": A reset to "automatic mode" is expected.
- "MV ramp active"

You can find additional information on this in the section Forcing operating modes (Page 41).

(19) Representation of neutral position

This representation shows the neutral position for the control valve:

- Green: Neutral position is "Open"
- Gray: Neutral position is "Closed"
- Light green: Neutral position is "Stop"

(20) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the valve would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

(21) Status display of the control valve

You can find more information about this in section Block icon for VlvAnL (Page 1485)

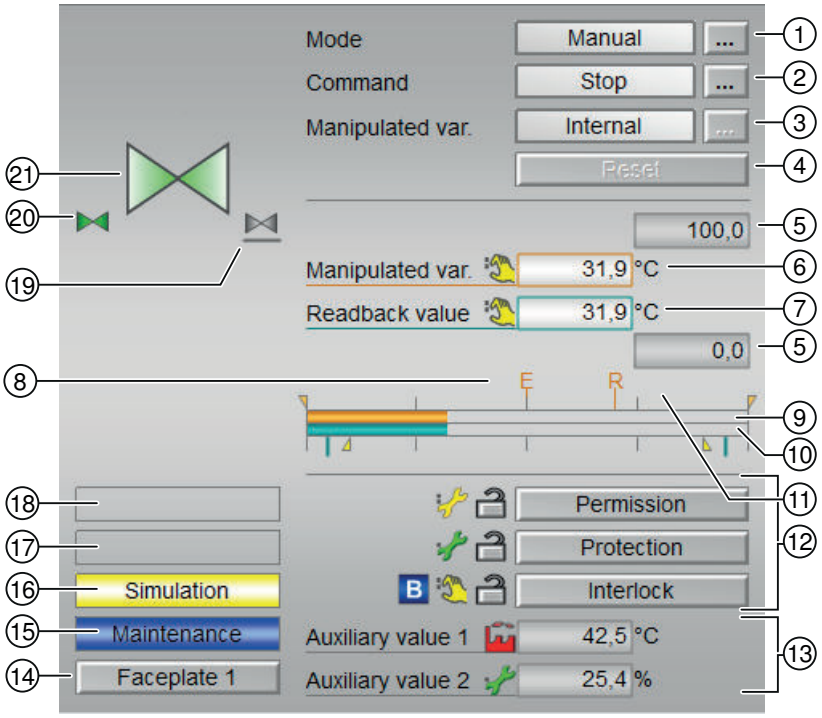
(22) Picture of auxiliary valve

The small auxiliary valve and the associated line are only shown when the control valve has an additional auxiliary valve and it can be controlled. This can be configured with `Feature` Bit 5 on the block.

The current status of the auxiliary valve is graphically displayed here.

7.13.8.3 VlvAnL standard view without auxiliary valve

Standard view without auxiliary valve of VlvAnL



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Opening, closing and stopping the control valve

This area shows you the default operating state for the valve. The following states can be shown and executed here:

- "Open"
- "Close"
- "Stop" (display only, no operation possible)

"Close"/"Open" command relates to the control valve.

The "Open" command sets the manipulated variable equal to the high limit of the control range (`MV_OpScale.High`).

The "Close" command sets the manipulated variable equal to the low limit of the control range (`MV_OpScale.Low`)

If there is no auxiliary valve (`Feature` Bit 5 = 0), the command line can be completely hidden with `Feature` Bit 6.

- 0: (default) line is displayed
- 1: Command line hidden.

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information about this in section Labeling of buttons and text (Page 205)

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

(3) Displaying and switching the default manipulated variable

This area shows how to specify the manipulated variable. The manipulated variable can be specified as follows:

- By the application ("External", CFC/SFC)
- By the user directly in the faceplate ("Internal").

If there is no auxiliary valve, the specification of the manipulated variable depends on the mode, "manual" or "automatic". You cannot switch between "internal" and "external" here in this case.

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the setpoint specification.

(4) Resetting the block

Click "Reset" for interlocks or errors. You can find additional information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(5) High and low scale range for the manipulated variable

These values provide information on the display range for the bar graph of the manipulated variable. The scale range is defined in the engineering system.

(6) Displaying and changing the manipulated variable including signal status:

This area shows the current manipulated variable with the corresponding signal status.

The manipulated variable can only be changed by internal manipulated variable specification (`MV_ExtAct` = 0).

For more information on changing the manipulated variable, refer to section Changing values (Page 253).

(7) Display of the position feedback including signal status

This area shows the current feedback of the manipulated variable with the corresponding signal status.

(8) Display for the target manipulated variable of the manipulated variable ramp

This display [R] shows the target setpoint and is only visible if you have enabled ramp generation in the Ramp view (Page 294).

(9) Bar graph for the manipulated variable

This area shows the current manipulated variable in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(10) Bar graph for position feedback

This area shows the current position feedback in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

The limits for the "open" and "close" positions are shown with 2 green lines.

(11) Display of external manipulated variable

This display [E] is only visible when you have selected "Internal" manipulated variable specification. It shows the external manipulated variable that would apply if you were to change the manipulated variable specification to "External".

(12) Operating range for the interlock functions of the block

This display is only visible when the corresponding block input is connected.

You can use this button to control the interlock functions of the block. You can find additional information on this in the section Interlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), e.g.:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), e.g.:

- Bypass information (see Forming the group status for interlock information (Page 104)):

  If there is a bypass, it is displayed instead of the signal status.

(13) Display of auxiliary values

This display is only visible when the corresponding block input is connected.

You can use this area to display two auxiliary values that have been configured in the engineering system (ES). You can find additional information on this in the section Displaying auxiliary values (Page 207).

(14) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

See also the section Opening additional faceplates (Page 203) for more on this.

(15) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

Additional information on this is available in section Release for maintenance (Page 64) Display area for block states.

(16) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"
- "Delay"

You will find more detailed information on this in the sections Simulating signals (Page 58) and Display of delay times (Page 250).

(17) Display area for block states

This area provides additional information on the operating state of the block:

- "External error"
- "End position error"
- "Control error"
- "Invalid signal"
- "Changeover error"

Additional information on these errors is available in the sections Monitoring the feedbacks (Page 97), Error handling (Page 119) (section "Invalid input signals" and "Mode changeover error").

(18) Display area for block states

This area provides additional information on the operating state of the block:

- "Forced open" (`OpenForce`)
- "Forced close" (`CloseForce`)
- "Forced tracking" (`MV_ForOn`)
- "Tracking" (`MV_TrkOn`)
- "Request 0/1": A reset to automatic mode is expected.
- "MV ramp active"

You can find additional information on this in the section Forcing operating modes (Page 41).

(19) Representation of neutral position

This representation shows the neutral position for the control valve:

- Green: Neutral position is "Open"
- Gray: Neutral position is "Closed"
- Light green: Neutral position is "Stop"

(20) Automatic preview

This display is only visible in "manual mode", in "local mode", or with a reset request in "automatic mode", when the current output signals are not identical to the control in "automatic mode".

The display shows what state the valve would assume if you switched from "manual" or "local" mode to "automatic mode", or performed a reset to "automatic mode".

(21) Picture of control valve

The current status of the control valve is graphically displayed here.

You can find more information about this in section Block icon for VlvAnL (Page 1485)

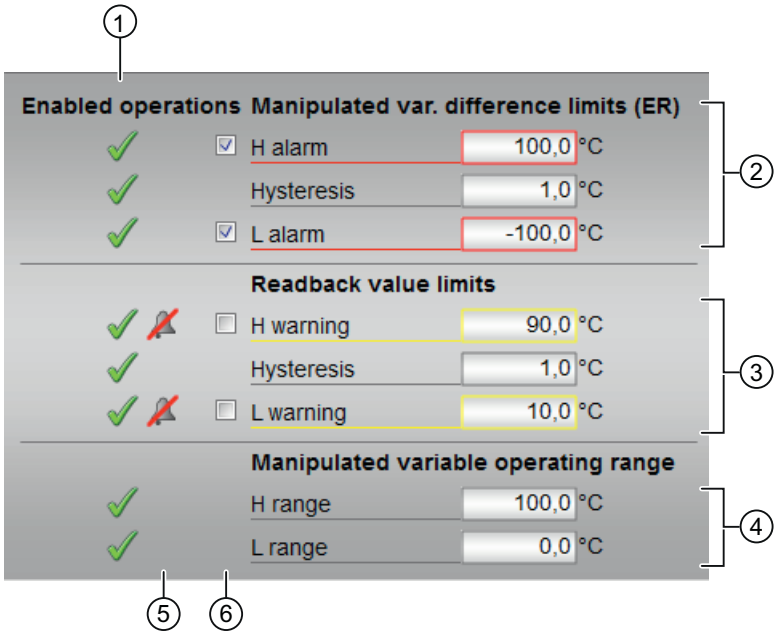
7.13.8.4 VlvAnL limit view

Limit view of VlvAnL

Several values are set in this view by default:

- Manipulated variable difference limits
- Readback value limits
- Manipulated variable operating range

The toolbars of the faceplate and the block icon indicate when the limits are reached or violated.



(1) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

(2) Manipulated variable difference limits

In this area, you can enter the limits for the manipulated variable error. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "Hysteresis"
- "L alarm": Alarm low

(3) Readback value limits (MV)

In this area, you can enter the limits for the readback value (position feedback). Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H warning": Warning high
- "Hysteresis"
- "L warning": Warning low

(4) Manipulated variable operating range (MV)

In this area, you can enter the limits for the manipulated variable operation range. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H range": Range limit high
- "L range": Range limit low

(5) Message suppression / delay

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

Alarm delays are also displayed in this position; for more on this see section

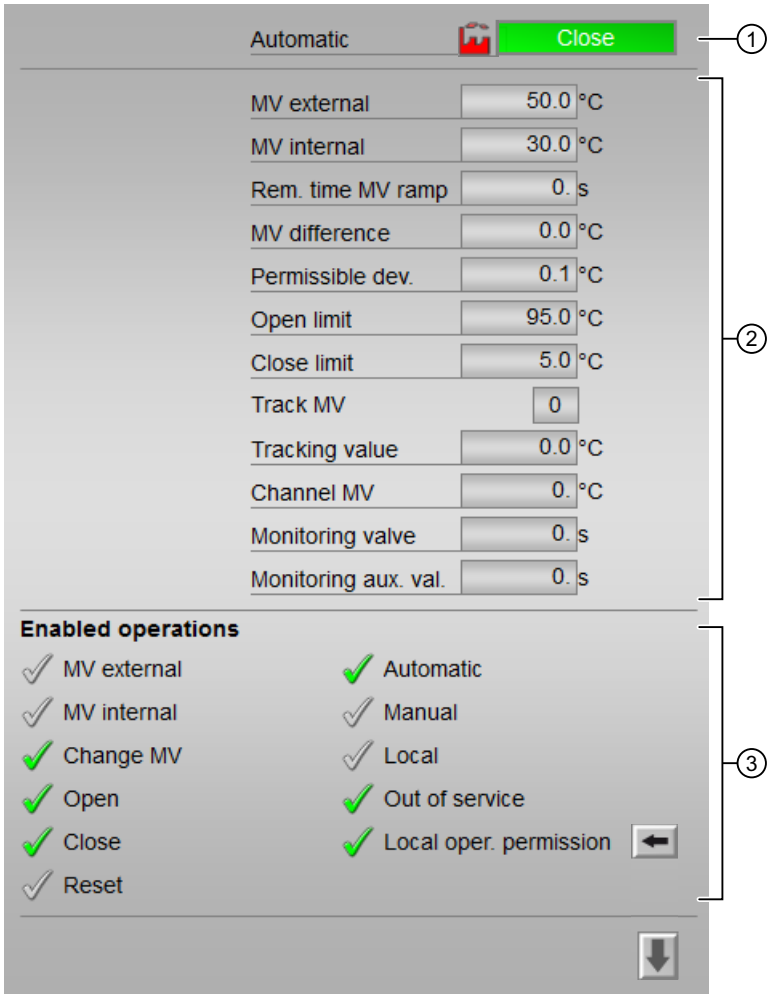
Area of application of the alarm delays (Page 195).

(6) Suppress messages

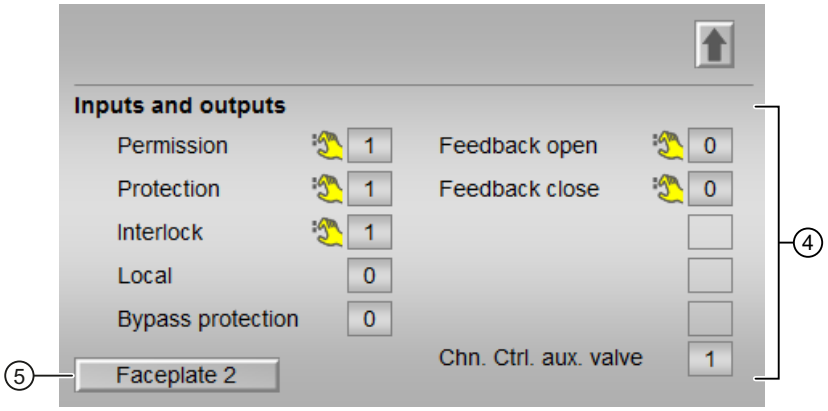
You can enable / disable messages by setting the check mark.

7.13.8.5 VlvAnL preview

Preview of VlvAnL



The preview has an upper half and a lower half. You can change between the two halves with the arrow keys.



(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `OpenAut`
- `CloseAut`
- `MV_Ext`

(2) Preview area

- "Manipulated variable external": Display current external manipulated variable (`MV_ExtOut`).
- "Manipulated variable internal": Display current internal manipulated variable (`MV_Int`).
- "Rem. time MV ramp": Remaining time to reach the ramp target value.
- "Manipulated variable difference": Current manipulated variable error (`ER`)
- "Permissible dev.": Permissible ± deviation (`PosDeadBand`) of manipulated variable that is output. If the manipulated variable feedback `Rbk` is within this range, the manipulated variable is considered reached.
- "Open limit": Limit (`PosDiOpen`) for forming the "Control valve Open" signal (`FbkOpenOut`). If the position feedback reaches this limit, the control valve is open.
- "Close limit": Limit (`PosDiClose`) for forming the "Control valve Closed" signal (`FbkCloseOut`). If the position feedback reaches this limit, the control valve is closed.
- "Manipulated variable tracking": (`MV_TrkOn` = 1) manipulated variable is corrected to the tracking value. Tracking value for the effective manipulated variable for "Track manipulated variable"
- "Channel MV" Display of the manipulated variable by the output channel block
- "Auxiliary valve monitoring" Display of the current monitoring time of the auxiliary valve.
- "Monitoring valve" Display of the current monitoring time of the valve.

(3) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Manipulated variable external": You can enable external manipulated variable specification
- "Manipulated variable internal": You can enable internal manipulated variable specification
- "Change MV": You can change the manipulated variable
- "Open": You can open the valve. The display always relates to the auxiliary valve and control valve. Neither the auxiliary valve nor the control valve can be opened if there is no enable.
  For the control valve this means: If a new manipulated variable is greater than the current valve position, this new manipulated variable take effect.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Close": You can close the valve. The display always relates to the auxiliary valve and control valve. Neither the auxiliary valve nor the control valve can be closed if there is no enable.
  For the control valve this means: If a new manipulated variable is less than the current valve position, this new manipulated variable does not take effect and the control valve retains its position.
  If text is configured for this command, it is also displayed in brackets. You can find additional information on this in the section Labeling of buttons and text (Page 205).
- "Reset": You can reset the valve if interlocks or errors occur.
- "Automatic": You can switch to "automatic mode".
- "Manual": You can switch to "manual mode".
- "Local": You can switch to "local mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the OpStations block. You can find additional information on this in the section Operator control permissions (Page 248) .

(4) Inputs and outputs

This area shows the most important parameters for this block with the current selection.

- "Permission":
  This display is only visible when the corresponding block input is connected.
  - 0 = Valve activation not enabled on OS
  - 1 = Enable for "opening"/"closing" from the neutral position
- "Protection":
  This display is only visible when the corresponding block input is connected.
  - 0 = Protective interlocking is in effect; once the interlocking condition in automatic mode has disappeared, you will have to reset the block
  - 1 = "Good" state
- "Interlock":
  This display is only visible when the corresponding block input is connected.
  - 0 = Interlocking without reset is active; you can operate the block without reset once the interlocking condition has disappeared
  - 1 = "Good" state
- "Local correct": 1 = Control signal for "Local mode" (LocalLi) is active
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypassing interlock in "local mode" and in "simulation"

Since, depending on the configuration, the block itself forms the digital feedback signals internally via the "Open" and "Closed" positions, the following reaction results from the "Feedback open" and "Feedback closed" signals.

- Control valve:
  - "Feedback open": The display is derived from the `FbkOpenOut` output. However, this output is formed from the configured limit for the "Open" position.
  - "Feedback close": The display is derived from the `FbkCloseOut` output. However, this output is formed from the configured limit for the "Closed" position.
- "Control auxiliary valve": only visible when there is a auxiliary valve
  - Control binary control `Ctrl.Out`
  - "Feedback open auxiliary valve": `FbkAuxVOpenOut`.
  - "Feedback closed auxiliary valve": `FbkAuxCloseOut`.
  - "Channel Control auxiliary valve": Signal for controlling the auxiliary valve by the output channel block

(5) Navigation button for switching to the standard view of any faceplate

This display is only visible when the corresponding block input is connected.

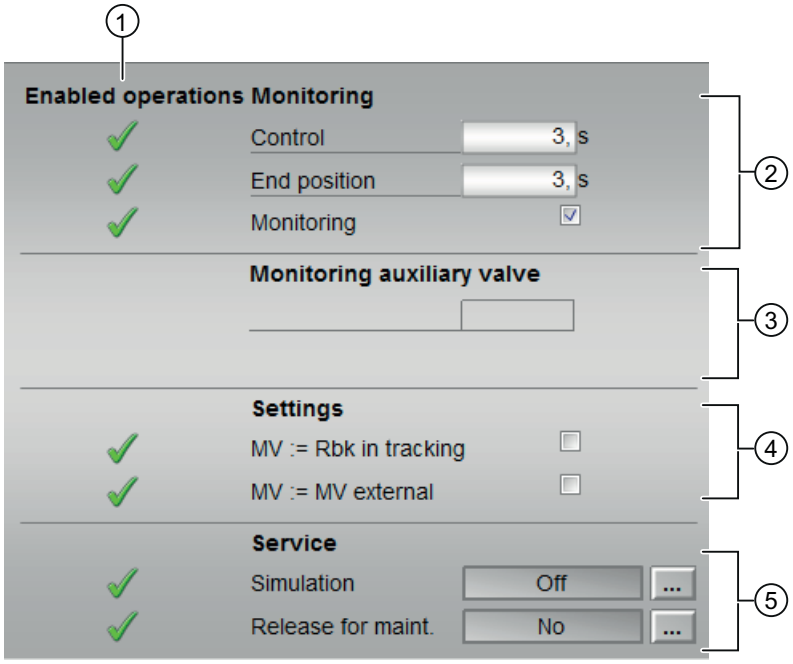
Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203)

.

7.13.8.6 VlvAnL parameter view

Parameter view of VlvAnL



(1) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

(2) Monitoring

In this area, you change parameters and therefore influence the control valve. Refer to the Changing values (Page 253) section for more on this.

You can influence the following parameters:

- "Control": Monitoring time during opening and closing of the control valve (dynamic)
- "End position": Monitoring time of the end position of the control valve (static)

Enable monitoring

You can enable monitoring by selecting the check box (☑)

You can find additional information on this in the section Monitoring the feedbacks (Page 97).

(3) Monitoring auxiliary valve

As under **(2)** Monitoring, but only visible when there is an auxiliary valve. Only one common monitoring time can be entered for dynamic and static monitoring.

(4) Settings

- `MV` = `Rbk` in tracking mode: Correction is performed with the position feedback `Rbk` instead of the `MV_Trk` tracking value. The switchover from "Track manipulated variable" (`MV_TrkOn` = 1) to "Do not track manipulated variable" is bumpless (`MV_TrkOn` = 0). Set `MV_TrkRbk` parameter.
- `MV` = `MV` extern: Bumpless switchover of manipulated variable from external to internal. The internal manipulated variable is tracked to the external one. Set `MV_TrkExt` parameter.

(5) Service

You can select the following functions in this area:

- "Simulation"
- "Release for maintenance" (with display for a maintenance request)

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Simulating signals (Page 58)
- Release for maintenance (Page 64)

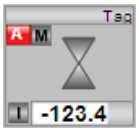
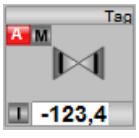
7.13.8.7 Block icon for VlvAnL

Block icons for VLvAnL

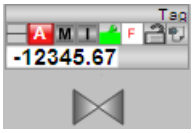
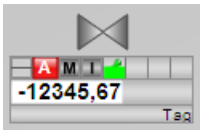
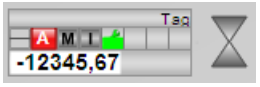
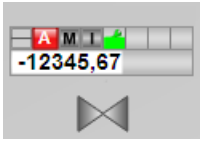
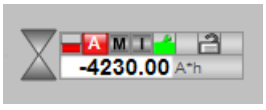
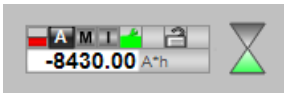
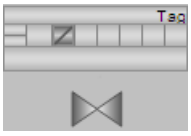
A variety of block icons are available with the following functions:

- Limits (high/low)
- Alarm violation. Warning and tolerance limits as well as control system faults
- Operating modes
- Signal status, release for maintenance
- Forcing states
- Displays for bypassing interlocks
- Interlocks
- Valve status display
- Display of feedback value (white, with decimal places)
- Operation of the manipulated variable
- Process tag type
- External/internal manipulated variable specification
- Memo display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Non-rotating block icon
	4	Non-rotating block icon

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Valve status display

The following valve states are shown here:

Icon	Meaning
	Valve open
	Valve closed
	Error at valve
	Valve is opening
	Valve is closing
	Valve stopped
	Valve closed
	Valve is closing

7.14 VlvPosL - Valve positioner

7.14.1 Description of VlvPosL

Object name (type + number) and family

Type + number: FB 1918

Family: Drives

Area of application for VlvPosL

The block is used for the following application:

- Control of a motor valve with analog position feedback.

How it works

Various operating modes are available for controlling the motorized valve. This enables you to change the valve states individually. All changes of modes or states, as well as errors occurring in this context, are monitored, visualized in the faceplate and reported to the operator. Operators with suitable permissions can use the block icon and the faceplate to view the current state of the motorized valve and operate it.

The motor valve with analog position feedback can be controlled via binary signals such as "Open", "Close", or "Stop" or via analog manipulated variable signals. An internal position controller is activated when analog manipulated variable signals are used. This enables even motor valves with delayed reaction in their position feedback to be position controlled.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Following startup, the messages are suppressed for the number of cycles assigned in the `RunUpCyc` parameter.

After a startup without control (`Open`, `Close` = 0) no monitoring of the feedback signals `FbkOpen` and `FbkClose` takes places during the `V_MonTiStatic` time. Changes to `FbkOpen` and `FbkClose` are applied. This means that the feedback is monitored again even in the stop state.

Status word assignment for `Status1` parameter

You can find a description of the individual parameters in the section VlvPosL I/Os (Page 1515).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`AutoAct.Value`
6	`LocalAct.Value`
7	0 = Open padlock in the block icon 1 = Closed padlock in block icon
8	`Open.Value`
9	Motor is stopped
10	`Close.Value`
11	Torque shutoff enabled (`TorqOpen` or `TorqClose` = 1) When the seal valve function is enabled, the torque shutoff is only active when the `TorqClose` signal is active before the valve feedback.
12	`WarnAct.Value`
13	Feedback error without control change
14	Feedback error due to control change
15	Mode switch fail
16	1 = `Intlock` is active
17	1 = `Permit` is active
18	1 = `Protect` is active
19	Displays forced operating state in the block icon: Force
20	Not used
21	1 = Position control is active
22	1 = Position control is not active
23	"Interlock" button is enabled
24	0 = Displays neutral position "Closed" 1 = Displays neutral position "Open"
25	1 = Displays neutral position "Stop"
26	Bypass information from the previous function block
27	Bypass active (`BypProt` = 1) and `Local.Act` = 1 or `SimOn` = 1
28	Invalid signal status
29	0 = Closed 1 = Open
30	"Permission" button is enabled
31	"Protection" button is enabled

Status word assignment for `Status2` parameter

Status bit	Parameter
0	`MsgLock`
1	`AV_AH_Act.Value`
2	`AV_WH_Act.Value`
3	`AV_TH_Act.Value`
4	`AV_TL_Act.Value`
5	`AV_WL_Act.Value`
6	`AV_AL_Act.Value`
7	`AV_AH_En`
8	`AV_WH_En`
9	`AV_TH_En`
10	`AV_TL_En`
11	`AV_WL_En`
12	`AV_AL_En`
13	`AV_AH_MsgEn`
14	`AV_WH_MsgEn`
15	`AV_TH_MsgEn`
16	`AV_TL_MsgEn`
17	`AV_WL_MsgEn`
18	`AV_AL_MsgEn`
19	1 = Input signals have no impact on "Local mode" when `LocalSetting` = 2 and `LocalSetting` = 4
20	1 = Valve is closing
21	1 = Valve is closed
22	1 = Valve is stopped
23	1 = Valve is opening
24	1 = Valve is open
25	For the error status display in the closed valve
26	For the error status display in the opened valve
27	Automatic preview for "Open"
28	Automatic preview for "Close"
29	Automatic preview for "Stop"
30	Display for interlocks in block icon
31	`MS_RelOp`

Status word assignment for `Status3` parameter

Status bit	Parameter
0	`M_MonStaErr.Value`
1	`M_MonDynErr.Value`
2	`V_MonStaErr.Value`
3	`V_MonDynErr.Value`

Status bit	Parameter
4	M_MonStopErr.Value
5	Automatic preview for "Position Control" on
6	Automatic preview for "Position Control" off
7	Torque shutoff opening enabled
8	Torque shutoff closing enabled
9	External fault generated by FaultExt or external control system fault CSF with set Feature.Bit18 Activating error state for external process control error CSF (Page 150)
10	Vibration successfully completed – Reset required
11	Display motor protection (Trip status, Trip.ST ≠ 16#FF)
12	1 = Input parameter FbkClose is interconnected
13	1 = Input parameter FbkClosing is interconnected
14	1 = Input parameter FbkOpen is interconnected
15	1 = Input parameter FbkOpening is interconnected
16	1 = Input parameter TorOpen is interconnected
17	1 = Input parameter TorClose is interconnected
18	SimLiOp.Value
19	1 = Enable for rapid stop (Feature.Bit14 Enabling rapid stop via faceplate (Page 167))
20	1 = Input parameter OpenChnST is interconnected
21	1 = Input parameter CloseChnST is interconnected
22	Not used
23	Command for rapid stop
24	"Open"/"Stop" command
25	"Close"/"Stop" command
26	Show automatic preview in the standard view
27	Not used
28	GrpErr.Value
29	RdyToStart.Value
30	Auxiliary value 1 visible
31	Auxiliary value 2 visible

Status word assignment for `Status4` parameter

Status bit	Parameter
0	Delay of message ER_AH_Lim
1	Delay of message ER_AL_Lim
2	Delay of message RbkWH_Lim
3	Delay of message RbkWL_Lim
4	Collection of message delays
5	1 = Monitor manipulated variable difference high limit (ER_AH_En)
6	1 = Monitor manipulated variable difference low limit (ER_AL_En)
7	1 = Signal manipulated variable difference high limit violation (ER_AH_MsgEn)

Status bit	Parameter
8	1 = Signal manipulated variable difference low limit violation (ER_AL_MsgEn)
9	1 = Monitor readback value high limit (RbkWH_En)
10	1 = Monitor readback value low limit (RbkWL_En)
11	1 = Signal readback value high limit violation (RbkWH_MsgEn)
12	1 = Signal readback value low limit violation (RbkWL_MsgEn)
13	OpenForce.Value
14	StopForce.Value
15	CloseForce.Value
16	1 = Forced manipulated variable MV_Forced is active
17	1 = Tracking value MV_Trk is active
18	Reset request in automatic mode
19	Trip.Value
20	1 = Vibrate enabled
21 - 22	Not used
23	Hidden bypass signal in "Permit"
24	Hidden bypass signal in "Interlock"
25	Hidden bypass signal in "Protect"
26	Feature2.Bit2 Separate evaluation for excluded and simulated interlock signals (Page 151)
27	Not used
28	Current position feedback monitoring time is visible
29	Current motor monitoring time is visible
30	Current valve monitoring time is visible
31	Feature.Bit13 Separate monitoring time for stopping the motor (Page 168)

Status word assignment for `Status5` parameter

Status bit	Parameter
0 - 7	Effective signal 1 to 8 of the message block connected via EventTsIn
8 - 15	Effective signal 9 to 16 of the message block connected via EventTsIn
16	AV is not connected
17	Delay of the message AV_AH_Lim
18	Delay of the message AV_WH_Lim
19	Delay of the message AV_TH_Lim
20	Delay of the message AV_TL_Lim
21	Delay of the message AV_WL_Lim
22	Delay of the message AV_AL_Lim
23	Collection of the message delays
24	1 = High limit of the manipulated variable difference is violated (ER_AH_Act.Value)
25	1 = Low limit of the manipulated variable difference is violated (ER_AL_Act.Value)
26	1 = High limit of the readback value is violated (RbkWH_Act.Value)
27	1 = Low limit of the readback value is violated (RbkWL_Act.Value)

Status bit	Parameter
28	1 = Valve still closes (`Status2` Bit 20) but the motor is not running
29	1 = Valve still opens (`Status2` Bit 23) but the motor is not running
30 - 31	Not used

Status word allocation for `Status6` parameter

Status bit	Parameter
0 - 15	Effective signal 1 to 16 of the message block connected via `EventTs2In`
16 - 31	Not used

See also

VlvPosL functions (Page 1495)

VlvPosL messaging (Page 1513)

VlvPosL block diagram (Page 1527)

VlvPosL error handling (Page 1510)

VlvPosL modes (Page 1493)

7.14.2 VlvPosL modes

VlvPosL operating modes

The block supports all standard operating modes:

- Local mode (Page 79)
- Automatic mode (Page 75)
- Manual mode (Page 75)
- Out of service (Page 71)

The next section provides additional block-specific information relating to the general descriptions.

"Local mode"

You can find general information on "Local mode", switching modes, and bumpless switchover in the section Local mode (Page 79).

When you put a block into "Local mode", it is controlled either by "Local" signals or by feedback signals. With `LocalSetting` = 1 or 3, you can control the motor valve with the following “Local” signals:

- "Open" (`OpenLocal` = 1)
- "Close" (`CloseLocal` = 1)
- "Stop" (`StopLocal` = 1)

The feedback signals `FbkOpening`, `FbkClosing`, `FbkOpen`, `FbkClose`, and the analog readback `Rbk` will be monitored.

With `LocalSetting` = 2 or 4, the control outputs `Open.Value`, `Close.Value`, and the internal status `Open`/`Close`/`Stop` of the block will be tracked.

In the case where motor feedback signals are available (`Feature.Bit12` = 0), the control outputs `Open.Value` and `Close.Value` will be tracked from the motor feedback signals `FbkOpening` and `FbkClosing`. The valve feedback signals `FbkOpen`, `FbkClose`, and the analog readback `Rbk` will be monitored.

In the case where no motor feedback signals are available (`Feature.Bit12` = 1), the internal status `Open`/`Close`/`Stop` of the block will be tracked from the valve feedback signals `FbkOpen`, `FbkClose`, and the analog readback `Rbk`. It will only be monitored if the analog readback and valve feedback signals are not suitable together.

If no position can be identified, the last valid position is assumed.

"Automatic mode"

You can find general information on "Automatic mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

You can control the motor valve in "Automatic mode" with analog position feedback:

- "Open" (`OpenAut` = 1)
- "Close" (`CloseAut` = 1)
- "Stop" (`StopAut` = 1)

Position control is terminated with "Open", "Close", or "Stop". It is enabled again with:

- "Activate position control" (`PosOnAut` = 1)

For position control, the motor valve is controlled via the `Open` and `Close` outputs such that the manipulated variable `MV` and position feedback `Rbk` come into agreement.

"Manual mode"

You can find general information on "Manual mode", switching modes and bumpless switchover in the section Manual and automatic mode for motors, valves and dosers (Page 75).

You can control the motor valve in "Manual mode" with analog position feedback:

- "Open" (`OpenMan` = 1)
- "Close" (`CloseMan` = 1)
- "Stop" (`StopMan` = 1)

Position control is terminated with "Open", "Close", or "Stop". It is enabled again with:

- "Activate position control" (`PosOnMan` = 1)

For position control, the motor valve is controlled via the `Open` and `Close` outputs such that the manual value `Man` and position feedback `Rbk` come into agreement.

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

VlvPosL block diagram (Page 1527)

VlvPosL I/Os (Page 1515)

VlvPosL error handling (Page 1510)

VlvPosL functions (Page 1495)

VlvPosL messaging (Page 1513)

Description of VlvPosL (Page 1488)

7.14.3 VlvPosL functions

Functions of VlvPosL

The functions for this block are listed below.

Motor valve control

The motor valve is controlled via the `Open` and `Close` outputs or via the pulse outputs `P_Open`, `P_Close`, and `P_Stop`. The outputs are set by either of the following methods:

- Directly via the corresponding inputs:
  - `OpenMan`, `CloseMan`, and `StopMan` in manual mode
  - `OpenAut`, `CloseAut`, and `StopAut` in automatic mode
  - `OpenLocal`, `CloseLocal`, and `StopLocal` in local mode
  - `OpenForce`, `CloseForce`, and `StopForce` for the force operating states function.
- Calculated by a control algorithm after the position controller is switched on in such a way that the position feedback `Rbk` reaches a specified manipulated variable.

Position control is switched on by:

- `PosOnMan` in manual mode
- `PosOnAut` in automatic mode
- `MV_ForOn` for the force operating states functions.

In the local mode, the manipulated variable is always tracked to the position feedback.

For direct control via the `Open`, `Close`, `Stop` inputs above, position control is terminated and must be reactivated via the `PosOn` inputs.

Motor valve control with position control

When controlling with enabled position control, the motor valve is moved to the desired valve position using the specified manipulated variable.

Depending on the mode and function, the manipulated variable is specified by:

- `MV_Forced`: Manipulated variable specification with the "Force" operating states function (`MV_ForOn` =1)
- `Man`: Manipulated variable specification in manual mode
- `MV`: Manipulated variable specification in automatic mode
- `MV_Trk`: Manipulated variable specification in automatic mode and for the "Track" manipulated variable function (`MV_TrkOn` =1)

Except for the "Force" operating states function (`MV_ForOn` =1), the manipulated variable is always limited by `MV_HiLim` and `MV_LoLim`. The effective manipulated variable is outputtted at the `MV_Out` output.

The control and manipulated variable difference `MV_Out` – `RbkOut` is controlled via a deadband with the deadband width `DeadBand` and displayed at the `ER` output.

Manipulated variable difference `ER`	Deadband range
(`MV_Out` – `RbkOut`) - `DeadBand`	`MV_Out` – `RbkOut` **>=** `DeadBand`
0.0	`-DeadBand` **<=** `MV_Out` – `RbkOut` **<=** `DeadBand`
(`MV_Out` – `RbkOut`) + `DeadBand`	`MV_Out` – `RbkOut` **<=** `-DeadBand`

The internal position controller is set with the following parameters:

- `Gain`: Controller gain
- `MotorTime`: Motor runtime from closed to fully open valve
- `TmLag`: Delay time caused, for example, by lag of the motor valve when switching on and off.

The pulse output is formed by a three-step element with hysteresis and set with the following parameters:

- `PulseTime`: Minimum pulse length of the outputs `Open` or `Close`
- `BreakTime`: Mminimum pulse break between two pulse outputs of `Open` or `Close`

Note

`PulseTime` and `BreakTime` are limited to `SampleTime`. If they are smaller parameterized than `SampleTime` they will be written back to the input with the value of `SampleTime`.

`PulseTime` and `BreakTime` values which are too large, can reduce the precision of position control.

The effective trip threshold and response threshold of the three-step element is displayed at the `ThreshOff` and `ThreshOn` output parameters.

The `PosReached` output shows whether the specified position has been reached. This is the case when:

- Intermediate position:
  `PosRbkHys` **>=** ABS(`MV_Out` - `RbkOut`)
- End position open:
  `MV_Out` **>=** `PosDiOpen` AND `RbkOut` **>=** `PosDiOpen` AND `FbkOpenOut` AND NOT `FbkCloseOut`
- End position closed:
  `MV_Out` **<=** `PosDiClose` AND `RbkOut` **<=** `PosDiClose` AND `FbkCloseOut` AND NOT `FbkOpenOut`

The control and feedback signals of the motor are at "Stopped" state.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the parameter `OS_Perm`:

Bit	Function
0	1 = Operator can switch to "Automatic mode"
1	1 = Operator can switch to "Manual mode"
2	1 = Operator can switch to "Local mode"

Bit	Function
3	1 = Operator can switch to "Out of service mode"
4	1 = Operator can stop the motor
5	1 = Operator can open the valve
6	1 = Operator can close the valve
7	1 = Operator can trigger rapid stop for motors
8	1 = Operator can activate position control
9	1 = Operator can reset the valve
10	1 = Operator can change the manual value `Man`
11	1 = Operator can change the simulation value `SimRbk`
12	1 = Operator can define the valve monitoring time for control
13	1 = Operator can define the valve monitoring time for end position
14	1 = Operator can activate the function valve monitoring time (bits 12 and 13)
15	1 = Operator can define the motor monitoring time for control start or stop
16	1 = Operator can define the monitoring time for control stop
17	1 = Operator can define the motor monitoring time for status
18	1 = Operator can activate the function motor monitoring time (bits 15,16, and 17)
19	1 = Operator can define the readback monitoring time
20	1 = Operator can activate the function readback monitoring time (bit 19)
21	Not used
22	1 = Operator can change proportional gain of the position control
23	Not used
24	1 = Operator can set the delay time of the motor valve
25	1 = Operator can set the deadband width of the manipulated variable difference
26	1 = Operator can set the motor run time
27	1 = Operator can set the minimum pulse duration
28	1 = Operator can set the minimum break duration
29	Not used
30	1 = Operator can activate the "Simulation" function
31	1 = Operator can activate the "Release for maintenance" function

The block has the following permissions for the parameter `OS1Perm`:

Bit	Function
0	1 = Operator can change the high operating limit of the manipulated variable `MV_HiLim`
1	1 = Operator can change the low operating limit of the manipulated variable `MV_LoLim`
2	Not used
3	1 = Operator can enable/disable messages via `ER_AH_MsgEn`
4	1 = Operator can enable/disable messages via `ER_AL_MsgEn`
5	1 = Operator can enable/disable messages via `RbkWH_MsgEn`
6	1 = Operator can enable/disable messages via `RbkWL_MsgEn`
7	1 = Operator can change the limit (manipulated variable difference) for the high alarm `ER_AH_Lim`
8	1 = Operator can change the hysteresis (manipulated variable difference) `ER_Hyst`

Bit	Function
9	1 = Operator can change the limit (manipulated variable difference) for the low alarm `ER_AL_Lim`
10	1 = Operator can change the limit (position feedback) for the high warning `RbkWH_Lim`
11	1 = Operator can change the hysteresis (position feedback) `RbkHyst`
12	1 = Operator can change the hysteresis (position feedback) for the low warning `RbkWL_Lim`
13	Not used
14	1 = Operator can change the simulation value `SimAV`
15	1 = Operator can change the limit (`AV`) for the high alarm
16	1 = Operator can change the limit (`AV`) for the high warning
17	1 = Operator can change the limit (`AV`) for the high tolerance
18	1 = Operator can change the limit (`AV`) for the hysteresis
19	1 = Operator can change the limit (`AV`) for the low alarm
20	1 = Operator can change the limit (`AV`) for the low warning
21	1 = Operator can change the limit (`AV`) for the low tolerance
22	1 = Operator can enable / disable messages via `AV_AH_MsgEn`
23	1 = Operator can enable / disable messages via `AV_WH_MsgEn`
24	1 = Operator can enable / disable messages via `AV_TH_MsgEn`
25	1 = Operator can enable / disable messages via `AV_TL_MsgEn`
26	1 = Operator can enable / disable messages via `AV_WL_MsgEn`
27	1 = Operator can enable / disable messages via `AV_AL_MsgEn`
28 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm`, `OS1Perm` as a parameter, you have to reset the corresponding `OS_Perm`, `OS1Perm` bit.

Limit monitoring for additional analog value

This block provides the standard function Limit monitoring of an additional analog value (Page 91).

Limit monitoring with hysteresis

This block provides the standard function Limit monitoring with hysteresis (Page 97). This is done using the input parameter `AV_Hyst`.

Suppress messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Interlocks

This block provides the following interlocks:

- Activation enable
- Interlock without reset ("Interlock")
- Interlock with reset ("Protection")

For more on this, see the sections Interlocks (Page 99) and Influence of the signal status on the interlock (Page 103).

When an interlock is pending and position control is enabled, the block remains in position control. The following manipulated variables are controlled based on the configured neutral position for an interlock with/without reset:

SafePos	Manipulated variable
0: Closed	MV_Out = MV_LoLim
1: Opened	MV_Out = MV_HiLim
2: Stopped	MV_Out = RbkOut

Note

The neutral position cannot be reached if the limit MV_HiLim is not configured greater than or equal to PosDiOpen or MV_LoLim is not configured less than or equal to PosDiClose.

With VlvPosL in position control, the neutral position is reached when:

SafePos	Neutral position is reached when
0: Closed	PosReached = 1, FBkOpenOut = 0, FBkCloseOut = 1
1: Opened	PosReached = 1, FBkOpenOut = 1, FBkCloseOut = 0
2: Stopped	PosReached = 1, independent of FBkOpenOut, FBkCloseOut

If VlvPosL is in the neutral position, it can only leave this position when an activation enable is set.

Motor protection function

This block provides the standard function Motor protection function (Page 99).

When motor protection is pending and position control is enabled, the block remains in position control. The manipulated variable is tracked to the feedback value (MV_Out = RbkOut) and with this, the motor is in the stop state.

Rapid stop for motors

This block provides the standard function Rapid stop for motors (Page 106).

When rapid stop is pending and position control is enabled, the block remains in position control. The manipulated variable is tracked to the feedback value (MV_Out = RbkOut) and with this, the motor is in the stop state.

Torque monitoring

This block provides torque monitoring.

The signals of the torque monitoring switches are interconnected to input parameters `TorqOpen` for opening and `TorqClose` for closing the motor valve.

The good state is indicated via this parameter with the value 1. In this case, the signal status cannot be 16#00 or 16#28.

Using `Feature2.Bit6` (Vibrate), you can define how to proceed in automatic mode when the torque has been reached. With `Feature2.Bit6` = 0 (no vibrating), position control is terminated and the motor is stopped. In the case where `TorqClose` is in end position close or `TorqOpen` is in end position open, the position control will not be terminated.

Vibration is disabled or VlvPosL is not in automatic mode:

If the torque monitoring is enabled, the motor is stopped. You can move the valve in the opposite direction.

If, for example, the torque shutdown is active when the valve opens, you can still close the valve.

Active torque shutoff appears in the standard view of the display area for block states.

When the "Seal valve" function is enabled by means of `Feature.Bit8`, the torque shutoff for the closing `TorqClose` is also evaluated (see section Sealing the valve (Page 175)).

Vibration is enabled and VlvPosL is in automatic mode:

If torque monitoring is enabled, the motor runs in the opposite direction. After traveling `VibrWidth` seconds or `VibrPerc` %, the motor runs in the opposite direction again to solve the cause of the torque shutoff using vibration. This procedure is repeated a maximum of `VibrNo` times.

With activated position control, the motor travels from the current position `VibrPerc` % in the opposite direction. When position control is disabled, the control outputs are immediately sent in the opposite direction for `VibrWidth` seconds.

If, for example, the torque shutdown is active when the valve opens with activated position control, the motor valve returns to the value:

`RbkOut` - (`MV_HiLim` - `MV_LoLim`) / 100.0 * `VibrPerc`

When this point is reached, the motor attempts to open to the previously set manipulated variable once again.

During reversing, the display area of the standard view, "Vibrate enabled" is displayed.

After attempting to correct the cause of the torque `VibrNo` times, the motor is stopped. The block must be reset before vibrating can be enabled again.

Sealing the valve

This function is enabled using `Feature.Bit8` Sealing the valve (Page 175). The seal valve function combines the query of the end position "Closed" via the input parameter `FbkClose` and `Rbk` with the limit violation of the configured torque via the input parameter `TorqClose`. This ensures that the valve is completely closed.

The valve is only considered completely closed when the feedback for the end position "Close" has been received (0->1), the position feedback `Rbk` is smaller than or equal to

PosDiClose, and the torque shutdown for "Closed" is enabled. The torque shutoff should not come before the feedback in this case. The FbkCloseOut output shows whether the valve is sealed tight with the following conditions:

- FbkCloseOut:= FbkClose.Value AND RbkOut.Value <= PosDiClose has come in (0->1)
- "Torque shutoff closed is enabled"
- "Torque shutoff closed is enabled" did not come before FbkCloseOut (0->1)

"Torque shutoff closed enabled" means TorqClose = 0 or the signal status is 16#00 or 16#28.

When "Torque shutoff closed is enabled" comes before the end position feedback closed, this is displayed in the faceplate in the standard view and the motor stops. Opening the valve is still possible.

Note

The command text for Close in the standard view and preview can be changed to Seal in the CFC at the CloseMan parameter in Text 1. See section Labeling of buttons and text (Page 205).

Disable interlocks

This block provides the standard function Disabling interlocks (Page 103).

Resetting the block in case of interlocks

This block provides the standard function Resetting the block in case of interlocks or errors (Page 43).

External fault (FaultExt), External control system fault (CSF)

This block provides the possibility to pass an external fault via the FaultExt parameter or an external control system fault via the CSF parameter. See VlvPosL error handling (Page 1510).

When an external fault or control system fault is pending (Feature.Bit18 = 1) and position control is enabled, the block remains in position control. The following manipulated variable is set based on the configured neutral position:

SafePos	Manipulated variable
0: Closed	MV_Out = MV_LoLim
1: Opened	MV_Out = MV_HiLim
2: Stopped	MV_Out = RbkOut

Note

The neutral position cannot be reached if the limit MV_HiLim is not configured greater than or equal to PosDiOpen or MV_LoLim is not configured less than or equal to PosDiClose.

Group error

This block provides the standard function, Outputting group errors (Page 122).

The following parameters are taken into consideration when forming the group error:

- CSF
- Trip
- V_MonDynErr
- V_MonStaErr
- M_MonDynErr
- M_MonStaErr
- RbkMonDynErr
- RbkMonStaErr
- FaultExt

Outputting a signal for start readiness

This block provides the standard function, Outputting a signal for start readiness (Page 53).

Forming the group status for interlocks

This block provides the standard function, Forming the group status for interlock information (Page 104).

Forming the signal status for blocks

This block provides the standard function, Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status for the block is formed from the following parameters:

- FbkClsgOut.ST
- FbkOpngOut.ST
- FbkOpenOut.ST
- FbkCloseOut.ST
- RbkOut.ST
- LocalLi.ST
- OpenLocal.ST
- StopLocal.ST
- CloseLocal.ST
- Trip.ST
- TorqOpen.ST
- TorqClose.ST

- `AV_Out.ST`
- `OpenChn.ST`
- `CloseChn.ST`
- `OpenAut.ST` (only if `Feature2.Bit10` = 1)
- `CloseAut.ST` (only if `Feature2.Bit10` = 1)
- `StopAut.ST` (only if `Feature2.Bit10` = 1)
- `MV.ST` (only if `Feature2.Bit10` = 1)
- `PosOnAut.ST` (only if `Feature2.Bit10` = 1)

Considering bad quality of automatic commands or external values

If the `Feature2` bit Considering bad quality of automatic commands or external values (Page 185) is set to 1 and one of the following parameters has bad signal status (16#00 or 16#28), the block goes to the neutral position in the "Automatic" mode:

- `OpenAut.ST`
- `CloseAut.ST`
- `StopAut.ST`
- `MV.ST`
- `PosOnAut.ST`

Force operating states

This block provides the standard function, Forcing operating modes (Page 41). The `OpenForce`, `CloseForce`, and `StopForce` inputs force the block to open, close, or stop. With activated position control, this mode is terminated and must be separately reactivated.

With `MV_ForOn` = 1, position control is enabled and set to the manipulated variable `MV_Forced`.

Feedback monitoring

This block provides the standard function, Monitoring the feedbacks (Page 97).

The `FbkOpen`, `FbkClose`, and `Rbk` feedback is monitored for the valve, the `FbkOpening` and `FbkClosing` feedback is monitored for the motor.

Position control is terminated if a monitoring error of the feedback occurs. The configured neutral position is targeted if a monitoring error of the valve feedback occurs.

The control signals are switched to motor stop if a monitoring error of the motor feedback occurs.

Monitoring valve feedback

The monitoring of valve feedback is set using the `V_Monitor` parameter.

Startup characteristics are monitored by setting the V_MonTiDynamic parameter. When starting the motor valve, the current position feedback Rbk is taken into consideration and is included in the calculation of the monitoring time as follows:

Monitoring of the position feedback is switched on (RbkMonitor =1):

- Monitoring time = V_MonTiDynamic – MotorTime + Abs(MV_Out - RbkOut) * MotorTime/100

Note

The dynamic monitoring time V_MonTiDynamic should be greater than motor actuating time MotorTime.

Monitoring of the position feedback is switched off (RbkMonitor =0):

- Open or close motor valve: Monitoring time = V_MonTiDynamic

Monitoring of the position feedback Rbk with activated position control can be set separately using RbkMonitor, RbkMonTi.

The V_MonTiStatic parameter sets the compliance with the valve position.

Feedback errors are displayed at the corresponding parameters V_MonDynErr or V_MonStaErr.

The valve travel range is divided into three ranges for monitoring:

- Valve open: RbkOut ≥ PosDiOpen
- Valve in intermediate position: PosDiClose < RbkOut < PosDiOpen
- Valve closed: RbkOut ≤ PosDiClose

To monitor the end positions, the FbkOpenOut and FbkCloseOut outputs are formed as follows:

When the motor valve reaches the "Open" position, the output is FbkOpenOut = 1:

- With a binary limit switch for the "Open" position (NoFbkOpen = 0):
  The valve is considered open if FbkOpenOut is set. FbkOpenOut is set if RbkOut ≥ PosDiOpen and FbkOpen = 1.
- Without binary end switch for the position "Open" (NoFbkOpen = 1):
  The valve is considered open if FbkOpenOut is set. FbkOpenOut is set if RbkOut ≥ PosDiOpen.

When the motor valve reaches the "Closed" position, the output is FbkCloseOut = 1:

- With a binary limit switch for the "Closed" position (NoFbkClose = 0):
  The valve is considered closed if FbkCloseOut is set. FbkCloseOut is set if RbkOut ≤ PosDiClose and FbkClose = 1.
- Without binary end switch for the position "Closed" (NoFbkClose = 1):
  The valve is considered closed if FbkCloseOut is set. FbkCloseOut is set if RbkOut ≤ PosDiClose.

Note

After the motor valve stops in the intermediate position or end position or after a CPU startup without control (`Open`, `Close` = 0), no monitoring of the feedback signals `FbkOpen` and `FbkClose` takes place during the time `V_MonTiStatic`. Changes to `FbkOpen` and `FbkClose` are applied. This means that the feedback is monitored again, also in the stop state.

Note

When the "Seal valve" function is enabled by means of `Feature.Bit8`, the torque shutoff for the closing `TorqClose` is also evaluated (see section Sealing the valve (Page 175)).

Monitoring the motor feedback

The monitoring of motor feedback is set using the `M_Monitor` parameter.

Startup characteristics are monitored by setting the `M_MonTiDynamic` parameter; the `M_MonTiStatic` parameter monitors maintenance of the position.

Feedback errors are displayed at the corresponding parameters `M_MonDynErr` or `M_MonStaErr`.

Monitoring of the position feedback Rbk with activated position control

The monitoring of position feedback `Rbk` is set using the `RbkMonitor` parameter.

The monitoring of the startup characteristics is set at the `RbkMonTi` parameter. The effective monitoring time is calculated as follows:

- Monitoring time = ABS(MV_Out_n - MV_Out_{n-1}) * `MotorTime`/100 + `RbkMonitor`
  MV_Out_n: Current manipulated variable to be reached
  MV_Out_{n-1}: Last manipulated variable reached

The monitoring of maintaining the valve position is set at the `RbkMonTi`

- Monitoring time = `RbkMonitor`

Feedback errors are displayed at the corresponding parameters `RbkMonDynErr` or `RbkMonStaErr`.

Release for maintenance

This block provides the standard function, Release for maintenance (Page 64).

Specify warning times for control functions

This block provides the standard function, Specifying warning times for control functions at motors and valves (Page 51).

You can generate warning signals when, for example, valves open. Warning signals can be generated in the following modes:

- Manual and automatic mode for motors, valves and dosers (Page 75) (input parameter WarnTiMan)
- Manual and automatic mode for motors, valves and dosers (Page 75) (input parameter WarnTiAut)

You specify the warning times in seconds using the input parameters WarnTiMan and WarnTiAut . If, for example, a valve opens, this is displayed at the output parameter with WarnAct = 1. The valve then opens after the set warning time has expired and WarnAct then returns to 0.

A corresponding warning is not outputted if the values specified for the warning times (WarnTiMan or WarnTiAut) are lower than the SampleTime parameter.

Note

The warning is activated for each actuation that causes the motor to start, even if this means that the valve is moved to the neutral position.

A parameterized warning time (WarnTiMan, WarnTiAut) is not active in the position control mode.

Simulating signals

This block provides the standard function, Simulating signals (Page 58).

You can simulate the following values:

- Additional value (SimAV, SimAV_Li)

With internal simulation with immediate tracking of feedback, it is possible to simulate a position between the open and closed state (FbkOpenOut = FbkCloseOut = 0) by means of a stop command.

Select unit of measure

This block provides the standard function, Selecting a unit of measure (Page 207).

Neutral position

This block provides the standard function, Neutral position for motors, valves and controllers (Page 48).

Output signal as pulse signal or static signal

This block provides the standard function Output signal as a static signal or pulse signal (Page 51). In addition to the static control outputs Open and Close, the block also has pulse outputs P_Open, P_Close, and P_Stop, which are dependent on the static control output.

Generate instance-specific messages

This block provides the standard function, Generating instance-specific messages (Page 200).

Configurable reactions using the `Feature` parameter

An overview of all the reactions that are provided by the `Feature` parameter is available in section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
3	Enabling resetting of commands for the control settings (Page 160)
4	Setting switch or button mode (Page 166)
8	Sealing the valve (Page 175)
9	Resetting via input signals in the event of interlocking (Protection) or errors (Page 162)
10	Exiting local mode (Page 176)
11	Activating the run time of feedback signals (Page 152)
12	Motor feedback is not available (Page 156)
13	Separate monitoring time for stopping the motor (Page 168)
14	Enabling rapid stop via faceplate (Page 167)
17	Enabling bumpless switchover to automatic mode for valves, motors, and dosers (Page 172)
18	Activating error state for external process control error CSF (Page 150)
19	Reset even with locked state (Page 164)
20	Disable calculation of impulse control in LocalSetting 2 and 4 (Page 185)
21	Enable bumpless switchover to "Automatic" mode for operator only (Page 171)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
27	Interlock display with LocalSetting 2 or 4 (Page 177)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)
31	Activating reset of protection / error in manual mode (Page 164)

In switching mode (`Feature.Bit4` = 1), the control is selected with the static signals `OpenAut` and `CloseAut`. If the `OpenAut` and `CloseAut` inputs are not set, the motor is stopped. Control via `StopAut` is not required. If the "Enable reset of commands for control" function (`Feature.Bit3` = 1) is also enabled, the `OpenAut` and `CloseAut` inputs are reset to 0 after evaluation in the block.

In switching mode (`Feature.Bit4` = 1), the `PosOnAut` control acts as a switch. The "Enable resetting of commands for control" function (`Feature.Bit3` = 1) has no influence on `PosOnAut`.

Display auxiliary values

This block provides the standard function, Displaying auxiliary values (Page 207).

Configurable reactions using the `Feature2` parameter

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
3	Control priority in the event of an invalid input command (Page 174)
4	Setting switch or button mode for local commands (Page 180)
5	Evaluation of the signal status of the interlock signals (Page 141)
6	Vibrate after torque monitoring (Page 177)
8	Forcing operating modes in the "Local" mode (Page 183)
10	Considering bad quality of automatic commands or external values (Page 185)

Connection of the time-stamped messages from EventTs or Event16Ts

If the output parameter `EventTsOut` of the block EventTs or Event16Ts is connected to the input parameter `EventTsIn` or `EventTs2In` of this block, the time-stamped messages of the block EventTs or Event16Ts will be displayed at this block. If you change the operating mode of this block to "Out of service" mode, the block EventTs or Event16Ts will also change to "Out of service" mode. In the batch mode, the batch parameter of this block will be used for generating the time-stamped messages in the block EventTs or Event16Ts.

SIMATIC BATCH functionality

This block provides the standard function, SIMATIC BATCH functionality (Page 67).

Disable feedback

This block provides the standard function, Disabling feedback for valves (Page 99). Feedback monitoring can be disabled separately for each feedback signal with `NoFbkOpen` or `NoFbkClose`.

Labeling of buttons

This block provides the standard function, Labeling of buttons and text (Page 205).

Instance-specific text can be configured for the following parameters:

- `OpenMan`
- `CloseMan`
- `StopMan`
- `RapidStp`

See also

EventTs functions (Page 1634)

Description of VlvPosL (Page 1488)

VlvPosL messaging (Page 1513)

VlvPosL I/Os (Page 1515)

VlvPosL block diagram (Page 1527)

VlvPosL modes (Page 1493)

7.14.4 VlvPosL error handling

Error handling of VlvPosL

For troubleshooting all blocks, see also the Error handling (Page 119) section in the basics.

The following errors can be displayed for this block:

- Error numbers
- Mode switchover error
- Invalid input signals
- Control system fault (`CSF`)

Overview of error numbers

The `ErrorNum` parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Pre-defined value when inserting the block; the block is not executed.
0	No error pending.
41	The value for the `LocalSetting` connection is not within the valid limit of 0 to 4.
42	`LocalSetting` = 0 or `LocalSetting` = 3 or `LocalSetting` = 4 and `LocalLi` = 1

Error number	Meaning of the error number
51	OpenLocal = 1 and StopLocal = 1 CloseLocal = 1 and StopLocal = 1 OpenLocal = 1 and CloseLocal = 1 OpenAut = 1 and StopAut = 1 CloseAut = 1 and StopAut = 1 OpenAut = 1 and CloseAut = 1 PosOnAut = 1 and StopAut = 1 PosOnAut = 1 and OpenAut = 1 PosOnAut = 1 and CloseAut = 1 AutModLi = 1 and ManModLi = 1 OpenForce = 1 and StopForce = 1 CloseForce = 1 and StopForce = 1 OpenForce = 1 and CloseForce = 1
52	LocalAct = 1 and LocalSetting = 2 or 4 and SimOn = 1

Mode switchover error

This error can be outputted by the block, see section Error handling (Page 119).

Invalid input signals

This error can be outputted by the block, see section Error handling (Page 119).

For the following invalid input signals, the control output can be stopped or switched to the neutral position. This depends on the function Control priority in the event of an invalid input command (Page 174).

Operating mode	Invalid input signals	Control reaction with **Feature2.Bit3 = 1**
Local: Localsetting = 1 or Localsetting = 3	Pushbutton operation for local mode (Feature2 bit 4 = 0): OpenLocal = 1 and CloseLocal = 1 or OpenLocal = 1 and StopLocal = 1 or StopLocal = 1 and CloseLocal = 1 Switching mode (Feature2 bit 4 = 1): OpenLocal = 1 and CloseLocal = 1	Motor is started in valve neutral position direction.
Local: Localsetting = 1 or Localsetting = 3 and forcing	OpenForce = 1 and CloseForce = 1 or OpenForce = 1 and StopForce = 1 or StopForce = 1 and CloseForce = 1	
Forcing and no "Local mode"	OpenForce = 1 and CloseForce = 1 or OpenForce = 1 and StopForce = 1 or StopForce = 1 and CloseForce = 1	
"Automatic mode" and no forcing	Pushbutton operation (Feature.Bit4 = 0): OpenAut = 1 and CloseAut = 1 or OpenAut = 1 and StopAut = 1 or StopAut = 1 and CloseAut = 1 or StopAut = 1 and PosOnAut = 1 or CloseAut = 1 and PosOnAut = 1 or OpenAut = 1 and PosOnAut = 1 Switch mode (Feature.Bit4 = 1): OpenAut = 1 and CloseAut = 1 or CloseAut = 1 and PosOnAut = 1 or OpenAut = 1 and PosOnAut = 1	
"Manual mode" and no forcing	OpenMan = 1 and CloseMan = 1 or OpenMan = 1 and StopMan = 1 or StopMan = 1 and CloseMan = 1	

Control system fault (CSF)

An external signal can be activated via the CSF input. If this signal CSF.Value = 1, a control system fault is triggered. For more information on this, see section Error handling (Page 119).

See also

VlvPosL block diagram (Page 1527)

VlvPosL I/Os (Page 1515)

VlvPosL functions (Page 1495)

VlvPosL modes (Page 1493)

Description of VlvPosL (Page 1488)

7.14.5 VlvPosL messaging

Message characteristics

The following messages can be generated for this block:

- Control system fault
- Process messages
- Instance-specific messages

Control system fault

The following control system fault messages can be outputted:

Message instance	Message ID	Message class	Event
MsgEvId1	SIG1	AS control system message - fault	$$BlockComment$$ Motor feedback error
	SIG2	AS control system message - fault	$$BlockComment$$ Motor protection triggered
	SIG3	AS control system message - fault	$$BlockComment$$ Valve feedback error
	SIG4	AS control system message - fault	$$BlockComment$$ External fault has occurred

Explanation:

$$BlockComment$$: Content of the instance-specific comment

You can interconnect an external fault (signal) to the input parameter CSF. If this CSF = 1, a control system fault is triggered (MsgEvId1, SIG4).

Process messages

Message instance	Message ID	Message class	Event
MsgEvId2	SIG1	Alarm - high	$$BlockComment$$ ER - Alarm high limit violated
	SIG2	Alarm - low	$$BlockComment$$ ER - Alarm low limit violated
	SIG3	Warning - high	$$BlockComment$$ Rbk - High warning limit violated
	SIG4	Warning - low	$$BlockComment$$ Rbk - Low warning limit violated

Instance-specific messages

You can use up to four instance-specific messages with this block.

Message instance	Message ID	Message class	Event
MsgEvId1	SIG5	AS control system message - fault	$$BlockComment$$ External message 1
	SIG6	AS control system message - fault	$$BlockComment$$ External message 2
MsgEvId2	SIG5	AS control system message - fault	$$BlockComment$$ External message 3
	SIG6	AS control system message - fault	$$BlockComment$$ External message 4

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for the message instance **MsgEvId1**

Associated value	Block parameter
1	BatchName
2	StepNo
3	BatchID
4	ExtVa104
5	ExtVa105
6	ExtVa106
7	Reserved
8	Reserved
9	Reserved
10	Reserved

The associated values 4 to 6 are assigned to the parameters ExtVa104 to ExtVa106, and are usable. For more information on this, refer to the manual "*Process Control System PCS 7 - Engineering System*".

Associated values for the message instance **MsgEvId2**

Associated value	Block parameter
1	BatchName
2	StepNo
3	BatchID
4	ExtVa204
5	ExtVa205
6	ExtVa206
7	Reserved
8	Reserved

Associated value	Block parameter
9	Reserved
10	Reserved

The associated values 4 to 6 are assigned to the parameters ExtVa204 to ExtVa206, and are usable. For more information on this, refer to the manual "*Process Control System PCS 7 - Engineering System*".

See also

Description of VlvPosL (Page 1488)

VlvPosL functions (Page 1495)

VlvPosL I/Os (Page 1515)

VlvPosL block diagram (Page 1527)

VlvPosL modes (Page 1493)

7.14.6 VlvPosL I/Os

I/Os of VlvPosL

Input parameters

Parameter	Description	Type	Default
AutModLi*	1= "Automatic mode" via interconnection or SFC (controlled via ModLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutModOp*	1 = "Automatic mode" via operator (controlled via ModLiOp = 1)	BOOL	0
AV	Input additional analog value, to be connected to AV_Tech of the AV block	ANY	
AV_AH_Lim	Limit high alarm	REAL	95.0
AV_AL_Lim	Limit low alarm	REAL	5.0
AV_Hyst	Hysteresis for alarm, warning and tolerance limits	REAL	1.0
AV_TH_Lim	Limit high tolerance	REAL	85.0
AV_TL_Lim	Limit low tolerance	REAL	15.0
AV_WH_Lim	Limit high warning	REAL	90.0
AV_WL_Lim	Limit low warning	REAL	10.0
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7 string	
BreakTime*	Minimum break duration [s]	REAL	1.0
BypProt	1 = Bypass interlock is enabled in "Local mode" and in simulation	BOOL	0

Parameter	Description	Type	Default
CloseAut*	1 = Select Close valve in "Automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseChnST	Signal status of the output channel of Close. It should be connected to an output channel block.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
CloseForce	1 = Force Close valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseLocal	1 = Select Close valve in "Local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CloseMan*	1 = Select Close valve in "Manual mode"	BOOL	0
CSF	1 = External fault (control system fault) Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EN	1 = Called block is executed	BOOL	1
ER_A_DC*	Delay for incoming alarms during monitoring of the manipulated variable difference	REAL	0.0
ER_A_DG*	Delay for outgoing alarms during monitoring of the manipulated variable difference	REAL	0.0
ER_AH_En	1 = Activate alarm (high) for manipulated variable difference monitoring	BOOL	1
ER_AH_Lim	Alarm limit (high) for manipulated variable difference monitoring	REAL	100.0
ER_AH_MsgEn	1 = Enable messages for alarm (high) for manipulated variable difference monitoring	BOOL	1
ER_AL_Lim	Alarm limit (low) for manipulated variable difference monitoring	REAL	-100.0
ER_AL_En	1 = Activate alarm (low) for manipulated variable difference monitoring	BOOL	1
ER_AL_MsgEn	1 = Enable messages for alarm (low) for manipulated variable difference monitoring	BOOL	1
ER_Hyst	Alarm hysteresis for manipulated variable difference monitoring	REAL	1.0
EventTsIn	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. Input parameter EventTsIn is used to interconnect output parameter EventTsOut of the EventTs, Event16TS block. When this interconnection is configured, the messages of the EventTs, Event16TS block are displayed and can also be acknowledged on the OS in the message view of the technological block.	ANY	

Parameter	Description	Type	Default
EventTs2In	For interconnecting data between a technology block and the message blocks EventTs, Event16Ts. The EventTs2In input parameter serves to interconnect the EventTsOut output parameter of the EventTs,Event16Ts block. When this interconnection is configured, the messages of the EventTs, Event16Ts block are displayed on the OS in the alarm view of the technology block and can also be acknowledged there.	ANY	
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg3	Binary input for freely selectable message 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg4	Binary input for freely selectable message 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvId1)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvId1)	ANY	
ExtVa106	Associated value 6 for messages (MsgEvId1)	ANY	
ExtVa204	Associated value 4 for messages (MsgEvId2)	ANY	
ExtVa205	Associated value 5 for messages (MsgEvId2)	ANY	
ExtVa206	Associated value 6 for messages (MsgEvId2)	ANY	
FaultExt	1 = External fault Error handling (Page 119)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkClose	1 = Valve closed feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkClosing	1 = Valve closing feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkOpen	1 = Valve open feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
FbkOpening	1 = Valve opening feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF

Parameter	Description	Type	Default
`Feature`	I/O for additional functions (Page 1495)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`Feature2`	I/O for additional functions (Page 1495)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`Gain`	Position control gain	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#FF
`Intlock`	0 = Interlock without reset is enabled; you can operate the block without reset once the interlock condition has cleared 1 = Interlock not effective	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
`Intl_En`	1 = Interlock without reset (interlock, `Intlock` parameter) is active	BOOL	1
`LocalLi`	1 = Activate "Local mode" via plant signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`LocalOp*`	1 = "Local mode" via operator	BOOL	0
`LocalSetting`	Properties for Local mode (Page 79)	INT	0
`Man*`	Manipulated variable in "manual mode"	REAL	0.0
`ManModLi*`	1 = "Manual mode" via interconnection or SFC (controlled via `ModLiOp` = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ManModOp*`	1= "Manual mode" via: OS operator (controlled via `ModLiOp` = 0)	BOOL	1
`ModLiOp`	Toggle operation between: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`MonSafePos`	1 = Go to neutral position in the event of monitoring errors	BOOL	1
`MotorTime*`	Motor actuating time [s]	REAL	30.0
`M_Monitor`	1 = Motor feedback monitoring	BOOL	1
`M_MonTiDynamic*`	Monitoring time for feedback errors or feedback start errors after successful operation in [s]	REAL	3.0
`M_MonTiStatic*`	Monitoring time for feedback errors without operation in [s]	REAL	3.0
`M_MonTiStop*`	Monitoring time for feedback stop errors after successful operation in [s]	REAL	3.0
`MS_RelOp*`	1= Release for maintenance via OS operator	BOOL	0
`MsgEvId1`	Message number (assigned automatically)	DWORD	16#00000000
`MsgEvId2`	Message number (assigned automatically)	DWORD	16#00000000

Parameter	Description	Type	Default
MsgLock	1 = Suppress process messages. For more on this, refer to the section Suppressing messages using the MsgLock parameter (Page 201).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV	Manipulated variable in "automatic mode"	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_Forced	Forced manipulated variable that is not limited and has top priority	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_ForOn	1 = Output forced manipulated variable MV_Forced unlimited at output MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_HiLim	Limit (high) for manipulated variable MV	REAL	100.0
MV_LoLim	Limit (low) for manipulated variable MV	REAL	0.0
MV_OpScale	OS display range for the manipulated variable MV	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
MV_Trk	Tracking value for manipulated variable MV	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
MV_TrkOn	1 = Tracking of manipulated variable MV	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NoFbkClose	1 = No feedback present for "valve closed"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NoFbkOpen	1 = No feedback present for "valve open"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = Occupied by a batch	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpenAut*	1 = Select Open valve in "Automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenChnST	Signal status of the output channel of Open. It should be connected to an output channel block.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF

Parameter	Description	Type	Default
OpenForce	1 = Force Open valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenLocal	1 = Select Open valve in "Local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpenMan*	1 = Select Open valve in "Manual mode"	BOOL	0
OpSt_In	Input parameter for local operator permission, to be connected with the Out output parameter of the upstream block OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for Operator control permissions (Page 248)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OS1Perm	I/O for Operator control permissions (Page 248)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
Permit	1 = Enable for opening/closing from neutral position 0 = Valve activation not enabled on OS	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Perm_En	1 = Activation enable (enable, Permit parameter) is active	BOOL	1
DeadBand	Deadband for forming manipulated variable difference	REAL	0.1
PosDiClose	Limit for control valve position "closed"	REAL	5.0
PosDiOpen	Limit for control valve position "open"	REAL	95.0
PosOnAut	1 = Select Activate position control in "Automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PosOnMan	1 = Select Activate position control in "Manual mode"	BOOL	0
Protect	0 = Protective interlock is in effect; once the interlock condition has cleared, you have to reset the block 1 = Protective interlock not activated	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Prot_En	1 = Protective interlock (protection, Protect parameter) is active	BOOL	1
PulseTime	Minimum pulse duration [s]	REAL	1.0
PulseWidth*	Pulse width of the control signal [s]	REAL	3.0
RapidStp*	Rapid stop for the motor 0 = Motor On 1 = Motor off	BOOL	0
Rbk	Position feedback	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
RbkHyst	Alarm hysteresis for position feedback	REAL	1.0
RbkMonitor	1 = Position feedback monitoring	BOOL	0
RbkMonTi	Monitoring time for the position feedback in [s]	REAL	5.0
RbkW_DC*	Delay time for incoming warnings [s]	REAL	0.0
RbkW_DG*	Delay time for outgoing warnings [s]	REAL	0.0
RbkWH_En	1 = Enable warning (high) for position feedback	BOOL	0
RbkWH_Lim	Limit for position feedback of warning (high)	REAL	90.0
RbkWH_MsgEn	1 = Enable messages for warning (high) for position feedback	BOOL	1
RbkWL_En	1 = Enable warning (low) for position feedback	BOOL	0
RbkWL_Lim	Limit for position feedback of warning (low)	REAL	10.0
RbkWL_MsgEn	1 = Enable messages for warning (low) for position feedback	BOOL	1
RstLi*	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SafePos	Neutral position for valve: 0 = Closed 1 = Open 2 = stop	INT	2
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	Call a block saved in this parameter as an Opening additional faceplates (Page 203) in the standard view	ANY	-
SelFp2	Call a block saved in this parameter as an Opening additional faceplates (Page 203) in the preview	ANY	-
SimAV*	Additional value used with SimOn = 1	REAL	0.0
SimAV_Li	Additional analog value used with SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimOnLi	1 = Simulation via interconnection or SFC (controlled via SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimRbk*	Position feedback used with SimOn = 1	REAL	0.0
SimRbkLi	Position feedback used with SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
StepNo	Batch step number	DWORD	16#00000000

Parameter	Description	Type	Default
StopAut*	1 = Stop the motor in "Automatic mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopForce	1 = Force motor stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopLocal	1 = Stop the motor in "Local mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StopMan*	1 = Stop the motor in "Manual mode"	BOOL	0
TmLag	Delay time of the LeadLag filter [s]	REAL	0.0
TorqClose	0 = Torque shutdown active when closing 1 = Good state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
TorqOpen	0 = Torque shutdown active when opening 1 = Good state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
Trip	1 = Motor is in "good" state	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#FF
UA1unit	Unit of measure for analog auxiliary value 1	INT	0
UA2unit	Unit of measure for analog auxiliary value 2	INT	0
UserAna1	Analog auxiliary value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UserAna2	Analog auxiliary value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
V_Monitor	1 = Valve feedback monitoring	BOOL	1
V_MonTiDynamic*	Valve monitoring time after operation in [s]	REAL	30.0
V_MonTiStatic*	Monitoring time for valve feedback errors without operation in [s]	REAL	5.0
WarnTiAut*	Prewarning of valve movement in "automatic mode" in [s]	REAL	0.0
WarnTiMan*	Prewarning of valve movement in "manual mode" in [s]	REAL	0.0

* Values can be written back to these inputs during execution of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AutAct	1 = "Automatic mode" enabled 0 = "Manual mode" is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AutoST	1 = Worst signal status of automatic commands	BYTE	16#80
AV_OpScale	Limit for scale in AV bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
AV_Out	Output additional analog value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
AV_Unit	Unit of measure for additional analog value	INT	0
CascaCut	Cascade connection: 1 = Control chain from master controller to secondary valve is interrupted	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Close	Control output 1= Close valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Closed	1 = Valve is closed	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Closing	1 = Valve is closing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CurrMonM	Current motor monitoring time [s]	DINT	0
CurrMonRbk	Current monitoring time of position feedback [s]	DINT	0
CurrMonV	Current valve monitoring time [s]	DINT	0
ENO	1 = Block algorithm completed without errors	BOOL	0
ER	Difference of the manipulated variable and the position feedback (ER = MV_Out - RbkOut)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ER_AH_Act	1 = Alarm limit (high) for manipulated variable difference violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ER_AL_Act	1 = Alarm limit (low) for manipulated variable difference violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see the section VlvPosL error handling (Page 1510)	INT	-1

Parameter	Description	Type	Default
FbkCloseOut	Valve closed feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkClsgOut	Valve closing feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpenOut	Valve open feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FbkOpngOut	Valve opening feedback	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GrpErr	1 = Group error pending	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LocalAct	1 = "Local mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LockAct	1 = Interlock (Intlock, Permit, Protect) or Trip is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManAct	1 = "Manual mode" enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
M_MonDynErr	1 = Feedback error or feedback start error of the motor due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
M_MonStaErr	1 = Motor feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
M_MonStopErr	1 = Feedback stop error of the motor due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Release	Release for maintenance: 1 = Enable for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1	Message - acknowledgment status 1 (ACK_STATE output of the first ALARM_8P)	WORD	16#0000
MsgAckn2	Message - acknowledgment status 2 (ACK_STATE output of the first ALARM_8P)	WORD	16#0000
MsgErr1	Message error 1 (ERROR output of the first ALARM_8P)	BOOL	0

Parameter	Description	Type	Default
MsgErr2	Message error 2 (ERROR output of the first ALARM_8P)	BOOL	0
MsgStat1	Message status 1 (STATUS output of the first ALARM_8P)	WORD	16#0000
MsgStat2	Message status 2 (STATUS output of the first ALARM_8P)	WORD	16#0000
MV_HiAct	1 = Limit (high) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_LoAct	1 = Limit (low) of manipulated variable violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MV_Aut	Manipulated variable in automatic mode	REAL	0.0
MV_Out	Manipulated variable	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Open	Control output: 1 = Open the valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Opened	1 = Valve is open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Opening	1 = Valve is opening	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the input parameter OpSt_In for further interconnection with other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS1PermOut	Display of OS1Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS1PermLog	Display of OS1Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
PosReached	1 = Control valve has reached specified position	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Close	1 = Pulse signal to close valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
P_Open	1 = Pulse signal to open valve	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Rst	1= Pulse output for reset The parameter persists for one cycle after a reset.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
P_Stop	0 = Pulse signal for stopping the valve	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
R_StpAct	1 = Rapid stop of the motor is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkMonDynErr	1 = Position feedback error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkMonStaErr	1 = Position feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkOut	Position feedback output	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RbkWH_Act	1 = Warning (high) enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkWL_Act	1 = Warning (low) enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToReset	1 = Ready for reset via RstLi input or commands in "local", "automatic", or "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RdyToStart	1 = Ready to start	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1488)	DWORD	16#00000000
Status2	Status word 2 (Page 1488)	DWORD	16#00000000
Status3	Status word 3 (Page 1488)	DWORD	16#00000000
Status4	Status word 4 (Page 1488)	DWORD	16#00000000
Status5	Status word 5 (Page 1488)	DWORD	16#00000000

Parameter	Description	Type	Default
`Stop`	1 = Motor stopped and valve is in intermediate position	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SumMsgAct`	1 = Active hardware interrupt	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`ThreshOff`	Trip threshold of the three-step element	REAL	0.0
`ThreshOn`	Response threshold of the three-step element	REAL	0.0
`V_MonDynErr`	1 = Valve feedback error due to control change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`V_MonStaErr`	1 = Valve feedback error due to unexpected feedback change	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`WarnAct`	1 = Prewarning for control valve movement away from active (`WarnTiAut` and `WarnTiMan` parameters)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

VlvPosL messaging (Page 1513)

VlvPosL block diagram (Page 1527)

VlvPosL modes (Page 1493)

Error handling (Page 119)

Disabling operating points (Page 144)

Signaling limit violation (Page 169)

7.14.7 VlvPosL block diagram

Block diagram of VlvPosL

A block diagram is not provided for this block.

See also

VlvPosL I/Os (Page 1515)

VlvPosL messaging (Page 1513)

VlvPosL error handling (Page 1510)

VlvPosL functions (Page 1495)

VlvPosL modes (Page 1493)

Description of VlvPosL (Page 1488)

7.14.8 Operator control and monitoring

7.14.8.1 VlvPosL views

Views of the VlvPosL block

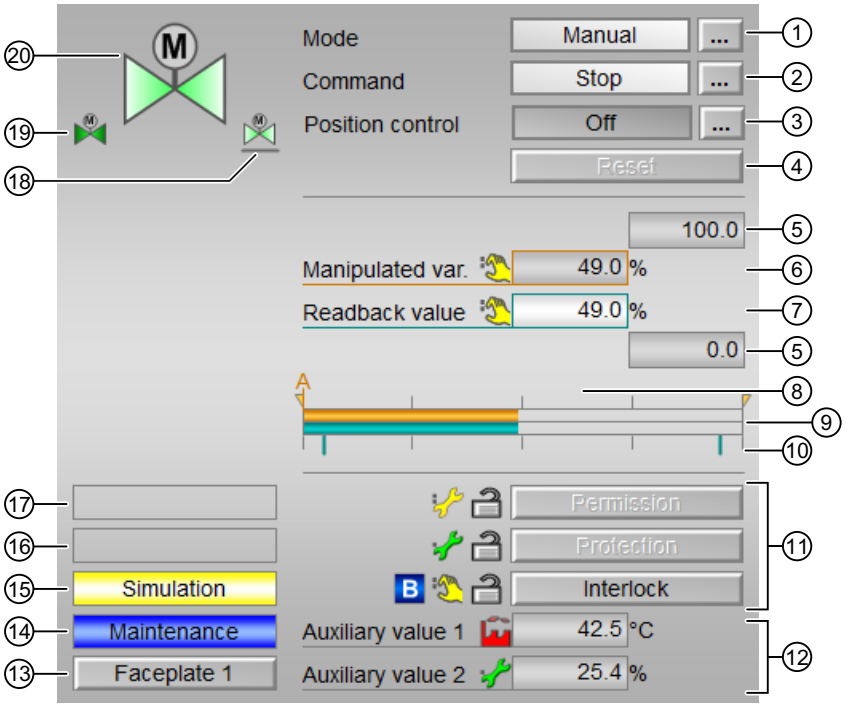
The VlvPosL block provides the following views:

- VlvPosL standard view (Page 1529)
- VlvPosL limit view (Page 1533)
- Alarm view (Page 296)
- Trend view (Page 299)
- VlvPosL parameter view (Page 1535)
- VlvPosL preview (Page 1539)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for VlvPosL (Page 1543)

You can find general information about the faceplate and block icon in the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226).

7.14.8.2 VlvPosL standard view

Standard view of VlvPosL



(1) Display and switch the operating mode

This area displays the current operating mode of the block. The following operating modes can be shown here:

- Manual and automatic mode for motors, valves and dosers (Page 75)
- Local mode (Page 79)
- Out of service (Page 71)

For information on switching the mode, see section Switching operating states and operating modes (Page 251).

(2) Open, close and stop the motor valve

This area displays the default operating state for the motor valve. The following states can be shown and executed here:

- "Open"
- "Close"
- "Stop"
- "Rapid stop"

Analog positioning mode is terminated if one of these commands is executed. Positioning mode can be reactivated using the **(3)** button.

For information on switching the mode, see section Switching operating states and operating modes (Page 251).

If text is configured for these commands, it is displayed as status text and as button labels for command selection. You can find more information on this in the section Labeling of buttons and text (Page 205).

(3) Activate positioning at an analog manipulated variable

This area shows you the default operating state for positioning to the analog manipulated variable. The following states can be shown here:

- "On"
- "Off"

Only the positioning mode is activated here:

- "On"

Positioning mode is terminated when the motor valve is operated using the **(2)** button.

(4) Reset the block

Click "Reset" in the event of interlocks or errors. You can find more information on this in the section Resetting the block in case of interlocks or errors (Page 43).

(5) High and low scale range for the manipulated variable

These values provide information on the display range for the bar graph of the manipulated variable. The scale range is defined in the engineering system.

(6) Display and change the manipulated variable including signal status

This area shows the current manipulated variable with the corresponding signal status.

The manipulated variable can only be changed in manual mode and active position control.

You can find more information about changing the manipulated variable in the section Changing values (Page 253).

(7) Display for position feedback including signal status

This area shows the current feedback of the manipulated variable with the corresponding signal status.

(8) Display of the manipulated variable for automatic mode

The display [A] is only visible when the block is not in automatic mode. In Automatic mode, it is displayed when it needs to be reset or when tracking or forced tracking is enabled.

This display shows the manipulated variable that would apply if you were to switch to automatic mode or if you were to reset in automatic mode or disable tracking or forced tracking.

(9) Bar display for manipulated variable

This area shows the current manipulated variable in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(10) Bar display for position feedback

This area shows the current position feedback in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

The limits for the "open" and "closed" positions are shown with 2 green lines.

(11) Area for operating the interlock functions of the block

This display is only visible when the corresponding block input is interconnected.

You can use these buttons to control the interlock functions of the block. You can find more information on this in the section Interlocking functions (Page 99).

The following is displayed in addition to the buttons:

- Interlock status (see Forming the group status for interlock information (Page 104)), for example:

- Signal status (see Forming and outputting the signal status for technologic blocks (Page 108)), for example:

- Bypass information (see Forming the group status for interlock information (Page 104)):

(12) Display for auxiliary values

This display is only visible when the corresponding block input is interconnected.

You can use this area to display two auxiliary values that have been configured in the engineering system. You can find more information on this in the section Displaying auxiliary values (Page 207).

(13) Jump key to standard view of any faceplate

This display is only visible when the corresponding block input is interconnected.

Use this navigation button to jump to the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find more information on this in the section Opening additional faceplates (Page 203).

(14) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

You can find more information on this in the section Release for maintenance (Page 64) display area for block states.

(15) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"
- "Time delay"

You can find more information on this in the sections Simulating signals (Page 58) and Display of delay times (Page 250).

(16) Display area for block states

This area provides additional information on the operating state of the block:

- "Motor protection"
- "External error"
- "Vibrate enabled"
- "End position error"
- "Control error"
- "Invalid signal"
- "Changeover error"
- "Torque active"

You can find more information on this in the sections Monitoring the feedbacks (Page 97), Error handling (Page 119) (headings "Invalid input signals" and "Mode switchover error") and Motor protection function (Page 99).

(17) Display area for block states

This area provides additional information on the operating state of the block:

- "Forced open" (`OpenForce`)
- "Forced closed" (`CloseForce`)
- "Forced stop" (`StopForce`)
- "Forced tracking" (`MV_ForOn`)

- "Tracking" (`MV_TrkOn`)
- "Request 0/1": A reset to "automatic mode" is expected.

You can find more information on this in the section Forcing operating modes (Page 41).

(18) Neutral position of the valve

This representation shows the neutral position for the valve:

- **Green**: Neutral position is "Open"
- **Gray**: Neutral position is "Closed"
- **Light green**: Neutral position is "Stop"

(19) Automatic preview

This display is only visible in "Manual mode", in "Local mode", or with a reset request in "Automatic mode", when the current output signals are not identical to the control in "Automatic mode".

The display shows what state the valve would assume if you switched from "Manual" or "Local" mode to "Automatic mode", or performed a reset to "Automatic mode".

(20) Status display of the motor valve

The current status of the motor valve is graphically displayed here.

You can find more information on this in the section Block icon for VlvPosL (Page 1543).

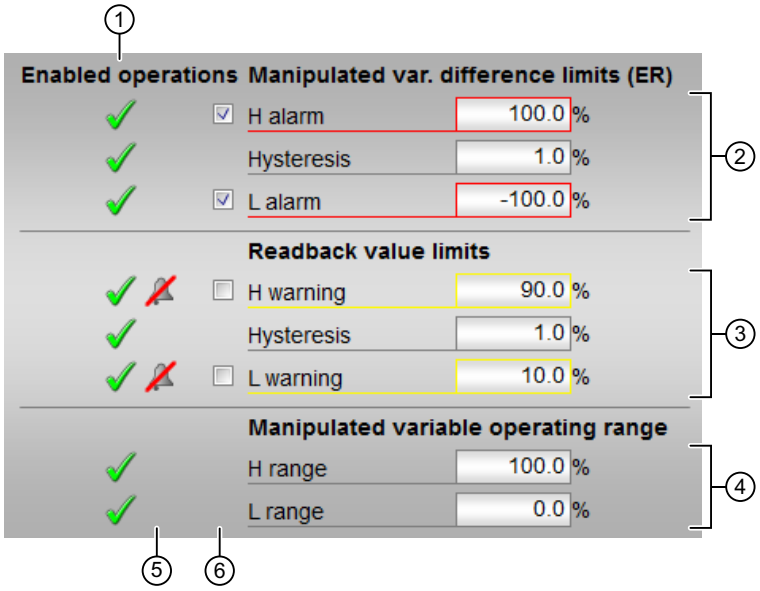
7.14.8.3 VlvPosL limit view

Limit view of VlvPosL

Several values are set in this view by default:

- Manipulated variable difference limits
- Readback value limits
- Manipulated variable operating range

The toolbar of the faceplate and the block icon indicate when the limits are reached or violated.



(1) Enabled operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for operation enable:

• **Green check mark**: The OS operator can control this parameter

• **Gray check mark**: The OS operator cannot control this parameter at this time due to the process

• **Red cross**: The OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

(2) Limits of the manipulated variable difference

In this area, you can enter the limits for the manipulated variable difference. For more on this, see section Changing values (Page 253).

You can change the following limits:

- "H alarm": Alarm high
- "Hysteresis"
- "L alarm": Alarm low

(3) Readback value limits (`Rbk`)

In this area, you can enter the limits for the readback value (position feedback). For more on this, see section Changing values (Page 253).

You can change the following limits:

- "H warning": Warning high
- "Hysteresis"
- "L warning": Warning low

(4) Manipulated variable operating range

In this area, you can enter the limits for the manipulated variable operation range. For more on this, see section Changing values (Page 253).

You can change the following limits:

- "H range": Range limit high
- "L range": Range limit low

(5) Message suppression / delay

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when you install the block (all `xx_MsgEn` parameters are assigned the default value 1). Messages can only be outputted if limit monitoring of the additional analog value has been enabled.

Alarm delays are also displayed in this position; for more on this, see section Area of application of the alarm delays (Page 195).

(6) Suppress messages

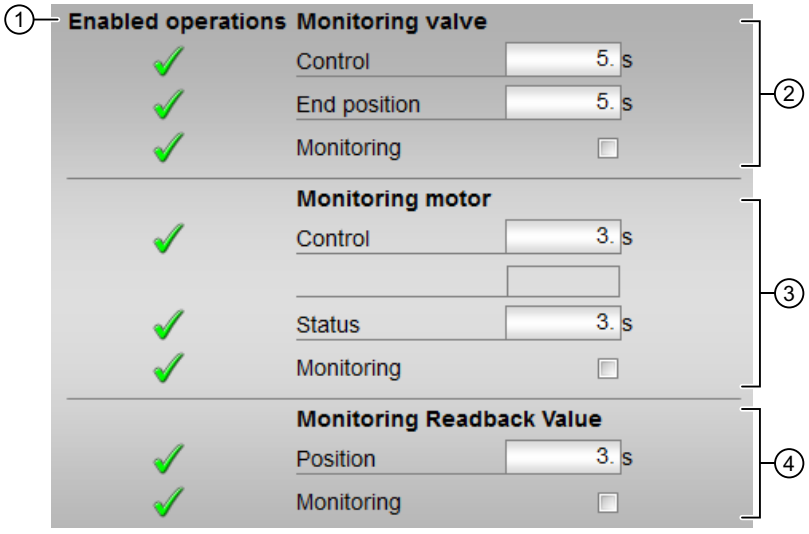
You can enable/disable messages by setting the check mark.

7.14.8.4 VlvPosL parameter view

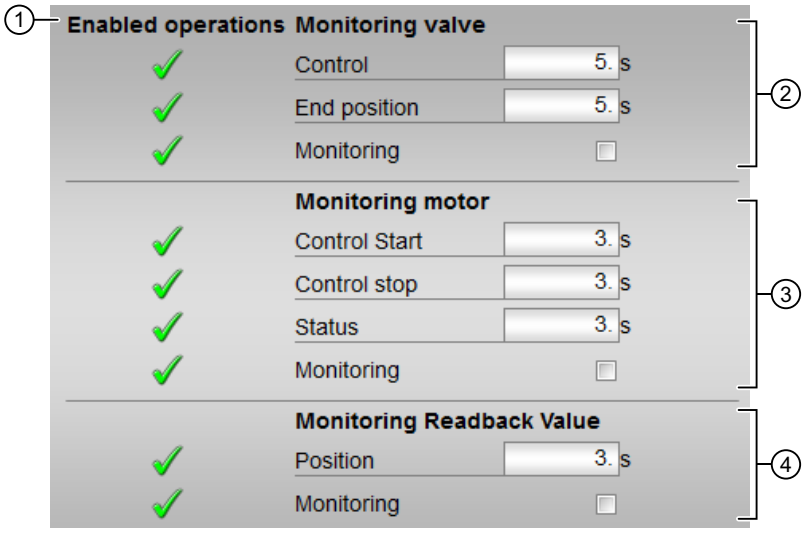
Parameter view of VlvPosL

The parameter has an upper half and a lower half. You can navigate between the two halves with the arrow keys.

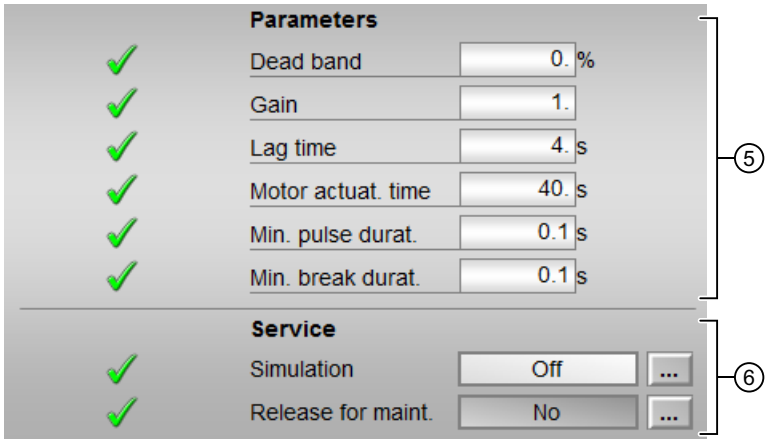
Upper half of the parameter view of VlvPosL with `Feature.Bit13` = 0:



Upper half of the parameter view of VlvPosL with `Feature.Bit13` = 1:



Lower half of the parameter view of VlvPosL:



(1) Enabled operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for operation enable:

- **Green check mark**: The OS operator can control this parameter.
- **Gray check mark**: The OS operator cannot control this parameter at this time due to the process.
- **Red cross**: The OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

(2) Monitor valve

In this area, you can change parameters and thereby influence the valve. You can find more information on this in the section Changing values (Page 253).

You can influence the following parameters:

- "Control": Monitoring time while "Opening"/"Closing" the valve (dynamic)
- "End position": Monitoring time for maintaining the valve position (static)

Enable monitoring

You can enable monitoring by selecting the check box (☑).

You can find more information on this in the section Monitoring the feedbacks (Page 97).

(3) Monitor motor

In this area, you change parameters and thereby influence the motor. You can find more information on this in the section Changing values (Page 253).

You can influence the following parameters:

- "Control": Monitoring time during startup and stopping of the motor (dynamic) `Feature.Bit13` = 0
- "Control start": Monitoring time during startup of the motor (dynamic) `Feature.Bit13` = 1
- "Control stop": Monitoring time during stopping of the motor (dynamic) `Feature.Bit13` = 1
- "Status": Monitoring time during permanent operation of the motor (static)

Enable monitoring

You can enable monitoring by selecting the check box (☑).

You can find more information on this in the section Monitoring the feedbacks (Page 97).

(4) Monitoring the valve position

In this area, you change parameters and thereby influence the valve. You can find more information on this in the section Changing values (Page 253).

You can influence the following parameter:

- "Position": Monitoring time for the valve

Enable monitoring

You can enable monitoring by selecting the check box (☑).

You can find more information on this in the section Monitoring the feedbacks (Page 97).

(5) "Parameters"

In this area, you change parameters and thereby influence the controller. For more on this, see section Changing values (Page 253).

You can influence the following parameters:

- "Deadband": Width of deadband
- "Gain": Gain
- "Delay": Delay time in [s]
- "Motor actuating time": Motor actuating time [s]
- "Minimum pulse duration": Minimum pulse duration [s]
- "Minimum break duration": Minimum break duration [s]

(6) Service

You activate the following functions in this area:

- "Simulation"
- "Release for maintenance" (with display for maintenance demanded)

You can find more information on this in the section Switching operating states and operating modes (Page 251).

You can find information on this area in the sections:

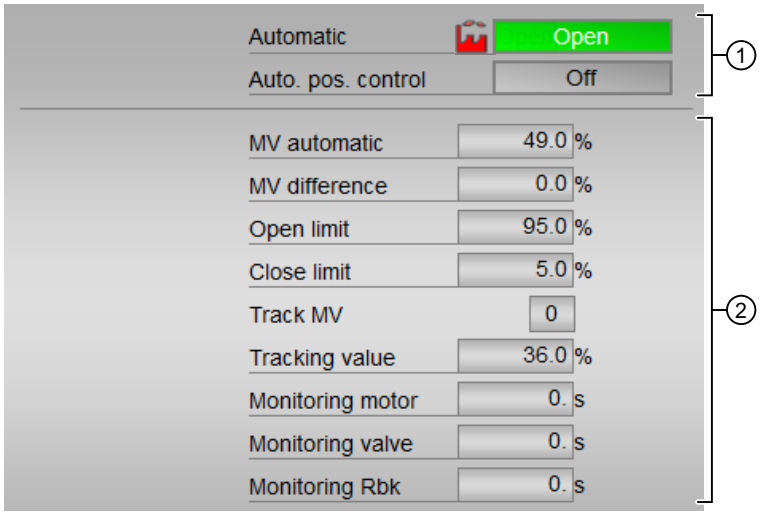
- Simulating signals (Page 58)
- Release for maintenance (Page 64)

7.14.8.5 VlvPosL preview

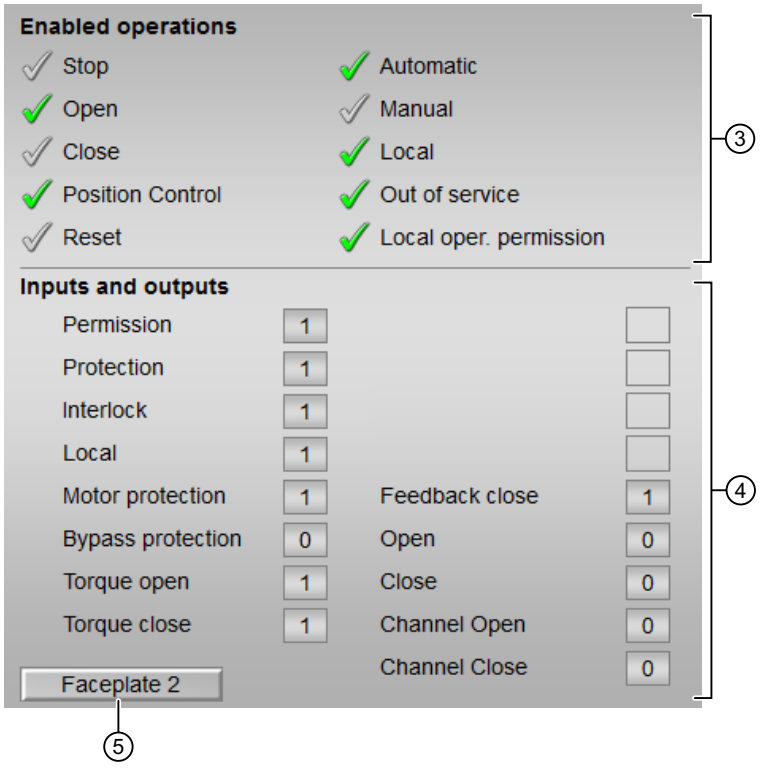
Preview of VlvPosL

The preview has an upper half and a lower half. You can navigate between the two halves with the arrow keys.

Upper half of the preview of VlvPosL:



Lower half of the preview of VlvPosL:



(1) Automatic preview

This area shows you the block status after it has switched from the "Manual" mode or "Local" mode to the "Automatic" mode.

If the block is in "Automatic" mode, the current block state is displayed.

This area also displays the worst signal status of the following automatic commands:

- `OpenAut`
- `CloseAut`
- `StopAut`
- `PosOnAut`
- `MV`

(2) Preview area

- "MV Automatic": Display of the current automatic manipulated variable (`MV`).
- "MV difference": Current manipulated variable difference (`ER`).
- "Open limit": Limit (`PosDiOpen`) for forming the "Control valve open" signal (`FbkOpenOut`). If the position feedback reaches this limit, the control valve is open.
- "Close limit": Limit (`PosDiClose`) for forming the "Control valve closed" signal (`FbkCloseOut`). If the position feedback reaches this limit, the control valve is closed.

- "Manipulated variable tracking": (`MV_TrkOn` = 1) Manipulated variable is tracked to the tracking value.
- "Tracking value": Effective manipulated variable for "Track manipulated variable"
- "Monitoring motor": Display of the current monitoring time of the motor.
  This display is only visible when monitoring is activated.
- "Monitor valve": Display of the current monitoring time of the valve.
  This display is only visible when monitoring is activated.
- "Monitoring Rbk": Display of the current monitoring time for the position feedback.
  This display is only visible when monitoring is activated.

(3) Enabled operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for operation enable:

- **Green check mark**: The OS operator can control this parameter.
- **Gray check mark**: The OS operator cannot control this parameter at this time due to the process.
- **Red cross**: The OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

The following enabled operations are shown here:

- "Stop": You can stop the motor of the valve.
  If text is configured for this command, it is also displayed in brackets. You can find more information on this in the section Labeling of buttons and text (Page 205).
- "Open": You can open the motor valve.
  If text is configured for this command, it is also displayed in brackets. You can find more information on this in the section Labeling of buttons and text (Page 205).
- "Close": You can close the motor valve.
  If text is configured for this command, it is also displayed in brackets. You can find more information on this in the section Labeling of buttons and text (Page 205).
- "Position Control": You can switch on the position control.
- "Reset": You can reset the motor valve if interlocks or errors occur.
- "Automatic": You can switch to "Automatic mode".
- "Manual": You can switch to "Manual mode".
- "Local": You can switch to "Local mode".
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to change to the standard view of the OpStations block. You can find more information on this in the section Operator control permissions (Page 248).

(4) Display current control signals

This area shows the most important parameters for this block with the current selection:

- "Permission":
  This display is only visible when the corresponding block input is interconnected.
  - 0 = Motor valve activation not enabled on OS
  - 1 = Enable for "opening"/"closing" from the neutral position
- "Protection":
  This display is only visible when the corresponding block input is interconnected.
  - 0 = Protective interlock is in effect; once the interlock condition has cleared, you must reset the block
  - 1 = Good state
- "Interlock":
  This display is only visible when the corresponding block input is interconnected.
  - 0 = Interlock without reset is enabled; you can operate the block without reset once the interlock condition has cleared
  - 1 = Good state
- "Local correct": 1 = Control signal for "Local mode" (`LocalLi`) is active
- "Interlock deact.":
  - 0 = Bypass disabled
  - 1 = Bypass interlock in "Local mode" and in simulation
- "Torque opening": 0 = Torque shutoff when opening
- "Torque closing": 0 = Torque shutoff when closing
- "Local stop": 1 = Stop the motor valve in "Local mode"
- "Local open": 1 = Open the motor valve in "Local mode"
- "Local close": 1 = Close the motor valve in "Local mode"
- "Feedback open": 1 = Motor valve is opened
- "Feedback closed": 1 = Motor valve is closed
- "Open": 1 = Motor valve is opening
- "Close": 1 = Motor valve is closing
- "Channel Open": Signal from the output channel block for "Open"
- "Channel Close": Signal from the output channel block for "Close"

(5) Jump key to standard view of any faceplate

This display is only visible when the corresponding block input is interconnected.

Use this navigation button to jump to the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find more information on this in the section Opening additional faceplates (Page 203).

7.14.8.6 Block icon for VlvPosL

Block icons for VlvPosL

A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits as well as control system faults
- Operating modes
- Signal status, release for maintenance
- Forcing states
- Operating the manipulated variable in manual mode
- Display of active position control
- Display of the position feedback (white, with decimal places)
- Display for bypassing interlocks
- Interlocks
- Memo display
- Valve status display

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Non-rotating block icon
	4	Non-rotating block icon

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234)

Valve status display

The following valve states are shown here:

Icon	Meaning
	Valve open
	Valve closed
	Error at valve
	Valve is opening
	Valve is closing
	Valve is stopping
	Valve closed
	Valve is closing

Interlock blocks

8

8.1 Intlk02 - Interlock display with 2 input signals

8.1.1 Description of Intlk02

Object name (type + number) and family

Type + number: FB 1824

Family: Interlck

Area of application for Intlk02

The block is used for the following applications:

- Standardized interlock with display

How it works

The block is used to calculate a standardized interlock that can be displayed on the OS. A maximum of 2 input signals can be supplied to the block. They are linked using selectable binary logic. The signal status of the output signal is also determined. You can assign an analog value with the signal status and unit to each input value for display in the faceplate.

The current state is displayed at the `Out` output parameter:

- `Out` = 0: Interlock
- `Out` = 1: "Good" state

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

For the Intlk02 block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Dosing (Dose_Lean) (Page 2332)
- Dosing with PA/FF devices (Dose_Lean_Fb) (Page 2333)
- Two-speed motor (Motor2Speed) (Page 2336)
- Reversing motor (MotorReversible) (Page 2337)
- Reversible motor with controllable speed (MotorSpeedControlled) (Page 2338)
- Two-way valve (Valve2Way) (Page 2342)

- Motor valve (ValveMotor) (Page 2343)
- Control valve (VlvAnL) (Page 2344)
- Control valve for PA/FF devices (ValveAnalog_Fb) (Page 2345)

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section Intlk02 I/Os (Page 1552).

Status bit	Parameter
0	1 = `Logic` = OR
1	1 = `Logic` = AND
2	Not used
3	Result of logic operation `Out.Value`
4	1 = At least one input value is excluded. See the section Excluding input values in Intlk02 functions (Page 1548).
5	1 = All input values are excluded
6	1 = No input value is interconnected or `NotUsed.Value`
7	Status display for simulation
8	Status display for not interlocked
9	Status display for interlocked
10	Activate OS_Perm bits
11	`Feature` bit 2: Separate evaluation for excluded and simulated interlock signals.
12 - 31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`In01.Value`
1	`In02.Value`
2 - 31	Not used

Status word allocation for `Status3` parameter

Status bit	Parameter
0	`InvIn01`
1	`InvIn02`
2 - 31	Not used

Status word allocation for `Status4` parameter

Status bit	Parameter
0	`In01` with inversion
1	`In02` with inversion
2 - 15	Not used
16	Bypass `In01` (via interconnection)
17	Bypass `In02` (via interconnection)
18 - 31	Not used

Status word allocation for `Status5` parameter

Status bit	Parameter
0	`BypIn01`
1	`BypIn02`
2 - 15	Not used
16	In01 hidden bypass information
17	In02 hidden bypass information
18 - 31	Not used

Status word allocation for `Status6` parameter

Status bit	Parameter
0	`In01` not connected
1	`In02` not connected
2 - 31	Not used

Status word allocation for `Status7` parameter

Status bit	Parameter
0	`AV01` not connected
1	`AV02` not connected
2 - 31	Not used

Status word allocation for `Status8` parameter

Identical to `FirstIn`.

See also

Intlk02 messaging (Page 1552)

Intlk02 block diagram (Page 1555)

Intlk02 error handling (Page 1551)

Intlk02 modes (Page 1548)

8.1.2 Intlk02 modes

Intlk02 modes

This block does not have any modes.

See also

Intlk02 block diagram (Page 1555)

Intlk02 I/Os (Page 1552)

Intlk02 messaging (Page 1552)

Intlk02 error handling (Page 1551)

Intlk02 functions (Page 1548)

Description of Intlk02 (Page 1545)

8.1.3 Intlk02 functions

Functions of Intlk02

The functions for this block are listed below.

Handling non-connected inputs

Inputs which are not interconnected are not evaluated. They are not displayed in the faceplate either.

Special situation: If no inputs are connected, the output value is defined using the `DefaultOut` parameter. The signal status is set to 16#FF.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203). However, only one additional faceplate can be called up using `SelFp1 = 1`.

First-in detection for interlock blocks

This block provides the standard function Recording the first signal for interlock blocks (Page 52). Note that a signal status change only has an effect on the first-in detection when the `Feature` Bit Evaluation of signal status (Page 141) and the input `FirstInEn` is set.

The response to deactivation via the `FirstInEn` input can be influenced by Feature bit First-in detection response to deactivation (Page 175).

Note

This function can only be executed in the faceplate with "process control" operating permission.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for interlock blocks (Page 115).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `Out.ST`

Bypass

Note

Bypassing the interlock means that the interlock signal (input signal) is excluded from the logic of the interlock block, in other words, this signal is ignored in the logic operation.

This function can only be executed in the faceplate with "high-level operating permission".

You can exclude input signals that are temporarily not to be used for the calculation in the block by setting the corresponding I/O:

- Independent `BypLix` is connected or not, the operator can bypass the input signal: `BypInx` = 1 (from operator over faceplate)
- `BypLix.ST` <> 16#FF (input is connected) and a change in `BypLix` value is detected: `BypLix` = 1 (from connection in CFC)

The I/O is shown in the faceplate by the following symbol:

Special situation: If all input parameters are excluded, the output value is defined using the `DefaultOut` parameter.

Depending on the configuration of the `Feature` bit Separate evaluation for excluded and simulated interlock signals (Page 151), the output "Bypass" is formed and the hidden bypass signal is set at the output `Out`.

Feature Bit =0:

If one of the interlock inputs is excluded (`BypIn01`...`BypIn0x` or `BypLi01`...`BypLix`), the output `BypAct` is set. The hidden bypass signal at the output `Out` is reset.

- `BypAct.Value` =
  `BypIn01` OR `BypLi01` (if `BypLi01` is connected and the value has changed)
  OR `BypInx` OR `BypLix` (if `BypLix` is connected and the value has changed)
- `Out.Bit1` = 0

An excluded interlock input that is switch-relevant after the bypass, sets the status of `Out` to Simulation.

Feature Bit =1:

If one of the interlock inputs is excluded (`BypIn01`...`BypIn0x` or `BypLi01`...`BypLix`) or a hidden signal is set at an interlock input (`In01`...`Inx`), the output `BypAct` is set. The hidden bypass signal at the input `In01`...`Inx` can be read from the upstream interlock block at the output `BypAct`. The hidden bypass signal at the output `Out` is set to the value of `BypAct`.

- `BypAct.Value` =
  `BypIn01` OR `BypLi01` (if `BypLi01` is connected and the value has changed)
  OR `BypInx` OR `BypLix` (if `BypLix` is connected and the value has changed)
  OR `In01.Bit1` OR `Inx.Bit1` (hidden lines)
- `Out.Bit1` = `BypAct.Value`

An excluded interlock input has no influence on the status of `Out`.

Note

Do **not** connect the input `BypLix` with the output `BypAct` from an interlock block because `BypAct` is calculated from the hidden bits.

Operating permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can exclude values
1	1 = Operator can reset the exclusion of input values
2	1 = Operator can reset the recording of the first signal
3 - 15	Not used
16	1 = Operator can set or reset the exclusion of input value In01
17	1 = Operator can set or reset the exclusion of input value In02
18 – 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the chapter Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
5	Activate OS_Perm bits (Page 156) 0 = OS_Perm bits 16...17 inactive (evaluation only in faceplate) 1 = OS_Perm bits 16...17 active (evaluation only in faceplate)
21	First-in detection response to deactivation (Page 175)
23	Evaluation of signal status (Page 141)
24	Enabling local operator authorization (Page 157)
31	Activating recording of the first signal (Page 149)

See also

Description of Intlk02 (Page 1545)

Intlk02 messaging (Page 1552)

Intlk02 I/Os (Page 1552)

Intlk02 block diagram (Page 1555)

Intlk02 error handling (Page 1551)

Intlk02 modes (Page 1548)

8.1.4 Intlk02 error handling

Error handling of Intlk02

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
13	The parameter `Logic` has not been set to 0 or 1.

See also

Intlk02 block diagram (Page 1555)
Intlk02 I/Os (Page 1552)
Intlk02 messaging (Page 1552)
Intlk02 functions (Page 1548)
Intlk02 modes (Page 1548)
Description of Intlk02 (Page 1545)

8.1.5 Intlk02 messaging

Messaging

This block does not offer messaging.

See also

Description of Intlk02 (Page 1545)
Intlk02 functions (Page 1548)
Intlk02 I/Os (Page 1552)
Intlk02 block diagram (Page 1555)
Intlk02 error handling (Page 1551)
Intlk02 modes (Page 1548)

8.1.6 Intlk02 I/Os

I/Os of Intlk02

Input parameters

Parameter	Description	Type	Default
AV01	Analog value of In01	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV02	Analog value of In02	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV01_Unit	Unit of measure for AV01	INT	0
AV02_Unit	Unit of measure for AV02	INT	0
BypIn01*	1 = Input In01 is not used	BOOL	0
BypIn02*	1 = Input In02 is not used	BOOL	0
BypLi01	Bypass In01 (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
BypLi02	Bypass In02 (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
DefaultOut	Output value for cases where all inputs are excluded or not interconnected. Refer to the section Intlk02 functions (Page 1548) of the block.	BOOL	1
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1548)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
FirstInEn	First-in detection	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In01	Input In01	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In02	Input In02	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
InvIn01	Invert input In01	BOOL	0
InvIn02	Invert input In02	BOOL	0
Logic	Logical operation: 0 = Logical OR 1 = Logical AND	INT	0

Parameter	Description	Type	Default
NotUsed	1 = Block is not used (only for display in the faceplate). You can find additional information on this in the section Interlock blocks standard view (Page 270) / Color of the field.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OS_Perm	I/O for operating permissions (Page 1548)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
RstBypOp*	1 = Reset via BypIn01and BypIn02	BOOL	0
RstLi	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Data type	Default
BypAct	1 = Bypass active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see Intlk02 error handling (Page 1551).	INT	-1
FirstIn	Bit-coded number of the first signal that has resulted in a change at the output	DWORD	16#00000000
OpSt_Out	Value of the OpSt_In input parameter for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF

Parameter	Description	Data type	Default
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1545)	DWORD	16#00000000
Status2	Status word 2 (Page 1545)	DWORD	16#00000000
Status3	Status word 3 (Page 1545)	DWORD	16#00000000
Status4	Status word 4 (Page 1545)	DWORD	16#00000000
Status5	Status word 5 (Page 1545)	DWORD	16#00000000
Status6	Status word 6 (Page 1545)	DWORD	16#00000000
Status7	Status word 7 (Page 1545)	DWORD	16#00000000
Status8	Status word 8 (Page 1545)	DWORD	16#00000000

See also

Intlk02 messaging (Page 1552)

Intlk02 block diagram (Page 1555)

Intlk02 modes (Page 1548)

8.1.7 Intlk02 block diagram

Intlk02 block diagram

A block diagram is not provided for this block.

See also

Intlk02 I/Os (Page 1552)

Intlk02 messaging (Page 1552)

Intlk02 error handling (Page 1551)

Intlk02 functions (Page 1548)

Intlk02 modes (Page 1548)

Description of Intlk02 (Page 1545)

8.1.8 Operator control and monitoring

8.1.8.1 Interlock block views

Views of the blocks Intlk02, Intlk04, Intlk08, Intlk16

The blocks provide the following views:

- Interlock blocks standard view (Page 270)
- Preview of interlock blocks (Page 293)
- Block icon for interlock blocks (Page 237)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

8.2 Intlk04 - Interlock display with 4 input signals

8.2.1 Description of Intlk04

Object name (type + number) and family

Type + number: FB 1825

Family: Interlck

Area of application for Intlk04

The block is used for the following applications:

- Standardized interlock with display

How it works

The block is used to calculate a standardized interlock that can be displayed on the OS. A maximum of 4 input signals can be supplied to the block. They are linked using selectable binary logic. The signal status of the output signal is also determined. You can assign an analog value with the signal status and unit to each input value for display in the faceplate.

The current state is displayed at the `Out` output parameter:

- `Out` = 0: Interlock
- `Out` = 1: "Good" state

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

For the Intlk02 (Intlk04) block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Refer to Description of Intlk02 (Page 1545) for more information.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section Intlk04 I/Os (Page 1565).

Status bit	Parameter
0	1= `Logic` = OR
1	1 = `Logic` = AND
2	Not used

Status bit	Parameter
3	Result of logic operation `Out.Value`
4	1 = At least one input value is excluded. See the section Excluding input values in Intlk04 functions (Page 1560).
5	1 = All input values are excluded
6	1 = No input value is interconnected or `NotUsed.Value`
7	Status display for simulation
8	Status display for not interlocked
9	Status display for interlocked
10	Not used
11	`Feature` bit 2: Separate evaluation for excluded and simulated interlock signals
12 - 31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`In01.Value`
1	`In02.Value`
2	`In03.Value`
3	`In04.Value`
4 - 31	Not used

Status word allocation for `Status3` parameter

Status bit	Parameter
0	`InvIn01`
1	`InvIn02`
2	`InvIn03`
3	`InvIn04`
4 - 31	Not used

Status word allocation for `Status4` parameter

Status bit	Parameter
0	`In01` with inversion
1	`In02` with inversion
2	`In03` with inversion
3	`In04` with inversion
4 - 15	Not used
16	Bypass `In01` (via interconnection)
17	Bypass `In02` (via interconnection)
18	Bypass `In03` (via interconnection)

Status bit	Parameter
19	Bypass `In04` (via interconnection)
20 - 31	Not used

Status word allocation for `Status5` parameter

Status bit	Parameter
0	`BypIn01`
1	`BypIn02`
2	`BypIn03`
3	`BypIn04`
4 - 15	Not used
16	In01hidden bypass information
17	In02hidden bypass information
18	In03hidden bypass information
19	In04hidden bypass information
20-31	Not used

Status word allocation for `Status6` parameter

Status bit	Parameter
0	`In01` not connected
1	`In02` not connected
2	`In03` not connected
3	`In04` not connected
4 - 31	Not used

Status word allocation for `Status7` parameter

Status bit	Parameter
0	`AV01` not connected
1	`AV02` not connected
2	`AV03` not connected
3	`AV04` not connected
4 - 31	Not used

Status word allocation for `Status8` parameter

Identical to `FirstIn`.

See also

Intlk04 messaging (Page 1564)
Intlk04 block diagram (Page 1568)
Intlk04 error handling (Page 1564)
Intlk04 modes (Page 1560)

8.2.2 Intlk04 modes

Intlk04 modes

This block does not have any modes.

See also

Intlk04 block diagram (Page 1568)
Intlk04 I/Os (Page 1565)
Intlk04 messaging (Page 1564)
Intlk04 error handling (Page 1564)
Intlk04 functions (Page 1560)
Description of Intlk04 (Page 1557)

8.2.3 Intlk04 functions

Functions of Intlk04

The functions for this block are listed below.

Inversion of logic signals

You can invert the input signals by setting the input parameter `InvInx` for the input concerned to `Inx` = 1, e.g. for input `In01` set the I/O `InvIn01`.

The inversion is displayed in the faceplate. If you invert signals using any other method this is not shown in the faceplate.

Bypass

Note

Bypassing the interlock means that the interlock signal (input signal) is excluded from the logic of the interlock block, in other words, this signal is ignored in the logic operation.

This function can only be executed in the faceplate with "high-level operating permission".

You can exclude input signals that are temporarily not to be used for the calculation in the block by setting the corresponding I/O:

- Independent `BypLix` is connected or not, the operator can bypass the input signal: `BypInx` = 1 (from operator over faceplate)
- `BypLix.ST` <> 16#FF (input is connected) and a change in `BypLix` value is detected: `BypLix` = 1 (from connection in CFC)

The I/O is shown in the faceplate by the following symbol:

Special situation: If all input parameters are excluded, the output value is defined using the `DefaultOut` parameter.

Depending on the configuration of the `Feature` bit Separate evaluation for excluded and simulated interlock signals (Page 151), the output "Bypass" is formed and the hidden bypass signal is set at the output `Out`.

Feature Bit =0:

If one of the interlock inputs is excluded (`BypIn01`...`BypIn0x` or `BypLi01`...`BypLix`), the output `BypAct` is set. The hidden bypass signal at the output `Out` is reset.

- `BypAct.Value` =
  `BypIn01` OR `BypLi01` (if `BypLi01` is connected and the value has changed)
  OR `BypInx` OR `BypLix` (if `BypLix` is connected and the value has changed)
- `Out.Bit1` = 0

An excluded interlock input that is switch-relevant after the bypass, sets the status of `Out` to Simulation.

Feature Bit =1:

If one of the interlock inputs is excluded (`BypIn01`...`BypIn0x` or `BypLi01`...`BypLix`) or a hidden signal is set at an interlock input (`In01`...`Inx`), the output `BypAct` is set. The hidden bypass signal at the input `In01`...`Inx` can be read from the upstream interlock block at the output `BypAct`. The hidden bypass signal at the output `Out` is set to the value of `BypAct`.

- `BypAct.Value` =
  `BypIn01` OR `BypLi01` (if `BypLi01` is connected and the value has changed)
  OR `BypInx` OR `BypLix` (if `BypLix` is connected and the value has changed)
  OR `In01.Bit1` OR `Inx.Bit1` (hidden lines)
- `Out.Bit1` = `BypAct.Value`

An excluded interlock input has no influence on the status of `Out`.

Note

Do **not** connect the input `BypLix` with the output `BypAct` from an interlock block because `BypAct` is calculated from the hidden bits.

Handling non-connected inputs

Inputs which are not interconnected are not evaluated. They are not displayed in the faceplate either.

Special situation: If no inputs are connected, the output value is defined using the `DefaultOut` parameter. The signal status is set to 16#FF.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203). However, only one additional faceplate can be called up using `SelFp1` = 1.

First-in detection for interlock blocks

This block provides the standard function Recording the first signal for interlock blocks (Page 52). Note that a signal status change only has an effect on the first-in detection when the `Feature` Bit Evaluation of signal status (Page 141) and the input `FirstInEn` is set.

The response to deactivation via the `FirstInEn` input can be influenced by Feature bit First-in detection response to deactivation (Page 175).

Note

This function can only be executed in the faceplate with "process control" operating permission.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for interlock blocks (Page 115).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `Out.ST`

Operating permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can exclude values
1	1 = Operator can reset the exclusion of input values
2	1 = Operator can reset the recording of the first signal
3 - 15	Not used
16	1 = Operator can set or reset the exclusion of input value In01
17	1 = Operator can set or reset the exclusion of input value In02
18	1 = Operator can set or reset the exclusion of input value In03
19	1 = Operator can set or reset the exclusion of input value In04
20 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable reactions using the `Feature` I/O

You can find an overview of all reactions provided by the `Feature` parameter in the chapter Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
5	Activate OS_Perm bits (Page 156) 0 = OS_Perm bits 16...17 inactive (evaluation only in faceplate) 1 = OS_Perm bits 16...17 active (evaluation only in faceplate)
21	First-in detection response to deactivation (Page 175)
23	Evaluation of signal status (Page 141)
24	Enabling local operator authorization (Page 157)
31	Activating recording of the first signal (Page 149)

See also

Description of Intlk04 (Page 1557)

Intlk04 messaging (Page 1564)

Intlk04 I/Os (Page 1565)

Intlk04 block diagram (Page 1568)

Intlk04 error handling (Page 1564)

Intlk04 modes (Page 1560)

8.2.4 Intlk04 error handling

Error handling of Intlk04

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
13	The parameter `Logic` has not been set to 0 or 1.

See also

Intlk04 block diagram (Page 1568)

Intlk04 I/Os (Page 1565)

Intlk04 messaging (Page 1564)

Description of Intlk04 (Page 1557)

Intlk04 modes (Page 1560)

Intlk04 functions (Page 1560)

8.2.5 Intlk04 messaging

Messaging

This block does not offer messaging.

See also

Description of Intlk04 (Page 1557)

Intlk04 functions (Page 1560)

Intlk04 I/Os (Page 1565)

Intlk04 block diagram (Page 1568)

Intlk04 error handling (Page 1564)

Intlk04 modes (Page 1560)

8.2.6 Intlk04 I/Os

I/Os of Intlk04

Input parameters

Parameter	Description	Type	Default
AV01	Analog value of In01	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV02	Analog value of In02	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV03	Analog value of In03	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV04	Analog value of In04	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV01_Unit	Unit of measure for AV01	INT	0
AV02_Unit	Unit of measure for AV02	INT	0
AV03_Unit	Unit of measure for AV03	INT	0
AV04_Unit	Unit of measure for AV04	INT	0
BypIn01*	1 = Input In01 is not used	BOOL	0
BypIn02*	1 = Input In02 is not used	BOOL	0
BypIn03*	1 = Input In03 is not used	BOOL	0
BypIn04*	1 = Input In04 is not used	BOOL	0
BypLi01...BypLi04	Bypass In01...In04 (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
DefaultOut	Output value for the case that all inputs are excluded or not connected. Refer to section Intlk04 functions (Page 1560) of the block.	BOOL	1
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1560)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0

Parameter	Description	Type	Default
`FirstInEn`	First-in detection	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
`In01`	Input `In01`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In02`	Input `In02`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In03`	Input `In03`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In04`	Input `In04`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`InvIn01`	Invert input `In01`	BOOL	0
`InvIn02`	Invert input `In02`	BOOL	0
`InvIn03`	Invert input `In03`	BOOL	0
`InvIn04`	Invert input `In04`	BOOL	0
`Logic`	Logical operation: 0 = Logical OR 1 = Logical AND	INT	0
`NotUsed`	1 = Block is not used (only for display in the faceplate). You can find additional information on this in the section Interlock blocks standard view (Page 270) / Color of the field.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`OpSt_In`	Input parameter for local operator authorization, connected with the `Out` output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
`OS_Perm`	I/O for operating permissions (Page 1560)	STRUCT • Bit 0: BOOL • .... • Bit 31: BOOL	- • 1 • 1 • 1
`RstBypOp*`	1 = Reset inputs `BypIn01` to `BypIn04`	BOOL	0
`RstLi`	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`RstOp*`	1 = Reset via operator	BOOL	0

Parameter	Description	Type	Default
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
BypAct	1 = Bypass active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see Intlk04 error handling (Page 1564)	INT	-1
FirstIn	Bit-coded number of the first signal that has resulted in a change at the output	DWORD	16#00000000
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1557)	DWORD	16#00000000
Status2	Status word 2 (Page 1557)	DWORD	16#00000000
Status3	Status word 3 (Page 1557)	DWORD	16#00000000
Status4	Status word 4 (Page 1557)	DWORD	16#00000000
Status5	Status word 5 (Page 1557)	DWORD	16#00000000
Status6	Status word 6 (Page 1557)	DWORD	16#00000000
Status7	Status word 7 (Page 1557)	DWORD	16#00000000
Status8	Status word 8 (Page 1557)	DWORD	16#00000000

See also

Intlk04 messaging (Page 1564)
Intlk04 block diagram (Page 1568)
Intlk04 modes (Page 1560)

8.2.7 Intlk04 block diagram

Intlk04 block diagram

A block diagram is not provided for this block.

See also

Intlk04 I/Os (Page 1565)
Intlk04 messaging (Page 1564)
Intlk04 error handling (Page 1564)
Intlk04 functions (Page 1560)
Intlk04 modes (Page 1560)
Description of Intlk04 (Page 1557)

8.2.8 Operator control and monitoring

8.2.8.1 Interlock block views

Views of the blocks Intlk02, Intlk04, Intlk08, Intlk16

The blocks provide the following views:

- Interlock blocks standard view (Page 270)
- Preview of interlock blocks (Page 293)
- Block icon for interlock blocks (Page 237)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

8.3 Intlk08 - Interlock display with 8 input signals

8.3.1 Description of Intlk08

Object name (type + number) and family

Type + number: FB 1826

Family: Interlck

Area of application of Intlk08

The block is used for the following applications:

- Standardized interlock with display

How it works

The block is used to calculate a standardized interlock that can be displayed on the OS. A maximum of 8 input signals can be supplied to the block. They are linked using selectable binary logic. The signal status of the output signal is also determined. You can assign an analog value with the signal status and unit to each input value for display in the faceplate.

The current state is displayed at the `Out` output parameter:

- `Out` = 0: Interlock
- `Out` = 1: "Good" state

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

For the Intlk02 (Intlk08) block, the Advanced Process Library contains process tag type templates; these serve as examples by providing various application scenarios for this block.

Refer to Description of Intlk02 (Page 1545) for more information.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section Intlk08 I/Os (Page 1577).

Status bit	Parameter
0	1 = `Logic` = OR
1	1 = `Logic` = AND
2	Not used

Status bit	Parameter
3	Result of logic operation `Out.Value`
4	1 = At least one input value is excluded. See the section Excluding input values in Intlk08 functions (Page 1573)
5	1 = All input values are excluded
6	1 = No input value is interconnected or `NotUsed.Value`
7	Status display for simulation
8	Status display for not interlocked
9	Status display for interlocked
10	Not used
11	`Feature` bit 2: Separate evaluation for excluded and simulated interlock signals
12 - 31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`In01.Value`
1	`In02.Value`
2	`In03.Value`
3	`In04.Value`
4	`In05.Value`
5	`In06.Value`
6	`In07.Value`
7	`In08.Value`
8 - 31	Not used

Status word allocation for `Status3` parameter

Status bit	Parameter
0	`InvIn01`
1	`InvIn02`
2	`InvIn03`
3	`InvIn04`
4	`InvIn05`
5	`InvIn06`
6	`InvIn07`
7	`InvIn08`
8 - 31	Not used

Status word allocation for `Status4` parameter

Status bit	Parameter
0	`In01` with inversion
1	`In02` with inversion
2	`In03` with inversion
3	`In04` with inversion
4	`In05` with inversion
5	`In06` with inversion
6	`In07` with inversion
7	`In08` with inversion
8 - 15	Not used
16	Bypass `In01` (via interconnection)
17	Bypass `In02` (via interconnection)
18	Bypass `In03` (via interconnection)
19	Bypass `In04` (via interconnection)
20	Bypass `In05` (via interconnection)
21	Bypass `In06` (via interconnection)
22	Bypass `In07` (via interconnection)
23	Bypass `In08` (via interconnection)
24 - 31	Not used

Status word allocation for `Status5` parameter

Status bit	Parameter
0	`BypIn01`
1	`BypIn02`
2	`BypIn03`
3	`BypIn04`
4	`BypIn05`
5	`BypIn06`
6	`BypIn07`
7	`BypIn08`
8 - 15	Not used
16 - 23	`In01..IN08` hidden bypass information
24 - 31	Not used

Status word allocation for `Status6` parameter

Status bit	Parameter
0	`In01` not connected
1	`In02` not connected
2	`In03` not connected

Status bit	Parameter
3	`In04` not connected
4	`In05` not connected
5	`In06` not connected
6	`In07` not connected
7	`In08` not connected
8 - 31	Not used

Status word allocation for `Status7` parameter

Status bit	Parameter
0	`AV01` not connected
1	`AV02` not connected
2	`AV03` not connected
3	`AV04` not connected
4	`AV05` not connected
5	`AV06` not connected
6	`AV07` not connected
7	`AV08` not connected
8 - 31	Not used

Status word allocation for `Status8` parameter

Identical to `FirstIn`.

See also

Intlk08 messaging (Page 1577)
Intlk08 block diagram (Page 1581)
Intlk08 error handling (Page 1576)
Intlk08 modes (Page 1572)

8.3.2 Intlk08 modes

Intlk08 modes

This block does not have any modes.

See also

Intlk08 block diagram (Page 1581)
Intlk08 I/Os (Page 1577)

Intlk08 messaging (Page 1577)
Intlk08 error handling (Page 1576)
Intlk08 functions (Page 1573)
Description of Intlk08 (Page 1569)

8.3.3 Intlk08 functions

Functions of Intlk08

The functions for this block are listed below.

Logic operators

Use the `Logic` input to specify the logic operator that the block should employ when determining the interlock state. Make the following settings:

- `Logic` = 0: OR
- `Logic` = 1: AND

Inversion of logic signals

You can invert the input signals by setting the input parameter `InvInx` for the input concerned to `Inx` = 1 , e.g. for input `In01` set the I/O `InvIn01`.

The inversion is displayed in the faceplate. If you invert signals using any other method, this is not shown in the faceplate.

Bypass

Note

Bypassing the interlock means that the interlock signal (input signal) is excluded from the logic of the interlock block, in other words, this signal is ignored in the logic operation.

This function can only be executed in the faceplate with "high-level operating permission".

You can exclude input signals that are temporarily not to be used for the calculation in the block by setting the corresponding I/O:

- Independent `BypLix` is connected or not, the operator can bypass the input signal: `BypInx` = 1 (from operator over faceplate)
- `BypLix.ST` <> 16#FF (input is connected) and a change in `BypLix` value is detected: `BypLix` = 1 (from connection in CFC)

The I/O is shown in the faceplate by the following symbol:

Special situation: If all input parameters are excluded, the output value is defined using the `DefaultOut` parameter.

Depending on the configuration of the `Feature` bit Separate evaluation for excluded and simulated interlock signals (Page 151), the output "Bypass" is formed and the hidden bypass signal is set at the output `Out`.

Feature Bit =0:

If one of the interlock inputs is excluded (`BypIn01`...`BypIn0x` or `BypLi01`...`BypLix`), the output `BypAct` is set. The hidden bypass signal at the output `Out` is reset.

- `BypAct.Value` =
  `BypIn01` OR `BypLi01` (if `BypLi01` is connected and the value has changed)
  OR `BypInx` OR `BypLix` (if `BypLix` is connected and the value has changed)
- `Out.Bit1` = 0

An excluded interlock input that is switch-relevant after the bypass, sets the status of `Out` to Simulation.

Feature Bit =1:

If one of the interlock inputs is excluded (`BypIn01`...`BypIn0x` or `BypLi01`...`BypLix`) or a hidden signal is set at an interlock input (`In01`...`Inx`), the output `BypAct` is set. The bypass signal at the input `In01`...`Inx` can be read from the upstream interlock block at the output `BypAct`. The hidden bypass signal at the output `Out` is set to the value of `BypAct`.

- `BypAct.Value` =
  `BypIn01` OR `BypLi01` (if `BypLi01` is connected and the value has changed)
  OR `BypInx` OR `BypLix` (if `BypLix` is connected and the value has changed)
  OR `In01.Bit1` OR `Inx.Bit1` (hidden lines)
- `Out.Bit1` = `BypAct.Value`

An excluded interlock input has no influence on the status of `Out`.

Note

Do **not** connect the input `BypLix` with the output `BypAct` from an interlock block because `BypAct` is calculated from the hidden bits.

Handling non-connected inputs

Inputs which are not interconnected are not evaluated. They are not displayed in the faceplate either.

Special situation: If no inputs are connected, the output value is defined using the `DefaultOut` parameter. The signal status is set to 16#FF.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203). However, only one additional faceplate can be called up using `SelFp1` = 1 .

First-in detection for interlock blocks

This block provides the standard function Recording the first signal for interlock blocks (Page 52). Note that a signal status change only has an effect on the first-in detection when the `Feature` Bit Evaluation of signal status (Page 141) and the input FirstInEn is set.

The response to deactivation via the `FirstInEn` input can be influenced by Feature bit First-in detection response to deactivation (Page 175).

Note

This function can only be executed in the faceplate with "process control" operating permission.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for interlock blocks (Page 115).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `Out.ST`

Operating permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	1 = Operator can exclude values
1	1 = Operator can reset the exclusion of input values
2	1 = Operator can reset the recording of the first signal
3 - 15	Not used
16	1 = Operator can set or reset the exclusion of input value In01
17	1 = Operator can set or reset the exclusion of input value In02
18	1 = Operator can set or reset the exclusion of input value In03
19	1 = Operator can set or reset the exclusion of input value In04
20	1 = Operator can set or reset the exclusion of input value In05
21	1 = Operator can set or reset the exclusion of input value In06
22	1 = Operator can set or reset the exclusion of input value In07
23	1 = Operator can set or reset the exclusion of input value In08
24 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the chapter Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
5	Activate OS_Perm bits (Page 156) 0 = OS_Perm bits 16...17 inactive (evaluation only in faceplate) 1 = OS_Perm bits 16...17 active (evaluation only in faceplate)
21	First-in detection response to deactivation (Page 175)
23	Evaluation of signal status (Page 141)
24	Enabling local operator authorization (Page 157)
31	Activating recording of the first signal (Page 149)

See also

Description of Intlk08 (Page 1569)

Intlk08 messaging (Page 1577)

Intlk08 I/Os (Page 1577)

Intlk08 block diagram (Page 1581)

Intlk08 error handling (Page 1576)

Intlk08 modes (Page 1572)

8.3.4 Intlk08 error handling

Error handling of Intlk08

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
13	The parameter `Logic` has not been set to 0 or 1

See also

Intlk08 block diagram (Page 1581)
Intlk08 I/Os (Page 1577)
Intlk08 messaging (Page 1577)
Intlk08 functions (Page 1573)
Intlk08 modes (Page 1572)
Description of Intlk08 (Page 1569)

8.3.5 Intlk08 messaging

Messaging

This block does not offer messaging.

See also

Description of Intlk08 (Page 1569)
Intlk08 functions (Page 1573)
Intlk08 I/Os (Page 1577)
Intlk08 block diagram (Page 1581)
Intlk08 error handling (Page 1576)
Intlk08 modes (Page 1572)

8.3.6 Intlk08 I/Os

I/Os of Intlk08

Input parameters

Parameter	Description	Type	Default
AV01	Analog value of In01	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV02	Analog value of In02	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV03	Analog value of In03	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV04	Analog value of In04	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV05	Analog value of In05	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV06	Analog value of In06	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV07	Analog value of In07	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV08	Analog value of In08	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV01_Unit	Unit of measure for AV01	INT	0
AV02_Unit	Unit of measure for AV02	INT	0
AV03_Unit	Unit of measure for AV03	INT	0
AV04_Unit	Unit of measure for AV04	INT	0
AV05_Unit	Unit of measure for AV05	INT	0
AV06_Unit	Unit of measure for AV06	INT	0
AV07_Unit	Unit of measure for AV07	INT	0
AV08_Unit	Unit of measure for AV08	INT	0
BypIn01*	1 = Input In01 is not used	BOOL	0
BypIn02*	1 = Input In02 is not used	BOOL	0
BypIn03*	1 = Input In03 is not used	BOOL	0
BypIn04*	1 = Input In04 is not used	BOOL	0
BypIn05*	1 = Input In05 is not used	BOOL	0
BypIn06*	1 = Input In06 is not used	BOOL	0
BypIn07*	1 = Input In07 is not used	BOOL	0

Parameter	Description	Type	Default
BypIn08*	1 = Input In08 is not used	BOOL	0
BypLi01...BypLi08	Bypass In01...In08 (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
DefaultOut	Output value for cases where all inputs are excluded or not interconnected. Refer to the section Intlk08 functions (Page 1573) of the block.	BOOL	1
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1573)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
FirstInEn	First-in detection	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In01	Input In01	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In02	Input In02	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In03	Input In03	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In04	Input In04	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In05	Input In05	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In06	Input In06	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In07	Input In07	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In08	Input In08	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
InvIn01	Invert Input In01	BOOL	0
InvIn02	Invert Input In02	BOOL	0

Parameter	Description	Type	Default
InvIn03	Invert Input In03	BOOL	0
InvIn04	Invert Input In04	BOOL	0
InvIn05	Invert Input In05	BOOL	0
InvIn06	Invert Input In06	BOOL	0
InvIn07	Invert Input In07	BOOL	0
InvIn08	Invert Input In08	BOOL	0
Logic	Logical operation: 0 = Logical OR 1 = Logical AND	INT	0
NotUsed	1 = Block is not used (only for display in the faceplate). You can find additional information on this in the section Interlock blocks standard view (Page 270) / Color of the field.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operating permissions (Page 1573)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RstBypOp*	1 = Reset inputs BypIn01to BypIn08	BOOL	0
RstLi	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
BypAct	1 = Bypass active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0

Parameter	Description	Type	Default
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see Intlk08 error handling (Page 1576)	INT	-1
FirstIn	Bit-coded number of the first signal that has resulted in a change at the output	DWORD	16#00000000
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1569)	DWORD	16#00000000
Status2	Status word 2 (Page 1569)	DWORD	16#00000000
Status3	Status word 3 (Page 1569)	DWORD	16#00000000
Status4	Status word 4 (Page 1577)	DWORD	16#00000000
Status5	Status word 5 (Page 1569)	DWORD	16#00000000
Status6	Status word 6 (Page 1569)	DWORD	16#00000000
Status7	Status word 7 (Page 1569)	DWORD	16#00000000
Status8	Status word 8 (Page 1569)	DWORD	16#00000000

See also

Intlk08 block diagram (Page 1581)

Intlk08 modes (Page 1572)

8.3.7 Intlk08 block diagram

Intlk08 block diagram

A block diagram is not provided for this block.

See also

Intlk08 I/Os (Page 1577)

Intlk08 messaging (Page 1577)

Intlk08 error handling (Page 1576)

Intlk08 functions (Page 1573)

Intlk08 modes (Page 1572)

Description of Intlk08 (Page 1569)

8.3.8 Operator control and monitoring

8.3.8.1 Interlock block views

Views of the blocks Intlk02, Intlk04, Intlk08, Intlk16

The blocks provide the following views:

- Interlock blocks standard view (Page 270)
- Preview of interlock blocks (Page 293)
- Block icon for interlock blocks (Page 237)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

8.4 Intlk16 - Interlock display with 16 input signals

8.4.1 Description of Intlk16

Object name (type + number) and family

Type + number: FB 1827

Family: Interlck

Area of application for Intlk16

The block is used for the following applications:

- Standardized interlock with display

How it works

The block is used to calculate a standardized interlock that can be displayed on the OS. A maximum of 16 input signals can be supplied to the block. They are linked using selectable binary logic. The signal status of the output signal is also determined. You can assign an analog value with the signal status and unit to each input value for display in the faceplate.

The current state is displayed at the `Out` output parameter:

- `Out` = 0: Interlock
- `Out` = 1: "Good" state

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

For the Intlk02 (Intlk16) block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Refer to Description of Intlk02 (Page 1545) for more information.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section Intlk16 I/Os (Page 1593).

Status bit	Parameter
0	1 = `Logic` = OR
1	1 = `Logic` = AND
2	Not used

Status bit	Parameter
3	Result of logic operation `Out.Value`
4	1 = At least one input value is excluded. See the section Excluding input values in Intlk16 functions (Page 1588)
5	1 = All input values are excluded
6	1 = No input value is interconnected or `NotUsed.Value`
7	Status display for simulation
8	Status display for not interlocked
9	Status display for interlocked
10	Not used
11	`Feature` bit 2: Separate evaluation for excluded and simulated interlock signals
12 - 31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`In01.Value`
1	`In02.Value`
2	`In03.Value`
3	`In04.Value`
4	`In05.Value`
5	`In06.Value`
6	`In07.Value`
7	`In08.Value`
8	`In09.Value`
9	`In10.Value`
10	`In11.Value`
11	`In12.Value`
12	`In13.Value`
13	`In14.Value`
14	`In15.Value`
15	`In16.Value`
16 - 31	Not used

Status word allocation for `Status3` parameter

Status bit	Parameter
0	`InvIn01`
1	`InvIn02`
2	`InvIn03`
3	`InvIn04`
4	`InvIn05`
5	`InvIn06`

Status bit	Parameter
6	InvIn07
7	InvIn08
8	InvIn09
9	InvIn10
10	InvIn11
11	InvIn12
12	InvIn13
13	InvIn14
14	InvIn15
15	InvIn16
16 - 31	Not used

Status word allocation for Status4 parameter

Status bit	Parameter
0	In01 with inversion
1	In02 with inversion
2	In03 with inversion
3	In04 with inversion
4	In05 with inversion
5	In06 with inversion
6	In07 with inversion
7	In08 with inversion
8	In09 with inversion
9	In10 with inversion
10	In11 with inversion
11	In12 with inversion
12	In13 with inversion
13	In14 with inversion
14	In15 with inversion
15	In16 with inversion
16	Bypass In01 (via interconnection)
17	Bypass In02 (via interconnection)
18	Bypass In03 (via interconnection)
19	Bypass In04 (via interconnection)
20	Bypass In05 (via interconnection)
21	Bypass In06 (via interconnection)
22	Bypass In07 (via interconnection)
23	Bypass In08 (via interconnection)
24	Bypass In09 (via interconnection)
25	Bypass In10 (via interconnection)
26	Bypass In11 (via interconnection)

Status bit	Parameter
27	Bypass `In12` (via interconnection)
28	Bypass `In13` (via interconnection)
29	Bypass `In14` (via interconnection)
30	Bypass `In15` (via interconnection)
31	Bypass `In16` (via interconnection)

Status word allocation for `Status5` parameter

Status bit	Parameter
0	`BypIn01`
1	`BypIn02`
2	`BypIn03`
3	`BypIn04`
4	`BypIn05`
5	`BypIn06`
6	`BypIn07`
7	`BypIn08`
8	`BypIn09`
9	`BypIn10`
10	`BypIn11`
11	`BypIn12`
12	`BypIn13`
13	`BypIn14`
14	`BypIn15`
15	`BypIn16`
16 - 31	`In01`..`In16`hidden bypass information

Status word allocation for `Status6` parameter

Status bit	Parameter
0	`In01` not connected
1	`In02` not connected
2	`In03` not connected
3	`In04` not connected
4	`In05` not connected
5	`In06` not connected
6	`In07` not connected
7	`In08` not connected
8	`In09` not connected
9	`In10` not connected
10	`In11` not connected
11	`In12` not connected

Status bit	Parameter
12	`In13` not connected
13	`In14` not connected
14	`In15` not connected
15	`In16` not connected
16 - 31	Not used

Status word allocation for `Status7` parameter

Status bit	Parameter
0	`AV01` not connected
1	`AV02` not connected
2	`AV03` not connected
3	`Av04` not connected
4	`AV05` not connected
5	`AV06` not connected
6	`AV07` not connected
7	`AV08` not connected
8	`AV09` not connected
9	`AV10` not connected
10	`AV11` not connected
11	`AV12` not connected
12	`AV13` not connected
13	`AV14` not connected
14	`AV15` not connected
15	`AV16` not connected
16 - 31	Not used

Status word allocation for `Status8` parameter

Identical to `FirstIn`.

See also

Intlk16 block diagram (Page 1599)
Intlk16 error handling (Page 1592)
Intlk16 modes (Page 1588)
Intlk16 messaging (Page 1593)

8.4.2 Intlk16 modes

Intlk16 modes

This block does not have any modes.

See also

Intlk16 block diagram (Page 1599)
Intlk16 I/Os (Page 1593)
Intlk16 messaging (Page 1593)
Intlk16 functions (Page 1588)
Intlk16 error handling (Page 1592)
Description of Intlk16 (Page 1583)

8.4.3 Intlk16 functions

Functions of Intlk16

The functions for this block are listed below.

Logic operators

Use the `Logic` input to specify the logic operator that the block should employ when determining the interlock state. Make the following settings:

- `Logic` = 0: OR
- `Logic` = 1: AND

Inversion of logic signals

You can invert the input signals by setting the input parameter `InvInx` for the input concerned to `Inx` = 1, e.g. for input `In01` set the I/O `InvIn01`.

The inversion is displayed in the faceplate. If you invert signals using any other method, this is not shown in the faceplate.

Bypass

Note

Bypassing the interlock means that the interlock signal (input signal) is excluded from the logic of the interlock block, in other words, this signal is ignored in the logic operation.

This function can only be executed in the faceplate with "high-level operating permission".

You can exclude input signals that are temporarily not to be used for the calculation in the block by setting the corresponding I/O:

- Independent `BypLix` is connected or not, the operator can bypass the input signal: `BypInx` = 1 (from operator over faceplate)
- `BypLix.ST` <> 16#FF (input is connected) and a change in `BypLix` value is detected: `BypLix` = 1 (from connection in CFC)

The I/O is shown in the faceplate by the following symbol:

Special situation: If all input parameters are excluded, the output value is defined using the `DefaultOut` parameter.

Depending on the configuration of the `Feature` bit Separate evaluation for excluded and simulated interlock signals (Page 151), the output "Bypass" is formed and the hidden bypass signal is set at the output `Out`.

Feature Bit =0:

If one of the interlock inputs is excluded (`BypIn01`...`BypIn0x` or `BypLi01`...`BypLix`), the output `BypAct` is set. The hidden bypass signal at the output `Out` is reset.

- `BypAct.Value` =
  `BypIn01` OR `BypLi01` (if `BypLi01` is connected and the value has changed)
  OR `BypInx` OR `BypLix` (if `BypLix` is connected and the value has changed)
- `Out.Bit1` = 0

An excluded interlock input that is switch-relevant after the bypass, sets the status of `Out` to Simulation.

Feature Bit =1:

If one of the interlock inputs is excluded (`BypIn01`...`BypIn0x` or `BypLi01`...`BypLix`) or a hidden signal is set at an interlock input (`In01`...`Inx`), the output `BypAct` is set. The bypass signal at the input `In01`...`Inx` can be read from the upstream interlock block at the output `BypAct`. The hidden bypass signal at the output `Out` is set to the value of `BypAct`.

- `BypAct.Value` =
  `BypIn01` OR `BypLi01` (if `BypLi01` is connected and the value has changed)
  OR `BypInx` OR `BypLix` (if `BypLix` is connected and the value has changed)
  OR `In01.Bit1` OR `Inx.Bit1` (hidden lines)
- `Out.Bit1` = `BypAct.Value`

An excluded interlock input has no influence on the status of `Out`.

Note

Do **not** connect the input `BypLix` with the output `BypAct` from an interlock block because `BypAct` is calculated from the hidden bits.

Handling non-connected inputs

Inputs which are not interconnected are not evaluated. They are not displayed in the faceplate either.

Special situation: If no inputs are connected, the output value is defined using the `DefaultOut` parameter. The signal status is set to 16#FF.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203). However, only one additional faceplate can be called up using `SelFp1` = 1 .

First-in detection for interlock blocks

This block provides the standard function Recording the first signal for interlock blocks (Page 52). Note that a signal status change only has an effect on the first-in detection when the `Feature` Bit Evaluation of signal status (Page 141) and the input FirstInEn is set.

The response to deactivation via the `FirstInEn` input can be influenced by Feature bit First-in detection response to deactivation (Page 175).

Note

This function can only be executed in the faceplate with "process control" operating permission.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for interlock blocks (Page 115).

The worst signal status `ST_Worst` for the block is formed from the following parameter:

- `OUT.ST`

Operating permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the OS_Perm parameter:

Bit	Function
0	1 = Operator can exclude values
1	1 = Operator can reset the exclusion of input values
2	1 = Operator can reset the recording of the first signal
3 - 15	Not used
16	1 = Operator can set or reset the exclusion of input value In01
17	1 = Operator can set or reset the exclusion of input value In02
18	1 = Operator can set or reset the exclusion of input value In03
19	1 = Operator can set or reset the exclusion of input value In04
20	1 = Operator can set or reset the exclusion of input value In05
21	1 = Operator can set or reset the exclusion of input value In06
22	1 = Operator can set or reset the exclusion of input value In07
23	1 = Operator can set or reset the exclusion of input value In08
24	1 = Operator can set or reset the exclusion of input value In09
25	1 = Operator can set or reset the exclusion of input value In10
26	1 = Operator can set or reset the exclusion of input value In11
27	1 = Operator can set or reset the exclusion of input value In12
28	1 = Operator can set or reset the exclusion of input value In13
29	1 = Operator can set or reset the exclusion of input value In14
30	1 = Operator can set or reset the exclusion of input value In15
31	1 = Operator can set or reset the exclusion of input value In16

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

Configurable reactions using the Feature parameter

You can find an overview of all reactions provided by the Feature parameter in the chapter Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
2	Separate evaluation for excluded and simulated interlock signals (Page 151)
5	Activate OS_Perm bits (Page 156) 0 = OS_Perm bits 16...17 inactive (evaluation only in faceplate) 1 = OS_Perm bits 16...17 active (evaluation only in faceplate)
21	First-in detection response to deactivation (Page 175)
23	Evaluation of signal status (Page 141)
24	Enabling local operator authorization (Page 157)
31	Activating recording of the first signal (Page 149)

See also

Intlk16 block diagram (Page 1599)
Intlk16 error handling (Page 1592)
Intlk16 modes (Page 1588)
Description of Intlk16 (Page 1583)
Intlk16 I/Os (Page 1593)
Intlk16 messaging (Page 1593)

8.4.4 Intlk16 error handling

Error handling of Intlk16

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
13	The parameter `Logic` has not been set to 0 or 1.

See also

Intlk16 block diagram (Page 1599)
Intlk16 I/Os (Page 1593)
Intlk16 messaging (Page 1593)
Intlk16 functions (Page 1588)
Intlk16 modes (Page 1588)
Description of Intlk16 (Page 1583)

8.4.5 Intlk16 messaging

Messaging

This block does not offer messaging.

See also

Intlk16 block diagram (Page 1599)
Intlk16 error handling (Page 1592)
Intlk16 modes (Page 1588)
Description of Intlk16 (Page 1583)
Intlk16 I/Os (Page 1593)
Intlk16 functions (Page 1588)

8.4.6 Intlk16 I/Os

I/Os of Intlk16

Input parameters

Parameter	Description	Type	Default
AV01	Analog value of In01	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV02	Analog value of In02	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV03	Analog value of In03	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV04	Analog value of In04	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV05	Analog value of In05	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV06	Analog value of In06	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF

Parameter	Description	Type	Default
AV07	Analog value of In07	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV08	Analog value of In08	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV09	Analog value of In09	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV10	Analog value of In10	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV11	Analog value of In11	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV12	Analog value of In12	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV13	Analog value of In13	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV14	Analog value of In14	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV15	Analog value of In15	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV16	Analog value of In16	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV01_Unit	Unit of measure for AV01	INT	0
AV02_Unit	Unit of measure for AV02	INT	0
AV03_Unit	Unit of measure for AV03	INT	0
AV04_Unit	Unit of measure for AV04	INT	0
AV05_Unit	Unit of measure for AV05	INT	0
AV06_Unit	Unit of measure for AV06	INT	0
AV07_Unit	Unit of measure for AV07	INT	0
AV08_Unit	Unit of measure for AV08	INT	0
AV09_Unit	Unit of measure for AV09	INT	0
AV10_Unit	Unit of measure for AV10	INT	0
AV11_Unit	Unit of measure for AV11	INT	0

Parameter	Description	Type	Default
AV12_Unit	Unit of measure for AV12	INT	0
AV13_Unit	Unit of measure for AV13	INT	0
AV14_Unit	Unit of measure for AV14	INT	0
AV15_Unit	Unit of measure for AV15	INT	0
AV16_Unit	Unit of measure for AV16	INT	0
BypIn01*	1 = Input In01 is not used	BOOL	0
BypIn02*	1 = Input In02 is not used	BOOL	0
BypIn03*	1 = Input In03 is not used	BOOL	0
BypIn04*	1 = Input In04 is not used	BOOL	0
BypIn05*	1 = Input In05 is not used	BOOL	0
BypIn06*	1 = Input In06 is not used	BOOL	0
BypIn07*	1 = Input In07 is not used	BOOL	0
BypIn08*	1 = Input In08 is not used	BOOL	0
BypIn09*	1 = Input In09 is not used	BOOL	0
BypIn10*	1 = Input In10 is not used	BOOL	0
BypIn11*	1 = Input In11 is not used	BOOL	0
BypIn12*	1 = Input In12 is not used	BOOL	0
BypIn13*	1 = Input In13 is not used	BOOL	0
BypIn14*	1 = Input In14 is not used	BOOL	0
BypIn15*	1 = Input In15 is not used	BOOL	0
BypIn16*	1 = Input In16 is not used	BOOL	0
BypLi01...BypLi16	Bypass In01...In16 (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
DefaultOut	Output value for cases where all inputs are excluded or not interconnected. Refer to the section Intlk16 functions (Page 1588) of the block.	BOOL	1
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1588)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
FirstInEn	First-in detection	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In01	Input In01	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In02	Input In02	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF

Parameter	Description	Type	Default
`In03`	Input `In03`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In04`	Input `In04`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In05`	Input `In05`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In06`	Input `In06`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In07`	Input `In07`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In08`	Input `In08`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In09`	Input `In09`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In10`	Input `In10`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In11`	Input `In11`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In12`	Input `In12`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In13`	Input `In13`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In14`	Input `In14`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`In15`	Input `In15`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF

Parameter	Description	Type	Default
In16	Input In16	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
InvIn01	Invert input In01	BOOL	0
InvIn02	Invert input In02	BOOL	0
InvIn03	Invert input In03	BOOL	0
InvIn04	Invert input In04	BOOL	0
InvIn05	Invert input In05	BOOL	0
InvIn06	Invert input In06	BOOL	0
InvIn07	Invert input In07	BOOL	0
InvIn08	Invert input In08	BOOL	0
InvIn09	Invert input In09	BOOL	0
InvIn10	Invert input In10	BOOL	0
InvIn11	Invert input In11	BOOL	0
InvIn12	Invert input In12	BOOL	0
InvIn13	Invert input In13	BOOL	0
InvIn14	Invert input In14	BOOL	0
InvIn15	Invert input In15	BOOL	0
InvIn16	Invert input In16	BOOL	0
Logic	Logical operation: 0 = Logical OR 1 = Logical AND	INT	0
NotUsed	1 = Block is not used (only for display in the faceplate). You can find additional information on this in the section Interlock blocks standard view (Page 270) / Color of the field.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operating permissions (Page 1588)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RstBypOp*	1 = Reset inputs BypIn01 to BypIn16	BOOL	0
RstLi	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0

Parameter	Description	Type	Default
SelFp1	Call a block saved in this parameter as Opening additional faceplates (Page 203) in standard view	ANY	
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
BypAct	1 = Bypass active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see Intlk16 error handling (Page 1592)	INT	-1
FirstIn	Bit-coded number of the first signal that has resulted in a change at the output	DWORD	16#00000000
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Prem with settings changed by the block algorithm	DWORD	16#FFFFFFFF
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1583)	DWORD	16#00000000
Status2	Status word 2 (Page 1583)	DWORD	16#00000000
Status3	Status word 3 (Page 1583)	DWORD	16#00000000
Status4	Status word 4 (Page 1583)	DWORD	16#00000000
Status5	Status word 5 (Page 1583)	DWORD	16#00000000
Status6	Status word 6 (Page 1583)	DWORD	16#00000000
Status7	Status word 7 (Page 1583)	DWORD	16#00000000
Status8	Status word 8 (Page 1583)	DWORD	16#00000000

See also

Intlk16 block diagram (Page 1599)

Intlk16 modes (Page 1588)

Intlk16 messaging (Page 1593)

8.4.7 Intlk16 block diagram

Intlk16 block diagram

A block diagram is not provided for this block.

See also

Intlk16 I/Os (Page 1593)

Intlk16 messaging (Page 1593)

Intlk16 functions (Page 1588)

Intlk16 error handling (Page 1592)

Intlk16 modes (Page 1588)

Description of Intlk16 (Page 1583)

8.4.8 Operator control and monitoring

8.4.8.1 Interlock block views

Views of the blocks Intlk02, Intlk04, Intlk08, Intlk16

The blocks provide the following views:

- Interlock blocks standard view (Page 270)
- Preview of interlock blocks (Page 293)
- Block icon for interlock blocks (Page 237)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

8.5 FirstIn - Transform output IntIck FirstIn for message associated value

8.5.1 Description of FirstIn

Object name (type + number) and family

Type + number: FC 1929

Family: Interlck

Area of application for FirstIn

The block is used for the following applications:

- To generate an external message that can be seen in the technological block.
- To generate an external message from EventTs or Event16Ts block.

How it works

FirstIn block works only with the combination of a technological block that has `ExtMsgx` and `ExtValxx` input parameters.

The block generates messages requiring acknowledgment that appear in the alarm view of the interconnected technological block.

The block must be connected to a technological block. Messages of the interconnected technological blocks are displayed in the message view.

FirstIn block also works with EventTs and Event16Ts blocks to generate an external message. External signals of FirstIn block should be connected with EventTs or Event16Ts block. When a fault occurs, EventTs or Event16Ts block is triggered and the message can be viewed in the interconnected technological block.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Connect the following output parameters with the input parameters as indicated in the following table:

Output parameter		Input parameter	
Parameter name	**Block**	**Parameter name**	**Block**
`FirstIn`	Interlock	`FirstIn`	FirstIn
`Out`	Interlock	`Interlock`	Technological
`FirstInAct`	FirstIn	`ExtMsgx`	Technological
`ExtVa1FI`	FirstIn	`ExtValxx`	Technological
`ExtVa2FI`	FirstIn	`ExtValxx`	Technological

For example:

- Output parameters ExtVa1FI and ExtVa2FI are connected to the input parameters ExtVa104 and ExtVa108 respectively.
- Output parameter FirstInAct is connected to the input parameter ExtMsg2.

In the PCS7 special message property of the external message 2 signal, add the following syntax:

"@4%s@@8%s@".

With EventTs/Event16Ts blocks:

You can connect FirstIn block with the block EventTs or Event16Ts. Connect the input parameters ExtVaxxx (ExtVal01...ExtVal03 in case of EventTs block and ExtVal01… ExtVa106 in case of Event16Ts block) with the output parameters ExtVa1FI or ExtVa2FI of the FirstIn block.

The output parameter FirstInAct of the FirstIn block should be connected with any Inx input parameters of the EventTs/Event16Ts blocks.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for Status parameter

This block does not have the Status parameter.

See also

FirstIn modes (Page 1601)
FirstIn functions (Page 1602)
FirstIn error handling (Page 1602)
FirstIn messaging (Page 1603)
FirstIn I/Os (Page 1603)
FirstIn block diagram (Page 1604)

8.5.2 FirstIn modes

FirstIn operating modes

This block does not have any operating modes.

See also

Description of FirstIn (Page 1600)
FirstIn functions (Page 1602)
FirstIn error handling (Page 1602)
FirstIn messaging (Page 1603)
FirstIn I/Os (Page 1603)
FirstIn block diagram (Page 1604)

8.5.3 FirstIn functions

Functions of FirstIn

This block does not have any special function.

See also

Description of FirstIn (Page 1600)
FirstIn modes (Page 1601)
FirstIn error handling (Page 1602)
FirstIn messaging (Page 1603)
FirstIn I/Os (Page 1603)
FirstIn block diagram (Page 1604)

8.5.4 FirstIn error handling

Error handling of FirstIn

This block does not have any error handling.

See also

Description of FirstIn (Page 1600)
FirstIn modes (Page 1601)
FirstIn functions (Page 1602)
FirstIn messaging (Page 1603)
FirstIn I/Os (Page 1603)
FirstIn block diagram (Page 1604)

8.5.5 FirstIn messaging

Messaging

This block does not offer messaging.

See also

Description of FirstIn (Page 1600)
FirstIn modes (Page 1601)
FirstIn functions (Page 1602)
FirstIn error handling (Page 1602)
FirstIn I/Os (Page 1603)
FirstIn block diagram (Page 1604)

8.5.6 FirstIn I/Os

I/Os of FirstIn

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
FirstIn	First active input	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
FirstInAct	First signal active for ExtMsgX or EventTs message	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
ExtVa1FI		STRUCT • Char1: CHAR • ... • Char15: CHAR	- • ' ' • ' ' • ' '
ExtVa2FI		STRUCT • Char1: CHAR • ... • Char23: CHAR	- • ' ' • ' ' • ' '

See also

Description of FirstIn (Page 1600)
FirstIn modes (Page 1601)
FirstIn functions (Page 1602)
FirstIn error handling (Page 1602)
FirstIn messaging (Page 1603)
FirstIn block diagram (Page 1604)

8.5.7 FirstIn block diagram

Block diagram of FirstIn

A block diagram is not provided for this block.

See also

Description of FirstIn (Page 1600)
FirstIn modes (Page 1601)
FirstIn functions (Page 1602)
FirstIn error handling (Page 1602)
FirstIn messaging (Page 1603)
FirstIn I/Os (Page 1603)

Message blocks

9

9.1 Event - Creating messages

9.1.1 Description of Event

Object name (type + number) and family

Type + number: FB 1811

Family: Report

Area of application for Event

The block is used for the following applications:

- Generation of messages requiring acknowledgment

How it works

The block is used to simultaneously output up to eight different messages that require acknowledgment.

The individual messages are assigned to the monitored signals via the inputs and the messages released or blocked depending on the process status data. If a change is made to one or more of the monitored signals enabled for reporting, a message is output.

The signals to be monitored are interconnected with inputs `In1` ... `In8`. Each `Inx` signal can also be inverted via input `InvInx`. A message is output if a signal value is changed (but taking inversion into account).

Each input is assigned a separate message text. A message received at input `In5`, for example, is output with the message text for the SIG 5 signal.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Further addressing is not required.

Startup characteristics

During startup, the message block suppresses all messages. The duration (number of cycles) of message suppression is set at the `RunUpCyc` parameter. During restart (OB100) an internal counter that is initialized with this value decrements each time the block is executed. Messages are not generated if the counter value does not equal zero.

Messages whose alarm delay has not elapsed during this period are then output.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section Event I/Os (Page 1613).

Status bit	Parameter
0	`In1.Value`
1	`In2.Value`
2	`In3.Value`
3	`In4.Value`
4	`In5.Value`
5	`In6.Value`
6	`In7.Value`
7	`In8.Value`
8	`InvIn1`
9	`InvIn2`
10	`InvIn3`
11	`InvIn4`
12	`InvIn5`
13	`InvIn6`
14	`InvIn7`
15	`InvIn8`
16	`In1` with inversion
17	`In2` with inversion
18	`In3` with inversion
19	`In4` with inversion
20	`In5` with inversion
21	`In6` with inversion
22	`In7` with inversion
23	`In8` with inversion
24	`In1` not connected
25	`In2` not connected
26	`In3` not connected
27	`In4` not connected
28	`In5` not connected
29	`In6` not connected
30	`In7` not connected
31	`In8` not connected

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`In1MsgEn`
1	`In2MsgEn`
2	`In3MsgEn`
3	`In4MsgEn`
4	`In5MsgEn`
5	`In6MsgEn`
6	`In7MsgEn`
7	`In8MsgEn`
8	`AV1` not connected
9	`AV2` not connected
10	`AV3` not connected
11	`AV4` not connected
12	`AV5` not connected
13	`AV6` not connected
14	`AV7` not connected
15	`AV8` not connected
16	Active signal 1 for messages
17	Active signal 2 for messages
18	Active signal 3 for messages
19	Active signal 4 for messages
20	Active signal 5 for messages
21	Active signal 6 for messages
22	Active signal 7 for messages
23	Active signal 8 for messages
24	`MsgLock`
25	`Occupied`
26	`BatchEn`
27	Batch - parameter exists
28 - 31	Not used

See also

Event functions (Page 1608)

Event messaging (Page 1611)

Event block diagram (Page 1617)

Event error handling (Page 1610)

Event modes (Page 1608)

9.1.2 Event modes

Event operating modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

You can find general information about the "On" mode in the On (Page 71) section.

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Event block diagram (Page 1617)

Event I/Os (Page 1613)

Event messaging (Page 1611)

Event error handling (Page 1610)

Event functions (Page 1608)

Description of Event (Page 1605)

9.1.3 Event functions

Functions of Event

The functions for this block are listed below.

Activation and deactivation of messages

Set the I/Os `In1MsgEn` to `In8MsgEn` accordingly to activate or deactivate the messages applied to inputs `In1` to `In8`. All messages are activated by default.

To deactivate messages received at I/O `In4`, for example, you set I/O `In4MsgEn` = 0 accordingly.

You can deactivate all messages via I/O `MsgLock` = 1 .

Delay of alarms

You can delay the alarms indicating signal changes.

For incoming alarms (signal change 0 - 1), set the delay at parameter `AlmOnDly`; for outgoing alarms (signal change 1 - 0) set it at parameter `AlmOffDly`.

Enter 0 or a negative value to deactivate the delay.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Operator control permissions

The block has the following Operator control permissions (Page 248) for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4 - 31	Not used

This block does not have a faceplate yet; the operator control permissions have already been assigned in the planning phase for these faceplates.

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
1	Reaction to the out of service mode (Page 176)
8	Reporting with BATCH parameters (Page 155)
22	Update acknowledgment and error status of the message call (Page 159)
27	Selecting values associated with messages (Page 155)

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The block determines the worst signal status via all interconnected binary and analog inputs, and outputs this value at `ST_Worst`.

- `In1`

etc. to

- `In8`
- `AV1`

etc. to

- `AV8`

See also

Description of Event (Page 1605)

Event messaging (Page 1611)

Event I/Os (Page 1613)

Event block diagram (Page 1617)

Event error handling (Page 1610)

Event modes (Page 1608)

9.1.4 Event error handling

Error handling of Event

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.

See also

Event block diagram (Page 1617)

Event I/Os (Page 1613)

Event messaging (Page 1611)

Event functions (Page 1608)

Event modes (Page 1608)

Description of Event (Page 1605)

9.1.5 Event messaging

Messaging

Messages which can be acknowledged are generated using ALARM_8P. The block uses the PMC communication channel and has 8 digital inputs and 8 associated values.

Every edge transition detected for one or more digital inputs results in a message. The associated values are assigned consistently to the message at the time of edge evaluation. All 8 signals are assigned a common message number, which is split at the OS into 8 messages. The Engineering System (ES) assigns the message number automatically by calling the message server.

Process messages

Message instance	Message identifier	Message class	Event
`MsgEvId`	SIG 1	AS process control message - error	Text 1
	SIG 2	AS process control message - error	Text 2
	SIG 3	AS process control message - error	Text 3
	SIG 4	AS process control message - error	Text 4
	SIG 5	AS process control message - error	Text 5
	SIG 6	AS process control message - error	Text 6
	SIG 7	AS process control message - error	Text 7
	SIG 8	AS process control message - error	Text 8

You can change the message class and the event to meet your needs at the block type and/or block instance.

Depending on `Feature` bit 27 "Select message associated values", either the signal status or the associated analog value is written as message associated value (`Feature` bit 8 = 0).

Associated values for message instance `MsgEvId`(`Feature` bit 27 = 0)

Associated value	Block parameters
1	`In1.ST`
2	`In2.ST`
3	`In3.ST`
4	`In4.ST`
5	`In5.ST`
6	`In6.ST`
7	`In7.ST`
8	`In8.ST`
9	Not allocated
10	Not allocated

Associated values for message instance `MsgEvId` (`Feature` bit 27 = 1)

Associated value	Block parameters
1	`AV1.Value`
2	`AV2.Value`
3	`AV3.Value`
4	`AV4.Value`
5	`AV5.Value`
6	`AV6.Value`
7	`AV7.Value`
8	`AV8.Value`
9	Not allocated
10	Not allocated

The batch information is transmitted with `Feature` bit 8 =1:

The first three associated values are described as follows and are followed by the signal status of the input signal or the associated analog value, depending on `Feature` bit 27 "Selecting message associated values":

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchId
4	In1.ST / AV1.Value
5	In2.ST / AV2.Value
6	In3.ST / AV3.Value
7	In4.ST / AV4.Value
8	In5.ST / AV5.Value

Associated value	Block parameters
9	In6.ST / AV6.Value
10	In7.ST / AV7.Value

Enter the batch ID @1%s@ under "Properties - Block - Special properties - Messages extended - Message texts block".

See also

Description of Event (Page 1605)

Event functions (Page 1608)

Event I/Os (Page 1613)

Event block diagram (Page 1617)

Event error handling (Page 1610)

Event modes (Page 1608)

Selecting values associated with messages (Page 155)

9.1.6 Event I/Os

I/Os of Event

Input parameters

Parameter	Description	Type	Default
AlmOnDly	Alarm delay time [s] for signal transition 0 → 1	REAL	0.0
AlmOffDly	Alarm delay time [s] for signal transition 1 → 0	REAL	0.0
AV1	Message associated value for In1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV1_Unit	Unit for AV1	INT	0
AV2	Message associated value for In2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV2_Unit	Unit for AV2	INT	0
AV3	Message associated value for In3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF

Parameter	Description	Type	Default
AV3_Unit	Unit for AV3	INT	0
AV4	Message associated value for In4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV4_Unit	Unit for AV4	INT	0
AV5	Message associated value for In5	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV5_Unit	Unit for AV5	INT	0
AV6	Message associated value for In6	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV6_Unit	Unit for AV6	INT	0
AV7	Message associated value for In7	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV7_Unit	Unit for AV7	INT	0
AV8	Message associated value for In8	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV8_Unit	Unit for AV8	INT	0
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00
BatchName	Batch name	S7-String	
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1608)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
In1	Input In1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In1MsgEn	1 = Activate message for input In1	BOOL	1
In2	Input In2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In2MsgEn	1 = Activate message for input In2	BOOL	1
In3	Input In3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In3MsgEn	1 = Activate message for input In3	BOOL	1

Parameter	Description	Type	Default
In4	Input In4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In4MsgEn	1 = Activate message for input In4	BOOL	1
In5	Input In5	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In5MsgEn	1 = Activate message for input In5	BOOL	1
In6	Input In6	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In6MsgEn	1 = Activate message for input In6	BOOL	1
In7	Input In7	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In7MsgEn	1 = Activate message for input In7	BOOL	1
In8	Input In8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In8MsgEn	1 = Activate message for input In8	BOOL	1
InvIn1	1 = Invert input In1	BOOL	0
InvIn2	1 = Invert input In2	BOOL	0
InvIn3	1 = Invert input In3	BOOL	0
InvIn4	1 = Invert input In4	BOOL	0
InvIn5	1 = Invert input In5	BOOL	0
InvIn6	1 = Invert input In6	BOOL	0
InvIn7	1 = Invert input In7	BOOL	0
InvIn8	1 = Invert input In8	BOOL	0
MS_RelOp	1 = Release for maintenance by OS operator	BOOL	0
MsgEvId	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. Refer also to the Suppressing messages using the MsgLock parameter (Page 201) section for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = In use by a batch	BOOL	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0

Parameter	Description	Type	Default
OS_Perm	I/O for operator control permissions (Page 1608)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
SelFp2	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	
StepNo	Batch step number	DWORD	16#00
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see Event error handling (Page 1610)	INT	-1
MsgAckn	Message acknowledgment status (output ACK_STATE of ALARM_8P)	WORD	16#0000
MsgErr	1 = Alarm error (output ERROR of ALARM_8)	BOOL	0
MsgStat	Alarm status (output STATUS of ALARM_8P)	WORD	16#0000
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF

Parameter	Description	Type	Default
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1605)	DWORD	16#00000000
Status2	Status word 2 (Page 1605)	DWORD	16#00000000

See also

Event messaging (Page 1611)

Event block diagram (Page 1617)

Event modes (Page 1608)

9.1.7 Event block diagram

Event block diagram

A block diagram is not provided for this block.

See also

Event I/Os (Page 1613)

Event messaging (Page 1611)

Event error handling (Page 1610)

Event functions (Page 1608)

Event modes (Page 1608)

Description of Event (Page 1605)

9.2 EventNck - Generating messages without acknowledgment

9.2.1 Description of EventNck

Object name (type + number) and family

Type + number: FB 1904

Family: Report

Area of application for EventNck

The block is used for the following applications:

- Generation of messages not requiring acknowledgment

How it works

The block is used to simultaneously output up to eight different messages that do not require acknowledgment.

The individual messages are assigned to the monitored signals via the inputs and the messages released or blocked depending on the process status data. If a change is made to one or more of the monitored signals enabled for reporting, a message is output.

The signals to be monitored are interconnected with inputs `In1` to `In8`. Each `Inx` signal can also be inverted via input `InvInx`. A message is output if a signal value is changed (but taking inversion into account).

Each input is assigned a separate message text. A message received at input `In5`, for example, is output with the message text for the SIG 5 signal.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Further addressing is not required.

Startup characteristics

During startup, the message block suppresses all messages. The duration (number of cycles) of message suppression is set at the `RunUpCyc` parameter. During restart (OB100) an internal counter that is initialized with this value decrements each time the block is executed. Messages are not generated if the counter value does not equal zero.

Messages whose alarm delay has not elapsed during this period are then output.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section EventNck I/Os (Page 1625).

Status bit	Parameter
0	`In1.Value`
1	`In2.Value`
2	`In3.Value`
3	`In4.Value`
4	`In5.Value`
5	`In6.Value`
6	`In7.Value`
7	`In8.Value`
8	`InvIn1`
9	`InvIn2`
10	`InvIn3`
11	`InvIn4`
12	`InvIn5`
13	`InvIn6`
14	`InvIn7`
15	`InvIn8`
16	`In1` with inversion
17	`In2` with inversion
18	`In3` with inversion
19	`In4` with inversion
20	`In5` with inversion
21	`In6` with inversion
22	`In7` with inversion
23	`In8` with inversion
24	`In1` not connected
25	`In2` not connected
26	`In3` not connected
27	`In4` not connected
28	`In5` not connected
29	`In6` not connected
30	`In7` not connected
31	`In8` not connected

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`In1MsgEn`
1	`In2MsgEn`
2	`In3MsgEn`

Status bit	Parameter
3	`In4MsgEn`
4	`In5MsgEn`
5	`In6MsgEn`
6	`In7MsgEn`
7	`In8MsgEn`
8	`AV1` not connected
9	`AV2` not connected
10	`AV3` not connected
11	`AV4` not connected
12	`AV5` not connected
13	`AV6` not connected
14	`AV7` not connected
15	`AV8` not connected
16	Active signal 1 for messages
17	Active signal 2 for messages
18	Active signal 3 for messages
19	Active signal 4 for messages
20	Active signal 5 for messages
21	Active signal 6 for messages
22	Active signal 7 for messages
23	Active signal 8 for messages
24	`MsgLock`
25 - 31	Not used

See also

EventNck modes (Page 1620)
EventNck functions (Page 1621)
EventNck error handling (Page 1623)
EventNck messaging (Page 1624)
EventNck block diagram (Page 1629)

9.2.2 EventNck modes

EventNck modes

The block provides the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of EventNck (Page 1618)

EventNck functions (Page 1621)

EventNck error handling (Page 1623)

EventNck messaging (Page 1624)

EventNck I/Os (Page 1625)

EventNck block diagram (Page 1629)

9.2.3 EventNck functions

Functions of EventNck

The functions for this block are listed below.

Activation and deactivation of messages

Set the I/Os `In1MsgEn` ... `In8MsgEn` accordingly to enable or disable the messages applied to inputs `In1` ... `In8`. All messages are activated by default.

To deactivate messages received at I/O `In4`, for example, you set I/O `In4MsgEn` = 0 accordingly.

You can deactivate all messages via I/O `MsgLock` = 1.

Delay of alarms

You can delay the alarms indicating signal changes.

For incoming alarms (signal change 0 - 1), set the delay at parameter `AlmOnDly`; for outgoing alarms (signal change 1 - 0) set it at parameter `AlmOffDly`.

Enter 0 or a negative value to deactivate the delay.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Operator control permissions

The block has the following Operator control permissions (Page 248) for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4 - 31	Not used

This block does not have a faceplate yet; the operator control permissions have already been assigned in the planning phase for these faceplates.

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
1	Reaction to the out of service mode (Page 176)
22	Update acknowledgment and error status of the message call (Page 159)
27	Selecting values associated with messages (Page 155)

Forming the signal status for blocks

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The block determines the worst signal status via all interconnected binary and analog inputs, and outputs this value at `ST_Worst`.

- `In1`

etc. to

- `In8`
- `AV1`

etc. to

- AV8

See also

Description of EventNck (Page 1618)

EventNck modes (Page 1620)

EventNck error handling (Page 1623)

EventNck messaging (Page 1624)

EventNck I/Os (Page 1625)

EventNck block diagram (Page 1629)

9.2.4 EventNck error handling

Error handling of EventNck

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The ErrorNum I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.

See also

Description of EventNck (Page 1618)

EventNck modes (Page 1620)

EventNck functions (Page 1621)

EventNck messaging (Page 1624)

EventNck I/Os (Page 1625)

EventNck block diagram (Page 1629)

9.2.5 EventNck messaging

Messaging

Messages not requiring acknowledgment are generated with NOTIFY_8P . The block uses the PMC communication channel and has 8 digital inputs and 10 associated values.

Every edge transition detected for one or more digital inputs results in a message. The associated values are assigned consistently to the message at the time of edge evaluation. All 8 signals are assigned a common message number, which is split at the OS into 8 messages. The Engineering System (ES) assigns the message number automatically by calling the message server.

Process messages

Message instance	Message identifier	Message class	Event
`MsgEvId`	SIG 1	Operating message – without acknowledgment	Text 1
	SIG 2	Operating message – without acknowledgment	Text 2
	SIG 3	Operating message – without acknowledgment	Text 3
	SIG 4	Operating message – without acknowledgment	Text 4
	SIG 5	Operating message – without acknowledgment	Text 5
	SIG 6	Operating message – without acknowledgment	Text 6
	SIG 7	Operating message – without acknowledgment	Text 7
	SIG 8	Operating message – without acknowledgment	Text 8

You can change the message class and the event to meet your needs at the block type and/or block instance.

Associated values for message instance `MsgEvId`

You can use the `Feature` Bit Selecting values associated with messages (Page 155) to define whether the signal status of the signal or the associated analog value is to be used as the associated value for the message.

Associated value	Block parameters
1	`In1.ST`
2	`In2.ST`
3	`In3.ST`
4	`In4.ST`

Associated value	Block parameters
5	In5.ST
6	In6.ST
7	In7.ST
8	In8.ST
9	Not allocated
10	Not allocated

Associated values for message instance MsgEvId

Associated value	Block parameters
1	AV1.Value
2	AV2.Value
3	AV3.Value
4	AV4.Value
5	AV5.Value
6	AV6.Value
7	AV7.Value
8	AV8.Value
9	Not allocated
10	Not allocated

See also

Description of EventNck (Page 1618)
EventNck modes (Page 1620)
EventNck functions (Page 1621)
EventNck error handling (Page 1623)
EventNck I/Os (Page 1625)
EventNck block diagram (Page 1629)

9.2.6 EventNck I/Os

I/Os of EventNck

Input parameters

Parameter	Description	Type	Default
AlmOnDly	Alarm delay time [s] for signal transition 0 → 1	REAL	0.0
AlmOffDly	Alarm delay time [s] for signal transition 1 → 0	REAL	0.0
AV1	Message associated value for In1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV1_Unit	Unit for AV1	INT	0
AV2	Message associated value for In2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV2_Unit	Unit for AV2	INT	0
AV3	Message associated value for In3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV3_Unit	Unit for AV3	INT	0
AV4	Message associated value for In4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV4_Unit	Unit for AV4	INT	0
AV5	Message associated value for In5	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV5_Unit	Unit for AV5	INT	0
AV6	Message associated value for In6	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV6_Unit	Unit for AV6	INT	0
AV7	Message associated value for In7	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV7_Unit	Unit for AV7	INT	0
AV8	Message associated value for In8	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
AV8_Unit	Unit for AV8	INT	0
EN	1 = Called block will be processed	BOOL	1

Parameter	Description	Type	Default
Feature	I/O for additional functions (Page 1621)	STRUCT	-
		• Bit 0: BOOL	• 0
		• ...	• 0
		• Bit 31: BOOL	• 0
In1	Input In1	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#FF
In1MsgEn	1 = Activate message for input In1	BOOL	1
In2	Input In2	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#FF
In2MsgEn	1 = Activate message for input In2	BOOL	1
In3	Input In3	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#FF
In3MsgEn	1 = Activate message for input In3	BOOL	1
In4	Input In4	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#FF
In4MsgEn	1 = Activate message for input In4	BOOL	1
In5	Input In5	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#FF
In5MsgEn	1 = Activate message for input In5	BOOL	1
In6	Input In6	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#FF
In6MsgEn	1 = Activate message for input In6	BOOL	1
In7	Input In7	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#FF
In7MsgEn	1 = Activate message for input In7	BOOL	1
In8	Input In8	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#FF
In8MsgEn	1 = Activate message for input In8	BOOL	1
InvIn1	1 = Invert input In1	BOOL	0
InvIn2	1 = Invert input In2	BOOL	0
InvIn3	1 = Invert input In3	BOOL	0
InvIn4	1 = Invert input In4	BOOL	0
InvIn5	1 = Invert input In5	BOOL	0

Parameter	Description	Type	Default
InvIn6	1 = Invert input In6	BOOL	0
InvIn7	1 = Invert input In7	BOOL	0
InvIn8	1 = Invert input In8	BOOL	0
MS_RelOp	1 = Release for maintenance by OS operator	BOOL	0
MsgEvId	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. Refer also to the Suppressing messages using the MsgLock parameter (Page 201) section for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OS_Perm	I/O for operator control permissions (Page 1621)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
SelFp2	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see EventNck error handling (Page 1623).	INT	-1
MsgErr	1 = Alarm error (output ERROR of NOTIFY_8P)	BOOL	0
MsgStat	Alarm status (output STATUS of NOTIFY_8P)	WORD	16#0000

Parameter	Description	Type	Default
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1618)	DWORD	16#00000000
Status2	Status word 2 (Page 1618)	DWORD	16#00000000

See also

EventNck modes (Page 1620)

EventNck messaging (Page 1624)

EventNck block diagram (Page 1629)

Configurable functions using the Feature I/O (Page 130)

9.2.7 EventNck block diagram

EventNck block diagram

A block diagram is not provided for this block.

See also

Description of EventNck (Page 1618)

EventNck modes (Page 1620)

EventNck functions (Page 1621)

EventNck error handling (Page 1623)

EventNck messaging (Page 1624)

EventNck I/Os (Page 1625)

9.3 EventTs - Creating messages with time stamp

9.3.1 Description of EventTs

Object name (type + number) and family

Type + number: FB 1812

Family: Report

Area of application for EventTs

The block is used for the following applications:

- Generating messages which have to be acknowledged for time-stamped signals

How it works

The block must be linked to a channel block and monitors up to eight different binary signals. From these, it generates time-stamped messages requiring acknowledgment that appear in the alarm view of the technological block to which it is connected.

The individual messages are assigned to the monitored signals via the inputs and the messages released or blocked depending on the process status data. If a change is made to one or more of the monitored signals enabled for reporting, a message is output containing a time stamp for the signal change.

`EventTs` can operate independently or in combination with one of the technological blocks that has the `EventTsIn` or `EventTs2In` input parameter. When this interconnection is configured, the messages of the `EventTs` block are displayed in the OS of the alarm view of the technological block and can also be acknowledged there. The following functions are controlled by the technological block:

- Switching to "Out of service" mode and switching back to "On" mode: `OosAct`, `OnAct`
- Release for maintenance: MS_Release
- Suppressing messages using the `MsgLock` parameter
- Batch parameters: `BatchEn`, `BatchID`, `BatchName`, `StepNo`, `Occupied`

Note

The `EventTs` instance must be newly created between deleting the interconnection of `EventTs` and the technological block and a delta download.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100) and in OB1 because of the time stamp.

If the EventTs operates together with a technological block, interconnect the output parameter EventTsOut of the EventTs block and the input parameter EventTsIn or EventTs2In of the technological block in the CFC.

Depending on how the time stamp is generated, the signals to be monitored for this are interconnected at inputs In1 ... In8 or InTS1 ... InTS8. Each Inx or InTSx signal can also be inverted via input Invx. A message is output if a signal value is changed (but taking inversion into account).

Each input is assigned a separate message text. A message received at input InTS5, for example, is output with the message text for the SIG 5 signal.

Configuration of the alarms for negative edges 1 → 0 in HW Config only effects the signal inputs InTS1 ... InTS8 of EventTs.

If TimeStrampOn = 0 , meaning the inputs In1 ... In8 are used, you will have to set the input Invx = 1 to generate an alarm 1 → 0 at input Inx for a negative edge.

Note

Interconnection of the block to multiple technological blocks is not permitted.

For the EventTs block, the Advanced Process Library contains process tag type templates; these serve as examples by providing various application scenarios for this block.

Examples of process tag types:

- Motor with an additional analog value and time-stamped signals (Motor_AV_EventTs) (Page 2339)

Startup characteristics

During startup, the message block suppresses all messages. The duration (number of cycles) of message suppression is set at the RunUpCyc parameter. During restart (OB100) an internal counter that is initialized with this value decrements each time the block is executed. Messages are not generated if the counter value does not equal zero.

Status word allocation for Status1 parameter

You can find a description for each parameter in section EventTs I/Os (Page 1640).

Status bit	Parameter
0	Active signal 1
1	Active signal 2
2	Active signal 3
3	Active signal 4
4	Active signal 5
5	Active signal 6
6	Active signal 7
7	Active signal 8
8	InvIn1
9	InvIn2

Status bit	Parameter
10	`InvIn3`
11	`InvIn4`
12	`InvIn5`
13	`InvIn6`
14	`InvIn7`
15	`InvIn8`
16	Active signal 1 with inversion
17	Active signal 2 with inversion
18	Active signal 3 with inversion
19	Active signal 4 with inversion
20	Active signal 5 with inversion
21	Active signal 6 with inversion
22	Active signal 7 with inversion
23	Active signal 8 with inversion
24	Active signal 1 not connected
25	Active signal 2 not connected
26	Active signal 3 not connected
27	Active signal 4 not connected
28	Active signal 5 not connected
29	Active signal 6 not connected
30	Active signal 7 not connected
31	Active signal 8 not connected

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`In1MsgEn`
1	`In2MsgEn`
2	`In3MsgEn`
3	`In4MsgEn`
4	`In5MsgEn`
5	`In6MsgEn`
6	`In7MsgEn`
7	`In8MsgEn`
8	Effective signal 1 for message
9	Effective signal 2 for message
10	Effective signal 3 for message
11	Effective signal 4 for message
12	Effective signal 5 for message
13	Effective signal 6 for message
14	Effective signal 7 for message
15	Effective signal 8 for message

Status bit	Parameter
16	MsgLock
17 - 24	Not used
25	Occupied
26	BatchEn
27	Batch - parameter exists
28 - 31	Not used

See also

EventTs block diagram (Page 1644)

EventTs messaging (Page 1637)

EventTs error handling (Page 1636)

EventTs functions (Page 1634)

EventTs modes (Page 1633)

9.3.2 EventTs modes

EventTs operating modes

The block provides the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

EventTs block diagram (Page 1644)

EventTs I/Os (Page 1640)

EventTs messaging (Page 1637)

EventTs error handling (Page 1636)

EventTs functions (Page 1634)

Description of EventTs (Page 1630)

9.3.3 EventTs functions

Functions of EventTs

The functions for this block are listed below.

Activation and deactivation of messages

You can set the I/Os `InMsgEn1` ... `InMsgEn8` to individually enable or disable the messages applied to inputs `In1` ... `In8` or `InTS1` ... `InTS8`. All messages are activated by default.

To deactivate messages received at I/O `InTS4`, for example, you set I/O `InMsgEn4` = 0 accordingly.

You can deactivate all messages via I/O `MsgLock` = 1 .

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Time stamp as associated value of a message

You can use input `TimeStampOn` to select how the time stamp for the signals of EventTs will be generated:

- If you want to use the high-precision time stamp from the I/O devices, set `TimeStampOn` = 1. Interconnect one of the `InTSx` inputs with output `TS_Out` of block Pcs7DiIT.
- If you want to use the time stamp from the CPU, set `TimeStampOn` = 0. Interconnect one of the `Inx` inputs with output `PV_Out` of block Pcs7DiIT or with an appropriate output of a different block.

For additional time stamp properties, refer to the description of the standard function Time stamp (Page 201).

Signal status as associated value of a message

The signal status as an associated value of the message is output for every signal, as is the time stamp.

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Operator permissions

The block has the following Operator control permissions (Page 248) for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4 - 31	Not used

This block does not have a faceplate yet; the operator permissions have already been assigned in the planning phase for these faceplates.

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable functions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
1	Reaction to the out of service mode (Page 176)
8	Reporting with BATCH parameters (Page 155)
9	Enable external message (Page 184)
22	Update acknowledgment and error status of the message call (Page 159)

Displaying and outputting the signal status

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The block determines the worst signal status via all interconnected binary inputs (according to `TimeStampOn`) and outputs this value at `ST_Worst`.

- `TimeStampOn` = 0
  - `In1`

  etc. to

  - `In8`
- `TimeStampOn` = 1
  - `InTS1`

  etc. to

  - `InTS8`

See also

EventTs block diagram (Page 1644)

EventTs I/Os (Page 1640)

EventTs messaging (Page 1637)

EventTs error handling (Page 1636)

EventTs modes (Page 1633)

Description of EventTs (Page 1630)

9.3.4 EventTs error handling

Error handling of EventTs

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.

See also

EventTs block diagram (Page 1644)

EventTs I/Os (Page 1640)

EventTs messaging (Page 1637)

EventTs functions (Page 1634)

EventTs modes (Page 1633)

Description of EventTs (Page 1630)

9.3.5 EventTs messaging

Messaging

Messages which can be acknowledged are generated using ALARM_8P. The ALARM_8P has 8 digital inputs and 10 associated values. Every edge transition detected for one or more digital inputs results in a message. The associated values are assigned consistently to the message at the time of edge evaluation. All 8 signals are assigned a common message number, which is split at the OS into 8 messages. The Engineering System (ES) assigns the message number automatically by calling the message server.

Process messages

Message instance	Message identifier	Message class	Event
`MsgEvId 1`	SIG 1	AS process control message - fault	$$BlockComment$$ `InTS1` Status 16#@1%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ `InTS2` Status 16#@2%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ `InTS3` Status 16#@3%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ `InTS1` Status 16#@4%x@
	SIG 5	AS process control message - fault	$$BlockComment$$ `InTS5` Status 16#@5%x@
	SIG 6	AS process control message - fault	$$BlockComment$$ `InTS6` Status 16#@6%x@
	SIG 7	AS process control message - fault	$$BlockComment$$ `InTS7` Status 16#@7%x@
	SIG 8	AS process control message - fault	$$BlockComment$$ `InTS8` Status 16#@8%x@

Explanation:

$$BlockComment$$: Content of the instance-specific comment

The following applies for `TimeStampOn` = 0 : 16#@n%x@ (n = 1 ... 8): The value contains the signal status of `In1` ... `In8`

The following applies for `TimeStampOn` = 1 : 16#@n%x@ (n = 1 ... 8): Statement on the validity of the time stamp from `InTS0` ... `InTS8`. If the value is 80, the time stamp is formed by the I/O. If the value is ≠ 80, the time stamp of the I/O is invalid; the time stamp is generated by the CPU as a substitute and is therefore not exact.

You can change the message class and the event to meet your needs at the block type and/or block instance.

Associated values for message instance **`MsgEvId`**

Associated value	Block parameters
1	`TimeStampOn` = 0: `In1.ST`
2	`TimeStampOn` = 0: `In2.ST`
3	`TimeStampOn` = 0: `In3.ST`
4	`TimeStampOn` = 0: `In4.ST`
5	`TimeStampOn` = 0: `In5.ST`
6	`TimeStampOn` = 0: `In6.ST`
7	`TimeStampOn` = 0: `In7.ST`
8	`TimeStampOn` = 0: `In8.ST`
9	Not allocated
10	Not allocated

Associated values for message instance **`MsgEvId`**

Associated value	Block parameters
1	`TimeStampOn` = 1: `InTS1.ST`
2	`TimeStampOn` = 1: `InTS2.ST`
3	`TimeStampOn` = 1: `InTS3.ST`
4	`TimeStampOn` = 1: `InTS4.ST`
5	`TimeStampOn` = 1: `InTS5.ST`
6	`TimeStampOn` = 1: `InTS6.ST`
7	`TimeStampOn` = 1: `InTS7.ST`
8	`TimeStampOn` = 1: `InTS8.ST`
9	Not allocated
10	Not allocated

The batch information is transmitted with `Feature` bit 8 =1:

The first three associated values are described as follows and are followed by the signal status of the input signals or the validity of the time stamp depending on `TimeStampOn`:

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchId
4	In1.ST / InTS1.ST
5	In2.ST / InTS2.ST
6	In3.ST / InTS3.ST
7	In4.ST / InTS4.ST
8	In5.ST / InTS5.ST
9	In6.ST / InTS6.ST

The tenth associated value is not available.

Enter the batch ID @1%s@ under "Properties - Block - Special properties - Messages extended - Message texts block".

The associated values in the process messages (event) must be incremented by 3:

Message instance	Message identifier	Message class	Event
`MsgEvId 1`	SIG 1	AS process control message - fault	$$BlockComment$$ `InTS1` Status 16#@4%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ `InTS2` Status 16#@5%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ `InTS3` Status 16#@6%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ `InTS1` Status 16#@7%x@
	SIG 5	AS process control message - fault	$$BlockComment$$ `InTS5` Status 16#@8%x@
	SIG 6	AS process control message - fault	$$BlockComment$$ `InTS6` Status 16#@9%x@
	SIG 7	AS process control message - fault	$$BlockComment$$ `InTS7` Status[1]
	SIG 8	AS process control message - fault	$$BlockComment$$ `InTS8` Status[1]

[1] You cannot specify a separate associated value here.

See also

EventTs block diagram (Page 1644)

EventTs I/Os (Page 1640)

EventTs error handling (Page 1636)

EventTs functions (Page 1634)

EventTs modes (Page 1633)

Description of EventTs (Page 1630)

Selecting values associated with messages (Page 155)

9.3.6 EventTs I/Os

I/Os of EventTs

Inputs

Parameter	Description	Type	Default
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00
BatchName	Batch name	S7-string	
EN	1 = Called block will be processed	BOOL	1
ExtVal01	External message 1	ANY	
ExtVal02	External message 2	ANY	
ExtVal03	External message 3	ANY	
Feature	I/O for additional functions (Page 1634)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
In1	Signal 1 for CPU time stamp	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In2	Signal 2 for CPU time stamp	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In3	Signal 3 for CPU time stamp	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In4	Signal 4 for CPU time stamp	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In5	Signal 5 for CPU time stamp	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF

Parameter	Description	Type	Default
In6	Signal 6 for CPU time stamp	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In7	Signal 7 for CPU time stamp	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
In8	Signal 8 for CPU time stamp	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
InTS1	Signal 1 with I/O-device time stamp	ANY	
InTS2	Signal 2 with I/O-device time stamp	ANY	
InTS3	Signal 3 with I/O-device time stamp	ANY	
InTS4	Signal 4 with I/O-device time stamp	ANY	
InTS5	Signal 5 with I/O-device time stamp	ANY	
InTS6	Signal 6 with I/O-device time stamp	ANY	
InTS7	Signal 7 with I/O-device time stamp	ANY	
InTS8	Signal 8 with I/O-device time stamp	ANY	
Inv1	1 = Invert input In1 or InTS1	BOOL	0
Inv2	1 = Invert input In2 or InTS2	BOOL	0
Inv3	1 = Invert input In3 or InTS3	BOOL	0
Inv4	1 = Invert input In4 or InTS4	BOOL	0
Inv5	1 = Invert input In5 or InTS5	BOOL	0
Inv6	1 = Invert input In6 or InTS6	BOOL	0
Inv7	1 = Invert input In7 or InTS7	BOOL	0
Inv8	1 = Invert input In8 or InTS8	BOOL	0
MsgEn1	1 = Activate message for input In1 or InTS1	BOOL	1
MsgEn2	1 = Activate message for input In2 or InTS2	BOOL	1
MsgEn3	1 = Activate message for input In3 or InTS3	BOOL	1
MsgEn4	1 = Activate message for input In4 or InTS4	BOOL	1
MsgEn5	1 = Activate message for input In5 or InTS5	BOOL	1
MsgEn6	1 = Activate message for input In6 or InTS6	BOOL	1

Parameter	Description	Type	Default
MsgEn7	1 = Activate message for input In7 or InTS7	BOOL	1
MsgEn8	1 = Activate message for input In8 or InTS8	BOOL	1
MsgEvId	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp	1 = Release for maintenance by OS operator	BOOL	0
Occupied	1 = In use by a batch	BOOL	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OS_Perm	I/O for EventTs functions (Page 1634)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
SelFp2	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
StepNo	Batch step number	DWORD	16#00
TimeStampOn	0 = Use time stamp of the CPU 1 = Use time stamp of the I/O devices	BOOL	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Outputs

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see EventTs error handling (Page 1636)	INT	-1
EventTsOut	For maneuvering of data between a technological block and the message blocks EventTs, Event16Ts.	STRUCT	
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn	Message acknowledgment status (output ACK_STATE of ALARM_8P)	WORD	16#0000
MsgErr	1 = Message error (output ERROR of ALARM_8P)	BOOL	0
MsgStat	Message status (output STATUS of ALARM_8P)	WORD	16#0000
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1630)	DWORD	16#00000000
Status2	Status word 2 (Page 1630)	DWORD	16#00000000

See also

EventTs block diagram (Page 1644)

EventTs messaging (Page 1637)

EventTs modes (Page 1633)

9.3.7 EventTs block diagram

EventTs block diagram

A block diagram is not provided for this block.

See also

EventTs I/Os (Page 1640)

EventTs messaging (Page 1637)

EventTs error handling (Page 1636)

EventTs functions (Page 1634)

EventTs modes (Page 1633)

Description of EventTs (Page 1630)

9.4 Event16Ts - Creating 16 messages with time stamp

9.4.1 Description of Event16Ts

Object name (type + number) and family

Type + number: FB 1887

Family: Report

Area of application for Event16Ts

The block is used for the following applications:

- Generating messages which have to be acknowledged for time-stamped signals

For Event16Ts block, use the templates for the process tag types of the EventTs block in the Advanced Process Library as examples with various use cases. Replace the EventTs block with the Event16Ts block.

How it works

The block monitors up to sixteen different binary signals, connect the block to a channel block.

The block generates time-stamped messages requiring acknowledgment that appear in the alarm view of the technological block to which it is connected.

The individual messages are assigned to the monitored signals via the inputs and the messages released or blocked depending on the process status data. If a change is made to one or more of the monitored signals enabled for reporting, a message is output containing a time stamp for the signal change.

`Event16Ts` can operate independently or in combination with one of the technological blocks that has the `EventTsIn` or `EventTs2In` input parameter. When the interconnection is configured, the messages of the `Event16Ts` block are displayed in the OS of the alarm view of the technological block and you can also acknowledge the messages. The following functions are controlled by the technological block:

- Switching to "Out of service" mode and switching back to "On" mode: `OosAct`, `OnAct`
- Release for maintenance: MS_Release
- Suppressing messages using the `MsgLock` parameter
- Batch parameters: `BatchEn`, `BatchID`, `BatchName`, `StepNo`, `Occupied`

Note

The `Event16Ts` instance must be newly created between deleting the interconnection of `Event16Ts` and the technological block and a delta download.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100) and in OB1 because for time stamping.

If the `Event16Ts` operates together with a technological block, interconnect the output parameter `EventTsOut` of the `Event16Ts` block and the input parameter `EventTsIn` or `EventTs2In` of the technological block in the CFC.

Depending on how the time stamp is generated, the signals to be monitored for this are interconnected at inputs `In1` ... `In16` or `InTS1` ... `InTS16`. Each `Inx` or `InTSx` signal can also be inverted via input `Invx`. If a signal value changes taking inversion into account, a message is output.

Note

1. The input parameters `ExtVal01`, `ExtVal02`, and `ExtVal03` will be used for `In1`...`In8` and `InTS1`...`InTS8`.
2. The input parameters `ExtVal04`, `ExtVal05`, and `ExtVal06` will be used for `In9`...`In16` and `InTS9`...`InTS16`.

Each input is assigned a separate message text. A message received at input `InTS5`, for example, is output with the message text for the SIG 5 signal.

Configuration of the alarms for negative edges 1 → 0 in HW Config only effects the signal inputs `InTS1` ... `InTS16` of Event16Ts.

If `TimeStrampOn` = 0 , meaning the inputs `In1` ... `In16` are used, you will have to set the input `Invx` = 1 to generate an alarm 1 → 0 at input `Inx` for a negative edge.

Note

Interconnection of the block to multiple technological blocks is not permitted.

For the Event16Ts block, the Advanced Process Library contains templates for process tag types as an example with various application scenarios for this block.

Examples of process tag types:

- Motor with an additional analog value and time-stamped signals (Motor_AV_EventTs) (Page 2339)

Startup characteristics

During startup, the message block suppresses all messages. The duration (number of cycles) of message suppression is set at the `RunUpCyc` parameter. During restart (OB100) an internal counter that is initialized with this value decrements each time the block is executed. Messages are not generated if the counter value does not equal zero.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section I/Os of Event16Ts.

Status bit	Parameter
0	Active signal 1 with inversion
1	Active signal 2 with inversion
2	Active signal 3 with inversion
3	Active signal 4 with inversion
4	Active signal 5 with inversion
5	Active signal 6 with inversion
6	Active signal 7 with inversion
7	Active signal 8 with inversion
8	Active signal 9 with inversion
9	Active signal 10 with inversion
10	Active signal 11 with inversion
11	Active signal 12 with inversion
12	Active signal 13 with inversion
13	Active signal 14 with inversion
14	Active signal 15 with inversion
15	Active signal 16 with inversion
16	Active signal 1 not connected
17	Active signal 2 not connected
18	Active signal 3 not connected
19	Active signal 4 not connected
20	Active signal 5 not connected
21	Active signal 6 not connected
22	Active signal 7 not connected
23	Active signal 8 not connected
24	Active signal 9 not connected
25	Active signal 10 not connected
26	Active signal 11 not connected
27	Active signal 12 not connected
28	Active signal 13 not connected
29	Active signal 14 not connected
30	Active signal 15 not connected
31	Active signal 16 not connected

Status word allocation for `Status2` parameter

Status bit	Parameter
0	Active signal 1
1	Active signal 2
2	Active signal 3

Status bit	Parameter
3	Active signal 4
4	Active signal 5
5	Active signal 6
6	Active signal 7
7	Active signal 8
8	Active signal 9
9	Active signal 10
10	Active signal 11
11	Active signal 12
12	Active signal 13
13	Active signal 14
14	Active signal 15
15	Active signal 16
16	InvIn1
17	InvIn2
18	InvIn3
19	InvIn4
20	InvIn5
21	InvIn6
22	InvIn7
23	InvIn8
24	InvIn9
25	InvIn10
26	InvIn11
27	InvIn12
28	InvIn13
29	InvIn14
30	InvIn15
31	InvIn16

Status word allocation for `Status3` parameter

Status bit	Parameter
0	In1MsgEn
1	In2MsgEn
2	In3MsgEn
3	In4MsgEn
4	In5MsgEn
5	In6MsgEn
6	In7MsgEn
7	In8MsgEn
8	In9MsgEn

Status bit	Parameter
9	In10MsgEn
10	In11MsgEn
11	In12MsgEn
12	In13MsgEn
13	In14MsgEn
14	In15MsgEn
15	In16MsgEn
16	Effective signal 1 for message
17	Effective signal 2 for message
18	Effective signal 3 for message
19	Effective signal 4 for message
20	Effective signal 5 for message
21	Effective signal 6 for message
22	Effective signal 7 for message
23	Effective signal 8 for message
24	Effective signal 9 for message
25	Effective signal 10 for message
26	Effective signal 11 for message
27	Effective signal 12 for message
28	Effective signal 13 for message
29	Effective signal 14 for message
30	Effective signal 15 for message
31	Effective signal 16 for message

Status word allocation for `Status4` parameter

Status bit	Parameter
0-7	Not used
8 - 15	Not used
16	`MsgLock`
7 - 24	Not used
25	`Occupied`
26	`BatchEn`
27	Batch - parameter exists
28 - 31	Not used

See also

Event16Ts block diagram (Page 1661)

Event16Ts I/Os (Page 1659)

Event16Ts messages (Page 1653)

Event16Ts error handling (Page 1653)

Event16Ts functions (Page 1650)

Event16Ts operating modes (Page 1650)

9.4.2 Event16Ts operating modes

Event16Ts operating modes

The block provides the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Event16Ts block diagram (Page 1661)

Event16Ts I/Os (Page 1659)

Event16Ts messages (Page 1653)

Event16Ts error handling (Page 1653)

Event16Ts functions (Page 1650)

Description of Event16Ts (Page 1645)

9.4.3 Event16Ts functions

Functions of Event16Ts

The functions for this block are listed below.

Activation and deactivation of messages

You can set the I/Os `InMsgEn1` ... `InMsgEn8` to individually enable or disable the messages applied to inputs `In1` ... `In8` or `InTS1` ... `InTS8`. All messages are activated by default.

To deactivate messages received at I/O `InTS4`, for example, you set I/O `InMsgEn4` = 0 accordingly.

You can deactivate all messages via I/O `MsgLock` = 1 .

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Time stamp as associated value of a message

You can use input `TimeStampOn` to select how the time stamp for the signals of Event16Ts will be generated:

- If you want to use the high-precision time stamp from the I/O devices, set `TimeStampOn` = 1. Interconnect one of the `InTSx` inputs with output `TS_Out` of block Pcs7DiIT.
- If you want to use the time stamp from the CPU, set `TimeStampOn` = 0. Interconnect one of the `Inx` inputs with output `PV_Out` of block Pcs7DiIT or with an appropriate output of a different block.

For additional time stamp properties, refer to the description of the standard function Time stamp (Page 201).

Signal status as associated value of a message

The signal status as an associated value of the message is output for every signal, as is the time stamp.

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Operator permissions

The block has the following Operator control permissions (Page 248) for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4 - 31	Not used

This block does not have a faceplate yet; the operator permissions have already been assigned in the planning phase for these faceplates.

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Configurable functions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
1	Reaction to the out of service mode (Page 176)
8	Reporting with BATCH parameters (Page 155)
9	Enable external message (Page 184)
22	Update acknowledgment and error status of the message call (Page 159)

Displaying and outputting the signal status

The block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The block determines the worst signal status via all interconnected binary inputs (according to `TimeStampOn`) and outputs this value at `ST_Worst`.

- `TimeStampOn` = 0
  - `In1`

  etc. to

  - `In16`
- `TimeStampOn` = 1
  - `InTS1`

  etc. to

  - `InTS16`

See also

Event16Ts block diagram (Page 1661)

Event16Ts I/Os (Page 1659)

Event16Ts messages (Page 1653)

Event16Ts error handling (Page 1653)

Event16Ts operating modes (Page 1650)

Description of Event16Ts (Page 1645)

9.4.4 Event16Ts error handling

Error handling of EventTs

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.

See also

Event16Ts block diagram (Page 1661)

Event16Ts I/Os (Page 1659)

Event16Ts messages (Page 1653)

Event16Ts functions (Page 1650)

Event16Ts operating modes (Page 1650)

Description of Event16Ts (Page 1645)

9.4.5 Event16Ts messages

Messaging

Messages which can be acknowledged are generated using ALARM_8P. The ALARM_8P has 8 digital inputs and 10 associated values. Every edge transition detected for one or more digital inputs results in a message. The associated values are assigned consistently to the message at the time of edge evaluation. All 8 signals are assigned a common message number, which is split at the OS into 8 messages. The Engineering System (ES) assigns the message number automatically by calling the message server.

Process messages (MsgEvId1)

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 1	AS process control message - fault	$$BlockComment$$ `InTS1` Status 16#@1%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ `InTS2` Status 16#@2%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ `InTS3` Status 16#@3%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ `InTS4` Status 16#@4%x@
	SIG 5	AS process control message - fault	$$BlockComment$$ `InTS5` Status 16#@5%x@
	SIG 6	AS process control message - fault	$$BlockComment$$ `InTS6` Status 16#@6%x@
	SIG 7	AS process control message - fault	$$BlockComment$$ `InTS7` Status 16#@7%x@
	SIG 8	AS process control message - fault	$$BlockComment$$ `InTS8` Status 16#@8%x@

Explanation:

$$BlockComment$$: Content of the instance-specific comment

The following applies for `TimeStampOn` = 0 : 16#@n%x@ (n = 1 ... 8): The value contains the signal status of `In1` ... `In8`

The following applies for `TimeStampOn` = 1 : 16#@n%x@ (n = 1 ... 8): Statement on the validity of the time stamp from `InTS0` ... `InTS8`. If the value is 80, the time stamp is formed by the I/O. If the value is ≠ 80, the time stamp of the I/O is invalid; the time stamp is generated by the CPU as a substitute and is therefore not exact.

You can change the message class and the event to meet your needs at the block type and/or block instance.

Process messages (MsgEvId2)

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 1	AS process control message - fault	$$BlockComment$$ InTS9 Status 16#@1%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ InTS10 Status 16#@2%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ InTS11 Status 16#@3%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ InTS12 Status 16#@4%x@
	SIG 5	AS process control message - fault	$$BlockComment$$ InTS13 Status 16#@5%x@
	SIG 6	AS process control message - fault	$$BlockComment$$ InTS14 Status 16#@6%x@
	SIG 7	AS process control message - fault	$$BlockComment$$ InTS15 Status 16#@7%x@
	SIG 8	AS process control message - fault	$$BlockComment$$ InTS16 Status 16#@8%x@

Explanation:

$$BlockComment$$: Content of the instance-specific comment

The following applies for TimeStampOn = 0 : 16#@n%x@ (n = 1 ... 8): The value contains the signal status of In9 ... In16

The following applies for TimeStampOn = 1 : 16#@n%x@ (n = 1 ... 8): Statement on the validity of the time stamp from InTS9 ... InTS16. If the value is 80, the time stamp is formed by the I/O. If the value is ≠ 80, the time stamp of the I/O is invalid; the time stamp is generated by the CPU as a substitute and is therefore not exact.

You can change the message class and the event to meet your needs at the block type and/or block instance.

Associated values for message instance MsgEvId1

Associated value	Block parameters
1	TimeStampOn = 0: In1.ST
2	TimeStampOn = 0: In2.ST
3	TimeStampOn = 0: In3.ST
4	TimeStampOn = 0: In4.ST
5	TimeStampOn = 0: In5.ST
6	TimeStampOn = 0: In6.ST
7	TimeStampOn = 0: In7.ST
8	TimeStampOn = 0: In8.ST
9	Not allocated
10	Not allocated

Associated values for message instance `MsgEvId2`

Associated value	Block parameters
1	`TimeStampOn` = 1: `In9.ST`
2	`TimeStampOn` = 1: `In10.ST`
3	`TimeStampOn` = 1: `In11.ST`
4	`TimeStampOn` = 1: `In12.ST`
5	`TimeStampOn` = 1: `In13.ST`
6	`TimeStampOn` = 1: `In14.ST`
7	`TimeStampOn` = 1: `In15.ST`
8	`TimeStampOn` = 1: `In16.ST`
9	Not allocated
10	Not allocated

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`TimeStampOn` = 0: `InTS1.ST`
2	`TimeStampOn` = 0: `InTS2.ST`
3	`TimeStampOn` = 0: `InTS3.ST`
4	`TimeStampOn` = 0: `InTS4.ST`
5	`TimeStampOn` = 0: `InTS5.ST`
6	`TimeStampOn` = 0: `InTS6.ST`
7	`TimeStampOn` = 0: `InTS7.ST`
8	`TimeStampOn` = 0: `InTS8.ST`
9	Not allocated
10	Not allocated

Associated values for message instance `MsgEvId2`

Associated value	Block parameters
1	`TimeStampOn` = 0: `InTS9.ST`
2	`TimeStampOn` = 0: `InTS10.ST`
3	`TimeStampOn` = 0: `InTS11.ST`
4	`TimeStampOn` = 0: `InTS12.ST`
5	`TimeStampOn` = 0: `InTS13.ST`
6	`TimeStampOn` = 0: `InTS614.ST`
7	`TimeStampOn` = 0: `InTS15.ST`
8	`TimeStampOn` = 0: `InTS16.ST`
9	Not allocated
10	Not allocated

The batch information is transmitted with `Feature` bit 8 =1:

The first three associated values are described as follows and are followed by the signal status of the input signals or the validity of the time stamp depending on `TimeStampOn`:

MsgEvId1

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchId
4	In1.ST / InTS1.ST
5	In2.ST / InTS2.ST
6	In3.ST / InTS3.ST
7	In4.ST / InTS4.ST
8	In5.ST / InTS5.ST
9	In6.ST / InTS6.ST

MsgEvId2

Associated value	Block parameters
1	BatchName
2	StepNo
3	BatchId
4	In7.ST / InTS7.ST
5	In8ST / InTS8.ST8
6	In9.ST / InTS9.ST
7	In10.ST / InTS10.ST
8	In11.ST / InTS11.ST
9	I12.ST / InTS12.ST

The tenth associated value is not available.

Enter the batch ID @1%s@ under "Properties - Block - Special properties - Messages extended - Message texts block".

The associated values in the process messages (event) must be incremented by 3:

Message instance	Message identifier	Message class	Event
MsgEvId1	SIG 1	AS process control message - fault	$$BlockComment$$ InTS1 Status 16#@4%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ InTS2 Status 16#@5%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ InTS3 Status 16#@6%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ InTS4 Status 16#@7%x@
	SIG 5	AS process control message - fault	$$BlockComment$$ InTS5 Status 16#@8%x@
	SIG 6	AS process control message - fault	$$BlockComment$$ InTS6 Status 16#@9%x@
	SIG 7	AS process control message - fault	$$BlockComment$$ InTS7 Status[1]
	SIG 8	AS process control message - fault	$$BlockComment$$ InTS8 Status[1]

[1] You cannot specify a separate associated value here.

Message instance	Message identifier	Message class	Event
MsgEvId2	SIG 1	AS process control message - fault	$$BlockComment$$ InTS9 Status 16#@4%x@
	SIG 2	AS process control message - fault	$$BlockComment$$ InTS10 Status 16#@5%x@
	SIG 3	AS process control message - fault	$$BlockComment$$ InTS11 Status 16#@6%x@
	SIG 4	AS process control message - fault	$$BlockComment$$ InTS12 Status 16#@7%x@
	SIG 5	AS process control message - fault	$$BlockComment$$ InTS13 Status 16#@8%x@
	SIG 6	AS process control message - fault	$$BlockComment$$ InTS14 Status 16#@9%x@
	SIG 7	AS process control message - fault	$$BlockComment$$ InTS15 Status[1]
	SIG 8	AS process control message - fault	$$BlockComment$$ InTS16 Status[1]

[1] You cannot specify a separate associated value here.

See also

Event16Ts block diagram (Page 1661)

Event16Ts I/Os (Page 1659)

Event16Ts error handling (Page 1653)

Event16Ts functions (Page 1650)

Event16Ts operating modes (Page 1650)

Description of Event16Ts (Page 1645)

Selecting values associated with messages (Page 155)

9.4.6 Event16Ts I/Os

I/Os of EventTs

Inputs

Parameter	Description	Type	Default
`BatchEn`	1 = Enable allocation	BOOL	0
`BatchID`	Batch ID	DWORD	16#00000000
`BatchName`	Batch name	S7-string	
`EN`	1 = Called block will be processed	BOOL	1
`Feature`	I/O for additional functions (Page 1650)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`ExtVal01`	External message 1	ANY	
`ExtVal02`	External message 2	ANY	
`ExtVal03`	External message 3	ANY	
`ExtVal04`	External message 4	ANY	
`ExtVal05`	External message 5	ANY	
`ExtVal06`	External message 6	ANY	
`In1..In16`	Signal 1..16 for CPU time stamp	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#FF
`InTS1..InTS16`	Signal 1..16 with I/O-device time stamp	ANY	
`Inv1..Inc16`	1 = Invert input `In1..In16` or `InTS1..InTS16`	BOOL	0
`MsgEn1..MsgEn16`	1 = Activate message for input `In1..In16` or `InTS1..InTS16`	BOOL	1

Parameter	Description	Type	Default
MsgEvId1	Message number (assigned automatically)	DWORD	16#00000000
MsgEvId2	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp	1 = Release for maintenance by OS operator	BOOL	0
Occupied	1 = In use by a batch	BOOL	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OS_Perm	I/O for Event16Ts functions (Page 1650)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
SelFp2	1 = Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
StepNo	Batch step number	DWORD	16#00000000
TimeStampOn	0 = Use time stamp of the CPU 1 = Use time stamp of the I/O devices	BOOL	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00000000

* Values can be written back to these inputs during processing of the block by the block algorithm.

Outputs

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see Event16Ts error handling (Page 1653)	INT	-1
EventTsOut	For maneuvering of data between a technological block and the message blocks EventTs, Event16Ts.	STRUCT	
MS_Release	Release for maintenance: 1 = Release for OS operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn1..MsgAckn2	Message acknowledgment status (output ACK_STATE of ALARM_8P)	WORD	16#0000
MsgErr1..MsgErr2	1 = Message error (output ERROR of ALARM_8P)	BOOL	0
MsgStat1..MsgStat2	Message status (output STATUS of ALARM_8P)	WORD	16#0000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1645)	DWORD	16#00000000
Status2	Status word 2 (Page 1645)	DWORD	16#00000000
Status3	Status word 3 (Page 1650)	DWORD	16#00000000
Status4	Status word 4 (Page 1650)	DWORD	16#00000000

See also

Event16Ts block diagram (Page 1661)

Event16Ts messages (Page 1653)

9.4.7 Event16Ts block diagram

EventTs block diagram

A block diagram is not provided for this block.

See also

Event16Ts I/Os (Page 1659)
Event16Ts messages (Page 1653)
Event16Ts error handling (Page 1653)
Event16Ts functions (Page 1650)
Event16Ts operating modes (Page 1650)
Description of Event16Ts (Page 1645)

Counter blocks

10

10.1 CountScL - Counter with up and down counting direction

10.1.1 Description of CountScL

Object name (type + number) and family

Type + number: FB 1806

Family: Count

Area of application for CountScL

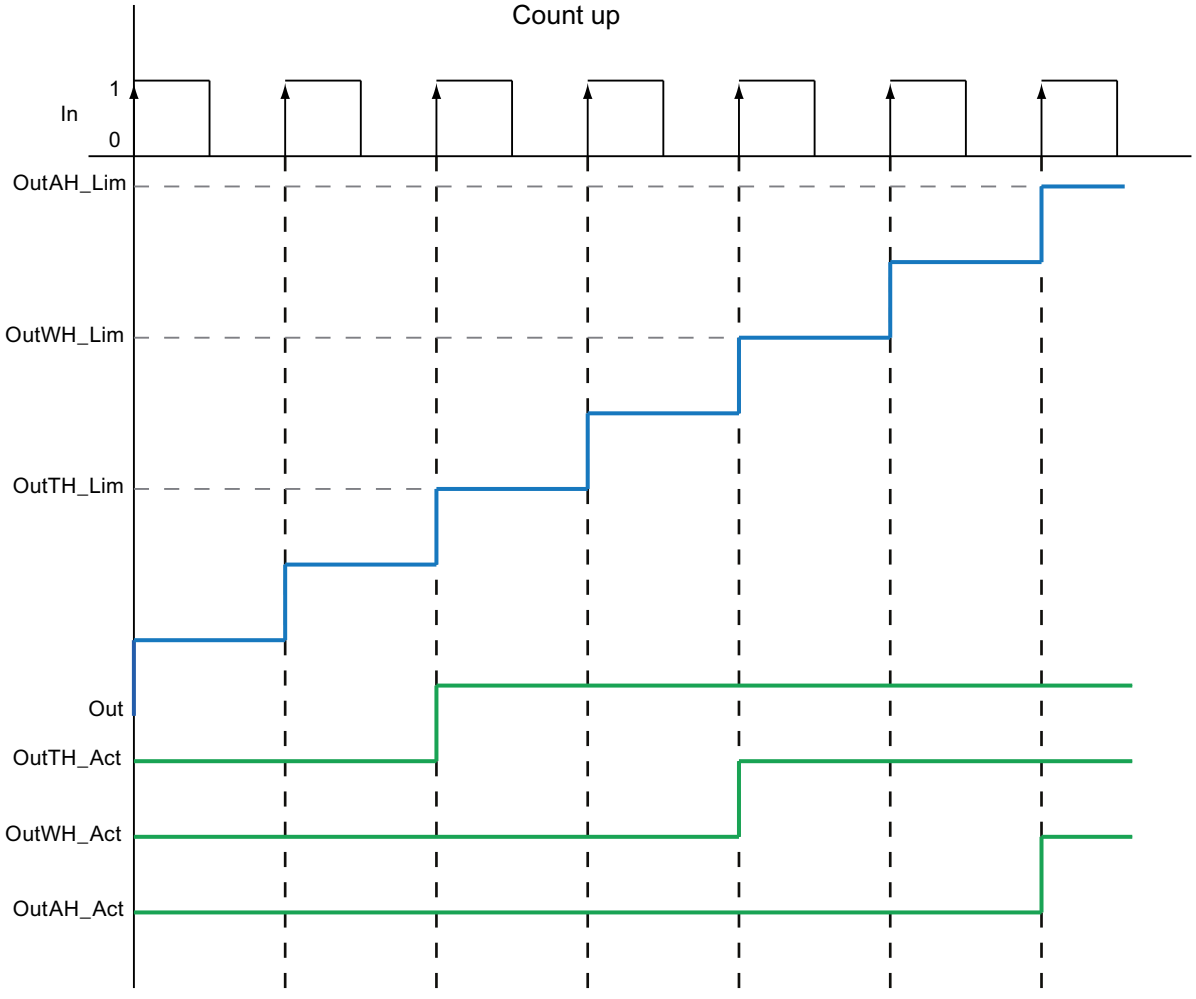
The block is used for the following applications:

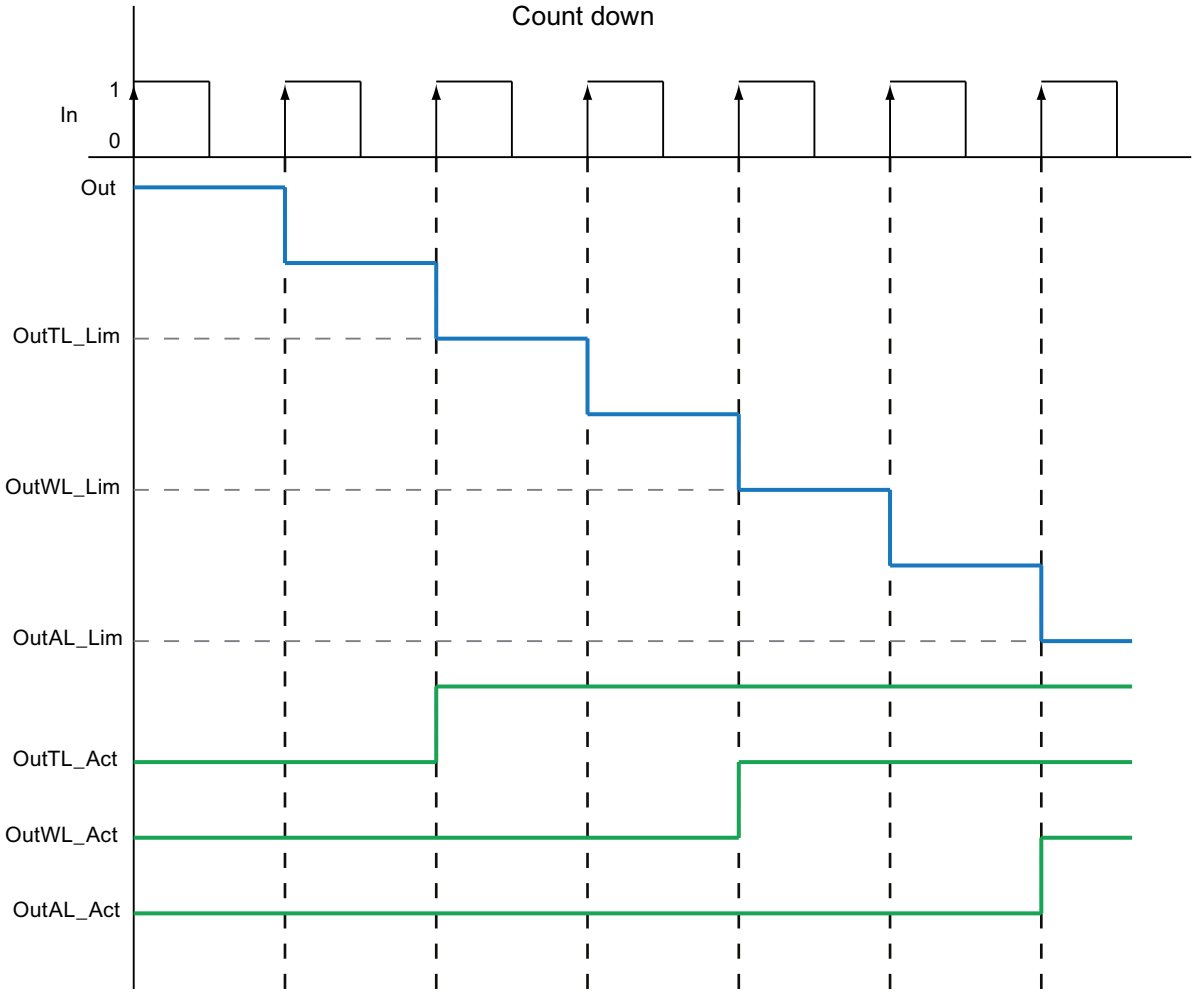
- Count up or count down with rising edge of the binary input signal.

How it works

Positive edges at the binary input signal In increment or decrement the count value at Out, depending on settings.

1. Count up (`UpOp` = 1 or `UpLi` =1)
   The block counts up for each rising edge of `In`. (Output parameter `CountMode` = 1)
2. Count down (`DnOp` = 1 or `DnLi` = 1)
   The block counts down for each rising edge of `In`. (Output parameter `CountMode` = 2)
3. Off (`OffOp` = 1 or `OffLi` = 1)
   The block is disabled (output parameter `CountMode` = 0). No counting takes place.





Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Define the startup characteristics for this block via two `Feature` Bits:

Bit 0: Set startup characteristics (Page 137)

Bit 5: Use the last value following a complete download as the current value during startup of the block (Page 153)

Time response

The block does not have any time response.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section CountScL I/Os (Page 1673).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`
7 - 10	Not used
11	`LiOp`
12 - 13	Not used
14	1 = Invalid signal status
15 - 31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock.Value`
1	`OutAH_Act.Value`
2	`OutWH_Act.Value`
3	`OutTH_Act.Value`
4	`OutTL_Act.Value`
5	`OutWL_Act.Value`
6	`OutAL_Act.Value`
7	`OutAH_En`
8	`OutWH_En`
9	`OutTH_En`
10	`OutTL_En`
11	`OutWL_En`
12	`OutAL_En`
13	`OutAH_MsgEn`
14	`OutWH_MsgEn`
15	`OutTH_MsgEn`
16	`OutTL_MsgEn`
17	`OutWL_MsgEn`
18	`OutAL_MsgEn`
19	Not used
20	Count up
21	Counter off
22	Count down

Status bit	Parameter
23 - 30	Not used
31	MS_RelOp

See also

CountScL functions (Page 1668)

CountScL messaging (Page 1672)

CountScL block diagram (Page 1676)

CountScL error handling (Page 1671)

CountScL modes (Page 1667)

10.1.2 CountScL modes

CountScL operating modes

This block provides the following operating modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

CountScL block diagram (Page 1676)

CountScL I/Os (Page 1673)

CountScL messaging (Page 1672)

CountScL error handling (Page 1671)

Description of CountScL (Page 1663)

CountScL functions (Page 1668)

10.1.3 CountScL functions

Functions of CountScL

The functions for this block are listed below.

Limit monitoring of the count value

This block provides the standard function Limit monitoring of the count value (Page 89).

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Read back the last counted value

When counting (visible at the `Out` output parameter), this count value is passed directly to the `OldOut` input parameter:

`OldOut` = `Out`If a warm restart is performed at this point, the count value (`Out`) is automatically reset to the default value, if the `Feature` Bit Set startup characteristics (Page 137) is set accordingly (= 0) (`OldOut` ≠ `Out`). In this case, the value from `OldOut` is only updated when the (`Out`) count value is changed again by a pulse. `OldOut` = `Out` now applies again.

Reset counter to zero

The counted value at the `Out` output parameter is reset with the interconnectable `ResetCount` parameter. The reset is made with a 0 - 1 edge.

The count cannot be reset to zero from the faceplate.

You can use Feature bit 30 "Set reset depending on operating mode or the `LiOp` parameter" to configure a reset depending on the `LiOp` parameter:

Feature bit 30 =0: The reset does not depend on `LiOp`

Feature bit 30 =1: The reset is only possible when `LiOp =1`.

Setting the count value to the default setting

You can use the `PresetVal` input parameter to enter a value at which the count should start, if the command has been given for setting the count to the default via the `PresetEn` input parameter with 1. This can also be done with the faceplate.

In this case, the `Out` output parameter is set to the `PresetVal` value.

You can use Feature bit 30 "Set reset depending on operating mode or the `LiOp` parameter" to configure the setting of the count value to a default depending on the `LiOp` parameter:

Feature bit 30 =0: Setting the count value to the default setting does not depend on `LiOp`

Feature bit 30 =1: Setting the count value to the default is only possible if `LiOp =0`.

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameter:

- `In.ST`
- OutST

The following signal status is formed by:

Signal status	Used status
`OutST`	Count up, count down: Formed from the signal status `In.ST` and the last signal status `OutST`. Counter off: Signal status `OutST` is frozen. Resetting CountScL also resets the signal status: `OutST` := 16#80

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` I/O in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
5	Use the last value following a complete download as the current value during startup of the block (Page 153)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can stop counting
5	1 = Operator can switch to incrementing mode
6	1 = Operator can switch to decrementing mode
7 - 11	Not used
12	1 = Operator can activate the Release for maintenance function
13	1 = Operator can change the limit (`OutAH_Lim`) for high alarm
14	1 = Operator can change the limit (`OutWH_Lim`) for high warning
15	1 = Operator can change the limit (`OutTH_Lim`) for high tolerance
16	Not used
17	1 = Operator can change the limit (`OutAL_Lim`) for low alarm
18	1 = Operator can change the limit (`OutWL_Lim`) for low warning
19	1 = Operator can change the limit (`OutTL_Lim`) for low tolerance
20	Not used
21	1 = Operator can set the count to the default value (`PresetEn`)
22	1 = Operator can specify the default value (`PresetTime`)
23	1 = Operator can activate / deactivate messages via `OutAH_MsgEn`
24	1 = Operator can activate / deactivate messages via `OutWH_MsgEn`
25	1 = Operator can activate / deactivate messages via `OutTH_MsgEn`
26	1 = Operator can activate / deactivate messages via `OutTL_MsgEn`
27	1 = Operator can activate / deactivate messages via `OutWL_MsgEn`
28	1 = Operator can activate / deactivate messages via `OutAL_MsgEn`
29 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of CountScL (Page 1663)

CountScL messaging (Page 1672)

CountScL I/Os (Page 1673)

CountScL block diagram (Page 1676)

CountScL error handling (Page 1671)

CountScL modes (Page 1667)

10.1.4 CountScL error handling

Error handling of CountScL

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
51	Invalid signal at `LiOp` = 1: • `OffLi` = 1 and `UpLi` = 1 and/or `DnLi` = 1 • `OffLi` = 0 and `UpLi` = 1 and `DnLi` = 1

See also

CountScL block diagram (Page 1676)

CountScL I/Os (Page 1673)

CountScL messaging (Page 1672)

Description of CountScL (Page 1663)

CountScL modes (Page 1667)

CountScL functions (Page 1668)

10.1.5 CountScL messaging

Messaging

The following messages can be generated for this block:

- Functions for displaying measured limits

Messages generated as a reaction to limit violations, can be suppressed by the settings `MsgEn` and `MsgLock`.

Associated values for message instance `MsgEvId`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`

Process messages

Message instance	Message identifier	Message class	Event
`MsgEvId`	SIG 1	Alarm - high	$$BlockComment$$ High alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ High warning limit violated
	SIG 3	Tolerance - high	$$BlockComment$$ High tolerance limit violated
	SIG 4	Tolerance - low	$$BlockComment$$ Low tolerance limit violated
	SIG 5	Warning - low	$$BlockComment$$ Low warning limit violated
	SIG 6	Alarm - low	$$BlockComment$$ Low alarm limit violated
	SIG 7	Reserved	$$BlockComment$$
	SIG 8	Reserved	$$BlockComment$$

Explanation:

$$BlockComment$$: Content of the instance-specific comment

See also

Description of CountScL (Page 1663)

CountScL functions (Page 1668)

CountScL I/Os (Page 1673)

CountScL block diagram (Page 1676)

CountScL modes (Page 1667)

CountScL error handling (Page 1671)

10.1.6 CountScL I/Os

I/Os of CountScL

Input parameters

Parameter	Description	Type	Default
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
DnLi	1 = Down counter, via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DnOp*	1 = Down counter, via operator	BOOL	0
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1668)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
In	Binary input value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LiOp	1 = Interconnection 0 = Operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgEvId	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp*	Operator can switch release for maintenance	BOOL	0
Occupied	1 = In use by a batch	BOOL	0
OffLi	1 = Counter disabled via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OffOp*	1 = Counter disabled via operator	BOOL	1
OldOut*	Previous output value	DINT	0
OnOp*	1 = "On" mode via operator	BOOL	0

Parameter	Description	Type	Default
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator permissions (Page 1668)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OutAH_En	1 = Enable alarm (high) for count	BOOL	1
OutAH_Lim	Limit for count alarm (high)	DINT	95
OutAH_MsgEn	1 = Enable message for count alarm (high)	BOOL	1
OutAL_En	1 = Enable alarm (low) for count	BOOL	1
OutAL_Lim	Limit for count alarm (low)	DINT	0
OutAL_MsgEn	1 = Enable message for count alarm (low)	BOOL	1
OutOpHiScale	High limit for scale in count bar graph of faceplate	DINT	100
OutOpLoScale	Low limit for scale in count bar graph of faceplate	DINT	0
OutTH_En	1 = Enable tolerance message (high)	BOOL	0
OutTH_Lim	Count tolerance limit message (high)	DINT	85
OutTH_MsgEn	1 = Enable message for count tolerance message (high)	BOOL	1
OutTL_En	1 = Enable count tolerance message (low)	BOOL	0
OutTL_Lim	Count tolerance message limit (low)	DINT	0
OutTL_MsgEn	1 = Enable message for count tolerance message (low)	BOOL	1
OutUnit	Unit of measure for count Out	INT	0
OutWH_En	1 = Enable count warning (high)	BOOL	1
OutWH_Lim	Count warning limit (high)	DINT	90
OutWH_MsgEn	1 = Enable message for count warning (high)	BOOL	1
OutWL_En	1 = Enable count warning (low)	BOOL	1
OutWL_Lim	Count warning limit (low)	DINT	0
OutWL_MsgEn	1 = Enable message for count warning (low)	BOOL	1
PresetEn*	1 = Set block to default count (PresetVal)	BOOL	0
PresetVal	Default setting for the count	DINT	0
ResetCount	1 = Counter restart	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RunUpCyc	Number of operating cycles for which all messages are suppressed	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	

Parameter	Description	Type	Default
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	
StepNo	Batch step number	DWORD	16#00000000
UpLi	1 = Forward counter, via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
UpOp*	1 = Forward counter, via operator	BOOL	0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
CountMode	0 = Counter off 1 = Count up 2 = Count down	INT	0
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see CountScL error handling (Page 1671)	INT	-1
MsgAckn	Message acknowledgement status (output ACK_STATE of first ALARM_8P)	WORD	16#0000
MsgErr	1 = Message error (output ERROR of the first ALARM_8P)	BOOL	0
MsgStat	Message status (output STATUS of the first ALARM_8P)	WORD	16#0000
MS_Release	Release for maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
O_MS_Ext	Reserved	DWORD	0
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
Out	Count	DINT	0

Parameter	Description	Type	Default
OutAH_Act	1 = Count alarm (high) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutAL_Act	1 = Count alarm (low) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutST	Signal status for Out	BYTE	16#80
OutTH_Act	1 = Count tolerance message (high) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutTL_Act	1 = Count tolerance message (low) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutWH_Act	1 = Count warning (high) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutWL_Act	1 = Count warning (low) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1663)	DWORD	16#00000000
Status2	Status word 2 (Page 1663)	DWORD	16#00000000

See also

CountScL messaging (Page 1672)

CountScL block diagram (Page 1676)

CountScL modes (Page 1667)

10.1.7 CountScL block diagram

CountScL block diagram

A block diagram is not provided for this block.

See also

Description of CountScL (Page 1663)

CountScL modes (Page 1667)

CountScL functions (Page 1668)

CountScL error handling (Page 1671)

CountScL messaging (Page 1672)

CountScL I/Os (Page 1673)

10.1.8 Operator control and monitoring

10.1.8.1 CountScL views

Views of the CountScL block

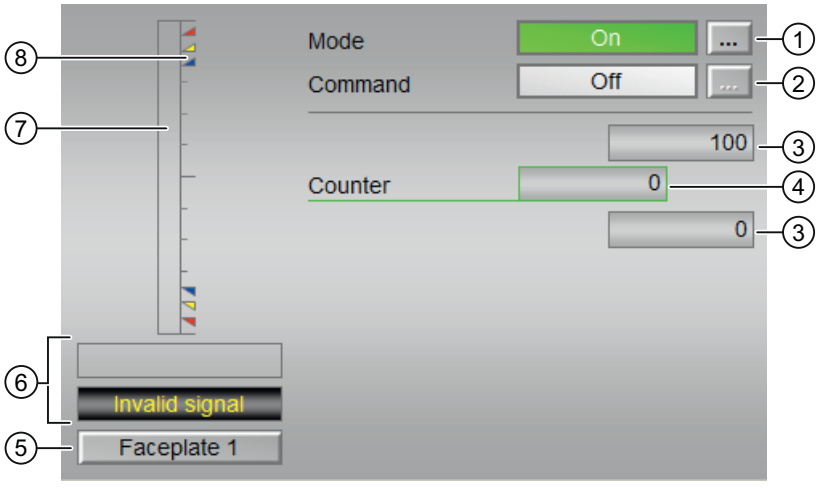
The block CountScL provides the following views:

- CountScL standard view (Page 1678)
- Alarm view (Page 296)
- CountScL limit view (Page 1680)
- Trend view (Page 299)
- CountScL parameter view (Page 1681)
- CountScL preview (Page 1682)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for CountScL (Page 1683)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

10.1.8.2 CountScL standard view

CountScL standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) Enabling and disabling the counter

This area shows you the default operating state for the counter. The following states can be shown and executed here:

- "On ⇑"
- "On ⇓"
- "Off"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

(3) High and low scale range for the count value

These values provide information on the display range for the bar graph of the count. The scale range is defined in the engineering system.

(4) Display of the count value

This area shows the current count value with the signal status `OutST`.

The name for the current count value can be set using the `s7_shortcut` attribute at the corresponding output parameter. The default text is displayed if nothing is specified.

You can find additional information on this in the section Labeling of buttons and text (Page 205) .

(5) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

(6) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"
- "Invalid signal"

(7) Graphic display of the current count value

This area shows you the current count in form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(8) Limit display

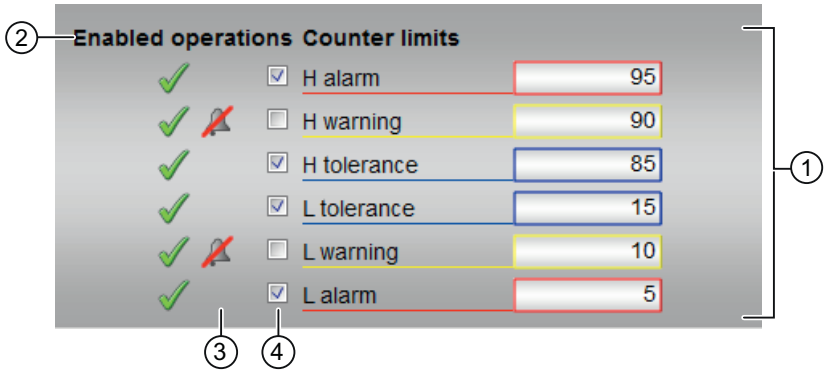
These colored triangles indicate the configured limits in the respective bar graph:

- Red: Alarm
- Yellow: Warning
- Blue: Tolerance

In counting mode "count up", only the colored triangles of the upper limits are visible; and in counting mode "count down", only the colored triangles of the lower limits are visible.

10.1.8.3 CountScL limit view

Limit view of CountScL



(1) Limits for the counter

In this area, you can enter the limits for the counter. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "H warning": Warning high
- "H tolerance": Tolerance high
- "L tolerance": Tolerance low
- "L warning": Warning low
- "L alarm": Alarm low

(2) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

(3) "Message suppression"

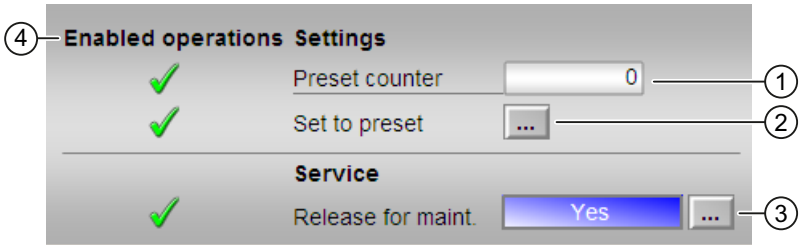
Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

(4) Suppress messages

You can enable / disable messages by setting the check mark.

10.1.8.4 CountScL parameter view

Parameter view of CountScL



(1) Preset counter

Enter the default setting here, where the counter should start. You can find additional information on this in the Changing values (Page 253) section.

(2) Set to default

Set the counter to the default value here. You can find additional information on this in the Switching operating states and operating modes (Page 251) section.

(3) Service

You can select the following function in this area:

- "Release for maintenance"

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the Release for maintenance (Page 64) section.

(4) Enabled operations

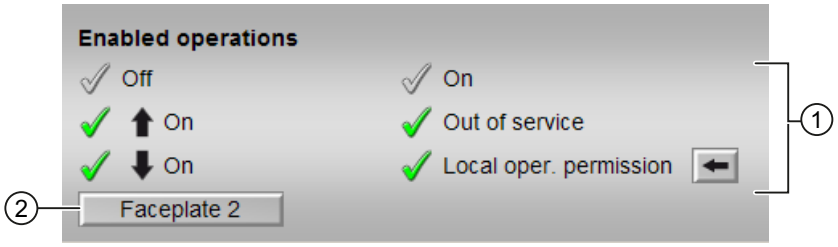
This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

10.1.8.5 CountScL preview

Preview of CountScL



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Off": You can disable the counter.
- "On ⇑": You can operate the incremental counter.
- "On ⇓": You can operate the decremental counter.
- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

(2) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the Opening additional faceplates (Page 203) section.

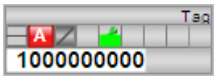
10.1.8.6 Block icon for CountScL

Block icons for CountScL

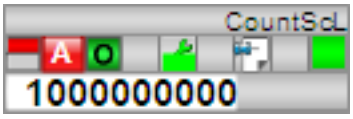
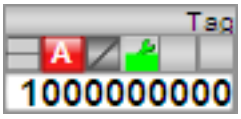
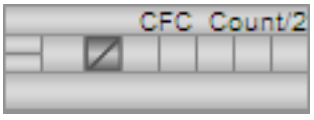
A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits
- Operating modes
- Signal status, release for maintenance
- Memo display
- Display counter running

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in the CFC	Special features
	1	
	2	
	3	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

10.2 CountOh - Determining runtime

10.2.1 Description of CountOh

Object name (type + number) and family

Type + number: FB 1864

Family: Count

Area of application for CountOh

The block is used for the following applications:

- Calculating the runtime of a unit

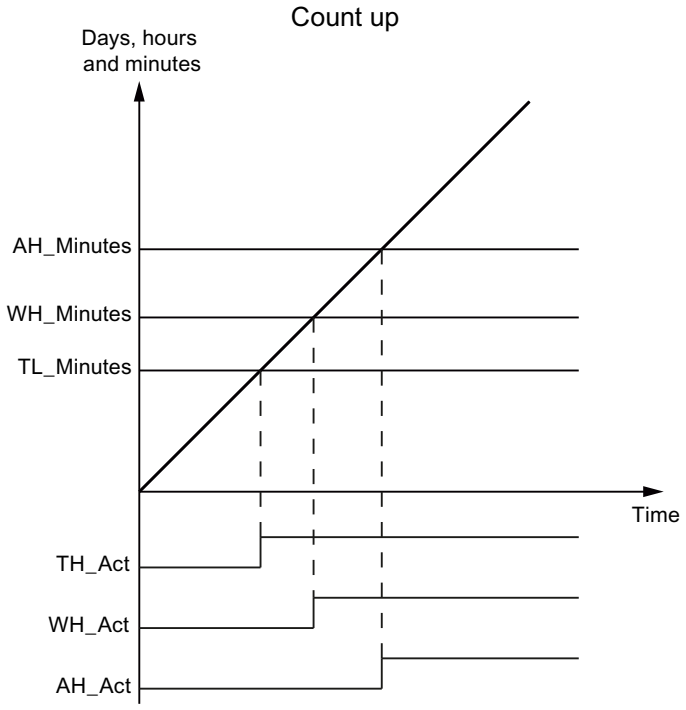
How it works

The block calculates the operating time of a unit.

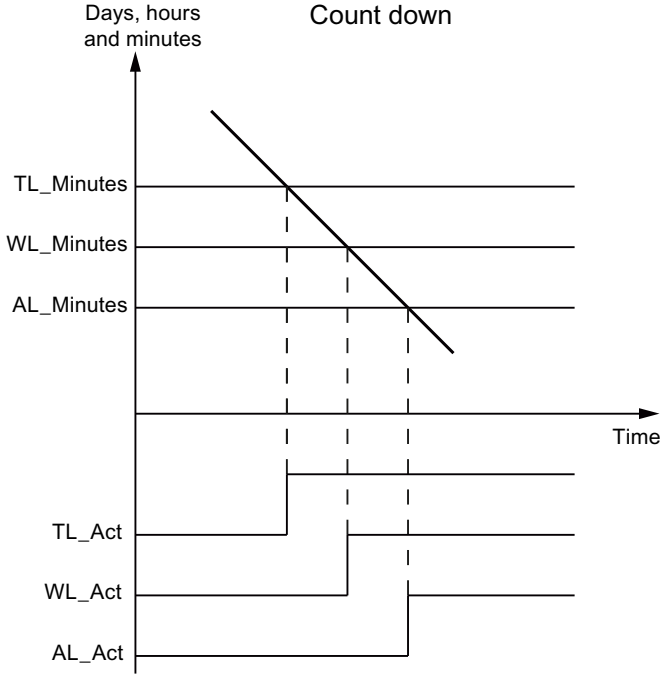
1. Off (`OffOp` = 1 or `OffLi` = 1)
   The block is disabled (output parameter `CountMode` = 0). No counting takes place.
2. Count up (`UpOp` = 1 or `UpLi` = 1)
   The operating time of the connected unit is incremented (output parameter `CountMode` = 1).
3. Count down (`DnOp` = 1 or `DnLi` = 1)
   The operating time of the connected unit is decremented (output parameter `CountMode` = 2).

The operating time is shown depending on the course of days, hours, minutes and seconds.

If the operating time exceeds the configured limits, an alarm is triggered.



When the block counts up and the total operating time `(TimeMin)` of the connected unit is greater than or equal to the limits (`TH_Minutes`, `WH_Minutes` and `AH_Minutes`), the alarms `TH_Act`, `WH_Act` and `AH_Act` are set.



When the block counts up and the total operating time `(TimeMin)` of the connected unit is greater than or equal to the limits (`TL_Minutes`, `WL_Minutes` and `AL_Minutes`), the alarms `TL_Act`, `WL_Act` and `AL_Act` are set.

The operating time can be preset. The value can be specified by the operator. The operator can set values for days, hours and minutes to begin counting, if the required operator control permission has been issues for this.

The maximum configurable value for the operating time is 24855 days, 3 hours and 14 minutes.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is first installed automatically in the startup OB (OB 100).

Startup characteristics

Define the startup characteristics for this block via two `Feature` Bits:

Bit 0: Set startup characteristics (Page 137)

Bit 5: Use the last value following a complete download as the current value during startup of the block (Page 153)

When `Feature` bit 5 is set to 1, then:

- `Days := OldDays`
- `Hours := OldHours`
- `Minutes := OldMinutes`
- `Seconds := OldSeconds`

When you set `Feature` bit 5 and `Feature` bit 0 to 1, then:

- `TotalTime := PresetTime`

The output parameters `Days`, `Hours`, `Minutes` are converted accordingly based on `PresetTime`.

When you set `Feature` bit 5 to 0 and `Feature` bit 0 to 1, no preset is made.

Time response

The block only functions properly in a cyclic interrupt OB. To ensure correct time acquisition, it should be installed (if configured in CFC) in the same runtime group as the control block of the monitored unit.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section CountOh I/Os (Page 1696).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	Not used
3	`OosAct.Value`
4	`OosLi.Value`

Status bit	Parameter
5	Not used
6	OnAct.Value
7 - 10	Not used
11	LiOp
12 - 13	Not used
14	1 = Invalid signal status
15 - 31	Not used

Status word allocation for Status2 parameter

Status bit	Parameter
0	MsgLock.Value
1	AH Act.Value
2	WH_Act.Value
3	TH_Act.Value
4	TL_Act.Value
5	WL_Act.Value
6	AL_Act.Value
7	AH_En
8	WH_En
9	TH_En
10	TL_En
11	WL_En
12	AL_En
13	AH_MsgEn
14	WH_MsgEn
15	TH_MsgEn
16	TL_MsgEn
17	WL_MsgEn
18	AL_MsgEn
19	Not used
20	Count up
21	Counter off
22	Count down
23 - 30	Not used
31	MS_RelOp

See also

CountOh functions (Page 1689)
CountOh messaging (Page 1694)
CountOh block diagram (Page 1700)

CountOh error handling (Page 1694)

CountOh modes (Page 1689)

10.2.2 CountOh modes

CountOh operating modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

You can find general information about the "On" mode in the On (Page 71) section.

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

CountOh block diagram (Page 1700)

CountOh I/Os (Page 1696)

CountOh messaging (Page 1694)

CountOh error handling (Page 1694)

Description of CountOh (Page 1685)

CountOh functions (Page 1689)

10.2.3 CountOh functions

Functions of CountOh

The block provides the following functions:

Limit monitoring of the operating time

The total time in minutes `(TimeIn)` is displayed for the following limits:

1. Alarm (`AH_Minutes` and `AL_Minutes`)
2. Warning (`WH_Minutes` and `WL_Minutes`)
3. Tolerance (`TH_Minutes` and `TL_Minutes`)

Note

Limit monitoring:

Limit monitoring depends on the counting direction:

- Operation mode 1 (counting up), limit monitoring top
  - `AH_Minutes`
  - `WH_Minutes`
  - `TH_Minutes`
- Operation mode 2 (counting down), limit monitoring bottom
  - `AL_Minutes`
  - `WL_Minutes`
  - `TL_Minutes`

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
5	Use the last value following a complete download as the current value during startup of the block (Page 153)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)

Bit	Function
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)

Read back the last counted value

When counting (`DeviceOn.Value = 1` and `CountMode = 1` or `2`), the inputs of the most recently valid operating time are updated:

- `OldDays := Days`
- `OldHours := Hours`
- `OldMinutes := Minutes`
- `OldSeconds := Seconds`

The inputs `OldDays`, `OldHours`, `OldMinutes`, `OldSeconds` are read back with a complete download.

After a warm restart, the outputs of the operating time are set depending on the configuration of the `Feature` parameter.

When `Feature` bit 5 is set to 1, then:

- `Days := OldDays`
- `Hours := OldHours`
- `Minutes := OldMinutes`
- `Seconds := OldSeconds`

When you set `Feature` bit 5 and `Feature` bit 0 to 1, then:

- `TotalTime := PresetTime`

The output parameters `Days`, `Hours`, `Minutes` are converted accordingly based on `PresetTime`.

When you set `Feature` bit 5 to 0 and `Feature` bit 0 to 1, no preset is made.

Display and operator input area for process values and setpoints

This block provides the standard function Display and operator input area for process values and setpoints (Page 203).

Reset counter to zero

The output parameters of the operating time are reset via the interconnectable `Reset` parameter. The reset is made with a 0 - 1 edge.

The count cannot be reset to zero from the faceplate.

You can use Feature bit 30 "Set reset depending on operating mode or the LiOp parameter" to configure a reset depending on the LiOp parameter:

Feature bit 30 =0: The reset does not depend on `LiOp`

Feature bit 30 =1: The reset is only possible when `LiOp =1`

Setting the count value to the default setting

You can enter the operating time at which counting should begin with the input parameters `PresetTime` and `PresetEn`.

When `PresetEn` is set, the following applies:

- `TotalTime := PresetTime`

The outputs `Days`, `Hours`, `Minutes` are converted accordingly based on `PresetTime`. `PresetTime` can also be entered in the faceplate.

The `PresetTime` time value is always indicated in seconds.

You can use the `Feature` Bit 30 "Set reset depending on operating mode or the `LiOp` parameter" to configure the setting of the count value to a default depending on the `LiOp` parameter:

`Feature` Bit 30 =0: Setting to the default does not depend on `LiOp`

`Feature` Bit 30 =1: Setting to the default is only possible if `LiOp` =0.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameter:

- `In.ST`
- `TmMinSt`

The following signal status is formed by:

Signal status	Used status
`TmMinST`	Count up, count down: Formed from the signal status `In.ST` and the last signal status `TmMinST`. Counter off: Signal status TmMinST is frozen. Resetting CountOh also resets the signal status: `TmMinST` := 16#80

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can start the block

Bit	Function
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can stop the block
5	1 = Operator can switch to incrementing mode
6	1 = Operator can switch to decrementing mode
7 - 11	Not used
12	1 = Operator can enable release for maintenance
13	1 = Operator can enter AH_Lim
14	1 = Operator can enter WH_Lim
15	1 = Operator can enter TH_Lim
16	Not used
17	1 = Operator can enter AL_Lim
18	1 = Operator can enter WL_Lim
19	1 = Operator can enter TL_Lim
20	Not used
21	1 = Operator can enable default
22	1 = Operator can apply configured time
23	1 = Operator can activate / deactivate messages via AH_MsgEn
24	1 = Operator can activate / deactivate messages via WH_MsgEn
25	1 = Operator can activate / deactivate messages via TH_MsgEn
26	1 = Operator can activate / deactivate messages via TL_MsgEn
27	1 = Operator can activate / deactivate messages via WL_MsgEn
28	1 = Operator can activate / deactivate messages via AL_MsgEn
29 - 31	Not used

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of CountOh (Page 1685)

CountOh messaging (Page 1694)

CountOh I/Os (Page 1696)

CountOh block diagram (Page 1700)

CountOh error handling (Page 1694)

CountOh modes (Page 1689)

10.2.4 CountOh error handling

Error handling of CountOh

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
51	Invalid signal at `LiOp` = 1 • `OffLi` = 1 and `UpLi` = 1 and/or `DnLi` = 1 • `OffLi` = 0 and `UpLi` = 1 and `DnLi` = 1

See also

CountOh block diagram (Page 1700)

CountOh I/Os (Page 1696)

CountOh messaging (Page 1694)

CountOh functions (Page 1689)

CountOh modes (Page 1689)

Description of CountOh (Page 1685)

10.2.5 CountOh messaging

Messaging

The following messages can be generated for this block:

- Functions for displaying measured limits

Messages generated as a reaction to limit violations, can be suppressed by the settings `xx_MsgEn` and `MsgLock`.

Process messages

Message instance	Message identifier	Message class	Event
`MsgEvId 1`	SIG 1	Alarm - high	$$BlockComment$$ High alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ High warning limit violated
	SIG 3	Tolerance - high	$$BlockComment$$ High tolerance limit violated
	SIG 4	Tolerance - low	$$BlockComment$$ Low tolerance limit violated
	SIG 5	Warning - low	$$BlockComment$$ Low warning limit violated
	SIG 6	Alarm - low	$$BlockComment$$ Low alarm limit violated
	SIG 7	Reserved	$$BlockComment$$ For internal use
	SIG 8	Reserved	$$BlockComment$$ For internal use

Explanation:

$$BlockComment$$: Content of the instance-specific comment

Associated values for message instance `MsgEvId1`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`

See also

Description of CountOh (Page 1685)

CountOh functions (Page 1689)

CountOh I/Os (Page 1696)

CountOh block diagram (Page 1700)

CountOh error handling (Page 1694)

CountOh modes (Page 1689)

10.2.6 CountOh I/Os

I/Os of CountOh

Input parameters

Parameter	Description	Type	Default
AH_En	High alarm enabled	BOOL	1
AH_MsgEn	Message for high alarm enabled	BOOL	1
AL_En	Low alarm enabled	BOOL	1
AL_MsgEn	Message for low alarm enabled	BOOL	1
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch number (Batch ID)	DWORD	16#00000000
BatchName	Batch name	S7-String	
DayOpHiScale*	Operating stage - High limit of bar display for OS	INT	100
DayOpLoScale*	Operating stage - Low limit of bar display for OS	INT	0
DaysAHLim*	Days - high limit alarm	INT	95
DaysALLim*	Days - low limit alarm	INT	0
DaysTHLim*	Days - high limit tolerance	INT	85
DaysTLLim*	Days - low limit tolerance	INT	0
DaysWHLim*	Days - limit for high warning	INT	90
DaysWLLim*	Days - limit for low warning	INT	0
DnLi	1 = Decrement via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DnOp*	1 = Decrement via OS operator	BOOL	0
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1689)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
HrsAHLim*	Hours - high limit alarm	INT	0
HrsALLim*	Hours - low limit alarm	INT	0
HrsOpHiScale*	Operating hours - high limit of bar display for OS	INT	23
HrsOpLoScale*	Operating hours - low limit of bar display for OS	INT	0
HrsTHLim*	Hours - high limit tolerance	INT	0
HrsTLLim*	Hours - low limit tolerance	INT	0
HrsWHLim*	Hours - high limit warning	INT	0
HrsWLLim*	Hours - low limit warning	INT	0

Parameter	Description	Type	Default
In	Device status: 1=On 0=Off	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LiOp	Switchover of operating mode between: 1 = Interconnection 0 = Operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MinOpHiScale*	Operating minutes - high limit of bar display for OS	INT	59
MinOpLoScale*	Operating minutes - low limit of bar display for OS	INT	0
MinsAHLim*	Minutes - high limit alarm	INT	0
MinsALLim*	Minutes - low limit alarm	INT	0
MinsTHLim*	Minutes - high limit tolerance	INT	0
MinsTLLim*	Minutes - low limit tolerance	INT	0
MinsWHLim*	Minutes - high limit warning	INT	0
MinsWLLim*	Minutes - low limit warning	INT	0
MS_RelOp*	Operator input for release for maintenance, 1: Release for maintenance demand	BOOL	0
MsgEvId	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1= Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Occupied	1 = In use by a batch	BOOL	0
OffLi	Counter disabled via interconnection: 1 = Off	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OffOp*	Counter disabled via operator: 1 = Off	BOOL	1
OldDays*	Previous day value	INT	0
OldHours*	Previous hour value	INT	0
OldMinutes*	Previous minute value	INT	0
OldSeconds*	Previous second value	INT	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000

Parameter	Description	Type	Default
OS_Perm	I/O for CountOh functions (Page 1689)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
PresetEn*	1 = Set block to default time time value (PresetTime)	BOOL	0
PresetTime*	Default setting for the time value [s]	DWORD	16#00000000
Reset	1 = Reset counter	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s]	REAL	0.1
SelFp1	Open faceplate 1	ANY	
SelFp2	Open faceplate 2	ANY	
StepNo	Batch step number	DWORD	16#00000000
TH_En	High tolerance enabled	BOOL	0
TH_MsgEn	Alarm for high tolerance enabled	BOOL	1
TL_En	Low tolerance enabled	BOOL	0
TL_MsgEn	Alarm for low tolerance enabled	BOOL	1
UpLi	1 = Increment (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
UpOp*	1 = Increment (via faceplate)	BOOL	0
WH_En	High warning enabled	BOOL	1
WH_MsgEn	Alarm for high warning enabled	BOOL	1
WL_En	Low warning enabled	BOOL	1
WL_MsgEn	Alarm for low warning enabled	BOOL	1

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
AH_Act	High alarm enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AH_Minutes	High alarm time [in min]	DINT	0

Parameter	Description	Type	Default
AL_Act	Low alarm enabled You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
AL_Minutes	Low alarm time [in min]	DINT	0
BarOpHiScale	High limit for bar display [min]	DINT	100
BarOpLoScale	Low limit for bar display [min]	DINT	0
CountMode	Count mode: 0 = Off 1 = Increment 2 = Count down	INT	0
Days	Operating stage	INT	0
DeviceOn	1 = Unit on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see CountOh error handling (Page 1694)	INT	-1
Hours	Operating hours	INT	0
Minutes	Operating minutes	INT	0
MS_Release	Release for maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgAckn	Message acknowledgment status (output ACK_STATE of ALARM_8P)	WORD	16#0000
MsgErr	1 = Message error (output ERROR of ALARM_8P)	BOOL	0
MsgStat	Message status (output STATUS of ALARM_8P)	WORD	16#0000
O_MS_Ext	Reserved	DWORD	16#00000000
OnAct	Block running	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	Block is "out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Operator permission: Output for OS	DWORD	16#FFFFFFFF
OS_PermOut	Operator permission: Output for OS	DWORD	16#FFFFFFFF
Seconds	Operating time [in sec]	INT	0

Parameter	Description	Type	Default
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1685)	DWORD	16#00000000
Status2	Status word 2 (Page 1685)	DWORD	16#00000000
TH_Act	High tolerance enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
TH_Minutes	Tolerance high	DINT	0
TimeMin	Operating period [min]	DINT	0
TL_Act	Low tolerance enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
TL_Minutes	Low tolerance time [in min]	DINT	0
TotalTime	Total operating time	DWORD	16#00000000
WH_Act	High warning enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
WH_Minutes	High warning time [in min]	DINT	0
WL_Act	Low warning enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
WL_Minutes	Low alarm time [in min]	DINT	0

See also

CountOh messaging (Page 1694)

CountOh block diagram (Page 1700)

CountOh modes (Page 1689)

10.2.7 CountOh block diagram

CountOh block diagram

A block diagram is not provided for this block.

See also

CountOh I/Os (Page 1696)

CountOh messaging (Page 1694)

CountOh error handling (Page 1694)

CountOh functions (Page 1689)

CountOh modes (Page 1689)

Description of CountOh (Page 1685)

10.2.8 Operator control and monitoring

10.2.8.1 CountOh views

Views of the CountOh block

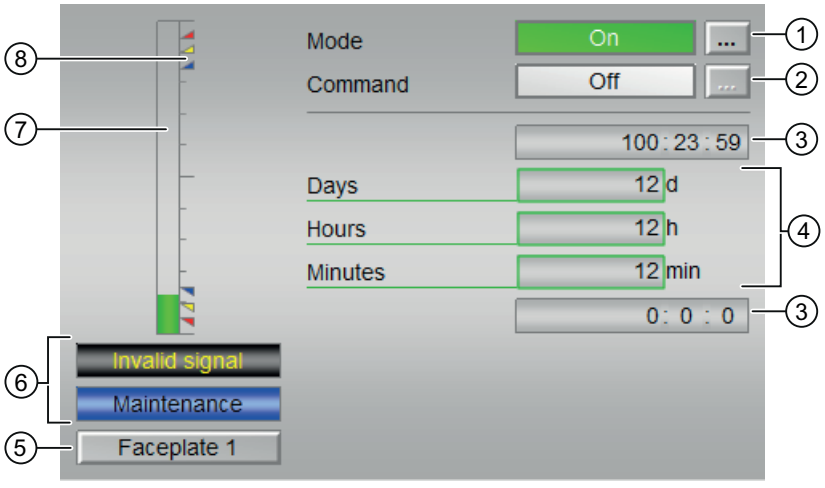
The block CountOh provides the following views:

- CountOh standard view (Page 1702)
- Alarm view (Page 296)
- CountOh limit view (Page 1704)
- Trend view (Page 299)
- CountOh parameter view (Page 1705)
- CountOh preview (Page 1706)
- Memo view (Page 298)
- Batch view (Page 296)
- Block icon for CountOh (Page 1707)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

10.2.8.2 CountOh standard view

CountOh standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) Enabling and disabling the counter

This area shows you the default operating state for the counter. The following states can be shown and executed here:

- "On ⇑"
- "On ⇓"
- "Off"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

(3) High and low scale range for the count value

These values provide information on the display range for the bar graph **(5)** of the count value. The scale range is defined in the engineering system.

(4) Displaying the counts values

The following counts are shown here:

- "Days" with signal status `TmMinST`
- "Hours" with signal status `TmMinST`
- "Minutes" with signal status `TmMinST`

(5) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

(6) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"
- "Invalid signal"

(7) Graphic display of the current count value

This area shows you the current count in form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(8) Limits

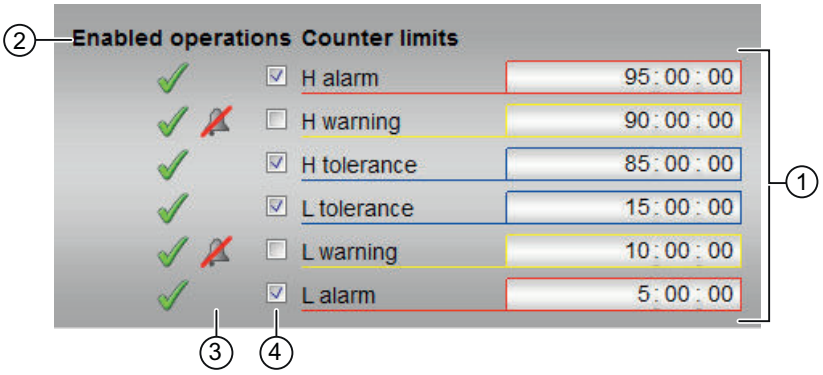
These colored triangles indicate the configured limits in the respective bar graph:

- Red: Alarm
- Yellow: Warning
- Blue: Tolerance

In counting mode "count up", only the colored triangles of the upper limits are visible; and in counting mode "count down", only the colored triangles of the lower limits are visible.

10.2.8.3 CountOh limit view

Limit view of CountOh



(1) Limits for the counter

In this area, you can enter the limits for the counter. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "H warning": Warning high
- "H tolerance": Tolerance high
- "L tolerance": Tolerance low
- "L warning": Warning low
- "L alarm": Alarm low

(2) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

(3) "Message suppression"

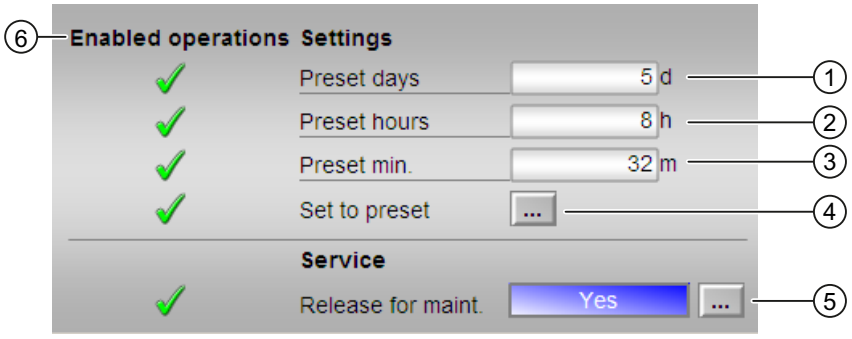
Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

(4) Suppress messages

You can enable / disable messages by setting the check mark.

10.2.8.4 CountOh parameter view

Parameter view of CountOh



(1), (2) and (3) preset counter

Enter the default setting here, where the counter should start.

You can change the following presets:

- "Days"
- "Hours"
- "Minutes"

You can find additional information on this in the Changing values (Page 253) section.

(4) Set to default

Set the counter to the default value here. You can find additional information on this in the Switching operating states and operating modes (Page 251) section.

(5)Service

You can select the following function in this area:

- "Release for maintenance"

Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the Release for maintenance (Page 64) section.

(6) Enabled operations

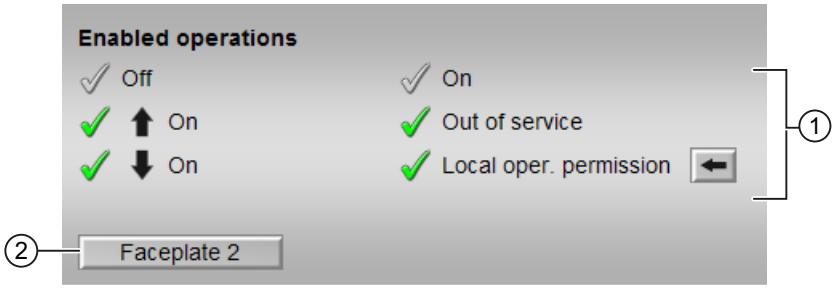
This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

10.2.8.5 CountOh preview

Preview of CountOh



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Off": You can disable the counter.
- "On ⇑": You can operate the incremental counter.
- "On ⇓": You can operate the decremental counter.

- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

(2) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the Opening additional faceplates (Page 203) section.

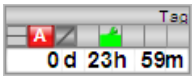
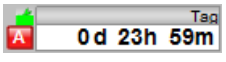
10.2.8.6 Block icon for CountOh

Block icons for CountOh

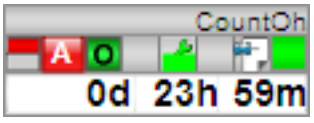
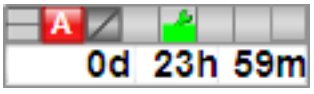
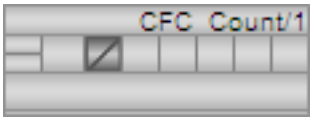
A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits
- Operating modes
- Signal status, release for maintenance
- Memo display
- Display counter running

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

10.3 TotalL - Additive counter with upward or downward counting direction (totalizer)

10.3.1 Description of TotalL

Object name (type + number) and family

Type + number: FB 1906

Family: Count

Area of application for TotalL

The block is used for the following applications:

- Triggered upward or downward summing
- Continuous upward or downward summing
- Upward or downward integration of an analog input value

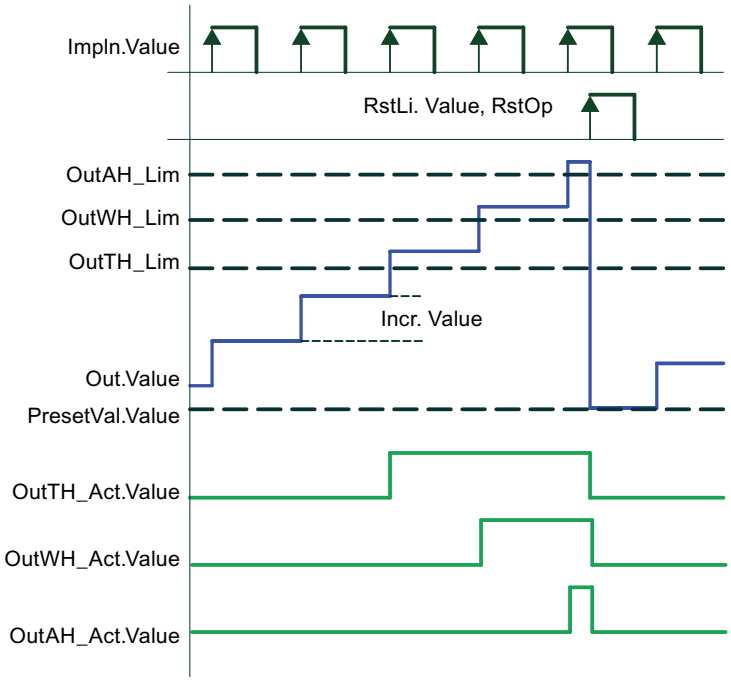
How it works

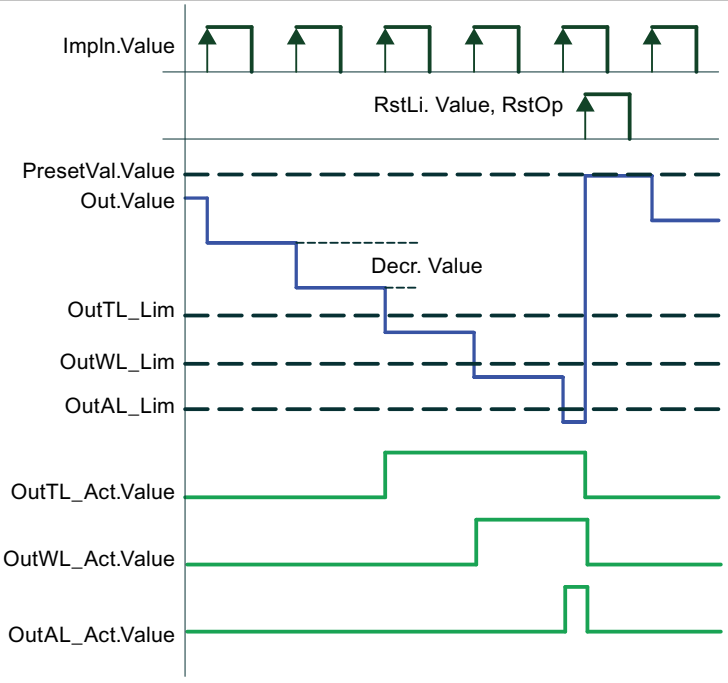
With a positive edge at the input parameter (pulse signal) `P_In` the output parameter `Out` is increased by a variable increment or reduced by a variable decrement.

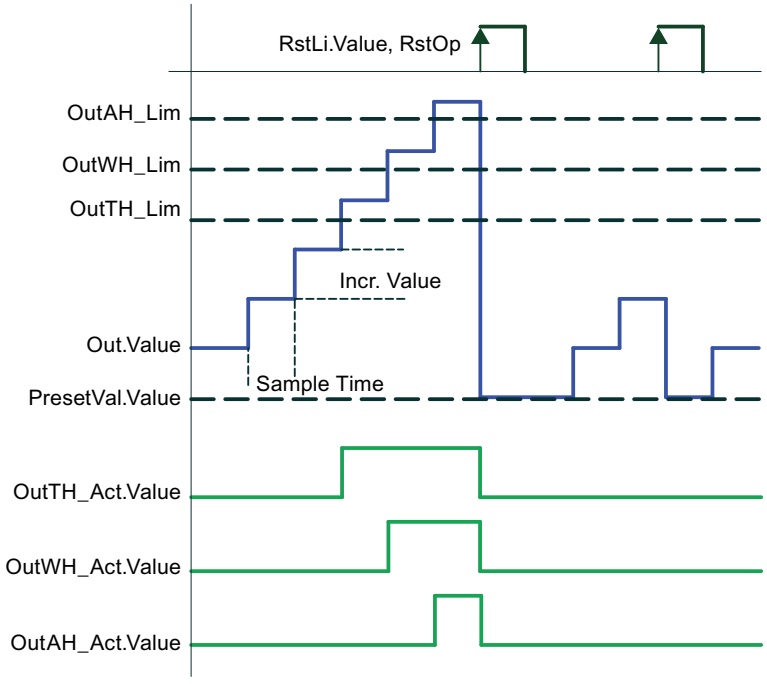
Configure how it works via the following `Feature` Bits:

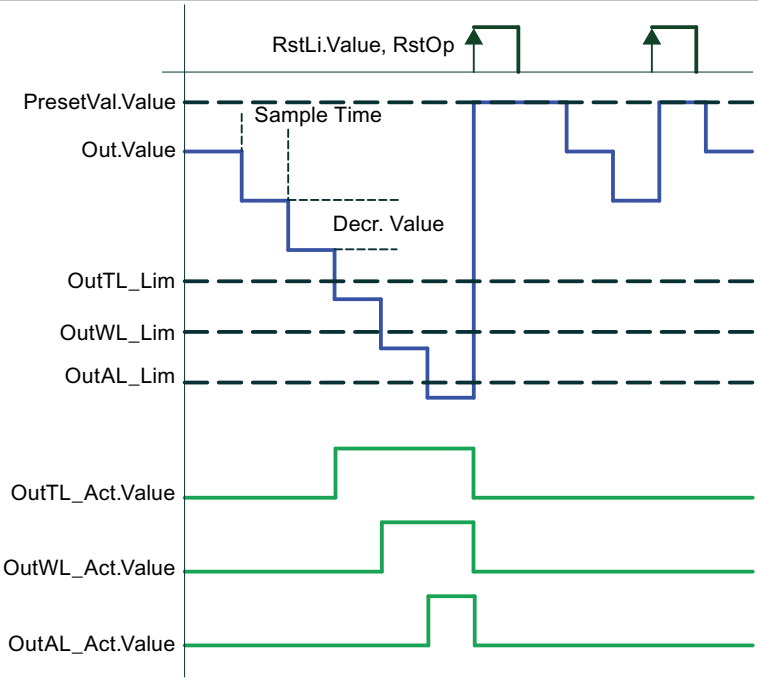
- `Feature` Bit 6: Block as summing unit or integrator (Page 142)
- `Feature` Bit 7: Summing characteristics continuous or triggered (Page 173)

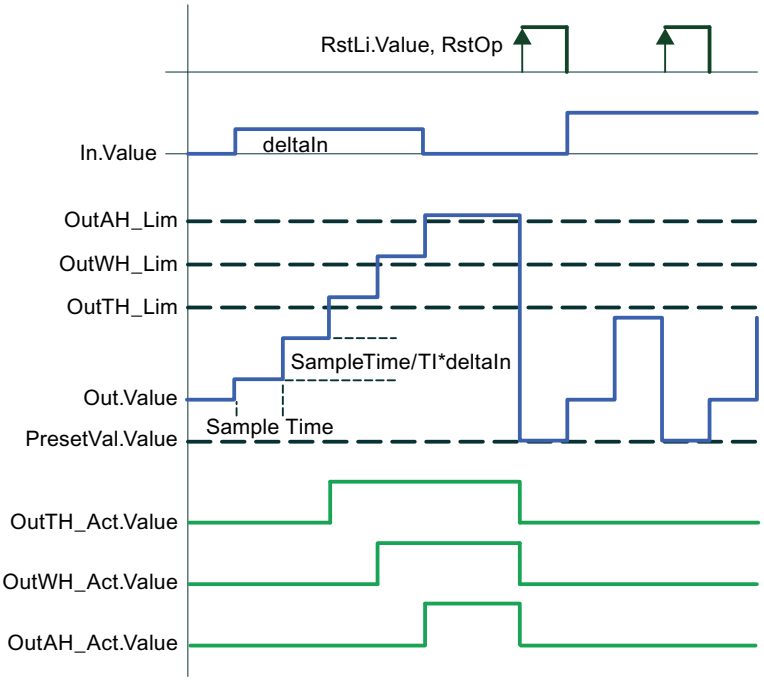
Using this parameterization, determine whether the block is to operate as an edge-triggered adder or summer, continuous adder or summer or as an integrator:

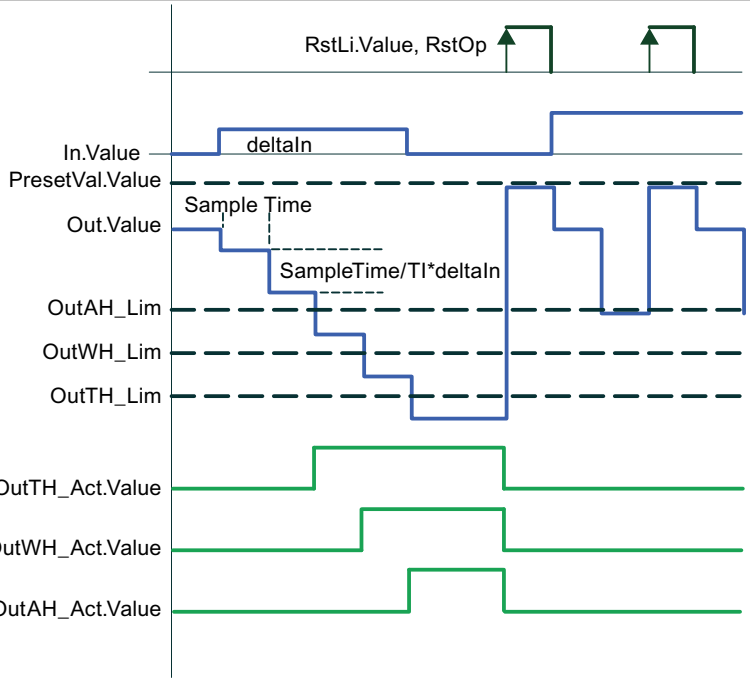
Parameterization on the `Feature` Bit	How the block operates
`Feature` Bit 6 = 0 `Feature` Bit 7 = 0	Block operates as an edge-triggered adder or summer. The calculation at the output parameter `Out` is performed with a 0 - 1 edge at input parameter `P_In`. The output parameter `Mode` shows whether the value is incremented or decremented. The pulses counted at the input parameter `P_In` are output to the output parameter `CntOut`.
	Triggered sum up `Out.Value` = $\texttt{Out.Value}_{(n-1)}$ + `Incr.Value` 
	Triggered sum down `Out.Value` = $\texttt{Out.Value}_{(n-1)}$ - `Decr.Value`

Parameterization on the `Feature` Bit	How the block operates
	

Parameterization on the `Feature` Bit	How the block operates
`Feature` Bit 6 = 0 `Feature` Bit 7 = 1	Block operates as a continuous adder or summer. The calculation at the output parameter `Out` is performed continuously. The output parameter `Mode` shows whether the value is incremented or decremented. The result is output to the output parameter `Out`. The output parameter `CntOut` always includes the value 0.
	Continuously sum up `Out.Value` = $\texttt{Out.Value}_{(n-1)}$ + `Incr.Value` 
	Continuously sum down `Out.Value` = $\texttt{Out.Value}_{(n-1)}$ - `Decr.Value`

Parameterization on the `Feature` Bit	How the block operates
	

Parameterization on the `Feature` Bit	How the block operates
`Feature` Bit 6 = 1 `Feature` Bit 7 = non-operational	Block operates as an continuous integrator. The direction of integration depends on the operation in the faceplate or from the interconnection. The input value `In` is integrated in accordance with the trapezoid rule dependent upon the counting direction according to the formula of the Integral block (Page 1831). The output parameter `Mode` shows whether the value is integrated up or down. The result is output to the output parameter `Out`. The output parameter `CntOut` always includes the value 0.
	Continuously integrate up `Mode` = 1: Out.Value = Out.Value n-1 + SampleTime/TI * (In.Value + In.Value n-1)/2 Set direction of integration: • Operation in the faceplate using the "On" command (`UpOp` = 1) Requirement: `LiOp.Value` = 0 • Interconnection `LiOp.Value` = 1 and `UpLi.Value` = 1 
	Continuously integrate down `Mode` = 2: Out.Value = Out.Value n-1 - SampleTime/TI * (In.Value + In.Value n-1)/2 Set direction of integration: • Operation in the faceplate using the "Off" command (`DnOp` = 1) Requirement: `LiOp.Value` = 0 • Interconnection `LiOp.Value` = 1 and `DnLi.Value` = 1

Parameterization on the `Feature` Bit	How the block operates
	

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Define the startup characteristics for this block via two `Feature` Bits:

Bit 0: Set startup characteristics (Page 137)

Bit 5: Use the last value following a complete download as the current value during startup of the block (Page 153)

The messages are suppressed after startup for the number of cycles specified in the `RunUpCyc` parameter.

Time response

The block does not have any time response.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section TotalL I/Os (Page 1726).

Status bit	Parameter
0	`Occupied`
1	`BatchEn`
2	`SimOn`
3	`OosAct.Value`
4	`OosLi.Value`
5	`BypassAct.Value`
6	`OnAct.Value`
7 - 8	Not used
9	Display of `BypassAct.Value` in faceplate (display and operator controls) and block icon
10	`SimLiOp.Value`
11	`LiOp`
12 - 13	Not used
14	1 = Invalid signal status
15	Triggered summing; `Feature` Bit 6 = 0, Bit 7 = 0
16	Continuous summing; `Feature` Bit 6 = 0, Bit 7 = 1
17	Integrate; `Feature` Bit 6 = 1
18 - 31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	`MsgLock.Value`
1	`OutAH_Act.Value`
2	`OutWH_Act.Value`
3	`OutTH_Act.Value`
4	`OutTL_Act.Value`
5	`OutWL_Act.Value`
6	`OutAL_Act.Value`
7	`OutAH_En`
8	`OutWH_En`
9	`OutTH_En`
10	`OutTL_En`
11	`OutWL_En`
12	`OutAL_En`
13	`OutAH_MsgEn`
14	`OutWH_MsgEn`
15	`OutTH_MsgEn`
16	`OutTL_MsgEn`
17	`OutWL_MsgEn`

Status bit	Parameter
18	`OutAL_MsgEn`
19	Not used
20	`Mode` = 1, upward summing or integration
21	`Mode` = 0, summer/integrator off
22	`Mode` = 2, downward summing or integration
23 - 30	Not used
31	`MS_RelOp`

See also

TotalL operating modes (Page 1718)

TotalL functions (Page 1719)

TotalL error handling (Page 1723)

TotalL messaging (Page 1724)

TotalL block diagram (Page 1731)

10.3.2 TotalL operating modes

TotalL operating modes

This block provides the following operating modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of TotalL (Page 1709)

TotalL functions (Page 1719)

TotalL error handling (Page 1723)

TotalL messaging (Page 1724)

TotalL I/Os (Page 1726)

TotalL block diagram (Page 1731)

10.3.3 TotalL functions

Functions of TotalL

The functions for this block are listed below.

Limit monitoring of the count value

This block provides the standard function Limit monitoring of the count value (Page 89).

Group display SumMsgAct for limit monitoring, CSF and ExtMsgx

The block provides the standard function Group display for limit monitoring, CSF and ExtMsgx (Page 85).

Suppressing messages using the `MsgLock` parameter

This block provides the standard function Suppressing messages using the MsgLock parameter (Page 201).

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

For `In` and `Out` different units of measure can be entered.

Simulating signals

The block provides the standard function Simulating signals (Page 58).

You can simulate the following values:

- Output value (`SimOut, SimOutLi`)

Note

Note on the simulation of the output value `Out`

For the presetting of the summing or integration value `Out` via the `RstLi` input, a 0 → 1 signal change at the input parameter `RstLi` is necessary during simulation.

Bypass function

This block provides the standard function Bypassing signals (Page 107).

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is always formed from the following parameters:

- `P_In.ST`
- `Incr.ST`
- `Decr.ST`,
- `PresetVal.ST`
- `In.ST`
- `Out.ST`

The following signal status is formed by:

Signal status	Used status
`Out.ST`	**Block operates as an edge-triggered adder or summer:** Formed from the signal status `Incr.ST, Decr.ST, P_In.ST` and the last signal status `Out.St`. **Block operates as continuous adder or summer:** Formed from the signal status `Incr.ST, Decr.ST` and the last signal status `Out.St`. **Block operates as continuous integrator:** Formed from the signal status `In.ST` and the last signal status `Out.St`. Resetting TotalL also resets the signal status: Out, ST := 16#80

With internal simulation, `ST_Worst` and the status of `Out` become `16#60`.

If there is a non-displayable floating-point number at one of the following parameters `In`, `PresetVal`, `TI` or `Out` when integrating `Feature` bit 6 = 1, the result is `ST_Worst` and the status of `Out` is set to `16#28`, provided the internally calculated status of `Out` does not result in `16#0` or the block is in internal simulation.

In addition, the status of `Out` is also written to the status of the output parameter `UpDnAct`.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` I/O in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)

Bit	Function
2	Resetting the commands for changing the mode (Page 160)
5	Use the last value following a complete download as the current value during startup of the block (Page 153)
6	Block as summing unit or integrator (Page 142)
7	Summing characteristics continuous or triggered (Page 173)
9	Substitution value is active if the block is in bypass (Page 184)
22	Update acknowledgment and error status of the message call (Page 159)
24	Enabling local operator authorization (Page 157)
25	Suppression of all messages (Page 173)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
28	Disabling operating points (Page 144)
29	Signaling limit violation (Page 169)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can stop the summing or integration
5	1 = Operator can activate the upward summing or integration
6	1 = Operator can activate the downward summing or integration
7	1 = Operator can preset the sum value (`Out`)
8 - 9	Not used
10	1 =Operator can change the simulation value `SimOut`
11	1 = Operator can activate the Simulation function
12	1 = Operator can activate the Release for maintenance function
13	1 = Operator can change the limit (`OutAH_Lim`) for high alarm
14	1 = Operator can change the limit (`OutWH_Lim`) for high warning
15	1 = Operator can change the limit (`OutTH_Lim`) for high tolerance
16	Not used
17	1 = Operator can change the limit (`OutAL_Lim`) for low alarm
18	1 = Operator can change the limit (`OutWL_Lim`) for low warning
19	1 = Operator can change the limit (`OutTL_Lim`) for low tolerance
20	1 = Operator can activate bypass functionality
21	1 = Operator can deactivate bypass functionality
22	1 =Operator can specify the default value (`PresetVal.Value`)
23	1 = Operator can specify the integration time constant (`TI`)

Bit	Function
24	1 = Operator can specify the value for the increment (Incr.Value)
25	1 = Operator can specify the value for the decrement (Decr.Value)
26	1 = Operator can activate / deactivate messages via OutAH_MsgEn
27	1 = Operator can activate / deactivate messages via OutAL_MsgEn
28	1 = Operator can activate / deactivate messages via OutWH_MsgEn
29	1 = Operator can activate / deactivate messages via OutTH_MsgEn
30	1 = Operator can activate / deactivate messages via OutTL_MsgEn
31	1 = Operator can activate / deactivate messages via OutWL_MsgEn

Note

If you interconnect a parameter that is also listed in OS_Perm as a parameter, you have to reset the corresponding OS_Perm bit.

Readback of the most recently calculated sum

As long as the output parameter Mode ≠ 0, the preceding sum value (OldOut) or the preceding count value (OldCntOut) is corrected to the current value:

- OldOut = Out.Value
- OldCntOut = CntOut.Value

After a warm restart, the sum value (Out) is automatically reset to the default value for the input parameter PresetVal if you have set the Feature Set startup characteristics (Page 137)Bit accordingly.

The count value CntOut in this case is always reset to 0.

The current sum and/or count value is set to the default predecessor values at startup via the Feature bit Use the last value following a complete download as the current value during startup of the block (Page 153).

- Old.Value = OldOut
- CntOut.Value = OldCntOut

Setting the summing/integrating value to the default

PresetVal is used to specify a summing/integrating value that is used in a warm restart.

As long as RstLi or RstOp is set, the following applies:

Out.Value = PresetVal.Value.

When the input PresetVal is not interconnected (PresetVal.ST = FF), the default setting for the count value can also be made from the parameter view of the block.

RstLi or RstOp reset the CntOut output to 0 at the same time.

You can use the Feature Bit 30 "Set reset depending on operating mode or the LiOp parameter" to configure the setting of the count value to a default depending on the LiOp parameter:

`Feature` Bit 30 =0: The reset does not depend on `LiOp`

`Feature` Bit 30 =1: Setting to the default depends on `LiOp`

LiOp =0: Setting to the default `PresetVal` or resetting of `CntOut` can only take place via the faceplate or at the `RstOp` parameter for the faceplate.

LiOp =1: Setting to the default `PresetVal` or resetting of `CntOut` can only take place via the interconnectable `RstLi` input.

Release for maintenance

This block provides the standard function Release for maintenance (Page 64).

SIMATIC BATCH functionality

This block provides the standard function SIMATIC BATCH functionality (Page 67).

See also

Description of TotalL (Page 1709)

TotalL operating modes (Page 1718)

TotalL error handling (Page 1723)

TotalL messaging (Page 1724)

TotalL I/Os (Page 1726)

TotalL block diagram (Page 1731)

10.3.4 TotalL error handling

Error handling of TotalL

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.

Error number	Meaning of the error number
11	The integration time constant TI is within the range: -SampleTime / 2 < TI < SampleTime / 2. The integration is stopped.
15	PresetVal.Value > OutOpScale.High PresetVal.Value < OutOpScale.Low
30	The value of In.Value can no longer be displayed in the REAL number field. The last valid value is provided at the Out.Value output.
31	The value of Out can no longer be displayed in the REAL number field. The last valid value is provided at the Out output.
32	The value of PresetVal.Value can no longer be displayed in the REAL number field. The last valid value is provided at the Out.Value output.
33	The value of TI can no longer be displayed in the REAL number field. The integration is stopped.
34	The value Incr.Value or Decr.Value can no longer be displayed in the REAL number field. The last valid value is provided at the Out.Value output.
51	Invalid signal at LiOp = 1: OffLi = 1 and UpLi = 1 and/or DnLi = 1 OffLi = 0 and UpLi = 1 and DnLi = 1

See also

Description of TotalL (Page 1709)
TotalL operating modes (Page 1718)
TotalL functions (Page 1719)
TotalL messaging (Page 1724)
TotalL block diagram (Page 1731)
TotalL I/Os (Page 1726)

10.3.5 TotalL messaging

Messaging

The following messages can be generated for this block:

- Functions for displaying measured limits

Messages generated as a reaction to limit violations, can be suppressed by the settings MsgEn and MsgLock.

Associated values for message instance `MsgEvId`

Associated value	Block parameters
1	`BatchName`
2	`StepNo`
3	`BatchID`
4	`ExtVa104`
5	`ExtVa105`

Process messages

Message instance	Message identifier	Message class	Event
`MsgEvId`	SIG 1	Alarm - high	$$BlockComment$$ High alarm limit violated
	SIG 2	Warning - high	$$BlockComment$$ High warning limit violated
	SIG 3	Tolerance - high	$$BlockComment$$ High tolerance limit violated
	SIG 4	Tolerance - low	$$BlockComment$$ Low tolerance limit violated
	SIG 5	Warning - low	$$BlockComment$$ Low warning limit violated
	SIG 6	Alarm - low	$$BlockComment$$ Low alarm limit violated
	SIG 7	AS process control message - fault	$$BlockComment$$ External message 1
	SIG 8	AS process control message - fault	$$BlockComment$$ External message 2

Explanation:

$$BlockComment$$: Content of the instance-specific comment

See also

Description of TotalL (Page 1709)

TotalL operating modes (Page 1718)

TotalL functions (Page 1719)

TotalL error handling (Page 1723)

TotalL I/Os (Page 1726)

TotalL block diagram (Page 1731)

10.3.6 TotalL I/Os

TotalL I/Os

Input parameters

Parameter	Description	Type	Default
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000
BatchName	Batch name	S7-String	
BypLiOp	1 = Bypass commands via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypLock	1 = Bypass activation or deactivation is locked for operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypOut	Substitution value if block is in bypass	REAL	0.0
BypOutLi	1 = Select bypass Out (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
BypOutOp	1 = Select bypass Out (via operator)	BOOL	0
Decr	Decrement for downward summing Decr.ST = FF via operator Decr.ST <> FF via interconnection	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
DnLi*	1 = Down counter, via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
DnOp*	1 = Down counter, via operator	BOOL	0
EN	1 = Called block will be processed	BOOL	1
ExtMsg1	Binary input for freely selectable message 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtMsg2	Binary input for freely selectable message 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ExtVa104	Associated value 4 for messages (MsgEvId)	ANY	
ExtVa105	Associated value 5 for messages (MsgEvId)	ANY	
Feature	I/O for additional functions (Page 1719)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0

Parameter	Description	Type	Default
In	Input for integral value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Incr	Increment for upward summing Incr.ST = FF via operator Incr.ST <> FF via interconnection	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
InUnit	Unit of measure for the input parameter In	INT	1351
LiOp	1 = Interconnection 0 = Operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MsgEvId	Message number (assigned automatically)	DWORD	16#00000000
MsgLock	1 = Suppress process messages. See also the section Suppressing messages using the MsgLock parameter (Page 201) for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_RelOp*	Operator can switch release for maintenance	BOOL	0
Occupied	1 = In use by a batch	BOOL	0
OffLi*	1 = Counter disabled via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OffOp*	1 = Counter disabled via operator	BOOL	1
OldCntOut*	Previous count value	DINT	0
OldOut*	Previous output value	REAL	0.0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operating permissions (Page 1719)	STRUCT • Bit 0: BOOL • Bit 10: BOOL • Bit 31: BOOL	- • 1 • 1 • 1
OutAH_En	1 = Enable alarm (high) for count	BOOL	1
OutAH_Lim	Limit for count alarm (high)	REAL	95.0
OutAH_MsgEn	1 = Enable message for count alarm (high)	BOOL	1
OutAL_En	1 = Enable alarm (low) for count	BOOL	1
OutAL_Lim	Limit for count alarm (low)	REAL	0.0
OutAL_MsgEn	1 = Enable message for count alarm (low)	BOOL	1

Parameter	Description	Type	Default
OutOpScale	Limit for scale in Out bar graph of faceplate	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
OutTH_En	1 = Enable tolerance message (high)	BOOL	0
OutTH_Lim	Count tolerance limit message (high)	REAL	85.0
OutTH_MsgEn	1 = Enable message for count tolerance message (high)	BOOL	1
OutTL_En	1 = Enable count tolerance message (low)	BOOL	0
OutTL_Lim	Count tolerance message limit (low)	REAL	0.0
OutTL_MsgEn	1 = Enable message for count tolerance message (low)	BOOL	1
OutUnit	Unit of measure for count Out	INT	1038
OutWH_En	1 = Enable count warning (high)	BOOL	1
OutWH_Lim	Count warning limit (high)	REAL	90.0
OutWH_MsgEn	1 = Enable message for count warning (high)	BOOL	1
OutWL_En	1 = Enable count warning (low)	BOOL	1
OutWL_Lim	Count warning limit (low)	REAL	0.0
OutWL_MsgEn	1 = Enable message for count warning (low)	BOOL	1
P_In	Input for sum value (pulse input)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PresetVal*	Default setting for the count	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
RstBypLi	1 = Reset bypass Out (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstBypOp	1 = Reset bypass Out (via operator)	BOOL	0
RstLi	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstOp*	1 = Reset via operator	BOOL	0
RunUpCyc	Number of cycles in startup; messages are suppressed during these cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	
SimLiOp	Activation/deactivation of the simulation by: 0 = Operator 1 = Interconnection or SFC	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
SimOnLi	1 = Simulation via interconnection or SFC (controlled by SimLiOp = 1)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn*	1 = Simulation on	BOOL	0
SimOut*	Output value used for SimOn = 1	REAL	0.0
SimOutLi	Output value that is used for SimOnLi.Value = 1 (SimLiOp.Value = 1)	STRUCT • Value: REAL • ST: BYTE	- • 0 • 16#80
StepNo	Batch step number	DWORD	16#00000000
TI*	Intagral component time	REAL	1.0
UpLi*	1 = Forward counter, via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
UpOp*	1 = Forward counter, via operator	BOOL	0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
BypassAct	1 = Bypass is activated in this block	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CntOut	Number of counted pulses	DINT	0
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see TotalL error handling (Page 1723)	INT	-1
Mode	0 = Totalizer off 1 = Totalizer summed/integrated upward 2 = Totalizer summed/integrated downward	INT	0
MsgAckn	Message acknowledgment status (output ACK_STATE of ALARM_8P)	WORD	16#0000
MsgErr	1 = Message error (output ERROR of ALARM_8P)	BOOL	0
MsgStat	Message status (output STATUS of ALARM_8P)	WORD	16#0000
MS_Release	Release for maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
O_MS_Ext	Reserved	DWORD	0
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80

Parameter	Description	Type	Default
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
Out	Count	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
OutAH_Act	1 = Count alarm (high) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutAL_Act	1 = Count alarm (low) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutTH_Act	1 = Count tolerance message (high) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutTL_Act	1 = Count tolerance message (low) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutWH_Act	1 = Count warning (high) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutWL_Act	1 = Count warning (low) enabled. You can change the reaction for this parameter with Feature bit 28 (Disabling operating points (Page 144)) and with Feature bit 29 (Signaling limit violation (Page 169)).	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 1709)	DWORD	16#00000000

Parameter	Description	Type	Default
Status2	Status word 2 (Page 1709)	DWORD	16#00000000
UpDnAct	1 = Block counts/integrates 0 = Block has no activity	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

TotalL operating modes (Page 1718)

TotalL messaging (Page 1724)

TotalL block diagram (Page 1731)

10.3.7 TotalL block diagram

TotalL block diagram

A block diagram is not provided for this block.

See also

Description of TotalL (Page 1709)

TotalL operating modes (Page 1718)

TotalL functions (Page 1719)

TotalL error handling (Page 1723)

TotalL I/Os (Page 1726)

TotalL messaging (Page 1724)

10.3.8 Operator control and monitoring

10.3.8.1 TotalL views

Views of the TotalL block

The block TotalL provides the following views:

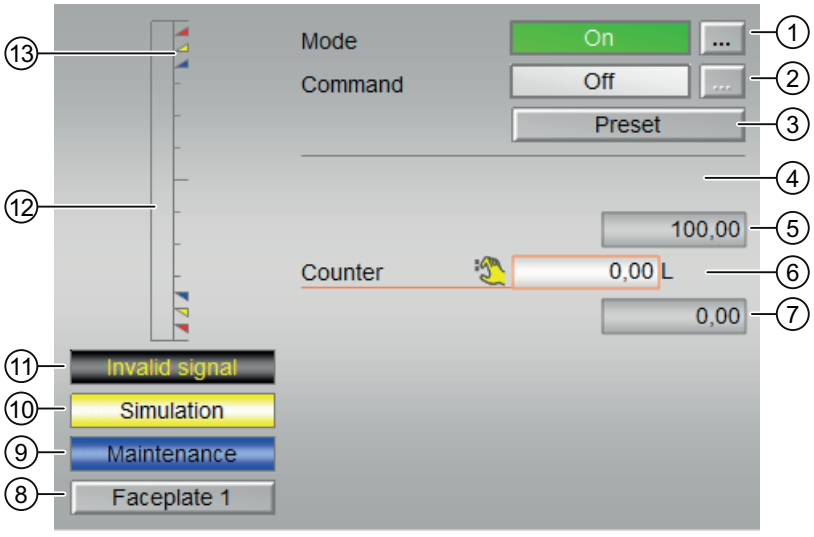
- TotalL standard view (Page 1732)
- Alarm view (Page 296)
- TotalL limit view (Page 1735)
- Trend view (Page 299)

- TotalL parameter view (Page 1736)
- TotalL preview (Page 1738)
- Memo view (Page 298)
- TotalL block icon (Page 1739)
- Batch view (Page 296)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

10.3.8.2 TotalL standard view

TotalL standard view



(1) Displaying and switching the operating mode

Operation of inputs `OosOp` and `OnOp`.

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71): `Status1` Bit 3 (`OosAct.Value`)
- Out of service (Page 71): `Status1` Bit 6 (`OnAct.Value`)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) Enabling and disabling the counter

Operation of the inputs `UpOp`, `OffOp` and `DnOp`.

This area shows you the default operating state for the counter. The following states can be shown and executed here:

- "On ⇑": `Status2` Bit 20 (`Mode` = 1)
- "On ⇓": `Status2` Bit 22 (`Mode` = 2)
- "Off": `Status2` Bit 21 (`Mode` = 0)

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

(3) Default setting

Operation of the input `RstOp`.

This button activates the preset value.

(4) Display of the count value

The current count values are displayed here:

Display of input In

Depending on `Feature` Bit 6 and `Feature` Bit 7, either input `In` or an empty frame is displayed:

- "Input": `Feature` Bit 6 = 1, otherwise no display

The display is not operational. Format of `In` like display `Out`.

(5) High scale range for the count value

This value provides information on the display range for the bar graph (above) of the count value. The scale range is defined in the engineering system.

(6) Display of the count value

The current count values are displayed here:

Display Out

Depending on `Feature` Bit 6, the text for the display is switched over:

- "Counter": `Feature` Bit 6 = 0
- "Integrator": `Feature` Bit 6 = 1

The "Counter" display is not operational, not even in simulation. The format of `In` is like the `Out` display.

The "Integrator" display is only operational in simulation (operation input `SimOut`). The format corresponds to the block icon of `AnalogValueFormat1`.

(7) Low scale range for the count value

This value provides information on the display range for the bar graph (below) of the count value. The scale range is defined in the engineering system.

(8) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(9) Display area for block states

This area provides additional information on the operating state of the block:

- "Maintenance"

You can find more information about this in the section Release for maintenance (Page 64).

(10) Display area for block states

This area provides additional information on the operating state of the block:

- "Simulation"

You can find additional information on this in the section Simulating signals (Page 58).

(11) Display area for block states

This area provides additional information on the operating state of the block:

- "Invalid signal"

You can find additional information on this in the section Error handling (Page 119).

(12) Graphic display of the current count value

This area shows you the current count in form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

(13) Limit display

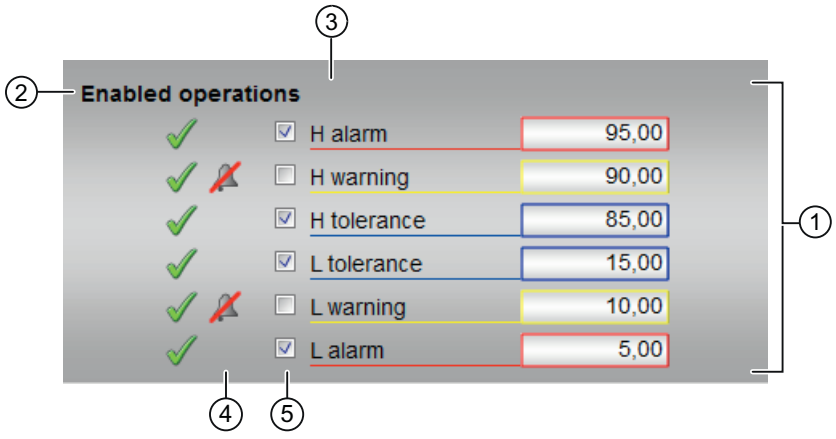
These colored triangles indicate the configured limits in the respective bar graph:

- Red: Alarm
- Yellow: Warning
- Blue: Tolerance

In counting mode "count up", only the colored triangles of the upper limits are visible; and in counting mode "count down", only the colored triangles of the lower limits are visible.

10.3.8.3 TotalL limit view

Limit view of TotalL



(1) Limits for the counter

In this area, you can enter the limits for the counter. Refer to the Changing values (Page 253) section for more on this.

You can change the following limits:

- "H alarm": Alarm high
- "H warning": Warning high
- "H tolerance": Tolerance high
- "L tolerance": Tolerance low
- "L warning": Warning low
- "L alarm": Alarm low

Format and unit is like the `Out` display in the standard view.

(2) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

(3) Dynamic text

Depending on `Feature` Bit 6, the text is switched:

- "Integrator limits (out)": Display for `Feature` Bit 6 = 1
- "Summer limits (out)": Display for `Feature` Bit 6 = 0

(4) "Message suppression"

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

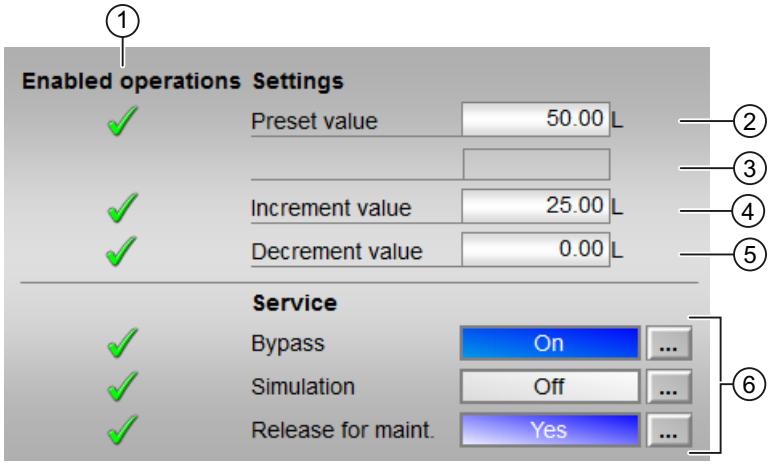
(5) Suppress messages

You can enable / disable messages by setting the check mark.

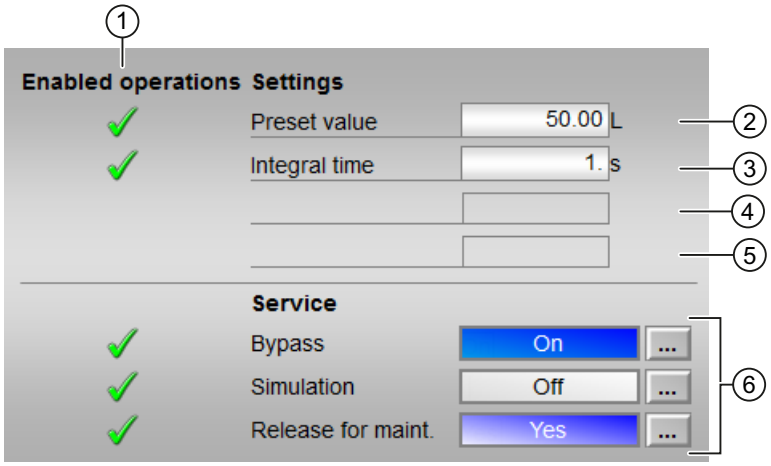
10.3.8.4 TotalL parameter view

Parameter view of TotalL

With `Feature` bit 6 = 0:



With `Feature` bit 6 = 1:



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`).

(2) Default value

In this section you can enter the value from which the counting should begin. Additional information is available in the section Changing values (Page 253).

(3) Integral time

The currently integral time in seconds is displayed in this area.

- "Integral time" Display for `Feature` Bit 6 = 1, otherwise empty.

(4) Increment value

The increment value is displayed in this area.

- "Increment value": Display for `Feature` Bit 6 = 0, otherwise empty.

(5) Decrement value

The decrement value is displayed in this area.

- "Decrement value": Display for `Feature` Bit 6 = 0, otherwise empty.

(6) Service

You can select the following function in this area:

- "Bypass"
- "Simulation"
- "Release for maintenance"

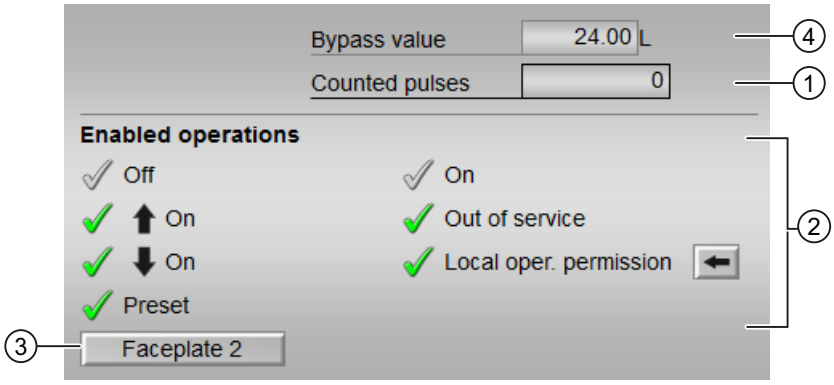
Refer to the Switching operating states and operating modes (Page 251) section for more on this.

You can find information on this area in the section:

- Bypassing signals (Page 107)
- Simulating signals (Page 58)
- Release for maintenance (Page 64)

10.3.8.5 TotalL preview

Preview of TotalL



(1) Counted pulses

The number of pulses already counted is displayed in this area.

- "Counted pulses": `Feature` Bit 6 = 0 and `Feature` Bit 7 = 0, otherwise empty

(2) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Off": You can disable the counter.
- "On ⇑": You can operate the incremental counter.
- "On ⇓": You can operate the decremental counter.
- "Default": You can change the default setting.
- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

Additional information is available in the section Opening additional faceplates (Page 203).

(4) Bypass value

This area displays the bypass value (`BypOut`).

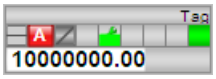
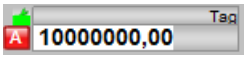
10.3.8.6 TotalL block icon

Block icons for TotalL

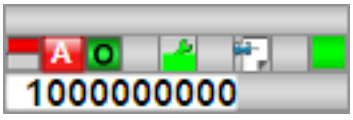
A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of alarm, warning, and tolerance limits
- Operating modes
- Signal status, release for maintenance
- Memo display
- Display counter running

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

10.4 CntOhSc - Runtime determination and counters with counting direction "up"

10.4.1 Description of CntOhSc

Object name (type + number) and family

Type + number: FB 1803

Family: Count

Area of application of CntOhSc

The block is used for the following applications:

- Incrementing of operating hours
- Incrementing a defined input value

How it works

The block determines the time in which a unit has been in operation and it counts a defined input value. The block can only count forwards.

1. Off (`OffOp = 1`)
   The block is disabled (output parameter `CountMode = 0`). No counting takes place.
2. Increment (`UpOp = 1`)
   The operating time of the connected unit is incremented (output parameter `CountMode = 1`).

The operating time is shown in days, hours, minutes and seconds. The maximum operating time is 32767 days and 23 hours.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is first installed automatically in the startup OB (OB 100).

Startup characteristics

Define the startup characteristics for this block via two feature bits:

Bit 0: Set startup characteristics (Page 137)

Bit 5: Use the last value following a complete download as the current value during startup of the block (Page 153)

When you set feature bit 5 to 1, then:

- `Days:= OldDays`
- `Hours:= OldHours`

- `Minutes:= OldMinutes`
- `Seconds:= OldSeconds`

Status word allocation for `Status1` parameter

Refer to the following section for a description of the individual parameters: CntOhSc I/Os (Page 1746)

Status bit	Parameter
0 - 2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`
7 - 10	Not used
11	`LiOp`
12 - 13	Not used
14	1 = Invalid signal status
15 - 31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	Not used
1	`HrsHiL1Act.Value`
2	`HrsHiL2Act.Value`
3	`CntHiL1Act.Value`
4	`CntHiL2Act.Value`
5 - 6	Not used
7	`HrsHiL1En`
8	`HrsHiL2En`
9	`CntHiL1En`
10	`CntHiL2En`
11 - 19	Not used
20	Count up
21	Counter off
22 - 31	Not used

10.4.2 CntOhSc operating modes

Operating modes of CntOhSc

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

10.4.3 CntOhSc functions

Functions of CntOhSc

The block provides the following functions:

Reset counter to zero

The output parameters of the operating time `Days`, `Hours`, `Minutes`, `Seconds`, `TimeHours` and `TotalTime` are reset via the interconnect `ResetOh` parameter. The reset is made with a 0 - 1 edge.

The counted value at the `Cnt` output parameter is reset with the interconnectable `ResetCnt` parameter. The reset is made with a 0 - 1 edge.

The count value and the output parameters of the operating time can also be reset to zero together in the standard view at the `ResetOp` parameter or the interconnectable `ResetLi` parameter via the faceplate.

You can use Feature bit 30 "Set reset depending on operating mode or the `LiOp` parameter" to configure a reset depending on the `LiOp` parameter:

Feature bit 30 =0: The reset does not depend on `LiOp`

Feature bit 30 =1: The reset depends on `LiOp`

`LiOp =0`: Separate reset via inputs `ResetOh` and `ResetCnt` is not possible. Simultaneous reset can only be made via the faceplate or at the `ResetOp` parameter.

`LiOp =1`: Separate reset is only possible via interconnectable inputs `ResetOh` and `ResetCnt`. Simultaneous reset can only be made via the interconnectable input `ResetLi`.

Configurable reactions using the Feature parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
5	Use the last value following a complete download as the current value during startup of the block (Page 153)
24	Enabling local operator authorization (Page 157)
26	Reaction of the switching points in the "Out of service" operating mode (Page 175)
29	Signaling limit violation (Page 169)
30	Set reset depending on the operating mode or the LiOp parameter (Page 162)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can stop counting
5	1 = Operator can switch to "Count up" mode
6	1 = Operator can reset the count
7 - 12	Not used
13	1 = Operator can change the operating hours high limit 1 (`HrsHi1Lim`)
14	1 = Operator can change the operating hours high limit 2 (`HrsHi2Lim`)
15	1 = Operator can change the counter high limit 1 (`CntHi1Lim`)
16	1 = Operator can change the counter high limit 2 (`CntHi2Lim`)
17 - 31	Not used

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `InOh.ST`
- `InCnt.ST`
- `TmHoursST`
- `CntST`

The following signal status is formed by:

Signal status	Used status
TmHoursST	Increment: Formed from the worst signal status of InOh.ST and the last signal status TmHoursST. This means that the worst signal status of InOh.ST is stored to TmHoursST during increment. Counter off: Signal status TmHoursST is frozen. Resetting CntOhSc or ResetOh also resets the signal status: TmHoursST := 16#80
CntST	Increment: Formed from the worst signal status of InCnt.ST and the last signal status CntST. This means that the worst signal status of InCnt.ST is stored to CntST during increment. Counter off: Signal status CntST is frozen. Resetting CntOhSc or ResetCnt also resets the signal status: CntST := 16#80

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

10.4.4 CntOhSc error handling

Error handling of CntOhSc

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The ErrorNum I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed
0	No active fault
51	Invalid signal at LiOp = 1: • OffLi = 1 and UpLi = 1

10.4.5 CntOhSc messaging

Messaging

This block does not offer messaging.

10.4.6 CntOhSc I/Os

I/Os of CntOhSc

Input parameters

Parameter	Description	Type	Default
`CntHiL1En`	1 = Counter high limit 1 is enabled	BOOL	0
`CntHiL2En`	1 = Counter high limit 2 is enabled	BOOL	0
`CntHi1Lim`	Counter high limit 1	DINT	95
`CntHi2Lim`	Counter high limit 2	DINT	90
`CntOpHiScale`	High limit for scale in count bar graph of face-plate	DINT	100
`CntOpLoScale`	Low limit for scale in count bar graph of face-plate	DINT	0
`CntUnit`	Unit of measure for count `Cnt`	INT	0
`Feature`	I/O for additional functions (Page 1743)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`HrsHiL1En`	1 = Operating hours high limit 1 is enabled	BOOL	0
`HrsHiL2En`	1 = Operating hours high limit 2 is enabled	BOOL	0
`HrsHi1Lim`	Operating hours high limit 1	DINT	2280
`HrsHi2Lim`	Operating hours high limit 2	DINT	2160
`HrsOpHiScale`	Operating hours high limit of bar display for OS	DINT	2400
`HrsOpLoScale`	Operating hours low limit of bar display for OS	DINT	0
`InOh`	Digital input value for operating hours counter	REAL	0.0
`InCnt`	Digital input value for up counter	REAL	0.0
`LiOp`	1= Interconnection 0 = Operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`OffLi`	Counter disabled via interconnection: 1 = Off	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
OffOp*	Counter disabled via operator: 1 = Off	BOOL	1
OldCnt*	Previous count value	DINT	0
OldDays	Previous day value	INT	0
OldHours	Previous hour value	INT	0
OldMinutes	Previous minute value	INT	0
OldSeconds*	Previous second value	INT	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via operator	BOOL	0
OpSt_In	Input parameter for local operator authorization, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for CntOhSc functions (Page 1743)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
ResetCnt	1 = Input valueCnt reset	BOOL	0
ResetLi	1 = Counter reset via interconnection	BOOL	0
ResetOh	1 = Operating hours reset	BOOL	0
ResetOp*	1 = Counter reset by operator	BOOL	0
SampleTime	Sampling time [s]	REAL	0.1
SelFp1	Open faceplate 1	ANY	
UpLi	1 = Increment (via interconnection)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
UpOp*	1 = Increment (via faceplate)	BOOL	0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
CountMode	Count mode: 0 = Off 1 = Increment	INT	0
Cnt	Count	DINT	0

Parameter	Description	Type	Default
CntHiL1Act	1 = Counter high limit 1 is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CntHiL2Act	1 = Counter high limit 2 is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CntST	Signal status for Cnt	BYTE	16#80
Days	Operating stage	INT	0
DeviceOn	1 = Unit on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see CntOhSc error handling (Page 1745)	INT	-1
Hours	Operating hours	INT	0
HrsHiL1Act	1 = Operating hours high limit 1 is active	BOOL	0
HrsHiL2Act	1 = Operating hours high limit 2 is active	BOOL	0
Minutes	Operating minutes	INT	0
O_MS_Ext	Reserved	DWORD	16#00000000
OnAct	Block running	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	Block is "out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Operator permission: Output for OS	DWORD	16#FFFFFFFF
OS_PermOut	Operator permission: Output for OS	DWORD	16#FFFFFFFF
Seconds	Operating time [in sec]	INT	0
ST_Worst	Worst signal status	BYTE	16#80
Status1	Status word 1 (Page 399)	DWORD	16#00000000
Status2	Status word 2 (Page 399)	DWORD	16#00000000
TimeHours	Service life [h]	DINT	0
TmHoursST	Signal status for TimeHours	BYTE	16#80
TotalTime	Total operating time [in sec]	DWORD	16#00000000

10.4.7 CntOhSc block diagram

Block diagram of CntOhSc

A block diagram is not provided for this block.

10.4.8 Operator control and monitoring

10.4.8.1 CntOhSc views

Views of the CntOhSc block

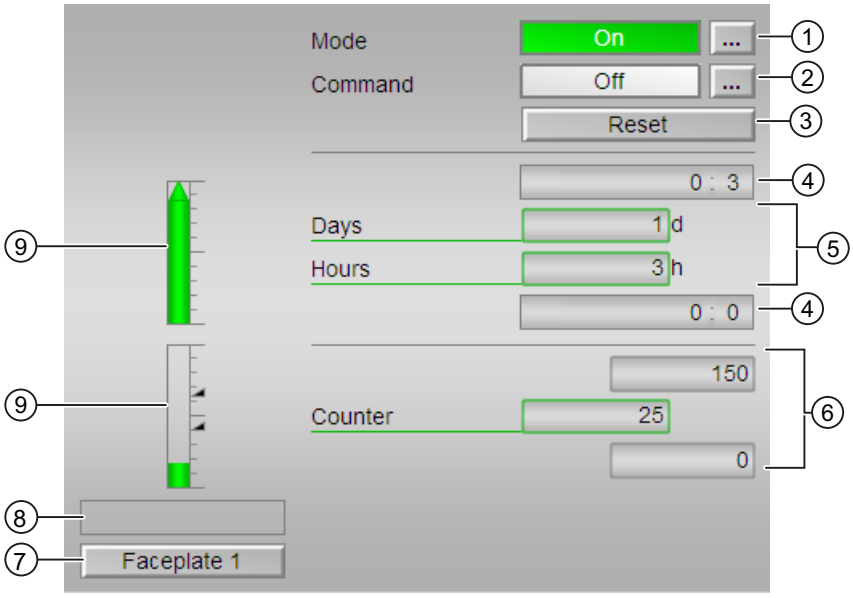
The CntOhSc block provides the following views:

- CntOhSc standard view (Page 1750)
- CntOhSc limit view (Page 1752)
- Trend view (Page 299)
- CntOhSc preview (Page 1753)
- Memo view (Page 298)
- Block icon for CntOhSc (Page 1754)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

10.4.8.2 CntOhSc standard view

Standard view of CntOhSc



(1) Display and switch the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to section Switching operating states and operating modes (Page 251) for information on switching the operating mode.

(2) Enable and disable the counter

This area shows you the default operating state for the counter. The following states can be shown and executed here:

- "On ⇑"
- "Off"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

(3) Reset

Operation of the input `RstOp`.

This button activates the preset value.

(4) High and low scale range for the count value

These values provide information on the display range for the bar graph of the count `HrsOpHiScale` or `HrsOpLoScale`. The scale range is defined in the engineering system.

(5) Display for counts

The following counts are shown here:

- "Days" with signal status `TmHoursST`
- "Hours" with signal status `TmHoursST`

(6) Counter

The following values are shown here:

- Current count value with the signal status `CntST`
- High and low scale range for the count

These values provide information on the display range for the bar graph of the count `CntOpHiScale` or `CntOpLoScale`. The scale range is defined in the engineering system.

(7) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

(8) Display area for states of the block

This area provides additional information on the operating state of the block:

- "Invalid signal"

You can find additional information on this in the section Error handling (Page 119).

(9) Graphic display for the current count

These areas show the `Cnt` and `TimeHours` counts in the form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

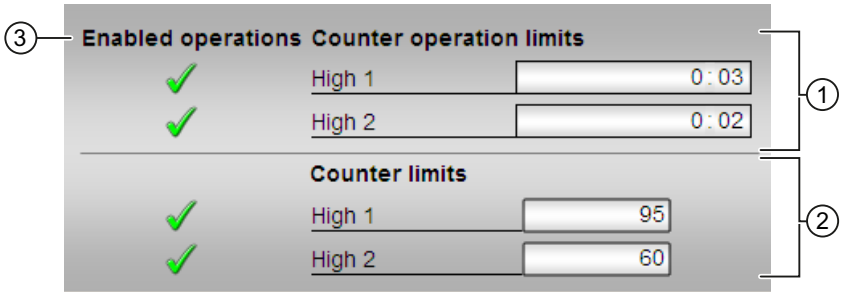
Limits

These colored triangles indicate the configured limits in the respective bar graph:

- Red: High limit 1
- Yellow: High limit 2

10.4.8.3 CntOhSc limit view

Limit view of CntOhSc



(1) Operating hours limits

In this area, you can enter the limits for the operating hours. You can find additional information on this in the section Changing values (Page 253) .

You can change the following limits:

- "High 1": Operating hours high limit 1
- "High 2": Operating hours high limit 2

(2) Counter limits

In this area, you can enter the limits for the counter. You can find additional information on this in the section Changing values (Page 253) .

You can change the following limits:

- "High 1": Counter high limit 1
- "High 2": Counter high limit 2

(3) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

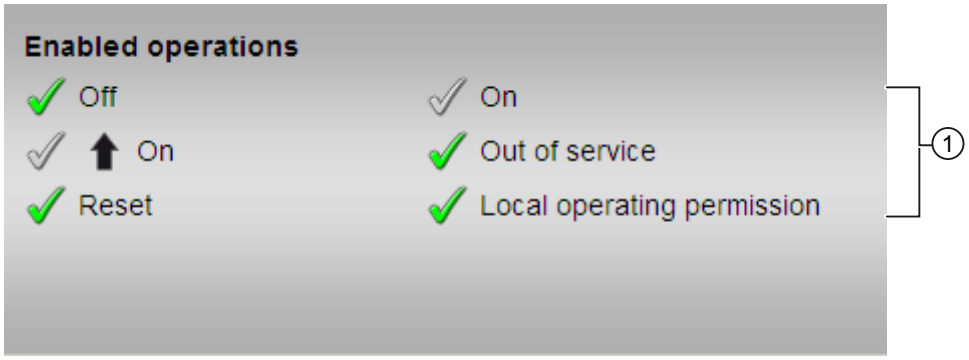
- **Green check mark:** The OS operator can control this parameter.
- **Gray check mark:** The OS operator cannot control this parameter at this time due to the process.
- **Red cross:** The OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm`).

(4) "Message suppression"

Message suppression indicates whether or not the suppression of the associated message in the AS block is activated with the `xx_MsgEn` parameter. The output of messages is not suppressed when the block is installed (all `xx_MsgEn` parameters are preset to 1). Messages can only be output if limit monitoring of the additional analog value has been enabled.

10.4.8.4 CntOhSc preview

Preview of CntOhSc



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark:** the OS operator can control this parameter
- **Gray check mark:** the OS operator cannot control this parameter at this time due to the process
- **Red cross:** the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm`)

The following enabled operations are shown here:

- "Off": You can disable the counter.
- "On ⇑": You can operate the counter.
- "Reset": You can reset the counter after interlocks or errors.
- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

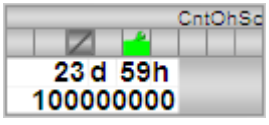
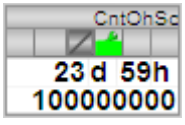
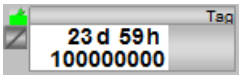
10.4.8.5 Block icon for CntOhSc

Block icons for CntOhSc

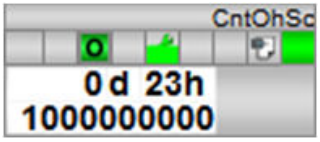
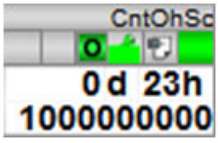
A variety of block icons are available with the following functions:

- Process tag type
- Limits (high/low)
- Violation of the limits
- Operating modes
- Signal status
- Memo display
- Display counter running

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in the CFC	Special features
	1	
	2	
	3	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in the CFC	Special features
	1	Block icon in the full display
	2	
	-	Block icon in "Out of service" mode (example with type block icon type 1)

Additional information on the block icon and the control options in the block icon is available in the following chapters:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

Timers

11

11.1 TimerP - Time delays signal forwarding / pulse generator

11.1.1 Description of TimerP

Object name (type+number) and family

Type + number: FB 1810

Family: TIME

Area of application for TimerP

The block is used for the following fields of applications:

- Pulse generator
- Extended pulse
- ON delay
- ON delay with memory
- OFF delay

How it works

The TimerP block is used for delayed forwarding of start or stop actions through the `Out` output parameter.

New values at the `Ti` I/O are only applied when a change is made to the `In` I/O.

The inverted signal for `Out` is also provided at the `InvOut` output parameter.

Use the Mode input parameter to define how the block should be used. Refer to the TimerP functions (Page 1758) section for more on this.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

A startup behavior can be reached with a connection of the input parameter `Init` to the output parameter `InitOut` of the Trigger block. For more information, refer to Startup characteristics over Trigger block (Page 69).

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

TimerP modes (Page 1758)
TimerP error handling (Page 1761)
TimerP messaging (Page 1762)
TimerP I/Os (Page 1762)
TimerP block diagram (Page 1764)

11.1.2 TimerP modes

TimerP operating modes

This block does not have any modes.

See also

Description of TimerP (Page 1757)
TimerP functions (Page 1758)
TimerP error handling (Page 1761)
TimerP messaging (Page 1762)
TimerP I/Os (Page 1762)
TimerP block diagram (Page 1764)

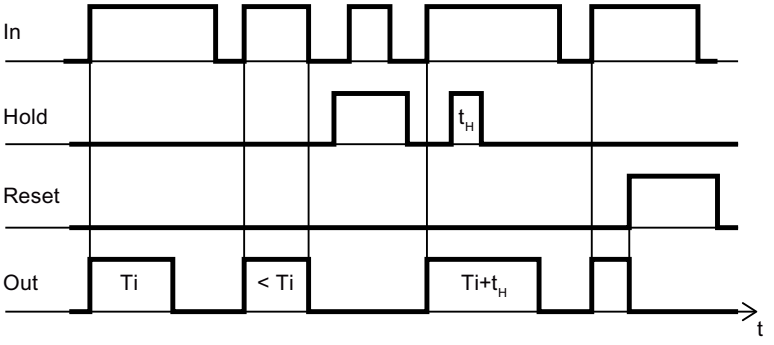
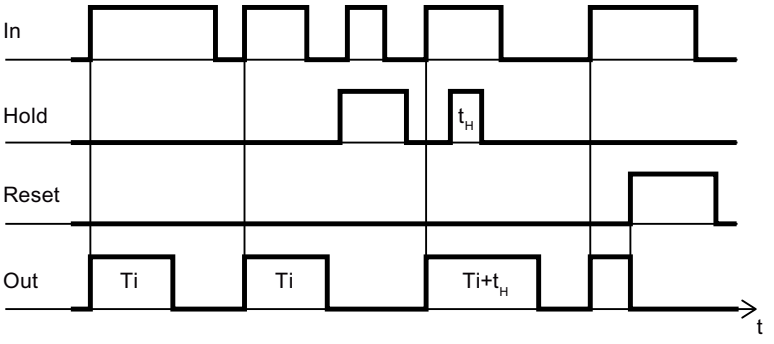
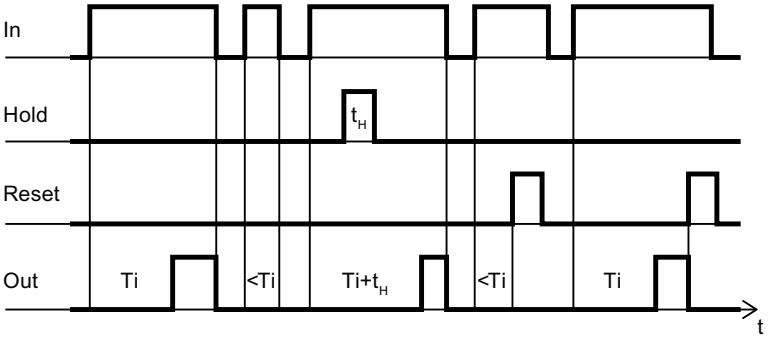
11.1.3 TimerP functions

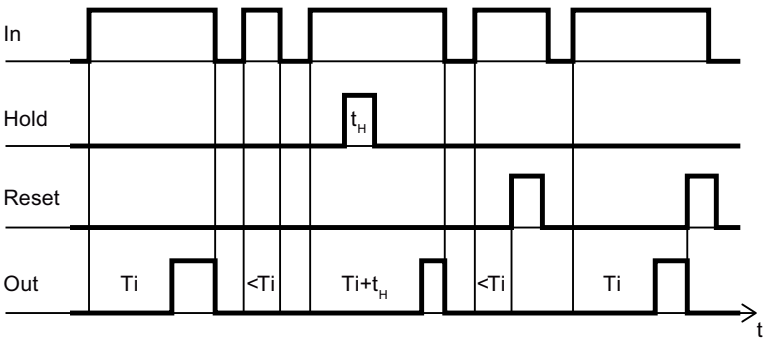
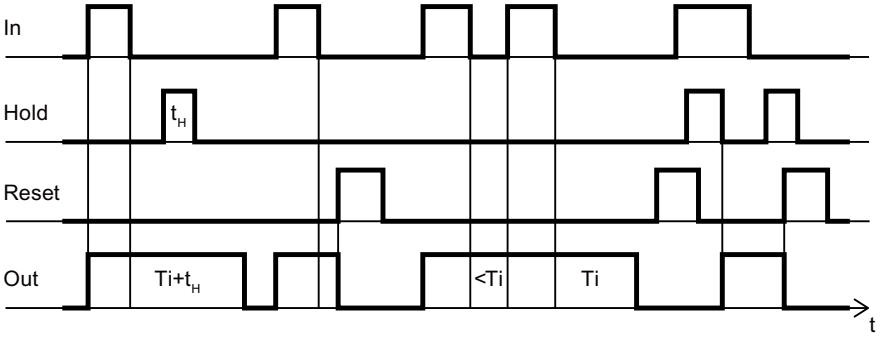
Functions of TimerP

The functions for this block are listed below.

Specifying how TimerP works

Use the `Mode` input parameter to specify how the block should work:

Mode =	How the block works	Graphic representation
0	Start timer as pulse	
1	Start extended pulse timer	
2	Start ON delay timer	

Mode =	How the block works	Graphic representation
3	Start ON delay timer with memory	
4	Start OFF delay timer	

Setting the time

You can use the `Ti` parameter to time for operating the block.

Note

Be aware when configuring this, that the time interval between `Ti` and `SampleTime` should not be more than 10^7.

Holding the `Out` and `TimeRemaining` output parameters

Use the `Hold.Value` = 1 parameter to freeze the calculation of the outputs `Out` and `TimeRemaining.Value`.

Resetting the `Out` and `TimeRemaining` output parameters

Use the `Reset.Value` = 1 parameter to reset the output parameters `Out` and `TimeRemaining.Value` to 0.

Initialization

With the input parameter `Init` = 1, the output parameters `Out` and `TimeRemaining` can be reset to 0 (same behavior as `Reset.Value` = 1). You can use the input parameter `Init` to realize a startup behavior.

To initialize the output parameters `Out` and `TimeRemaining` after a startup, connect the input parameter `Init` with the output parameter `InitOut` of the Trigger block. For more information, refer to Startup characteristics over Trigger block (Page 69).

Forming the signal status for blocks

The signal status for the block is formed using the following parameters and output at the `Out` and `InvOut` output parameters:

- `In.ST`

See also

Description of TimerP (Page 1757)

TimerP modes (Page 1758)

TimerP error handling (Page 1761)

TimerP messaging (Page 1762)

TimerP I/Os (Page 1762)

TimerP block diagram (Page 1764)

11.1.4 TimerP error handling

Error handling of TimerP

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
1	Invalid function type for the block set at the Mode input parameter
11	`Ti.Value` < `SampleTime` This error causes the `TimeRemaining` and `Out` output parameters to be set to 0.
33	The value of `Ti` can no longer be displayed in the REAL number field.

See also

Description of TimerP (Page 1757)
TimerP modes (Page 1758)
TimerP functions (Page 1758)
TimerP messaging (Page 1762)
TimerP I/Os (Page 1762)
TimerP block diagram (Page 1764)

11.1.5 TimerP messaging

Messaging

This block does not offer messaging.

See also

Description of TimerP (Page 1757)
TimerP modes (Page 1758)
TimerP functions (Page 1758)
TimerP error handling (Page 1761)
TimerP I/Os (Page 1762)
TimerP block diagram (Page 1764)

11.1.6 TimerP I/Os

I/Os of TimerP

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Hold	Hold calculation process	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
In	Input signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Init	Initialize function block	BOOL	0
Mode	How the block works: 0 = Start timer as pulse 1 = Start timer as extended pulse 2 = Start timer with ON delay 3 = Start timer with latching ON delay 4 = Start timer with OFF delay	INT	2
Reset	1 = Reset output Out	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
Ti*	Time [s]	REAL	0.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see TimerP error handling (Page 1761).	INT	-1
InvOut	Inverted output signal	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
Out	Output signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
TimeRemaining	Remaining time [s]	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of TimerP (Page 1757)

TimerP modes (Page 1758)

TimerP functions (Page 1758)

TimerP messaging (Page 1762)
TimerP block diagram (Page 1764)

11.1.7 TimerP block diagram

TimerP block diagram

A block diagram is not provided for this block.

See also

Description of TimerP (Page 1757)
TimerP modes (Page 1758)
TimerP functions (Page 1758)
TimerP error handling (Page 1761)
TimerP messaging (Page 1762)
TimerP I/Os (Page 1762)

11.2 TimeTrig - Calculations with the date formats DT and TIME

11.2.1 Description of TimeTrig

Object name (type + number) and family

Type + number: FB 1802

Family: Time

Area of application of TimeTrig

The block is used for the following applications:

- Daily, weekly or monthly generation of a pulse (trigger) using a definable time in hours (0 to 23h) and day of the week (Sun to Sat) or one day (1 to 31) per month.
- Delayed generation a pulse (trigger) based on the current CPU time and a delay time in seconds (0 to 20 days).

How it works

When a trigger is enabled, the next trigger time point and the duration until the next trigger time point is determined based on the CPU time. A periodic trigger and additionally a single trigger can be generated:

1. Generate periodic trigger (`PerTrigOn.Value = 1`)
2. Generate single trigger (`(SglTrigOn.Value = 1`)

The single trigger can be delayed by a specified period of time. The maximum delay is 20 days.

The period of time until the next trigger time point is displayed in days, hours, and minutes.

For calculations with the time formats DT and TIME, the block calls the following functions:

Name	APL V8.1	Family	Symbol Comment
AD_DT_TM_APL	FC455	IEC	DT + add time
GE_DT_APL	FC456	IEC	Greater than or equal to DT
LE_DT_APL	FC452	IEC	Less than or equal to DT
LT_DT_APL	FC453	IEC	Smaller than DT
SB_DT_DT_APL	FC454	IEC	DT - DT subtract

They come from the standard library "IEC function blocks" and are now also applied in the APL.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (`OB30` to `OB38`). The block is first installed automatically in the startup OB (`OB100`).

Startup characteristics

Define the startup characteristics for this block via Feature bit 0:

- Bit 0: Set startup characteristics (Page 137)

Status word allocation for `Status` parameter

Refer to the following section for a description of the individual parameters: TimeTrig I/Os (Page 1772).

Status bit	Parameter
0	Occupied
1	`BatchEn`
2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`
7	All triggers are switched off: NOT `PerTrigOn.Value OR SglTrigOn.Value OR Trigger.Value`
8	`PerTrigOn.Value`
9	NOT `PerTrigOn.Value`
10	`SglTrigOn.Value`
11	NOT `SglTrigOn.Value`
12	`Trigger.Value`
13	Not used
14	Next trigger is a daily trigger
15	Next trigger is a weekly trigger
16	Next trigger is a monthly trigger
17	Next trigger is a single trigger
18	`DailyOn`: Periodic trigger set to daily
19	`WeeklyOn`: Periodic trigger set to weekly
20	`MonthlyOn`: Periodic trigger set to monthly
21	Weekday is Sunday
22	Weekday is Monday
23	Weekday is Tuesday
24	Weekday is Wednesday
25	Weekday is Thursday
26	Weekday is Friday
27	Weekday is Saturday
28	`LiOp.Value`
29	1 = Invalid signal status `SetPerLi.Value` and `RstPerLi.Value`, `SetSglLi.Value` and `RstSglLi.Value`
30 - 31	Not used

See also

TimeTrig modes (Page 1767)

TimeTrig functions (Page 1767)

TimeTrig error handling (Page 1771)

TimeTrig messaging (Page 1772)

TimeTrig block diagram (Page 1776)

11.2.2 TimeTrig modes

TimerTrig operating modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

General information on the "On" mode is available in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

Description of TimeTrig (Page 1765)

TimeTrig functions (Page 1767)

TimeTrig error handling (Page 1771)

TimeTrig messaging (Page 1772)

TimeTrig I/Os (Page 1772)

TimeTrig block diagram (Page 1776)

11.2.3 TimeTrig functions

Functions of TimeTrig

The functions for this block are listed below.

Periodic output of a trigger

Use the LiOp input parameter to define whether the periodic trigger can be enabled (0 - 1, SetPerOp or SetPerLi parameter) or disabled (1 - 0, RstPerOp or RstPerLi parameter) via the faceplate or an interconnection.

LiOp = 0: Enable/disable via faceplate (SetPerOp or RstPerOp)

LiOp = 1: Enable/disable via interconnection (SetPerLi or RstPerLi)

You can select the faceplate from three periodic trigger settings:

- Periodic daily output DailyOn =1
  You set the time in hours with the HourOfDay parameter.
- Periodic weekly output WeeklyOn =1
  You set the day of the week and time of day with the HourOfDay and DayOfWeek parameters.
- Periodic monthly output MonthlyOn =1
  You set the day of the month and time of day with the HourOfDay and DayOfMonth parameters. If the day of a month is greater than the last day, the last day is used.

Delayed output of a single trigger

Use the LiOp input parameter to define whether the single trigger can be set (0 - 1, SetPerOp or SetPerLi parameter) or reset (1 - 0, RstPerOp or RstPerLi parameter) via the faceplate or an interconnection.

LiOp = 0: Set/reset via faceplate (SetPerOp or RstPerOp)

LiOp = 1: Set/reset via interconnection (SetPerLi or RstPerLi)

The output is delayed by the delay time DelayTime starting from the current CPU time. The currently effective delay time is displayed at the DlyTmEff output.

Calculation of the next trigger time point

The next trigger time point is displayed at the NxTrgDT, NxTrgYear, NxTrgMon, NxTrgDay, NxTrgHour, NxTrgMin and NxTrgSec outputs . The next trigger time point is calculated from an enabled periodic trigger and an activated single trigger.

If no trigger is activated, the the minimum DT#1990-01-01-0:0:0.0 is output at the NxTrgDT output.

Calculating the time period until the next trigger time point

The period of time until the next trigger time point based on the current CPU time is output at the DiffTmDay, DiffTmHour, DiffTmMin and DiffTmSec outputs.

If no trigger is activated, 0 is set at the outputs.

Reading back the trigger states and the single trigger time point for a complete download

The `PerTrigOn`, `SglTrigOn` trigger states and the single trigger time point are written back to inputs:

- `InPerTrigOn := PerTrigOn`
- `InSglTrigOn := SglTrigOn`
- `InSglTrigDT := SglTrigDT`

If you want to use the reset for a complete download, you must read back the marked parameters in addition to the operated and monitored parameters before a complete download.

The values are written back to the outputs at startup and when feature bit 0 = 1.

Also, in section Set startup characteristics (Page 137), see under "Set startup characteristics for the TimeTrig block".

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
1	Reaction to the out of service mode (Page 176)
24	Enabling local operator authorization (Page 157)

Operator permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 = Operator can switch to "On" mode
2	Not used
3	1 = Operator can switch to "Out of service" mode
4	1 = Operator can enable the periodic trigger
5	1 = Operator can disable the periodic trigger
6	1 = Operator can set the single trigger
7	1 = Operator can reset the single trigger
8	Not used
9	Not used
10	1 = Operator can enter `HourOfDay`
11	1 = Operator can enter `DayOfWeek`
12	1 = Operator can enter `DayOfMonth`
13	1 = Operator can enter `DelayDay`

Bit	Function
14	1 = Operator can enter DelayHour
15	1 = Operator can enter DelayMin
16	1 = Operator can enter PulseWidth
17	1 = Operator can set LocalTime
18	1 = Operator can enter DailyOn
19	1 = Operator can enter WeeklyOn
20	1 = Operator can enter MonthlyOn
21 - 27	used in OSPermLog for the weekdays Sun, Mon … Sat
28 - 31	Not used

Recognizing daylight saving time

The block reads the identifier for daylight saving time from the CPU and writes it to the SummerTime output. The update occurs only when

- The HourOfDay, DayOfWeek, DayOfMonth, DelayTime, DailyOn, WeeklyOn, MonthlyOn, or LocalTime parameters are changed
- A PerTrigOn or SglTrigOn trigger is enabled
- At startup or leaving Out of Service

If an error occurs when reading the daylight saving time, the status SummerTime.ST := 16#00 is set and the display is not updated. The status is not used in the calculation of ST_Worst.

Local time

In every cycle, the block reads the CPU time. With LocalTime = 1 the block calculates a local time. To this a correction value will be read from CPU and build a difference time, which is added to the CPU time. This correction takes into account the time zone and the time difference in summer time (daylight savings time) and winter time (standard time).

The update of the correction value occurs only when:

- The HourOfDay, DayOfWeek, DayOfMonth, DelayTime, DailyOn, WeeklyOn, MonthlyOn, or LocalTime parameters are changed.
- A PerTrigOn or SglTrigOn trigger is enabled.
- At startup or leaving "Out of Service" mode.

If an error occurs while reading the correction value, the status SummerTime.ST is set to 16#00 and the block does not changes the CPU time.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The worst signal status `ST_Worst` for the block is formed from the following parameter:

- `SetSglLi.ST`
- `RstSglLi.ST`
- `SetPerLi.ST`
- `RstPerLi.ST`

The worst signal status `ST_Worst` is written to the following outputs:

- `SglTrigOn.ST`
- `PerTrigOn.ST`
- `Trigger.ST`

If an error occurs when reading the daylight saving time, the status `SummerTime.ST := 16#00` is set and the display is not updated. The status is not used in the calculation of `ST_Worst`.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

See also

Description of TimeTrig (Page 1765)

TimeTrig modes (Page 1767)

TimeTrig error handling (Page 1771)

TimeTrig messaging (Page 1772)

TimeTrig I/Os (Page 1772)

TimeTrig block diagram (Page 1776)

11.2.4 TimeTrig error handling

Error handling for TimeTrig

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

Overview of error numbers

The ErrorNum I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.
51	Invalid signal at LiOp =1 • SetPerLi = 1 and RstPerLi = 1 • SetSglLi = 1 and RstSglLi = 1

See also

Description of TimeTrig (Page 1765)

TimeTrig modes (Page 1767)

TimeTrig functions (Page 1767)

TimeTrig I/Os (Page 1772)

TimeTrig block diagram (Page 1776)

11.2.5 TimeTrig messaging

Messaging

This block does not offer messaging.

See also

TimeTrig modes (Page 1767)

TimeTrig functions (Page 1767)

TimeTrig error handling (Page 1771)

TimeTrig I/Os (Page 1772)

TimeTrig block diagram (Page 1776)

Description of TimeTrig (Page 1765)

11.2.6 TimeTrig I/Os

I/Os of TimeTrig

Input parameters

Parameter	Description	Type	Default
InPerTrigOn	Periodic trigger 'On' (stored in the input variable)	BOOL	0
InSglTrigOn	Single trigger 'On' (stored in the input variable)	BOOL	0
InSglTrigDT	Single trigger data and time (stored in the input variable)	DATE_AND_TIME	1990-01-01-0:00:00
LiOp	Switchover of operating mode between: 1 = Interconnection 0 = Operator	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SetPerOp	Set periodic trigger via operator	BOOL	0
RstPerOp	Reset periodic trigger via operator	BOOL	0
SetPerLi	Set periodic trigger via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstPerLi	Reset periodic trigger via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SetSglOp	Set single trigger via operator	BOOL	0
RstSglOp	Reset single trigger via operator	BOOL	0
SetSglLi	Set single trigger via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstSglLi	Reset single trigger via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PerMode	Periodic trigger; 1: daily, 2: weekly; 3: monthly	INT	1
DayOfWeek	Setpoint weekday for periodic trigger [1..7]	INT	1
DayOfMonth	Setpoint day of month for periodic trigger [1..31]	INT	1
HourOfDay	Setpoint hour of day for periodic trigger [0..23]	INT	0
DelayTime	Delay time [sec]	DINT	0
PulseWidth	Pulse width of trigger signal [s]	REAL	3.0
SampleTime	Sampling time [s]	REAL	0.1
LocalTime	1 = Local time ON	BOOL	0
OnOp*	1 = "On" mode via operator	BOOL	0
OosOp*	1 = "Out of service", via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00
BatchEn	1 = Enable allocation	BOOL	0
BatchID	Batch ID	DWORD	16#00000000

Parameter	Description	Type	Default
BatchName	Batch name	STRING[32]	''
StepNo	Batch step number	DWORD	16#00000000
Occupied	Occupied by batch	BOOL	0
SelFp1	Open faceplate 1	ANY	
SelFp2	Open faceplate 2	ANY	
OS_Perm	I/O for operator permissions (Page 1767)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
OpSt_In	Input parameter for local operator authorization, to be connected with the Out output parameter of the upstream block, "OpStations"	DWORD	16#00000000
Feature	I/O for additional functions (Page 1767)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
PerTrigOn	1 = Periodic trigger On	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SglTrigOn	1 = Single trigger On	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Trigger	Trigger signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CPU_Year	CPU time, year	INT	0
CPU_Mon	CPU time, month	INT	0
CPU_Day	CPU time, day	INT	0
CPU_Hour	CPU time, hour	INT	0
CPU_Min	CPU time, minute	INT	0
CPU_Sec	CPU time, minute	INT	0
CPU_DayOfWeek	CPU time, weekday	INT	0
SglTrigDT	Single trigger date and time	DATE_AND_TIME	1990-01-01-0:00:00
NxTrgDT	Next trigger time point date and time	DATE_AND_TIME	1990-01-01-0:00:00

Parameter	Description	Type	Default
NxTrgYear	Next trigger time point year	INT	0
NxTrgMon	Next trigger time point months	INT	0
NxTrgDay	Next trigger time point days	INT	0
NxTrgHour	Next trigger time point hours	INT	0
NxTrgMin	Next trigger time point minutes	INT	0
NxTrgSec	Next trigger time point seconds		
DiffTmDay	Day: Period of time starting from the current CPU time until the next trigger time point	INT	0
DiffTmHour	Hours: Period of time starting from the current CPU time until the next trigger time point	INT	0
DiffTmMin	Minutes: Period of time starting from the current CPU time until the next trigger time point	INT	0
DiffTmSec	Seconds: Period of time starting from the current CPU time until the next trigger time point	INT	0
DlyTmEff	Current effective delay time [sec]	DINT	0
BarTmSec	Bar graph seconds	DINT	0
BarOpHiScale	Bar graph minutes	DINT	60
DayOpHiScale	Scale graph days	INT	1
HrsOpHiScale	Scale graph hours	INT	0
MinOpHiScale	Scale graph minutes	INT	0
SummerTime	Daylight saving time	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnAct	1 = Block is in service	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OS_PermOut	Operator permission: Output for OS	DWORD	16#FFFFFFFF
OS_PermLog	Operator permission: Output for OS	DWORD	16#FFFFFFFF
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
ST_Worst	Worst signal status	BYTE	16#80
Status	Status word (Page 1765)	DWORD	16#00000000
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see TimeTrig error handling (Page 1771).	INT	-1

See also

TimeTrig modes (Page 1767)

TimeTrig messaging (Page 1772)

TimeTrig block diagram (Page 1776)

11.2.7 TimeTrig block diagram

Block diagram of TimeTrig

A block diagram is not provided for this block.

See also

TimeTrig modes (Page 1767)

TimeTrig functions (Page 1767)

TimeTrig error handling (Page 1771)

TimeTrig messaging (Page 1772)

TimeTrig I/Os (Page 1772)

11.2.8 Operator control and monitoring

11.2.8.1 TimeTrig views

Views of the TimeTrig block

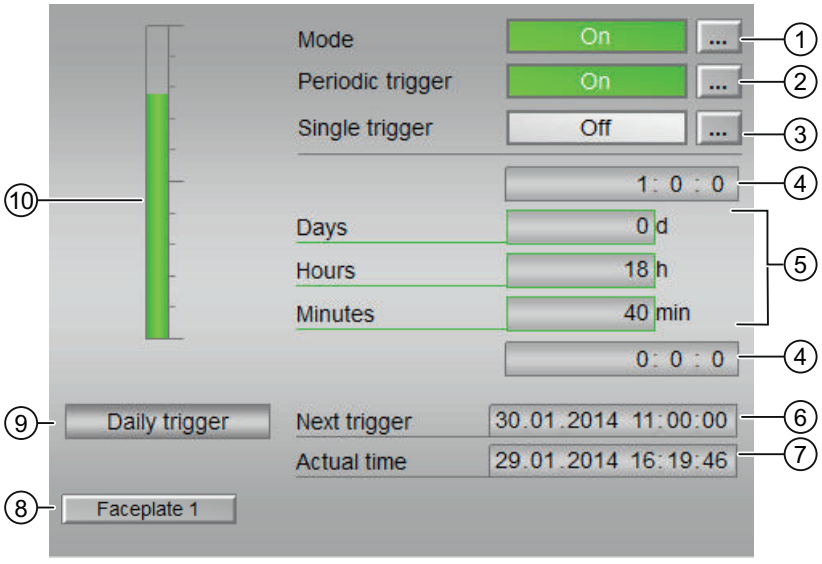
The block TimeTrig provides the following views:

- TimeTrig standard view (Page 1777)
- TimeTrig parameter view (Page 1779)
- TimeTrig preview (Page 1781)
- Memo view (Page 298)
- Block icon for TimeTrig (Page 1782)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

11.2.8.2 TimeTrig standard view

Standard view of TimeTrig



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

(2) Enabling and disabling the period trigger

This area shows the default operating mode of the trigger. The following operating modes can be displayed and set here:

- "On"
- "Off"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

(3) Enabling and disabling the single trigger

This area shows the default operating mode of the trigger. The following operating modes can be displayed and set here:

- "On"
- "Off"

Refer to the Switching operating states and operating modes (Page 251) section for information on changing the state.

(4) High and low scale range for the count value

This value provides information on the display range for the bar graph (above) of the count value. The scale range is defined in the engineering system.

(5) Display of the count value

The following counts are shown here: The amount of time until the next trigger is shown here.

- "Days"
- "Hours"
- "Minutes"

(6) Display of the time point of the next trigger

This area shows the time of the next trigger.

(7) Displaying of the actual time of day

This area shows the actual time of day.

(8) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system (ES). The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203) .

(9) Displaying of the actual trigger operating mode

- "Daily trigger"
- "Weekly trigger"
- "Monthly trigger"
- "Single trigger"

(10)Graphic display of the current count value

This area shows you the current count in form of a bar graph. The visible area in the bar graph depends on the configuration in the engineering system (ES).

See also

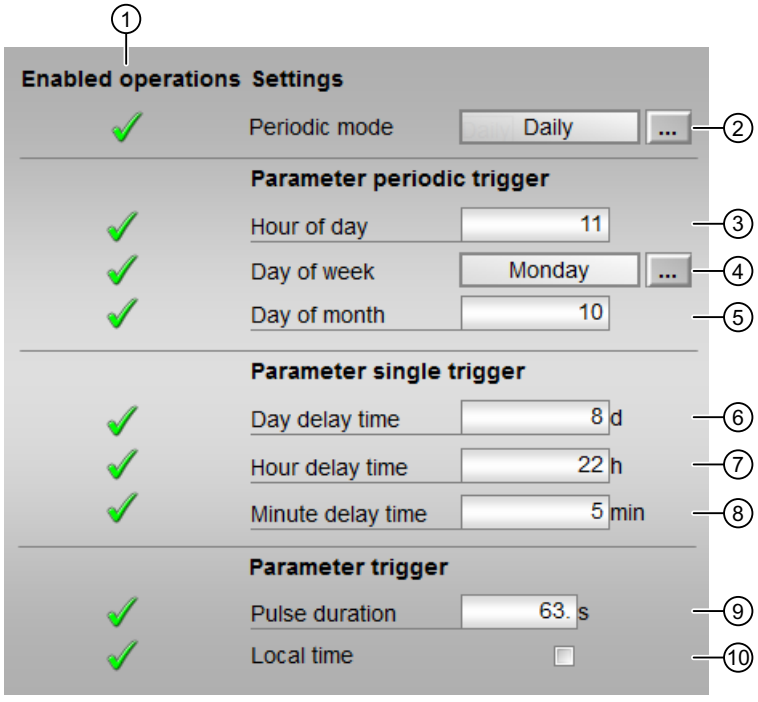
Release for maintenance (Page 64)

Simulating signals (Page 58)

Error handling (Page 119)

11.2.8.3 TimeTrig parameter view

Parameter view of TimeTrig



(1) Enabled operation

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`).

(2) Periodic operating mode

You can change the operating mode of the periodic trigger:

- "Daily": The trigger is activated once a day.
- "Weekly": The trigger is activated once a week.
- "Monthly": The trigger is activated once a month.

(3), (4) and (5) parameters for the periodic trigger

Enter the relevant parameters for the periodic mode:

- "Time": relevant for daily, weekly and monthly trigger
- "Weekday": relevant for weekly and monthly trigger
- "Day of the month": Relevant for monthly trigger. If the day of a month is greater than the last day, the last day is used.

(6), (7) and (8) parameters for single triggers

Enter the parameters for the single trigger:

- "Delay time day"
- "Delay time hour"
- "Delay time minute"

(9) and (10) parameters for the trigger

- "Pulse duration": Enter the duration of the trigger signal.
- "Local time": Activate local time by selecting the check box (☑).

See also

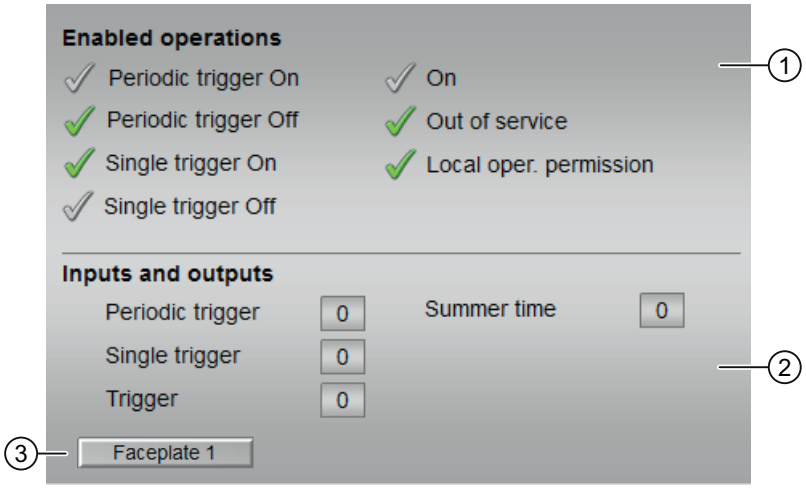
Changing values (Page 253)

Switching operating states and operating modes (Page 251)

Simulating signals (Page 58)

11.2.8.4 TimeTrig preview

Preview of TimeTrig



(1) Enable operations

This area shows all operations for which special operator permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Icons for enabled operation:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- "Periodic trigger on": You can enable the periodic trigger.
- "Periodic trigger off": You can disable the periodic trigger.
- "Single trigger on": You can enable the single trigger.
- "Single trigger off": You can disable the single trigger.
- "On": You can operate the trigger.
- "Out of service": You can switch to "Out of service" operating mode.
- "Local operator permission": Use the ← button to switch to the standard view of the "OpStations" block.
  You can find additional information on this in the section Operator control permissions (Page 248).

(2) Display of current operating signals

This area shows the most important parameters for this block with the current selection.

- "Periodic trigger":
  - 0 = Periodic trigger is enabled
  - 1 = Periodic trigger is disabled
- "Single trigger":
  - 0 = Single trigger is disabled
  - 1 = Single trigger is enabled
- "Trigger":
  - 0 = Output trigger signal is disabled
  - 1 = Output trigger signal is enabled
- "Daylight saving time":
  - 0 = Daylight saving time is disabled
  - 1 = Daylight saving time is enabled

(3) Navigation button for switching to the standard view of any faceplate

Use this navigation button to open the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the section Opening additional faceplates (Page 203).

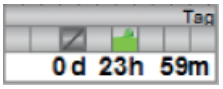
11.2.8.5 Block icon for TimeTrig

Block icons for TimeTrig

A variety of block icons are available with the following functions:

- Process tag type
- Operating modes
- Signal status, release for maintenance
- Memo display
- "Time until the next trigger" display is running

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

Mathematical blocks

12

12.1 AbsR - Absolute value of a real value

12.1.1 Description of AbsR

Object name (type + number) and family

Type + number: FC 395

Family: Math

Area of application for AbsR

The block is used for getting the absolute value of a real value.

How it works

The AbsR block provides the absolute value of a real value. The following equation is used for the calculation:

`Out` = |`In`|

Where:

`In` = Input value

`Out` = Absolute value

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

12.1.2 AbsR modes

AbsR operating modes

This block does not have any modes.

12.1.3 AbsR functions

Functions of AbsR

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The worst signal status for the block is formed by the following parameter and output at the `Out` output parameter:

- `In.ST`

12.1.4 AbsR error handling

AbsR error handling

The block does not report any errors.

12.1.5 AbsR messaging

Messaging

This block does not offer messaging.

12.1.6 AbsR I/Os

AbsR I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In	Analog input value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Analog output value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

12.1.7 AbsR block diagram

AbsR block diagram

A block diagram is not provided for this block.

12.2 Add04 - Adder with 4 values

12.2.1 Description of Add04

Object name (type + number) and family

Type + number: FC 351

Family: Math

Area of application for Add04

The block is used for the following applications:

- Adding values
- Output of the total value for further processing

How it works

The Add04 block calculates the sum of up to 4 values:

`Out` = `In1` +...+ `Inn` (`n` ≤ 4),

Where:

`Out` = sum value

`In1` ... `In4`= values to be added

The sum of all input parameters and the worst input parameter signal status is always output.

The output value is checked to determine if it is within the value range of REAL. If it is outside the value range, the highest or lowest possible REAL value is output.

If the value is NAN, the last valid output value is output and the status of the output value is set to 16#28 (if none of the input statuses is worse).

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Add04 functions (Page 1789)
Add04 messaging (Page 1790)
Add04 I/Os (Page 1791)
Add04 block diagram (Page 1792)
Add04 error handling (Page 1790)
Add04 modes (Page 1789)

12.2.2 Add04 modes

Add04 operating modes

This block does not have any modes.

See also

Add04 block diagram (Page 1792)
Add04 I/Os (Page 1791)
Add04 messaging (Page 1790)
Add04 error handling (Page 1790)
Description of Add04 (Page 1788)
Add04 functions (Page 1789)

12.2.3 Add04 functions

Functions of Add04

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The worst signal status for the block is formed by the following parameters and output at the `Out` output parameter:

- `In1.ST`
- `In2.ST`

- `In3.ST`
- `In4.ST`

See also

Description of Add04 (Page 1788)
Add04 messaging (Page 1790)
Add04 I/Os (Page 1791)
Add04 block diagram (Page 1792)
Add04 error handling (Page 1790)
Add04 modes (Page 1789)

12.2.4 Add04 error handling

Add04 error handling

The block does not report any errors.

See also

Add04 block diagram (Page 1792)
Add04 I/Os (Page 1791)
Add04 messaging (Page 1790)
Description of Add04 (Page 1788)
Add04 modes (Page 1789)
Add04 functions (Page 1789)

12.2.5 Add04 messaging

Messaging

This block does not offer messaging.

See also

Description of Add04 (Page 1788)
Add04 functions (Page 1789)
Add04 I/Os (Page 1791)

Add04 block diagram (Page 1792)
Add04 modes (Page 1789)
Add04 error handling (Page 1790)

12.2.6 Add04 I/Os

Add04 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Value 1 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In2	Value 2 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In3	Value 3 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In4	Value 4 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Output of the sum	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of Add04 (Page 1788)
Add04 functions (Page 1789)
Add04 messaging (Page 1790)
Add04 block diagram (Page 1792)

Add04 modes (Page 1789)
Add04 error handling (Page 1790)

12.2.7 Add04 block diagram

Add04 block diagram

A block diagram is not provided for this block.

See also

Description of Add04 (Page 1788)
Add04 modes (Page 1789)
Add04 functions (Page 1789)
Add04 error handling (Page 1790)
Add04 messaging (Page 1790)
Add04 I/Os (Page 1791)

12.3 Add08 - Adder with 8 values

12.3.1 Description of Add08

Object name (type + number) and family

Type + number: FC 352

Family: Math

Area of application for Add08

The block is used for the following applications:

- Adding values
- Output of the total value for further processing

How it works

The Add08 block calculates the sum of up to 8 values:

`Out` = `In1` +...+ `Inn (n` ≤ 8),

Where:

`Out` = sum value

`In1` ... `In8` = values to be added

The sum of all input parameters and the worst input parameter signal status is always output.

The output value is checked to determine if it is within the value range of REAL. If it is outside the value range, the highest or lowest possible REAL value is output.

If the value is NAN, the last valid output value is output and the status of the output value is set to 16#28 (if none of the input statuses is worse).

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Add08 functions (Page 1794)
Add08 messaging (Page 1795)
Add08 I/Os (Page 1796)
Add08 block diagram (Page 1797)
Add08 error handling (Page 1795)
Add08 modes (Page 1794)

12.3.2 Add08 modes

Add08 operating modes

This block does not have any modes.

See also

Description of Add08 (Page 1793)
Add08 functions (Page 1794)
Add08 error handling (Page 1795)
Add08 messaging (Page 1795)
Add08 I/Os (Page 1796)
Add08 block diagram (Page 1797)

12.3.3 Add08 functions

Functions of Add08

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The worst signal status for the block is formed by the following parameters and output at the `Out` output parameter:

- `In1.ST`
- `In2.ST`
- `In3.ST`

- `In4.ST`
- `In5.ST`
- `In6.ST`
- `In7.ST`
- `In8.ST`

See also

Description of Add08 (Page 1793)
Add08 messaging (Page 1795)
Add08 I/Os (Page 1796)
Add08 block diagram (Page 1797)
Add08 error handling (Page 1795)
Add08 modes (Page 1794)

12.3.4 Add08 error handling

Add08 error handling

The block does not report any errors.

See also

Add08 block diagram (Page 1797)
Add08 I/Os (Page 1796)
Add08 messaging (Page 1795)
Description of Add08 (Page 1793)
Add08 functions (Page 1794)
Add08 modes (Page 1794)

12.3.5 Add08 messaging

Messaging

This block does not offer messaging.

See also

Description of Add08 (Page 1793)
Add08 functions (Page 1794)
Add08 I/Os (Page 1796)
Add08 error handling (Page 1795)
Add08 modes (Page 1794)
Add08 block diagram (Page 1797)

12.3.6 Add08 I/Os

Add08 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Value 1 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In2	Value 2 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In3	Value 3 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In4	Value 4 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In5	Value 5 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In6	Value 6 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
In7	Value 7 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In8	Value 8 to add	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Output of the sum	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of Add08 (Page 1793)
Add08 functions (Page 1794)
Add08 messaging (Page 1795)
Add08 block diagram (Page 1797)
Add08 error handling (Page 1795)
Add08 modes (Page 1794)

12.3.7 Add08 block diagram

Add08 block diagram

A block diagram is not provided for this block.

See also

Description of Add08 (Page 1793)
Add08 functions (Page 1794)
Add08 error handling (Page 1795)
Add08 I/Os (Page 1796)
Add08 modes (Page 1794)
Add08 messaging (Page 1795)

12.4 Average - Mean value calculation

12.4.1 Description of Average

Object name (type + number) and family

Type + number: FB1804

Family: Math

Area of application for Average

The block is used for the following applications:

- Calculation of a time-based mean value

How it works

The block calculates the time-based mean value of an analog value `In` based on the time which has expired since its start. The equation below is used:

$$Out = \frac{Out_{(n-1)} \cdot NumCycles + In}{NumCycles + 1}$$

Where:

`In` = input variable

`Out` = present average value

$Out_{(n-1)}$ = calculated average upon successful startup

`NumCycles` = Number of cycles for creating the mean value from the 0 - 1 edge transition of the Run input parameter.

The calculation is started by a 0 - 1 edge at the `Run` input parameter. The `Out` output parameter is overwritten by the `In` input parameter.

The result at output value `Out` is recalculated in the next cycles, and cycle counter `NumCycles` is incremented.

The `NumCycles` cycle counter has the DINT data type and can therefore assume the maximum value of 2147483647. When this value is reached, the `NumCycles` cycle counter is reset to 1. The formation of the mean is then started once again. At a cycle time of 100 ms, this case occurs every 6.8 years.

Calculation is terminated by resetting (1 - 0 edge) the `Run` input parameter. The last results set at `Out` and `NumCycles` are saved.

The parameter `Feature` can be used to specify the startup characteristics of the block.

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

Average functions (Page 1800)

Average messaging (Page 1801)

Average I/Os (Page 1802)

Average modes (Page 1799)

Average error handling (Page 1800)

Average block diagram (Page 1803)

12.4.2 Average modes

Average operating modes

This block does not have any modes.

See also

Description of Average (Page 1798)

Average functions (Page 1800)

Average error handling (Page 1800)

Average messaging (Page 1801)

Average I/Os (Page 1802)

Average block diagram (Page 1803)

12.4.3 Average functions

Functions of Average

The functions for this block are listed below.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Functions
0	Set startup characteristics (Page 137)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The signal status for the block is formed using the following parameters and output at the `Out` output parameter:

- `In.ST`

See also

Description of Average (Page 1798)

Average messaging (Page 1801)

Average I/Os (Page 1802)

Average modes (Page 1799)

Average error handling (Page 1800)

Average block diagram (Page 1803)

12.4.4 Average error handling

Error handling of Average

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Default value displayed after installation in the block.
0	There is no error.
30	The `In` value can no longer be displayed in the REAL number field, the 16#28 status is also output. The last valid value is provided at the `Out` output and calculation is halted.
31	The `Out` value can no longer be displayed in the REAL number field, the 16#28 status is also output. The last valid value is provided at the `Out` output.

See also

Average modes (Page 1799)
Average functions (Page 1800)
Description of Average (Page 1798)
Average messaging (Page 1801)
Average I/Os (Page 1802)
Average block diagram (Page 1803)

12.4.5 Average messaging

Messaging

This block does not offer messaging.

See also

Description of Average (Page 1798)
Average functions (Page 1800)
Average I/Os (Page 1802)
Average modes (Page 1799)
Average error handling (Page 1800)
Average block diagram (Page 1803)

12.4.6 Average I/Os

I/Os of Average

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1800)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
In*	Analog input value for mean value calculation	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Run	1 = Start time-based mean value calculation	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
EndTime	1 = Time for ending time-based mean value calculation	DT	1990-01-01-0:00:00
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see Average error handling (Page 1800)	INT	-1
NumCycles	Cycle counter	DINT	1
Out	Output for average value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
StartTime	1 = Time for starting time-based mean value calculation	DT	1990-01-01-0:00:00

See also

Description of Average (Page 1798)

Average messaging (Page 1801)

Average modes (Page 1799)
Average block diagram (Page 1803)

12.4.7 Average block diagram

Average block diagram

A block diagram is not provided for this block.

See also

Average modes (Page 1799)
Average functions (Page 1800)
Average error handling (Page 1800)
Average messaging (Page 1801)
Description of Average (Page 1798)
Average I/Os (Page 1802)

12.5 DeadTime - Delayed signal output

12.5.1 Description of DeadTime

Object name (type + number) and family

Type + number: FB 1807

Family: Math

Area of application for DeadTime

The block provides the following functions:

- Delay [s] of signal output

How it works

This block delays the output of an input value by a time `DeadTime` [s] selected by the user.

Dead times up to 100 times the `SampleTime` can be realized. If you want to set a longer dead time, you need to connect several dead time blocks in sequence. You can also insert the dead time block in a runtime group with a longer sampling time or scale down the runtime group, but this would lead to loss of input values.

The block operates as follows:

`Out` = `In` (t - `DeadTime`)

Where:

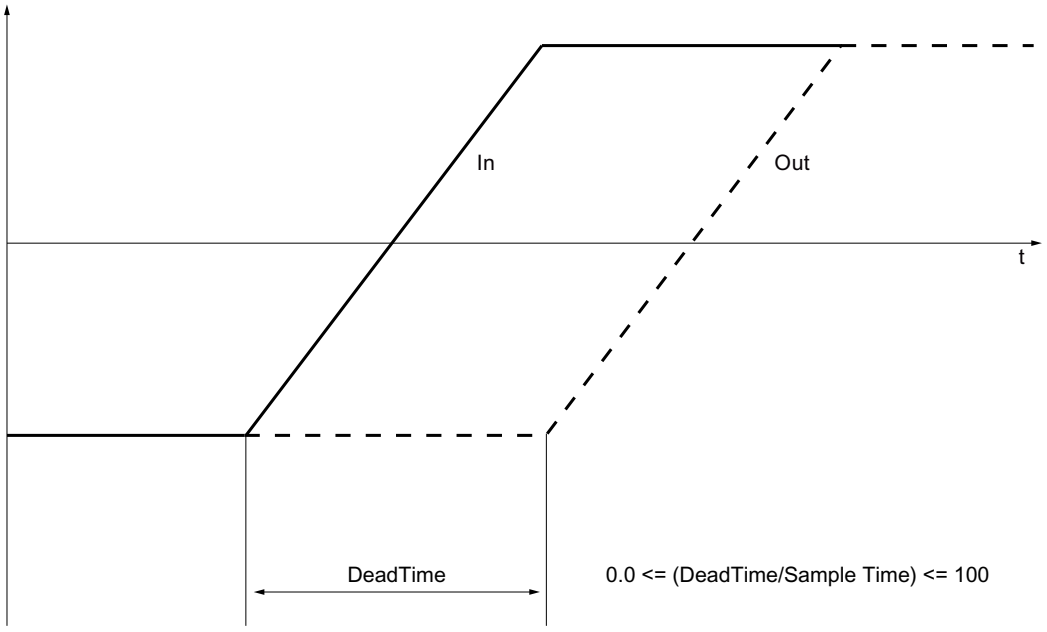
`Out` = output value

t = current time

`DeadTime` = `DeadT_Cyc` · `SampleTime` where:

`DeadT_Cyc` = Number of cycles [0...100], by which the analog input value is delayed

An analog value of input `In` is only output after the variable dead time at `DeadTime` has expired. It is output at `Out`.



You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Note

If you change the value for the dead time during the runtime of the block, the input value is written to the output. The change to the dead time then takes effect.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

The resulting sampling time of the block must be long enough for the desired dead time to be specified at the `DeadTime` parameter.

For the DeadTime block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303)
- PID controller with Smith predictor (SmithPredictorControl) (Page 2306)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

DeadTime functions (Page 1806)
DeadTime messaging (Page 1808)
DeadTime I/Os (Page 1809)
DeadTime block diagram (Page 1810)
DeadTime modes (Page 1806)
DeadTime error handling (Page 1807)

12.5.2 DeadTime modes

DeadTime operating modes

This block does not have any modes.

See also

DeadTime block diagram (Page 1810)
DeadTime I/Os (Page 1809)
DeadTime messaging (Page 1808)
DeadTime error handling (Page 1807)
DeadTime functions (Page 1806)
Description of DeadTime (Page 1804)

12.5.3 DeadTime functions

Functions of DeadTime

The functions for this block are listed below.

Configurable reactions using the `Feature` I/O

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) section. The following functionality is available for this block at the relevant bits:

Bit	Functions
0	Set startup characteristics (Page 137)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The signal status for the block is formed using the following parameters and output at the `Out` output parameter:

- `In.ST`

See also

Description of DeadTime (Page 1804)

DeadTime messaging (Page 1808)

DeadTime I/Os (Page 1809)

DeadTime block diagram (Page 1810)

DeadTime modes (Page 1806)

DeadTime error handling (Page 1807)

12.5.4 DeadTime error handling

Error handling of DeadTime

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed.
0	There is no error.

Error number	Meaning of the error number
30	The value of `In` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.
32	The value of `DeadTime` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.
45	The value of `DeadTime` is outside the permitted range `0` ≤ `INT` (`DeadTime` / `SampleTime`) ≤ 100 If the value of `DeadTime` is greater than 100 · `SampleTime`, the block works internally with the maximum dead time of 100 · `SampleTime`.

See also

DeadTime block diagram (Page 1810)
DeadTime I/Os (Page 1809)
DeadTime messaging (Page 1808)
DeadTime modes (Page 1806)
Description of DeadTime (Page 1804)
DeadTime functions (Page 1806)

12.5.5 DeadTime messaging

Messaging

This block does not offer messaging.

See also

Description of DeadTime (Page 1804)
DeadTime functions (Page 1806)
DeadTime I/Os (Page 1809)
DeadTime block diagram (Page 1810)
DeadTime modes (Page 1806)
DeadTime error handling (Page 1807)

12.5.6 DeadTime I/Os

I/Os of DeadTime

Input parameters

Parameter	Description	Type	Default
DeadTime*	Dead time [s]: Time by which the analog input value is delayed	REAL	1.0
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1806)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
In*	Analog input value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see DeadTime error handling (Page 1807).	INT	-1
DeadT_Cyc	Number of cycles [0 to 100] by which the analog input value is delayed	INT	0
Out	Analog output value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of DeadTime (Page 1804)

DeadTime messaging (Page 1808)

DeadTime block diagram (Page 1810)

DeadTime modes (Page 1806)

12.5.7 DeadTime block diagram

DeadTime block diagram

A block diagram is not provided for this block.

See also

DeadTime I/Os (Page 1809)
DeadTime messaging (Page 1808)
DeadTime error handling (Page 1807)
DeadTime functions (Page 1806)
DeadTime modes (Page 1806)
Description of DeadTime (Page 1804)

12.6 Derivative - Obtaining a derivative

12.6.1 Description of Derivative

Object name (type + number) and family

Type + number: FB 1808

Family: Math

Area of application for Derivative

The block is used for the following applications:

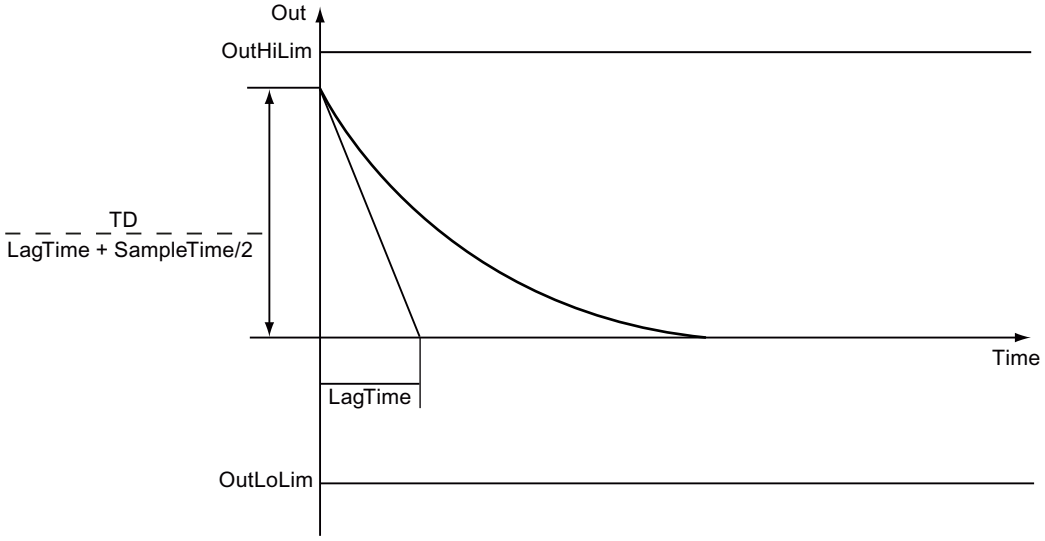
- Forming the D component for the controller design
- Forming a temporal derivative of the input signal

How it works

The block can be used as part of user-designed closed-loop controller. The block calculates a D component with delay. The derivative of the input signal is formed using the "trapezoid rule".

If only the derivative signal is to be calculated, `TD` = 1 [s] must be set. The speed signal is then formed from the travel signal.

The output signal can be damped using the `LagTime` parameter. Damping is disabled by configuring `LagTime` = 0. The following figure shows the step response from the unit step of the `In` input variable.



The block works according to the following formula:

$$Out_{(n)} = \left(\frac{TD}{LagTime + \frac{SampleTime}{2}} \right) \left(In_{(n)} - PrevIn_{(n)} \right)$$

The $PrevIn_{(n)}$ value is calculated for the next sampling step by means of a first order low pass filter:

$$PrevIn_{(n+1)} = PrevIn_{(n)} + \frac{SampleTime}{TD} Out_{(n)}$$

The following applies for both formulas:

- `Out`$_{(n)}$ = output value
- `TD` = derivative time
- `In`$_{(n)}$ = input value
- `PrevIn`$_{(n)}$ = memory variable of the low pass filter
- `SampleTime` = sampling time [s]
- `LagTime` = delay time [s]

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB3x). The block is then installed automatically in startup OB (OB100).

Further addressing is not required.

For the Derivative block, the Advanced Process Library contains a template for process tag types as an example with an application scenario for this block.

Example of process tag types:

- PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Derivative messaging (Page 1815)

Derivative error handling (Page 1815)

Derivative modes (Page 1813)

Derivative I/Os (Page 1816)

Derivative functions (Page 1814)

12.6.2 Derivative modes

Derivative operating modes

This block does not have any modes.

See also

Derivative block diagram (Page 1817)

Derivative I/Os (Page 1816)

Derivative messaging (Page 1815)

Derivative error handling (Page 1815)

Derivative functions (Page 1814)

Description of Derivative (Page 1811)

12.6.3 Derivative functions

Functions of Derivative

The functions for this block are listed below.

Monitoring the output parameter for limits

The `Out` output parameter can be checked for limit values:

- `OutHiLim`: High limit
- `OutLoLim`: Low limit

If the limits are infringed, this is indicated to you at the corresponding output parameters (output parameter `OutHiAct` or `OutLoAct` = 1).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in section Configurable functions using the Feature I/O (Page 130). The following behavior patterns are available for this block at the relevant bits

Bit	Function
0	Set startup characteristics (Page 137)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The signal status for the block is formed using the following parameters and output at the `Out`, `OutHiAct` and `OutLoAct` output parameters:

- `In.ST`

See also

Description of Derivative (Page 1811)

Derivative messaging (Page 1815)

Derivative I/Os (Page 1816)

Derivative block diagram (Page 1817)

Derivative error handling (Page 1815)

Derivative modes (Page 1813)

12.6.4 Derivative error handling

Error handling of Derivative

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed
0	There is no error.
30	The value of `In` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.
31	The value of `Out` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.
32	The value of `LagTime` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.
33	The value of `TD` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.

See also

Derivative block diagram (Page 1817)

Derivative I/Os (Page 1816)

Derivative messaging (Page 1815)

Description of Derivative (Page 1811)

Derivative modes (Page 1813)

Derivative functions (Page 1814)

12.6.5 Derivative messaging

Messaging

This block does not offer messaging.

See also

Description of Derivative (Page 1811)
Derivative functions (Page 1814)
Derivative I/Os (Page 1816)
Derivative block diagram (Page 1817)
Derivative modes (Page 1813)
Derivative error handling (Page 1815)

12.6.6 Derivative I/Os

I/Os of Derivative

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1814)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
In*	Analog input value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
LagTime*	Delay time [s]	REAL	10.0
OutHiLim	Limit (high) for output value	REAL	100.0
OutLoLim	Limit (low) for output value	REAL	0.0
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
TD*	Derivative time [s]	REAL	1.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see Derivative error handling (Page 1815)	INT	-1

Parameter	Description	Type	Default
Out	Output value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
OutHiAct	1= Limit (high) violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutLoAct	1= Limit (low) violated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Description of Derivative (Page 1811)
Derivative messaging (Page 1815)
Derivative block diagram (Page 1817)
Derivative modes (Page 1813)

12.6.7 Derivative block diagram

Derivative block diagram

A block diagram is not provided for this block.

See also

Description of Derivative (Page 1811)
Derivative modes (Page 1813)
Derivative functions (Page 1814)
Derivative error handling (Page 1815)
Derivative messaging (Page 1815)
Derivative I/Os (Page 1816)

12.7 Div02 - Division of two values

12.7.1 Description of Div02

Object name (type + number) and family

Type + number: FC 358

Family: Math

Area of application for Div02

The block is used for the following applications:

- Dividing two values
- Output of the division for further processing

How it works

The block is used to divide two values as follows:

`Out` = `In1` / `In2`

If the input parameter is `In2` = 0, the last valid output value is retained until another division is permitted mathematically.

The worst signal status is also always output at the output parameter.

The output value is checked to determine if it is within the value range of REAL. If it is outside the value range, the highest or lowest possible REAL value is output.

If the value is NAN or divided by 0, the last valid output value is output and the status of the output value is set to 16#28 (if none of the input statuses is worse).

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

Div02 block diagram (Page 1822)

Div02 I/Os (Page 1821)

Div02 messaging (Page 1821)

Div02 error handling (Page 1820)

Div02 functions (Page 1819)

Div02 modes (Page 1819)

12.7.2 Div02 modes

Div02 operating modes

This block does not have any modes.

See also

Div02 block diagram (Page 1822)

Div02 I/Os (Page 1821)

Div02 messaging (Page 1821)

Div02 error handling (Page 1820)

Div02 functions (Page 1819)

Description of Div02 (Page 1818)

12.7.3 Div02 functions

Functions of Div02

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The worst signal status for the block is formed by the following parameters and output at the `Out` output parameter:

- `In1.ST`
- `In2.ST`

See also

Div02 block diagram (Page 1822)
Div02 I/Os (Page 1821)
Div02 messaging (Page 1821)
Div02 error handling (Page 1820)
Div02 modes (Page 1819)
Description of Div02 (Page 1818)

12.7.4 Div02 error handling

Error handling of Div02

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed
0	There is no error.
30	It has been divided by 0.

See also

Div02 block diagram (Page 1822)
Div02 messaging (Page 1821)
Div02 functions (Page 1819)
Div02 modes (Page 1819)
Description of Div02 (Page 1818)
Div02 I/Os (Page 1821)

12.7.5 Div02 messaging

Messaging

This block does not offer messaging.

See also

Div02 block diagram (Page 1822)
Div02 I/Os (Page 1821)
Description of Div02 (Page 1818)
Div02 modes (Page 1819)
Div02 functions (Page 1819)
Div02 error handling (Page 1820)

12.7.6 Div02 I/Os

I/Os of Div02

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Dividend	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In2	Divisor	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see Div02 error handling (Page 1820).	INT	-1
Out	Output of division	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Div02 block diagram (Page 1822)

Description of Div02 (Page 1818)

Div02 modes (Page 1819)

Div02 functions (Page 1819)

Div02 messaging (Page 1821)

12.7.7 Div02 block diagram

Div02 block diagram

A block diagram is not provided for this block.

See also

Div02 I/Os (Page 1821)

Div02 messaging (Page 1821)

Div02 error handling (Page 1820)

Div02 functions (Page 1819)

Div02 modes (Page 1819)

Description of Div02 (Page 1818)

12.8 FlowCorr - Flow correction

12.8.1 Description of FlowCorr

Object name (type + number) and family

Type + number: FB 1916

Family: Math

Area of application for FlowCorr

The block is used for the following applications:

- Pressure and/or temperature compensation for measurements of the differential pressure method

The block calculates the pressure and/or temperature compensation for measurements with the differential pressure method. This refers to measurements from restrictors, standard flow nozzles, venturi nozzles and pitot tubes. The block can be used for gases and vapors, saturated steam and liquids to calculate the volume flow rate or, for liquids, to calculate the mass flow rate.

How it works

The block cyclically calculates the volume/mass flow rate.

Mode 1: Gas and steam:

In mode 1, the flow is calculated as shown in the following equation:

$$Q_{cal} = Q_0 * \sqrt{\frac{\Delta P}{\Delta P_0} * \frac{(P + P_B)}{(P_0 + P_B)} * \frac{(T_0 + T_{abs})}{(T + T_{abs})}}$$

Mode 2: Saturated steam, mass flow in liquids:

In mode 2, the flow is calculated as shown in the following equation:

$$Q_{cal} = Q_0 * \sqrt{\frac{\Delta P}{\Delta P_0} * \frac{\rho}{\rho_0}}$$

Mode 3: Volume flow for liquids:

In mode 3, the flow is calculated as shown in the following equation:

$$Q_{cal} = Q_0 * \sqrt{\frac{\Delta P}{\Delta P_0} * \frac{\rho_0}{\rho}}$$

Block I/O	Icon	Type	Description
Flow	Q_{cal}	Result	Calculated flow
Flow0	Q_0	Parameter	Design flow
DiffP	ΔP	Measurement	Differential pressure [1]
DiffP0	ΔP_0	Parameter	Design differential pressure
P	P	Parameter or measurement	Pressure [bar (abs. or relative)] [2]
P0	P_0	Parameter	Design pressure [bar (abs. or relative)] [3]
P_Atmos	P_B	Parameter or measurement	Barometric pressure [mbar (abs.)] [4]
Temp0	T_0	Parameter	Design temperature [0 °C]
Temp	T	Parameter or measurement	Temperature [0 °C]
TempAbs	T_{abs}	Parameter	[0 °C] in Kelvin (corresponds to 273.15K)
Density	ρ	Parameter or measurement	Density [5]
Density0	ρ_0	Parameter	Design density

1 When the sensor extracts the square root of the differential pressure, the P_SqrtOn parameter must be set to 'true'. In this case, the differential pressure value is squared before the flow has been calculated by the above equation.

2 If a sensor is used for absolute pressure, P_Atmos must be set to '0'. If a sensor is used for differential pressure, P_Atmos must either be set to the local average ambient pressure or linked to the ambient pressure measurement.

3 'P0' must be the same size as 'P' (e.g.: if a sensor for absolute pressure is used for 'P', there must be a matching value for the absolute pressure 'P0').

4 Unit of P_Atmos = unit of P / 1000 (e.g.: P: bar; P_Atmos: mbar)

5 Real density measurement can be connected. An alternative method is often used in practice to determine the density (a process pressure measurement for saturated steam or a temperature measurement for liquids).

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

FlowCorr block diagram (Page 1830)
FlowCorr I/Os (Page 1828)
FlowCorr messaging (Page 1828)
FlowCorr error handling (Page 1827)
FlowCorr functions (Page 1826)
FlowCorr modes (Page 1825)
Forming and outputting the signal status for mathematical blocks (Page 117)

12.8.2 FlowCorr modes

FlowCorr operating modes

This block does not have any operating modes.

See also

FlowCorr block diagram (Page 1830)
FlowCorr I/Os (Page 1828)
FlowCorr messaging (Page 1828)
FlowCorr error handling (Page 1827)
FlowCorr functions (Page 1826)
Description of FlowCorr (Page 1823)

12.8.3 FlowCorr functions

Functions of FlowCorr

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The worst signal status for the block is formed from the following parameters and output at the `Flow.ST` output parameter:

- `DiffP.ST`
- `DiffP0.ST`
- `P.ST`
- `P0.ST`
- `P_Atmos.ST`
- `Temp.ST`
- `Temp0.ST`
- `Density.ST`
- `Density0.ST`
- `Flow0.ST`
- `TempAbs.ST`

When one of the following conditions applies and depending on `Feature Bit` 8 (Switch to substitute value (Page 148)), the last valid value is retained at the `Flow` output or the substitute value `SubsFlow` is applied:

- One of the inputs relevant to the calculation or the `Flow` output is outside the `REAL` counter range
- A division by zero occurs in the calculation
- The Mode parameter is not 1, 2 or 3

`Feature Bit` 8 =0: Last valid value is retained and the signal status is set to bad, process-related `16#28` (assuming the signal status is not `16#0`).

`Feature Bit` 8 =1: Substitute value `SubsFlow` is applied and the signal status is set to manipulated value (for example, substitute value, simulation, last valid value) `16#60`.

See also

FlowCorr block diagram (Page 1830)

FlowCorr I/Os (Page 1828)

FlowCorr messaging (Page 1828)

FlowCorr error handling (Page 1827)

FlowCorr modes (Page 1825)

Description of FlowCorr (Page 1823)

12.8.4 FlowCorr error handling

Error handling of FlowCorr

The block does not report any errors.

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

When one of the following conditions applies and depending on feature bit 8 (Switch to substitute value (Page 148)) at the `Flow` output, the last valid value is retained or the substitute value `SubsFlow` is applied:

- One of the inputs relevant to the calculation or the `Flow` output is outside the `REAL` counter range
- A division by zero occurs in the calculation
- The Mode parameter is not 1, 2 or 3

`Feature Bit` 8 =0: Last valid value is retained and the signal status is set to bad, process-related `16#28` (assuming the signal status is not `16#0`).

`Feature Bit` 8 =1: Substitute value `SubsFlow` is applied and the signal status is set to manipulated value (for example, substitute value, simulation, last valid value) `16#60`.

The `Bad.Value` output is set to `true` when the following applies:

- When the signal status is formed, the value is bad, device-related or process-related (`16#00`, `16#28`)
- One of the inputs relevant to the calculation or the `Flow` output is outside the `REAL` counter range
- A division by zero occurs in the calculation
- The Mode parameter is not 1, 2 or 3

Overview of error numbers

The block has no `ErrorNum` I/O.

See also

FlowCorr block diagram (Page 1830)

FlowCorr I/Os (Page 1828)

FlowCorr messaging (Page 1828)

FlowCorr functions (Page 1826)
FlowCorr modes (Page 1825)
Description of FlowCorr (Page 1823)

12.8.5 FlowCorr messaging

Messaging

This block does not offer messaging.

See also

FlowCorr block diagram (Page 1830)
FlowCorr I/Os (Page 1828)
FlowCorr error handling (Page 1827)
FlowCorr functions (Page 1826)
FlowCorr modes (Page 1825)
Description of FlowCorr (Page 1823)

12.8.6 FlowCorr I/Os

I/Os of FlowCorr

Input parameters

Parameter	Description	Type	Default
`Density`	Density	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Density0`	Design density	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`DiffP`	Differential pressure [bar]	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
`DiffP0`	Design differential pressure [bar]	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80

Parameter	Description	Type	Default
Feature	I/O for additional functions (Page 1826)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Flow0	Design flow	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Mode	Mode 1..3	INT	1
P_Atmos	Barometric pressure [mbar (abs)]	STRUCT • Value: REAL • ST: BYTE	- • 100.0 • 16#80
P_SqrtOn	1 = Sensor calculates the square root	BOOL	0
P	Pressure [bar (abs or relative)]	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
P0	Design pressure [bar (abs or relative)]	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
SubsFlow	Substitute value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Temp	Temperature [°C]	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Temp0	Design temperature [°C]	STRUCT • Value: REAL • ST: BYTE	- • 20.0 • 16#80
TempAbs	Absolute temperature [K] at 0°C (corresponds to 273.15K)	STRUCT • Value: REAL • ST: BYTE	- • 273.15 • 16#80

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Flow	Calculated flow	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

FlowCorr block diagram (Page 1830)
FlowCorr messaging (Page 1828)
FlowCorr error handling (Page 1827)
FlowCorr modes (Page 1825)
Description of FlowCorr (Page 1823)

12.8.7 FlowCorr block diagram

FlowCorr block diagram

A block diagram is not provided for this block.

See also

FlowCorr I/Os (Page 1828)
FlowCorr messaging (Page 1828)
FlowCorr error handling (Page 1827)
FlowCorr functions (Page 1826)
FlowCorr modes (Page 1825)
Description of FlowCorr (Page 1823)

12.9 Integral - Generating a time integral

12.9.1 Description of Integral

Object name (type + number) and family

Type + number: FB 1823

Family: Math

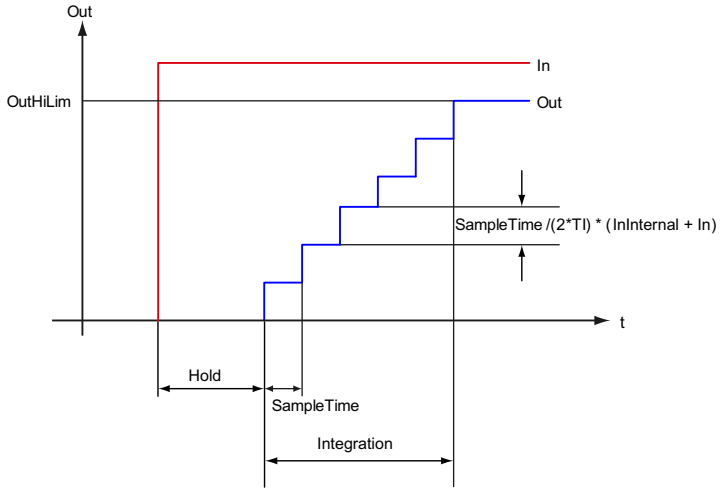
Area of application for Integral

The block is used for the following application:

- Forming the time integral of the connected input signal
- Forming the D component for the controller design

How it works

The block can be used as part of user-designed closed-loop controller. It integrates the input signal `In` based on the trapezoid rule and outputs the result, the integral action component, at `Out`. If the block should calculate the pure time integral, `TI` = 1 [s] must be set.



The block works according to the following formula:

$$Out_{(n)} = Out_{(n-1)} + \frac{SampleTime}{2 \cdot TI} \cdot \left(In_{(n-1)} + In_{(n)} \right)$$

Where:

- `Out` = Integrated value between high and low limits
- `SampleTime` = Sampling time [s]

- `TI` = Integration time constant [s]
- `In` = Input value
- $\texttt{In}_{(n-1)}$ = Last input value
- $\texttt{Out}_{(n-1)}$ = Last output value

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

There is an example project for the Integral block (APL_Example_xx, xx refers to the language variant) with an application scenario for this block, which explains how the block works.

Application scenario in the example project:

- Process simulation including noise generator (ProcSimC; ProcSimS) (Page 2347)

Startup characteristics

Use the feature bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Integral functions (Page 1833)

Integral messaging (Page 1836)

Integral I/Os (Page 1836)

Integral block diagram (Page 1837)

Integral error handling (Page 1835)

Integral modes (Page 1832)

12.9.2 Integral modes

Integral modes

This block does not have any modes.

See also

Integral block diagram (Page 1837)

Integral I/Os (Page 1836)

Integral messaging (Page 1836)

Integral error handling (Page 1835)

Integral functions (Page 1833)

Description of Integral (Page 1831)

12.9.3 Integral functions

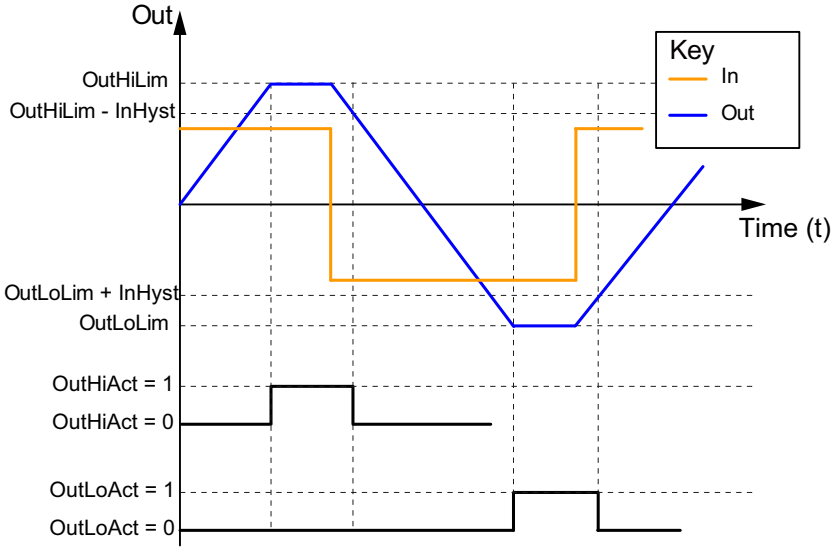
Functions of Integral

The functions for this block are listed below.

Monitoring limits

Use the `OutHiLim` and `OutLoLim` I/Os to define the limits of the integrated value `Out`.

If limits are reached or exceeded (`OutHiAct` or `OutLoAct` = 1 I/Os), the output value is set to user-defined limits and output at `Out`.



Tracking values

You use input parameter `OutTrkOn` = 1 to activate tracking of a value, which is in turn defined at input parameter `OutTrk`.

After tracking mode has been terminated, the block uses the value currently set at `Out` as the value to be integrated.

In tracking mode, the signal status of `Out` is set to the signal status of `OutTrk`.

Note

If the integration is stopped, the function (`Hold` = 1) has priority over the tracking.

Stopping integration

Set input parameter `Hold` = 1 if you want to stop the integration. Limits are no longer monitored and the updating of the `OutHiAct` and `OutLoAct` limit outputs stops. A change to the monitoring limits remains ineffective when integration is stopped. If you start the integration again, the value currently pending at `Out` output parameter is used for the integration process.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for technologic blocks (Page 108).

The signal status for the block is formed using the following parameters and output at the `Out`, `OutHiAct` and `OutLoAct` output parameters:

- `In.ST`
- `Out.ST` Signal status from the last cycle

Note

Special notes for the Integral mathematical block

Due to its application area (time integral, I-component for the configuration of a controller), the Integral block generates the signal status like the technological blocks.

See also

Description of Integral (Page 1831)

Integral messaging (Page 1836)

Integral I/Os (Page 1836)

Integral block diagram (Page 1837)

Integral error handling (Page 1835)

Integral modes (Page 1832)

Forming and outputting the signal status for mathematical blocks (Page 117)

12.9.4 Integral error handling

Error handling of Integral

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
11	The integration time constant `TI` is within the range: -`SampleTime` / 2 < `TI` < `SampleTime` / 2. The integration is stopped.
15	`OutTrk` > `OutHiLim` `OutTrk` < `OutLoLim`
30	The value of `In` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.
32	The value of `OutTrk` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.
33	The value of `TI` can no longer be displayed in the REAL number field. The integration is stopped.

See also

Integral I/Os (Page 1836)

Integral messaging (Page 1836)

Integral functions (Page 1833)

Integral modes (Page 1832)

Description of Integral (Page 1831)

Integral block diagram (Page 1837)

12.9.5 Integral messaging

Messaging

This block does not offer messaging.

See also

Description of Integral (Page 1831)
Integral functions (Page 1833)
Integral I/Os (Page 1836)
Integral block diagram (Page 1837)
Integral error handling (Page 1835)
Integral modes (Page 1832)

12.9.6 Integral I/Os

I/Os of Integral

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`Feature`	I/O for additional functions (Page 1833)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`Hold`	1 = Integration is held	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`In*`	Analog input value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`InHyst*`	Hysteresis	REAL	0.0
`OutHiLim`	High limit of output value	REAL	100.0
`OutLoLim`	Low limit of output value	REAL	0.0
`OutTrkOn`	1 = Output `Out` tracks the set value (`OutTrk` I/O)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
OutTrk*	Predefined value used for OutTrkOn = 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
TI*	Integral time constant [s]	REAL	1.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see Integral error handling (Page 1835)	INT	-1
Out	Integral value output	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
OutHiAct	1 = High limit (OutHiLim) reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutLoAct	1 = Low limit (OutLoLim) reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Description of Integral (Page 1831)
Integral messaging (Page 1836)
Integral block diagram (Page 1837)
Integral modes (Page 1832)

12.9.7 Integral block diagram

Integral block diagram

A block diagram is not provided for this block.

See also

Integral I/Os (Page 1836)
Integral messaging (Page 1836)
Integral functions (Page 1833)
Integral modes (Page 1832)
Description of Integral (Page 1831)
Integral error handling (Page 1835)

12.10 Lag - Low-pass filter

12.10.1 Description of Lag

Object name (type + number) and family

Type + number: FB 1828

Family: Math

Area of application for Lag

The block is used for the following application:

- Smoothing the Input value (low-pass filter)

How it works

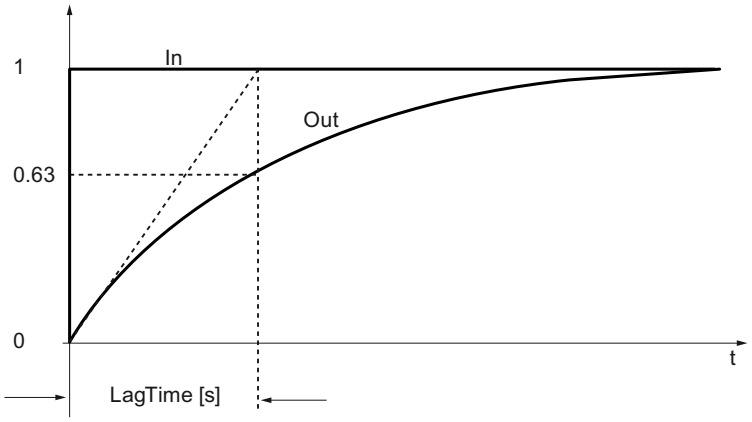
The block smoothes the input variable (`In` input) using a 1. order time delay. This delay time can be configured (`LagTime` connection). The block works according to the following formula:

$$Out_{(n)} = In_{(n)} + (Out_{(n-1)} - In_{(n)}) \cdot e^{\left(\frac{-SampleTime}{LagTime}\right)}$$

Where:

- `Out` = Output value
- `LagTime` = Delay time
- `SampleTime` = Sampling time
- `In` = Input value

The formula only applies to `LagTime` > 0. If `LagTime` = 0, the input is passed directly to the output. If the input value is outside the REAL range limits, the calculation is stopped. If the input value is outside the range limits again, the calculation is resumed automatically.



You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB3x). The block is also installed automatically in the startup OB (OB100).

Further addressing is not required.

For the Lag block, the Advanced Process Library contains templates for process tag types as examples and there is a example project (APL_Example_xx, xx designates the language variant) containing different application cases for this block. Several application cases are simulated in the example project and serve to explain how the block works.

Examples of process tag types:

- PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303)
- PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301)
- Model-based predictive control (ModPreCon) (Page 2325)
- PIDConR with safety logic and control loop monitoring (PIDConR_ConPerMon) (Page 2299)
- PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298)
- Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312)
- Ratio control with PIDConR (RatioR) (Page 2315)
- PID controller with Smith predictor (SmithPredictorControl) (Page 2306)
- Cascade control with control loop monitoring through ConPerMon (CascadeControl) (Page 2316)
- Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl) (Page 2309)

Application scenario in the example project:

- Process simulation including noise generator (ProcSimC; ProcSimS) (Page 2347)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Lag functions (Page 1841)

Lag messaging (Page 1843)

Lag I/Os (Page 1844)

Lag block diagram (Page 1845)

Lag error handling (Page 1842)

Lag modes (Page 1841)

12.10.2 Lag modes

Lag modes

This block does not have any modes.

See also

Lag block diagram (Page 1845)

Lag I/Os (Page 1844)

Lag messaging (Page 1843)

Lag error handling (Page 1842)

Lag functions (Page 1841)

Description of Lag (Page 1839)

12.10.3 Lag functions

Functions of Lag

The functions for this block are listed below.

Hold and restart calculation

You can interrupt the calculation by setting `Hold` = 1 . The output value is frozen. Continue calculation by setting `Hold` = 0 . The calculation is continued with the last output value.

Reset values

You need to set the `Reset` = 1 I/O if you want to reset the output value back to the input value. The output is reset by a rising 0 - 1 edge.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) section. The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The signal status for the block is formed using the following parameters and output at the `Out` output parameter:

- `In.ST`

See also

Description of Lag (Page 1839)

Lag messaging (Page 1843)

Lag I/Os (Page 1844)

Lag block diagram (Page 1845)

Lag error handling (Page 1842)

Lag modes (Page 1841)

12.10.4 Lag error handling

Error handling of Lag

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed
0	There is no error.
11	`LagTime` < 0
30	The value of `In` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.
32	The value of `LagTime` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.

See also

Lag block diagram (Page 1845)

Lag I/Os (Page 1844)

Lag messaging (Page 1843)

Lag functions (Page 1841)

Lag modes (Page 1841)

Description of Lag (Page 1839)

12.10.5 Lag messaging

Messaging

This block does not offer messaging.

See also

Description of Lag (Page 1839)

Lag functions (Page 1841)

Lag I/Os (Page 1844)

Lag block diagram (Page 1845)

Lag modes (Page 1841)

Lag error handling (Page 1842)

12.10.6 Lag I/Os

I/Os of Lag

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`Feature`	I/O for additional functions (Page 1841)	STRUCT	-
		• Bit 0: BOOL	• 0
		• ...	• 0
		• Bit 31: BOOL	• 0
`Hold`	1 = Hold calculation	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
`In*`	Analog input value	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
`LagTime*`	Delay time [s]	REAL	1.0
`Reset`	1 = Reset `Out` output to the value of the `In` input	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
`SampleTime`	Sampling time [s] (assigned automatically)	REAL	0.1

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
`ENO`	1 = Block algorithm completed without errors	BOOL	0
`ErrorNum`	Output of current error number. For error numbers that can be output by this block, see Lag error handling (Page 1842)	INT	-1
`Out`	Delayed value output	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80

See also

Description of Lag (Page 1839)

Lag messaging (Page 1843)

Lag block diagram (Page 1845)
Lag modes (Page 1841)

12.10.7 Lag block diagram

Lag block diagram

A block diagram is not provided for this block.

See also

Lag I/Os (Page 1844)
Lag messaging (Page 1843)
Lag error handling (Page 1842)
Lag functions (Page 1841)
Lag modes (Page 1841)
Description of Lag (Page 1839)

12.11 MeanTime - Averaging

12.11.1 Description of MeanTime

Object name (type + number) and family

Type + number: FB 1832

Family: Math

Area of application for MeanTime

The block is used for the following applications:

- Averaging an analog value over a previous, definable period

How it works

The MeanTime block is used to calculate the time-based mean value of an analog input signal `In` over a previous, definable period (`TimeWindow` input), according to the formula:

`Out` = (`In1` + ... + `Inn`) / (`TimeWindow` / `SampleTime`)

Where:

- `In1` ... `Inn` the n are the detected values used for averaging.
- The time window for the averaging is set at the `TimeWindow` parameter.
- The block determines the number of values to be saved based on the integer part of the `TimeWindow` / `SampleTime` quotient.
- The block can save up to 32 previous values in its internal memory. By interconnecting the `Mem` input parameter with the `Mem` output parameter of the MemR256 block, the memory size can be expanded to 256 previous values. If the time window is longer than memory spaces are available, an internal message is triggered in addition.

If `SampleTime` or `TimeWindow` is changed, the mean time value is reset.

The signal status is passed from the input parameter directly to the output parameter.

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (all OB3x blocks). The block is also installed automatically in the startup OB (OB 100). When using the MemR256 block, the `Mem` output parameter of the MemR256 block must be interconnected with the `Mem` input parameter of the MeanTime block. The MemR256 block can be inserted in any cyclic interrupt level.

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

MeanTime functions (Page 1847)

MeanTime messaging (Page 1850)

MeanTime I/Os (Page 1851)

MeanTime block diagram (Page 1852)

MeanTime error handling (Page 1850)

MeanTime modes (Page 1847)

12.11.2 MeanTime modes

MeanTime operating modes

This block does not have any operating modes.

See also

MeanTime block diagram (Page 1852)

MeanTime I/Os (Page 1851)

MeanTime messaging (Page 1850)

MeanTime error handling (Page 1850)

MeanTime functions (Page 1847)

Description of MeanTime (Page 1846)

12.11.3 MeanTime functions

Functions of MeanTime

The functions for this block are listed below.

Stopping the calculation of mean values

Mean value calculation can be stopped by setting the `Hold` input. To do this, make the following parameter settings:

- `Hold` = 1 and calculation is stopped. The output value is retained for the duration of this period.
- `Hold` = 0 and calculation is resumed

If the hold functionality is active, the output `HoldAct` will be set by the block.

With `Feature` bit 5 Retain last output value in case of bad input signal status (Page 183), you can specify that the calculation of mean value is stopped and the last output value is retained if the signal status of `In` is bad (`In.ST` = 16#00 or 16#28). In this case, the block sets both the outputs `HoldAct` and `Inbad`.

Setting a mean value constant

You can set the mean value using the `Reset` input. To do this, make the following parameter setting:

- `Reset` = 0 -> 1

The value at the `In` input is now set directly at the `Out` output. All internal values of the block are also adapted to the input value.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
5	Retain last output value in case of bad input signal status (Page 183)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The signal status for the block is formed using the following parameters and output at the `Out` output parameter:

- `In.ST`

Increasing the internal memory to 256 previous values

MeanTime can save up to 32 previous values internally. If the ratio of time window and sampling time gets too large, the number of internal memory slots can be expanded to 256 previous values by using the MemR256 block. To do so insert the MemR256 block into a CFC chart. Interconnect the `Mem` output parameter of the MemR256 block with the `Mem` input parameter of the MeanTime block. Each MeanTime block must be interconnected with its own separate MemR256 block. An internal averaging is automatically activated if the time window is still too large. You detect whether internal averaging is active at the `ExcessCnt` output parameter:

- `ExcessCnt` = 0: no internal averaging active.
- `ExcessCnt > 0`: internal averaging active.
  Without MemR256: ExcessCnt = TimeWindow / SampleTime Div 32 + 1
  Where MemR256: ExcessCnt = TimeWindow / SampleTime Div 256 + 1

When internal averaging is activated, the `In.Value` input is summed using ExcessCnt values (SumIntern) and after ExcessCnt cycles the internal average is stored as a past value at a memory location (1..UsedBuffer). The current value of the `Out` output is then calculated as follows:

- Out:=(ExcessCnt * Sum + SumIntern) / (UsedBuffer * ExcessCnt + Index)
- UsedBuffer: TimeWindow / SampleTime Div ExcessCnt
- Sum: Sum of all storage locations (UsedBuffer)
- SumIntern: Sum of the internal average value from 1 to index
- Index: Running index from 1 to ExcessCnt

Note

The ratio of the time window and sampling time is limited internally to 25600. If the time window is set too large, an error number is output at the `ErrorNum` output parameter.

See also

Description of MeanTime (Page 1846)

MeanTime messaging (Page 1850)

MeanTime I/Os (Page 1851)

MeanTime block diagram (Page 1852)

MeanTime error handling (Page 1850)

MeanTime modes (Page 1847)

12.11.4 MeanTime error handling

Error handling of MeanTime

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.
30	The value of `In` can no longer be displayed in the REAL number field. The last valid value is provided at the `Out` output.
31	The value of `Out` can no longer be displayed in the REAL number field. The `In` value is output unmodified at the `Out` output.
50	The quotient TimeWindow / SampleTime is greater than 25600. The block calculates with a smaller time window, so that the quotient TimeWindow / SampleTime does not exceed the value 25600.

See also

MeanTime block diagram (Page 1852)

MeanTime I/Os (Page 1851)

MeanTime messaging (Page 1850)

Description of MeanTime (Page 1846)

MeanTime modes (Page 1847)

MeanTime functions (Page 1847)

12.11.5 MeanTime messaging

Messaging

This block does not offer messaging.

See also

Description of MeanTime (Page 1846)

MeanTime functions (Page 1847)

MeanTime I/Os (Page 1851)

MeanTime block diagram (Page 1852)

MeanTime modes (Page 1847)

MeanTime error handling (Page 1850)

12.11.6 MeanTime I/Os

I/Os of MeanTime

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1847)	STRUCT	-
		• Bit 0: BOOL	• 0
		• ...	• 0
		• Bit 31: BOOL	• 0
Hold	1 = Hold calculation of mean value	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
In*	Analog input value	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
Mem	Input for Mem Block	ANY	-
Reset	1 = Reset output Out	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
TimeWindow	Size of the time window [s]	REAL	32.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see MeanTime error handling (Page 1850)	INT	-1
ExcessCnt	Counter for internal averaging; 0: no internal averaging. All values of In can be saved as previous values with the current TimeWindow. x: internal averaging of In across x cycles. The internal averaging is saved as previous value.	INT	0
HoldAct	The block is on hold.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
InBad	1 = Signal status of the analog input In is bad (16#00 or 16#28)	BOOL	0
Out	Output for the average time	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of MeanTime (Page 1846)

MeanTime messaging (Page 1850)

MeanTime block diagram (Page 1852)

MeanTime modes (Page 1847)

12.11.7 MeanTime block diagram

MeanTime block diagram

This block is not provided a block diagram.

See also

Description of MeanTime (Page 1846)

MeanTime modes (Page 1847)

MeanTime functions (Page 1847)

MeanTime error handling (Page 1850)

MeanTime messaging (Page 1850)

MeanTime I/Os (Page 1851)

12.12 Mul04 - Multiplier with 4 values

12.12.1 Description of Mul04

Object name (type + number) and family

Type + number: FC 360

Family: Math

Area of application for Mul04

The block is used for the following applications:

- Multiplication of values
- Output of the product for further processing

How it works

The Mul04 block calculates the product of up to 4 values and returns the result at the `Out` output.

`Out` = `In1` ·...· `Inn` (`n` ≤ 4),

Where:

`Out` = Product

`In1` ... `In4` = values to be multiplied

The product of all input parameters and the worst input parameter signal status is always output.

The output value is checked to determine if it is within the value range of REAL. If it is outside the value range, the highest or lowest possible REAL value is output.

If the value is NAN, the last valid output value is output and the status of the output value is set to 16#28 (if none of the input statuses is worse).

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

For the Mul04 block, the Advanced Process Library contains templates for process tag types as examples and there is a example project (APL_Example_xx, xx designates the language variant) containing an application case for this block. An application case is simulated in the example project and serves to explain how the block works.

Examples of process tag types:

- PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303)
- PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301)
- Model-based predictive control (ModPreCon) (Page 2325)
- PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298)
- PIDConR with safety logic and control loop monitoring (PIDConR_ConPerMon) (Page 2299)
- Ratio control with PIDConR (RatioR) (Page 2315)
- Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312)
- PID controller with Smith predictor (SmithPredictorControl) (Page 2306)
- Cascade control with control loop monitoring through ConPerMon (CascadeControl) (Page 2316)
- Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl) (Page 2309)

Application scenario in the example project:

- Process simulation including noise generator (ProcSimC; ProcSimS) (Page 2347)

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

Mul04 functions (Page 1855)

Mul04 messaging (Page 1856)

Mul04 I/Os (Page 1857)

Mul04 block diagram (Page 1858)

Mul04 error handling (Page 1856)

Mul04 modes (Page 1855)

12.12.2 Mul04 modes

Mul04 operating modes

This block does not have any modes.

See also

Mul04 block diagram (Page 1858)

Mul04 I/Os (Page 1857)

Mul04 messaging (Page 1856)

Mul04 error handling (Page 1856)

Mul04 functions (Page 1855)

Description of Mul04 (Page 1853)

12.12.3 Mul04 functions

Functions of Mul04

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The worst signal status for the block is formed by the following parameters and output at the `Out` output parameter:

- `In1.ST`
- `In2.ST`
- `In3.ST`
- `In4.ST`

See also

Description of Mul04 (Page 1853)

Mul04 messaging (Page 1856)

Mul04 I/Os (Page 1857)

Mul04 error handling (Page 1856)

Mul04 block diagram (Page 1858)

Mul04 modes (Page 1855)

12.12.4 Mul04 error handling

Mul04 error handling

The block does not report any errors.

See also

Mul04 functions (Page 1855)
Mul04 block diagram (Page 1858)
Mul04 I/Os (Page 1857)
Mul04 messaging (Page 1856)
Description of Mul04 (Page 1853)
Mul04 modes (Page 1855)

12.12.5 Mul04 messaging

Messaging

This block does not offer messaging.

See also

Description of Mul04 (Page 1853)
Mul04 functions (Page 1855)
Mul04 I/Os (Page 1857)
Mul04 block diagram (Page 1858)
Mul04 modes (Page 1855)
Mul04 error handling (Page 1856)

12.12.6 Mul04 I/Os

Mul04 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Value 1 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
In2	Value 2 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
In3	Value 3 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
In4	Value 4 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Product version	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of Mul04 (Page 1853)
Mul04 functions (Page 1855)
Mul04 messaging (Page 1856)
Mul04 block diagram (Page 1858)
Mul04 modes (Page 1855)
Mul04 error handling (Page 1856)

12.12.7 Mul04 block diagram

Mul04 block diagram

A block diagram is not provided for this block.

See also

Description of Mul04 (Page 1853)
Mul04 modes (Page 1855)
Mul04 functions (Page 1855)
Mul04 error handling (Page 1856)
Mul04 messaging (Page 1856)
Mul04 I/Os (Page 1857)

12.13 Mul08 - Multiplier with 8 values

12.13.1 Description of Mul08

Object name (type + number) and family

Type + number: FC 361

Family: Math

Area of application for Mul08

The block is used for the following applications:

- Multiplication of values
- Output of the product for further processing

How it works

The Mul08 block calculates the product of up to 8 values and returns the result at the `Out` output.

`Out` = `In1` ·...· `Inn` (`n` ≤ 8),

Where:

`Out` = Product

`In1` ... `In8` = values to be multiplied

The product of all input parameters and the worst input parameter signal status is always output.

The output value is checked to determine if it is within the value range of REAL. If it is outside the value range, the highest or lowest possible REAL value is output.

If the value is NAN, the last valid output value is output and the status of the output value is set to 16#28 (if none of the input statuses is worse).

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

For the Mul04 (Mul08) block, the Advanced Process Library contains process tag type templates; these serve as examples by providing various application scenarios for this block.

Refer to Description of Mul04 (Page 1853) for more information.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

Mul08 functions (Page 1860)
Mul08 messaging (Page 1862)
Mul08 I/Os (Page 1862)
Mul08 block diagram (Page 1863)
Mul08 error handling (Page 1861)
Mul08 modes (Page 1860)

12.13.2 Mul08 modes

Mul08 operating modes

This block does not have any modes.

See also

Mul08 block diagram (Page 1863)
Mul08 I/Os (Page 1862)
Mul08 messaging (Page 1862)
Mul08 error handling (Page 1861)
Description of Mul08 (Page 1859)
Mul08 functions (Page 1860)

12.13.3 Mul08 functions

Functions of Mul08

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The worst signal status for the block is formed by the following parameters and output at the `Out` output parameter:

- `In1.ST`
- `In2.ST`
- `In3.ST`
- `In4.ST`
- `In5.ST`
- `In6.ST`
- `In7.ST`
- `In8.ST`

See also

Description of Mul08 (Page 1859)
Mul08 messaging (Page 1862)
Mul08 I/Os (Page 1862)
Mul08 block diagram (Page 1863)
Mul08 error handling (Page 1861)
Mul08 modes (Page 1860)

12.13.4 Mul08 error handling

Mul08 error handling

The block does not report any errors.

See also

Mul08 block diagram (Page 1863)
Mul08 I/Os (Page 1862)
Mul08 messaging (Page 1862)
Mul08 modes (Page 1860)
Mul08 functions (Page 1860)
Description of Mul08 (Page 1859)

12.13.5 Mul08 messaging

Messaging

This block does not offer messaging.

See also

Description of Mul08 (Page 1859)
Mul08 functions (Page 1860)
Mul08 I/Os (Page 1862)
Mul08 block diagram (Page 1863)
Mul08 modes (Page 1860)
Mul08 error handling (Page 1861)

12.13.6 Mul08 I/Os

Mul08 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Value 1 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
In2	Value 2 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
In3	Value 3 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
In4	Value 4 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
In5	Value 5 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80

Parameter	Description	Type	Default
In6	Value 6 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
In7	Value 7 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80
In8	Value 8 to multiply	STRUCT • Value: REAL • ST: BYTE	- • 1.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Product output	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of Mul08 (Page 1859)
Mul08 functions (Page 1860)
Mul08 messaging (Page 1862)
Mul08 block diagram (Page 1863)
Mul08 modes (Page 1860)
Mul08 error handling (Page 1861)

12.13.7 Mul08 block diagram

Mul08 block diagram

A block diagram is not provided for this block.

See also

Mul08 I/Os (Page 1862)
Mul08 messaging (Page 1862)
Mul08 error handling (Page 1861)
Mul08 functions (Page 1860)

Mul08 modes (Page 1860)
Description of Mul08 (Page 1859)

12.14 Polygon - Converting the first signal (non-linear)

12.14.1 Description of Polygon

Object name (type + number) and family

Type + number: FB 1881

Family: Math

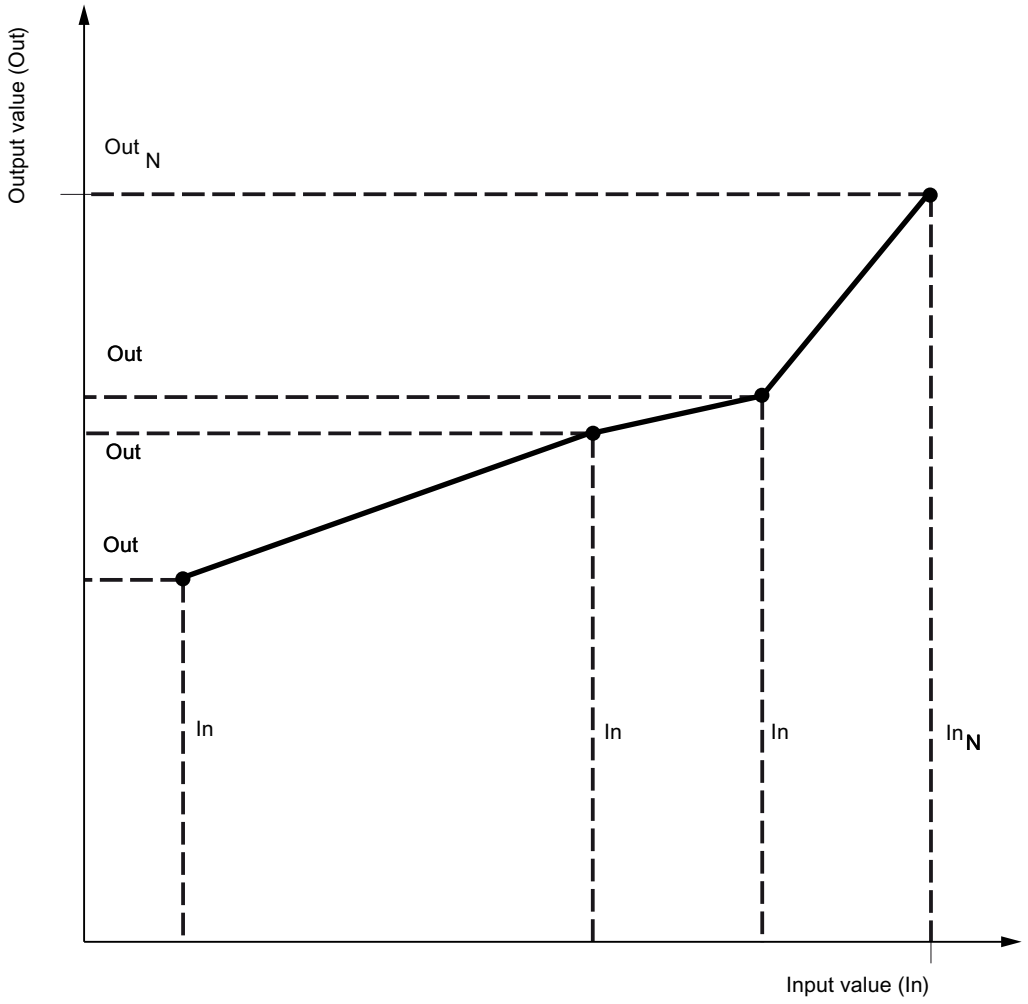
Area of application for Polygon

The block is used for the following applications:

- Conversion of the input signal according to a non-linear characteristic

How it works

An input `In` is converted to output `Out` based on a non-linear characteristic with up to 16 interpolation points per polygon block. The number of interpolation points can be increased by cascading multiple blocks. In cascade mode, the `Cascaded = 1` output is set.



Num (N)	- Number of intermediate points in the curve Minimum number of points = 2 Maximum number of points = 16
In	- Analog input value
Out	- Corresponding output value

After you have defined the N interpolation points (coordinate pairs In_i, Out_i with i = 1 ... N in a sequence with no gaps) and configured the number Num, the block operates as follows:

- Interpolation between the interpolation points is linear
- Beyond the end interpolation points, extrapolation is based on the first two or last two interpolation points.

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

The block is installed in the CFC editor in a cyclic interrupt OB (OB30 to OB38). The block is installed automatically in OB100.

Cascade mode

For the cascading of multiple polygon blocks, the `CasOut` output is interconnected to the `CasIn` input of the following block.

In cascade mode, the interpolation points in all polygon blocks must be configured in ascending order and without gaps. The cascaded polygon blocks must be configured in the same runtime group in ascending run sequence.

The analog input value `In` may only be interconnected at the first polygon block. The `In` input of the next block is tracked to this value.

For the Polygon block, the Advanced Process Library contains templates for process tag types as an example with various application scenarios for this block.

Example of process tag types:

- Source chart for GainSched function block (gain scheduling) (Page 2321)

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Polygon functions (Page 1868)

Polygon messaging (Page 1870)

Polygon I/Os (Page 1871)

Polygon block diagram (Page 1874)

Polygon error handling (Page 1869)

Polygon modes (Page 1868)

12.14.2 Polygon modes

Polygon modes

This block does not have any modes.

See also

Polygon block diagram (Page 1874)

Polygon I/Os (Page 1871)

Polygon messaging (Page 1870)

Polygon error handling (Page 1869)

Polygon functions (Page 1868)

Description of Polygon (Page 1865)

12.14.3 Polygon functions

Functions of Polygon

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The signal status for the block is formed using the following parameters and output at the `Out` output parameter:

- `In.ST`

Configurable reactions using the `Feature` parameter

An overview of all the reactions that are provided by the parameter Feature is available in the section Configurable functions with the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
5	Limit output Out (Page 143)
6	Substitute value switch in the event of an error (Page 148)

Cascading

- Specifying the number of interpolation points
  When n polygon blocks are cascaded, the interpolation points can be increased by the number set for the Num parameter. The configuration of the interpolation points over all cascaded blocks is subject to the same configuration rules as for an individual block, i.e. the interpolation points must be configured in sequence in ascending order and without gaps.
- Data communication
  The data communication 2 interconnected polygon blocks is made using an output -> input interconnection.

See also

Description of Polygon (Page 1865)

Polygon messaging (Page 1870)

Polygon I/Os (Page 1871)

Polygon block diagram (Page 1874)

Polygon error handling (Page 1869)

Polygon modes (Page 1868)

12.14.4 Polygon error handling

Error handling of Polygon

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The ErrorNum I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed
0	There is no error.
30	The value In or SubV_In can no longer be displayed in the REAL- number field. The last valid value is provided at the Out output.
31	The calculation of the output Out leads to a result that can no longer be displayed in a REAL- numerical field. The last valid value is provided at the Out output.
48	No valid interconnection to CasIn. The last valid value is provided at the Out output.

Error number	Meaning of the error number
49	`Num` < 2 or `Num` > 16 $In_{(N)} \leq In_{(N-1)}$, in the event of an error, the last valid value is output
≥ 1xx	Error in one of the upstream `xx = ErrorNum` block. The last valid value is provided at the `Out` output.

Note

The substitute value `SubV_In` can be output instead of the last valid value via the `Feature` bit 6 = 1 .

Substitute values that cannot be displayed in the REAL- number format have no effect. The last valid value is provided at the `Out` output in this case.

See also

Polygon block diagram (Page 1874)

Polygon I/Os (Page 1871)

Polygon messaging (Page 1870)

Description of Polygon (Page 1865)

Polygon modes (Page 1868)

Polygon functions (Page 1868)

12.14.5 Polygon messaging

Messaging

This block does not offer messaging.

See also

Description of Polygon (Page 1865)

Polygon functions (Page 1868)

Polygon I/Os (Page 1871)

Polygon block diagram (Page 1874)

Polygon error handling (Page 1869)

Polygon modes (Page 1868)

12.14.6 Polygon I/Os

I/Os of Polygon

Input parameters

Parameter	Description	Type	Default
CasIn*	Input for cascade, to be interconnected with the output parameter CasOut of the preceding polygon block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1868)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
In*	Analog input value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In1	Interpolation point In1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In2	Interpolation point In2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In3	Interpolation point In3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In4	Interpolation point In4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In5	Interpolation point In5	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In6	Interpolation point In6	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In7	Interpolation point In7	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
`In8`	Interpolation point `In8`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`In9`	Interpolation point `In9`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`In10`	Interpolation point `In10`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`In11`	Interpolation point `In11`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`In12`	Interpolation point `In12`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`In13`	Interpolation point `In13`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`In14`	Interpolation point `In14`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`In15`	Interpolation point `In15`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`In16`	Interpolation point `In16`	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Num`	Number of interpolation points	INT	16
`Out1`	Sampling point 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Out2`	Sampling point 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Out3`	Sampling point 3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`Out4`	Sampling point 4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
Out5	Sampling point 5	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out6	Sampling point 6	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out7	Sampling point 7	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out8	Sampling point 8	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out9	Sampling point 9	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out10	Sampling point 10	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out11	Sampling point 11	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out12	Sampling point 12	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out13	Sampling point 13	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out14	Sampling point 14	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out15	Sampling point 15	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Out16	Sampling point 16	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SubV_In*	Substitute value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
`Cascaded`	1 = Block is cascaded	BOOL	0
`CasOut`	Output parameter for cascade formation, to be interconnected with the input parameter `CasIn` of the following polygon block	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`ENO`	1 = Block algorithm completed without errors	BOOL	0
`ErrorNum`	Output of current error number. For error numbers that can be output by this block, see Polygon error handling (Page 1869)	INT	-1
`Out`	Output value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of Polygon (Page 1865)

Polygon messaging (Page 1870)

Polygon block diagram (Page 1874)

Polygon modes (Page 1868)

12.14.7 Polygon block diagram

Polygon block diagram

A block diagram is not provided for this block.

See also

Polygon I/Os (Page 1871)

Polygon messaging (Page 1870)

Polygon error handling (Page 1869)

Polygon functions (Page 1868)

Polygon modes (Page 1868)

Description of Polygon (Page 1865)

12.15 Smooth - Low pass filter

12.15.1 Description of Smooth

Object name (type + number) and family

Type + number: FB 1890

Family: Math

Area of application for Smooth

The block is used for the following applications:

- Butterworth low-pass filter, 2nd order with maverick detection

How it works

The block is used as a low pass filter. This filter allows the signal portions with frequencies below the cutoff frequency to pass practically unattenuated, whereas portions with high frequencies are attenuated. The enables you filter out high frequency interference in the signal (for example, signal noise) and smooth the signal.

In comparison to a first order low pass filter, the Butterworth filter has the advantage that the transition from the passing section to the blocking section is sharper in the Bode diagram. If the frequency area of the interference is known, it can be filtered out with minimal influence on the wanted signal.

The maverick (Page 2361) detection monitors adjacent signals. If signal mavericks are detected, they are not processed any further. The block outputs the last valid signal.

The signal status is passed from the input directly to the output.

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

You then set the filter time constant for the low pass filter to achieve the desired effect.

To set the filter time constant, it is helpful to observe the original and filtered signals in the CFC trend plotter. The filtered signal should be smoothed as required but not too delayed. An increase in the filter time constant increases the smoothing effect but also increases the delay. Typical starting values for the time constant are around ten times the sample rate of the signal.

There is a sample project for the Smooth block (APL_Example_xx, xx refers to the language variant) with an application case for this block:

- Filtering of noisy measured values in a control loop (SigSmoothSim) (Page 2354)

Startup characteristics

When OB100 is called, the state variables of the Butterworth filter are initialized based on the current process values.

Status word allocation for `Status1` parameter

The block does not have a status word allocation.

See also

Smooth functions (Page 1876)
Smooth messaging (Page 1878)
Smooth I/Os (Page 1879)
Smooth block diagram (Page 1880)
Smooth error handling (Page 1877)
Smooth modes (Page 1876)

12.15.2 Smooth modes

Smooth operating modes

This block does not have any modes.

See also

Smooth block diagram (Page 1880)
Smooth I/Os (Page 1879)
Smooth messaging (Page 1878)
Smooth error handling (Page 1877)
Smooth functions (Page 1876)
Description of Smooth (Page 1875)

12.15.3 Smooth functions

Smooth functions

The block provides the following functions:

- Restart low pass filter
- Activate and deactivate maverick detection

Restart low pass filter

You can recalculated the coefficients of the Butterworth filter. To do this, you must restart filter (`Restart` = 1).

The filter algorithm is then reinitialized, exactly as it is when the CPU is restarted or a change is made to the count value at the input parameter `TimeConstant`. The coefficients of the Butterworth filter are recalculated and the internal state memory of the filter is initialized so that the `CleanPV` output parameter is equal to the `PV` input parameter.

Activate and deactivate maverick detection

The maverick detection (`OutlDetOn` = 1) monitors the process value `PV` for the difference between two sequential sample values. These have to be within a tolerance range you have specified (`OutlTreshold`).

If the tolerance is violated, the maverick is set to the most recent valid value. The most recent valid measured value is then held constant by the block for a maximum of `OutlCycles` sample steps. If the maverick continues longer than this, it is accepted as a valid measured value.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The signal status for the block is formed using the following parameters and output at the `CleanPV` output parameter:

- `PV.ST`

See also

Description of Smooth (Page 1875)

Smooth messaging (Page 1878)

Smooth I/Os (Page 1879)

Smooth block diagram (Page 1880)

Smooth error handling (Page 1877)

Smooth modes (Page 1876)

12.15.4 Smooth error handling

Error handling of Smooth

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
2	`SampleTime` < 0.001 [s]
30	The value of `PV` can no longer be displayed in the REAL number field. The last valid value is provided at the `CleanPV` output.
61	`TimeConstant` < 5 · `SampleTime`

See also

Smooth block diagram (Page 1880)
Smooth I/Os (Page 1879)
Smooth messaging (Page 1878)
Smooth functions (Page 1876)
Smooth modes (Page 1876)
Description of Smooth (Page 1875)

12.15.5 Smooth messaging

Messaging

This block does not offer messaging.

See also

Description of Smooth (Page 1875)
Smooth functions (Page 1876)
Smooth I/Os (Page 1879)
Smooth block diagram (Page 1880)
Smooth error handling (Page 1877)
Smooth modes (Page 1876)

12.15.6 Smooth I/Os

I/Os of Smooth

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	This parameter currently has no use and is reserved for future functions.	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
FilterOn	1 = Filter active	BOOL	1
OutlCycles	Number of sampling steps to bridge mavericks	INT	3
OutlDetOn	1 = Maverick detection is activated	BOOL	0
OutlThreshold	Limit (trigger threshold) for maverick detection	REAL	10.0
PV*	Analog input (process value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Restart*	Restart the filter algorithm	BOOL	1
SampleTime	Sampling time [s] (assigned automatically)	REAL	1.0
TimeConstant	Butterworth filter time constant [s]	REAL	10.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
CleanPV	Output of the corrected process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum*	Output of current error number. For error numbers that can be output by this block, see Smooth error handling (Page 1877).	INT	0
OutlDetected	1 = Maverick detected	BOOL	0

* Values can be written back to these inputs during processing of the block by the block algorithm.

See also

Description of Smooth (Page 1875)
Smooth functions (Page 1876)
Smooth messaging (Page 1878)
Smooth block diagram (Page 1880)
Smooth modes (Page 1876)

12.15.7 Smooth block diagram

Smooth block diagram

A block diagram is not provided for this block.

See also

Smooth I/Os (Page 1879)
Smooth messaging (Page 1878)
Smooth error handling (Page 1877)
Smooth functions (Page 1876)
Smooth modes (Page 1876)
Description of Smooth (Page 1875)

12.16 SqrRoot - Derive the root of a value

12.16.1 Description of SqrRoot

Object name (type + number) and family

Type + number: FC 394

Family: Math

Area of application for SqrRoot

The block is used to derive the root from a value as follows:

- `Out.Value = SquareRoot(In.Value)`

How it works

The input value is checked as to whether it is within the REAL value range and is not negative.

- If the input value is not a number (NaN), the last valid output value is displayed at the output. If the input status is not 16#00, the status of the output value is set to 16#28 and `ErrorNum` is set to 30
- If the input value is negative, the last valid output value is displayed at the output. If the input status is not 16#00, the status of the output value is set to 16#28 and `ErrorNum` is set to 30
- If the input value is positive infinity, the the greatest REAL value is displayed at the output. If the input status is not 16#00, the status of the output value is set to 16#28 and `ErrorNum` is set to 30
-
- The status of the output value is set to 16#28 (if the input status is not below this). ErrorNum is also set to 30.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

SqrRoot block diagram (Page 1885)
SqrRoot I/Os (Page 1884)
SqrRoot messaging (Page 1883)
SqrRoot functions (Page 1882)
SqrRoot modes (Page 1882)
Source chart for GainSched function block (gain scheduling) (Page 2321)
PID controller with Smith predictor (SmithPredictorControl) (Page 2306)
Override control (Page 2322)
Forming and outputting the signal status for mathematical blocks (Page 117)

12.16.2 SqrRoot modes

Operating modes of SqrRoot

This block does not have any operating modes.

See also

SqrRoot block diagram (Page 1885)
SqrRoot I/Os (Page 1884)
SqrRoot messaging (Page 1883)
SqrRoot functions (Page 1882)
Description of SqrRoot (Page 1881)

12.16.3 SqrRoot functions

Functions of SqrRoot

The functions for this block are listed below.

Forming the signal status for blocks

The signal status of the block is formed using the following parameters and output at the Out output parameter:

- `Out.ST = In.ST`

See also

SqrRoot block diagram (Page 1885)
SqrRoot I/Os (Page 1884)
SqrRoot messaging (Page 1883)
SqrRoot modes (Page 1882)
Description of SqrRoot (Page 1881)
Forming and outputting the signal status for mathematical blocks (Page 117)

12.16.4 SqrRoot error handling

Error handling of SqrRoot

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed.
0	There is no error.
30	The input value is a negative number or it cannot be displayed in the REAL number field.

See also

Sub02 block diagram (Page 1890)
Sub02 I/Os (Page 1889)
Sub02 messaging (Page 1888)
Sub02 functions (Page 1887)
Sub02 modes (Page 1887)
Description of Sub02 (Page 1886)

12.16.5 SqrRoot messaging

Messaging

This block does not offer messaging.

See also

SqrRoot block diagram (Page 1885)
SqrRoot I/Os (Page 1884)

SqrRoot functions (Page 1882)

SqrRoot modes (Page 1882)

Description of SqrRoot (Page 1881)

12.16.6 SqrRoot I/Os

I/Os of SqrRoot

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
IN	Signal status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of the pending error number. For error numbers that can be output by this block, see SqrRoot error handling (Page 1883).	INT	• -1
Out	Signal status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

SqrRoot block diagram (Page 1885)

SqrRoot messaging (Page 1883)

SqrRoot functions (Page 1882)

SqrRoot modes (Page 1882)

Description of SqrRoot (Page 1881)

12.16.7 SqrRoot block diagram

SqrRoot block diagram

A block diagram is not provided for this block.

See also

SqrRoot I/Os (Page 1884)

SqrRoot messaging (Page 1883)

SqrRoot functions (Page 1882)

SqrRoot modes (Page 1882)

Description of SqrRoot (Page 1881)

12.17 Sub02 - Subtracting two values

12.17.1 Description of Sub02

Object name (type + number) and family

Type + number: FC 381

Family: Math

Area of application for Sub02

The block is used for the following applications:

- Subtracting two values

How it works

The block subtracts one value from another and outputs the subtraction at the `Out` output parameter as follows:

- `Out` = `In1` - `In2`

The worst signal status is also always output at the output parameter.

The output value is checked to determine if it is within the value range of REAL. If it is outside the value range, the highest or lowest possible REAL value is output.

If the value is NAN, the last valid output value is output and the status of the output value is set to 16#28 (if none of the input statuses is worse).

You can find additional information on forming the signal status under Forming and outputting the signal status for mathematical blocks (Page 117)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

For the Sub02 block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Source chart for GainSched function block (gain scheduling) (Page 2321)
- PID controller with Smith predictor (SmithPredictorControl) (Page 2306)
- Override control (Page 2322)

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

Sub02 block diagram (Page 1890)
Sub02 I/Os (Page 1889)
Sub02 messaging (Page 1888)
Sub02 error handling (Page 1888)
Sub02 functions (Page 1887)
Sub02 modes (Page 1887)

12.17.2 Sub02 modes

Sub02 operating modes

This block does not have any modes.

See also

Sub02 block diagram (Page 1890)
Sub02 I/Os (Page 1889)
Sub02 messaging (Page 1888)
Sub02 error handling (Page 1888)
Sub02 functions (Page 1887)
Description of Sub02 (Page 1886)

12.17.3 Sub02 functions

Functions of Sub02

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for mathematical blocks (Page 117).

The worst signal status for the block is formed by the following parameters and output at the `Out` output parameter:

- `In1.ST`
- `In2.ST`

See also

Sub02 block diagram (Page 1890)
Sub02 I/Os (Page 1889)
Sub02 messaging (Page 1888)
Sub02 error handling (Page 1888)
Sub02 modes (Page 1887)
Description of Sub02 (Page 1886)

12.17.4 Sub02 error handling

Sub02 error handling

The block does not report any errors.

See also

Sub02 block diagram (Page 1890)
Sub02 I/Os (Page 1889)
Sub02 messaging (Page 1888)
Sub02 functions (Page 1887)
Sub02 modes (Page 1887)
Description of Sub02 (Page 1886)

12.17.5 Sub02 messaging

Messaging

This block does not offer messaging.

See also

Sub02 block diagram (Page 1890)
Sub02 I/Os (Page 1889)

Sub02 error handling (Page 1888)
Sub02 functions (Page 1887)
Sub02 modes (Page 1887)
Description of Sub02 (Page 1886)

12.17.6 Sub02 I/Os

Sub02 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Value 1 to subtract	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In2	Value 2 to subtract	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Product version	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Sub02 block diagram (Page 1890)
Sub02 messaging (Page 1888)
Sub02 error handling (Page 1888)
Sub02 functions (Page 1887)
Sub02 modes (Page 1887)
Description of Sub02 (Page 1886)

12.17.7 Sub02 block diagram

Sub02 block diagram

A block diagram is not provided for this block.

See also

Sub02 I/Os (Page 1889)
Sub02 messaging (Page 1888)
Sub02 error handling (Page 1888)
Sub02 functions (Page 1887)
Sub02 modes (Page 1887)
Description of Sub02 (Page 1886)

Analog logic blocks

13

13.1 CompAn02 - Comparison of two analog values

13.1.1 Description of CompAn02

Object name (type + number) and family

Type + number: FC 387

Family: LogicAn

Area of application for CompAn02

The block is used for the following applications:

- Comparison of the analog input values `In1` and `In2`

How it works

The block compares the two analog input values `In1` and `In2` to see if they are "less than", "less than or equal to", "greater than", "greater than or equal to" and "equal".

There is an independent result output parameter for every comparison operation.

These outputs are formed as follows.

- `GT.Value = 1`, if `In1` > `In2`
- `GE.Value = 1`, if `In1` ≥ `In2`
- `EQ.Value = 1`, if `In1` = `In2`
- `LT.Value = 1`, if `In1` < `In2`
- `LE.Value = 1`, if `In1` ≤ `In2`

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Further addressing is not required.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

CompAn02 modes (Page 1892)
CompAn02 functions (Page 1892)
CompAn02 error handling (Page 1893)
CompAn02 messaging (Page 1894)
CompAn02 I/Os (Page 1894)
CompAn02 block diagram (Page 1895)

13.1.2 CompAn02 modes

CompAn02 operating modes

This block does not have any modes.

See also

Description of CompAn02 (Page 1891)
CompAn02 functions (Page 1892)
CompAn02 messaging (Page 1894)
CompAn02 error handling (Page 1893)
CompAn02 I/Os (Page 1894)
CompAn02 block diagram (Page 1895)

13.1.3 CompAn02 functions

Functions of CompAn02

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status of analog logic blocks (Page 111).

See also

Description of CompAn02 (Page 1891)
CompAn02 modes (Page 1892)
CompAn02 error handling (Page 1893)
CompAn02 messaging (Page 1894)
CompAn02 I/Os (Page 1894)
CompAn02 block diagram (Page 1895)

13.1.4 CompAn02 error handling

CompAn02 error handling

The block does not report any errors.

Behavior of the block leaving the REAL number field

If one of the two input parameters `In1` or `In2` is outside the REAL number field, it is treated like a REAL number.

Example

- `In1 = #e+INF` and `In2 < #e+INF: GT/GE := 1`, the other output parameters are then 0
- `In1 = #e-INF` and `In2 > #e-INF: LT/LE := 1`, the other output parameters are then 0
- `In1 = #NAN#` or `In2 = #NAN#:` all output parameters are set to 0

See also

Description of CompAn02 (Page 1891)
CompAn02 modes (Page 1892)
CompAn02 functions (Page 1892)
CompAn02 messaging (Page 1894)
CompAn02 I/Os (Page 1894)
CompAn02 block diagram (Page 1895)

13.1.5 CompAn02 messaging

Messaging

This block does not offer messaging.

See also

Description of CompAn02 (Page 1891)
CompAn02 modes (Page 1892)
CompAn02 functions (Page 1892)
CompAn02 error handling (Page 1893)
CompAn02 block diagram (Page 1895)
CompAn02 I/Os (Page 1894)

13.1.6 CompAn02 I/Os

CompAn02 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Analog input value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
In2	Analog input value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
EQ	1= In1 = In2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
GE	1 = In1 ≥ In2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
GT	1 = In1 > In2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LE	1 = In1 ≤ In2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LT	1 = In1 < In2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Description of CompAn02 (Page 1891)
CompAn02 modes (Page 1892)
CompAn02 functions (Page 1892)
CompAn02 error handling (Page 1893)
CompAn02 messaging (Page 1894)
CompAn02 block diagram (Page 1895)

13.1.7 CompAn02 block diagram

CompAn02 block diagram

A block diagram is not provided for this block.

See also

Description of CompAn02 (Page 1891)
CompAn02 modes (Page 1892)
CompAn02 functions (Page 1892)
CompAn02 error handling (Page 1893)
CompAn02 messaging (Page 1894)
CompAn02 I/Os (Page 1894)

13.2 Limit - Limiting an analog value

13.2.1 Description of Limit

Object name (type + number) and family

Type + number: FB 1829

Family: LogicAn

Area of application for Limit

The block is used for the following applications:

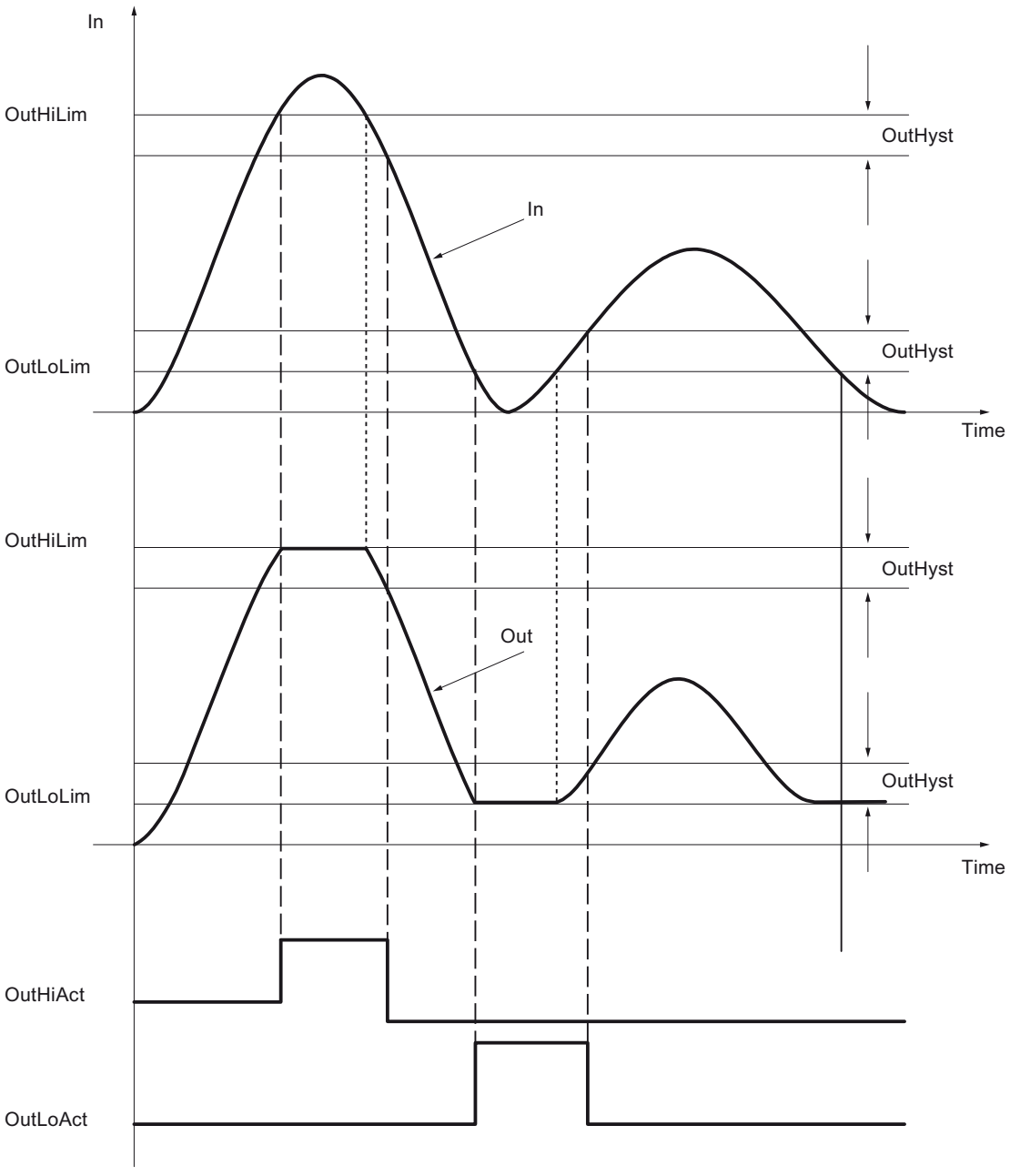
- Limiting of values

How it works

The Limit block is used to limit an analog value to an adjustable range.

The limits are set at the `OutHiLim` (high limit) and `OutLoLim` (low limit) input parameters. If a limit is violated, the limit you entered is output. The limit violation is also shown at the two output parameters `OutHiAct` (high) or `OutLoAct` (low).

You can configure hysteresis (`OutHyst` input parameter) to suppress signal flutters close to the limits.



Active conditions for setting limit values	Active conditions for resetting limit values
High limit active: In >= OutHiLim	High limit active: In < OutHiLim - OutHyst
Low limit active: In <= OutLoLim	Low limit active: In > OutLoLim + OutHyst

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Further addressing is not required.

Startup characteristics

The block does not have any startup characteristics.

A startup behavior can be reached with a connection of the input parameter `Init` to the output parameter `InitOut` of the Trigger block. For more information, refer to Startup characteristics over Trigger block (Page 69).

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

Limit block diagram (Page 1901)

Limit I/Os (Page 1900)

Limit messaging (Page 1900)

Limit error handling (Page 1899)

Limit functions (Page 1898)

Limit modes (Page 1898)

13.2.2 Limit modes

Limit modes

This block does not have any modes.

See also

Limit block diagram (Page 1901)

Limit I/Os (Page 1900)

Limit messaging (Page 1900)

Limit error handling (Page 1899)

Limit functions (Page 1898)

Description of Limit (Page 1896)

13.2.3 Limit functions

Functions of Limit

The function for this block is listed below.

Initialization

With the input parameter `Init`, the output parameters `OutHiAct` and `OutLoAct` can be initialized:

- `Init` = 0 (default): Normal limitation behavior
  If the input parameter `In` is in the hysteresis range, the output parameters `OutHiAct` and `OutLoAct` will remain unchanged.
- `Init` = 1:
  If the input parameter `In` is in the hysteresis range, the output parameters `OutHiAct` and `OutLoAct` will be reset.

To initialize the output parameters `OutHiAct` and `OutLoAct` after a startup, connect the input parameter `Init` with the output parameter `InitOut` of the Trigger block. For more information, refer to Startup characteristics over Trigger block (Page 69).

See also

Limit block diagram (Page 1901)
Limit I/Os (Page 1900)
Limit messaging (Page 1900)
Limit error handling (Page 1899)
Limit modes (Page 1898)
Description of Limit (Page 1896)

13.2.4 Limit error handling

Limit error handling

The block does not report any errors.

See also

Limit block diagram (Page 1901)
Limit I/Os (Page 1900)
Limit messaging (Page 1900)
Limit functions (Page 1898)
Limit modes (Page 1898)
Description of Limit (Page 1896)

13.2.5 Limit messaging

Messaging

This block does not offer messaging.

See also

Limit block diagram (Page 1901)

Limit I/Os (Page 1900)

Limit error handling (Page 1899)

Limit functions (Page 1898)

Limit modes (Page 1898)

Description of Limit (Page 1896)

13.2.6 Limit I/Os

I/Os of Limit

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 1898)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
In	Analog input value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Init	Initialize function block	BOOL	0
OutHyst*	Hysteresis in %	REAL	0.0
OutHiLim	High limit of output value	REAL	100.0
OutLoLim	Low limit of output value	REAL	0.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see Limit error handling (Page 1899)	INT	-1
Out	Analog output value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
OutHiAct	1 = High limit overshoot	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OutLoAct	1 = Low limit undershoot	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Limit block diagram (Page 1901)

Limit messaging (Page 1900)

Limit modes (Page 1898)

Description of Limit (Page 1896)

13.2.7 Limit block diagram

Limit block diagram

A block diagram is not provided for this block.

See also

Limit I/Os (Page 1900)

Limit messaging (Page 1900)

Limit error handling (Page 1899)

Limit functions (Page 1898)

Limit modes (Page 1898)

Description of Limit (Page 1896)

13.3 MuxAn03 - Selection of an analog value to increase availability / safety

13.3.1 Description of MuxAn03

Object name (type + number) and family

Type + number: FB 1860

Family: LogicAn

Area of application for MuxAn03

The block is used for the following applications:

- Selection of an analog value to increase availability or safety in the input of analog values

How it works

The block uses up to three process values `PV1` ... `PV3` to determine an output value and outputs this with the corresponding signal status at the `PV` output parameter.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any special startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

MuxAn03 block diagram (Page 1909)

MuxAn03 I/Os (Page 1907)

MuxAn03 messaging (Page 1907)

MuxAn03 error handling (Page 1906)

MuxAn03 functions (Page 1903)

MuxAn03 modes (Page 1903)

13.3.2 MuxAn03 modes

MuxAn03 operating modes

This block does not have any modes.

See also

MuxAn03 block diagram (Page 1909)

MuxAn03 I/Os (Page 1907)

MuxAn03 messaging (Page 1907)

MuxAn03 error handling (Page 1906)

MuxAn03 functions (Page 1903)

Description of MuxAn03 (Page 1902)

13.3.3 MuxAn03 functions

Functions of MuxAn03

The functions for this block are listed below.

Selection of output signal

Use the `SelValue` input parameter to select whether the output value is to offer higher availability or higher safety.

Increasing availability

- **Selection 1 of 2** (SelValue = 0):The block determines the input parameter with the higher priority from the two input parameters PV1 and PV2 based on their signal status. The value determined is written to the PV output parameter.
  If both input parameters have the same signal status, the PV1input parameter is written to the PVoutput parameter.
  Display of process parameters with bad signal status
  If the signal status of PV1 is bad (16#00, 16#28), the PV1Bad output is set.
  If the signal status of PV2 is bad (16#00, 16#28), the PV2Bad output is set.
  Regardless of PV3, the PV3Bad output is always set to 0.
  Mean, median, maximum and minimum
  They are determined from PV1 and PV2:
  - PV_Mean = (PV1 + PV2) /2
  - PV_Median = PV1
  - PV_Max = Max(PV1, PV2)
  - PV_Min = Min(PV1, PV2)

  The status is formed by PV1.ST and PV2.ST. If the status is the same, it is output at PV_Mean.ST, PV_Median.ST, PV_Max.ST and PV_Min.ST. If the statuses are different, the status with the lower priority is output.
- **Selection 1 of 3** (SelValue = 1): The block determines the input parameter with the highest priority from the three input parameters PV1, PV2 and PV3 based on their signal status. The value determined is written to the PV output parameter.
  If two or more of the input parameters have the same signal status, the one with the lowest index is written to the PV output parameter.
  Display of process parameters with bad signal status
  If the signal status of PV1 is bad (16#00, 16#28), the PV1Bad output is set.
  If the signal status of PV2 is bad (16#00, 16#28), the PV2Bad output is set.
  If the signal status of PV3 is bad (16#00, 16#28), the PV3Bad output is set
  Mean, median, maximum and minimum
  They are determined from PV1, PV2 and PV3:
  - PV_Mean = (PV1 + PV2 + PV3) /3
  - PV_Median = mean PV value of PV1, PV2 and PV3. If 2 of the 3 PV values are the same, the value of the same PVs is output as the mean PV value
    Examples:
    PV1 =1, PV2 =2, PV3 =3: thus PV_Median =2.
    PV1 =1, PV2 =1, PV3 =3: thus PV_Median =1.
  - PV_Max = Max(PV1, PV2, PV3)
  - PV_Min = Min(PV1, PV2, PV3)
    The status is formed by PV1.ST, PV2.ST and PV3.ST. If the status is the same, it is output at PV_Mean.ST, PV_Max.ST and PV_Min.ST. If the statuses are different, the status with the second highest priority is output.

Increasing safety

- **Selection 2 of 2** (SelValue = 2): The block determines whether the two input parameters PV1 and PV2 have the same signal status and whether they deviate from one another by no more than the setting at input parameter PlDiff. Only then does the PV output parameter also have this signal status. The analog value of PVis set to a value of PV1.
  If PV1 and PV2 deviate from one another by more than PlDiff, the PV signal status is set to "Bad, device-related". The analog value of PVis set to a value of PV1.
  If PV1 and PV2 do not deviate from one another by more than PlDiff, but have a different signal state, the PV output parameter is generated from the lower priority signal status of the two input parameters and the associated analog value.
  Display of non-plausible or bad process parameters
  If the process value PV1 is non-plausible or bad (16#00, 16#28), the PV1Bad output is set.
  If the process value PV2 is non-plausible or bad (16#00, 16#28), the PV2Bad output is set.
  Regardless of PV3, the PV3Bad output is always set to 0
  Mean, median, maximum and minimum
  They are determined from PV1 and PV2. If PV1 and PV2 deviate from one another by more than PlDiff, the signal status of PV_Mean, PV_Median, PV_Max and PV_Min is set to "Bad, device-related" (16#00) and the process values of PV_Mean, PV_Median, PV_Max and PV_Min are frozen. If the absolute deviation is less than or equal to PlDiff, the mean value, maximum and minimum are calculated as follows:
  - PV_Mean = (PV1 + PV2) /2
  - PV_Median = PV1
  - PV_Max = Max(PV1, PV2)
  - PV_Min = Min(PV1, PV2)

  The status is formed by PV1.ST and PV2.ST. If the status is the same, it is output at PV_Mean.ST, PV_Max.ST and PV_Min.ST. If the statuses are different, the status with the lower priority is output.
- **Selection 2 of 3** (SelValue = 3): The block determines whether two of the three input parameters PV1, PV2 and PV3 have the same high priority signal status and whether they deviate from one another by no more than the setting at input parameter PlDiff. Only if this is the case, does the PV output parameter also have this signal status. The analog value of PV is set to the value of the input parameter with the lowest index (PV1 or PV2) of the values determined.
  If all input parameters PV1, PV2 and PV3 deviate from one another by more than PlDiff, the signal status of PV is set to "Bad, device-related". The analog value of PVis set to the value of the PV1 input parameter.
  If at least two input parameters do not deviate from one another by more than PlDiff, but have a different signal state, the value from these input parameters with the signal status of the second highest priority is written to the PV output parameter.
  Display of non-plausible or bad process parameters
  If the process value PV1 is non-plausible or bad (16#00, 16#28), the PV1Bad output is set.
  If the process value PV2 is non-plausible or bad (16#00, 16#28), the PV2Bad output is set.
  If the process value PV3 is non-plausible or bad (16#00, 16#28), the PV3Bad output is set.
  Mean value, maximum and minimum
  They are determined from PV1, PV2 and PV3. An internal mean value is calculated first from PV1, PV2 and PV3.
  This mean value is used to determine the deviations for each PV value. If all absolute deviations are less than or equal to PlDiff, the mean, median, maximum and minimum

are formed from PV1, PV2 and PV3 and the status with the second highest priority is output (see section Increasing availability, selection 1 to 3).
If at least one of the deviations is outside of PlDiff, the PV with the greatest deviation is excluded from further calculation.
If the two remaining process values (PVx1, PVx2) deviate from one another by more than PlDiff, the signal status of PV_Mean, PV_Median, PV_Max and PV_Min is set to "Bad, device-related" (16#00) and the process values of PV_Mean, PV_Median, PV_Max and PV_Min are frozen. If the absolute deviation is less than or equal to PlDiff, the mean value, maximum and minimum are calculated as follows:

- PV_Mean = (PVx1 + PVx2) /2
- PV_Median = PVx1
- PV_Max = Max(PVx1, PVx2)
- PV_Min = Min(PVx1, PVx2)

The status is formed by PVx1.ST and PVx2.ST. If the status is the same, it is output at PV_Mean.ST, PV_Median, PV_Max.ST and PV_Min.ST. If the statuses are different, the status with the lower priority is output.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for blocks with configurable status prioritization (Page 114).

See also

MuxAn03 block diagram (Page 1909)
MuxAn03 I/Os (Page 1907)
MuxAn03 messaging (Page 1907)
MuxAn03 error handling (Page 1906)
MuxAn03 modes (Page 1903)
Description of MuxAn03 (Page 1902)

13.3.4 MuxAn03 error handling

MuxAn03 error handling

Parameter assignment errors are handled as follows:

- If the input parameter is SelValue < 0, the parameter is automatically set to SelValue = 0.
- If the input parameter is SelValue > 3, the parameter is automatically set to SelValue = 3 .
- If the input parameter is SelPrio < 0, the parameter is automatically set to SelPrio = 0 .
- If the input parameter is SelPrio > 7, the parameter is automatically set to SelPrio = 7 .

An error number is not output in any of these cases.

See also

MuxAn03 block diagram (Page 1909)
MuxAn03 I/Os (Page 1907)
MuxAn03 messaging (Page 1907)
MuxAn03 functions (Page 1903)
MuxAn03 modes (Page 1903)
Description of MuxAn03 (Page 1902)

13.3.5 MuxAn03 messaging

Messaging

This block does not offer messaging.

See also

MuxAn03 block diagram (Page 1909)
MuxAn03 I/Os (Page 1907)
MuxAn03 error handling (Page 1906)
MuxAn03 functions (Page 1903)
MuxAn03 modes (Page 1903)
Description of MuxAn03 (Page 1902)

13.3.6 MuxAn03 I/Os

MuxAn03 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
PV1	Process value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV2	Process value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
PV3	Process value 3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PlDiff	Plausibility check value	REAL	0.0
SelPrio*	Setting of the prioritization for forming the best signal status	INT	6
SelValue*	Selection criterion for the search	INT	0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
PV	Process value determined	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Max	Process value maximum	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Mean	Process value mean	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Median	Process value median	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Min	Process value minimum	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV1Bad	Process value PV1 is not plausible or bad (16#00 or 16#28)	BOOL	0
PV2Bad	Process value PV2 is not plausible or bad (16#00 or 16#28)	BOOL	0
PV3Bad	Process value PV3 is not plausible or bad (16#00 or 16#28)	BOOL	0

See also

MuxAn03 block diagram (Page 1909)
MuxAn03 messaging (Page 1907)
MuxAn03 error handling (Page 1906)
MuxAn03 functions (Page 1903)

MuxAn03 modes (Page 1903)
Description of MuxAn03 (Page 1902)

13.3.7 MuxAn03 block diagram

MuxAn03 block diagram

A block diagram is not provided for this block.

See also

MuxAn03 I/Os (Page 1907)
MuxAn03 messaging (Page 1907)
MuxAn03 error handling (Page 1906)
MuxAn03 functions (Page 1903)
MuxAn03 modes (Page 1903)
Description of MuxAn03 (Page 1902)

13.4 MuxAn08 - Selection of an analog value to increase availability / safety

13.4.1 Description of MuxAn08

Object name (type + number) and family

Type + number: FB 1831

Family: LogicAn

Area of application of MuxAn08

The block is used for the following applications

- Selection of an analog value to increase availability or safety in the input of analog values

How it works

The block uses up to eight `PV1 ... PV8` process values to determine an output value and outputs this with the corresponding signal status at the PV output parameter.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any special startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

13.4.2 MuxAn08 modes

MuxAn08 operating modes

This block does not have any operating modes.

13.4.3 MuxAn08 functions

Functions of MuxAn08

The functions for this block are listed below.

Selection of output signal

Use the `SelValue` input parameter to select whether the output value is to offer higher availability or higher safety.

Increasing availability

- **Selection 1of 2**, **1 of 3, ... 1 of 8** (`SelValue` = 0..6):
  The input parameter `SelValue` defines the relevant input parameters of the process values to be considered for the determination of PV.

SelValue	Relevant input parameter	Irrelevant input parameter
0	PV1..PV2	PV3..PV8
1	PV1..PV3	PV4..PV8
2	PV1..PV4	PV5..PV8
3	PV1..PV5	PV6..PV8
4	PV1..PV6	PV7..PV8
5	PV1..PV7	PV8
6	PV1..PV8	-

The block determines the highest priority of the two input parameters from the relevant input parameters based on their signal status on and writes it to the output parameter PV. If several input parameters have the same level of signal status priority, the one with the lowest index is written to the output parameter.

Display of process parameters with "bad" signal status

When the relevant input parameters (`PV1`, `PV2`, .., `PV8`) have a "bad" signal status (`16#00` or `16#28`), the corresponding output parameters (`PV1Bad`, `PV2Bad`..`PV8Bad`) are set. The output parameters of the irrelevant input parameters are always reset.

Mean value, maximum and minimum

- PV_Mean = (PV1 + PV2 + ... +PVn) / n
- PV_Max = Max(PV1, PV2... PVn)
- PV_Min = Min(PV1, PV2... PVn)
  n depends on SelValue
  The status is formed from the `PV1..PV`n input parameters. If the status is the same, it is output at the `PV_Mean`, `PV_Max` and `PV_Min` output parameters. If the states are different, the status with the second highest priority is output at the `PV_Mean`, `PV_Max` and `PV_Min` output parameters.

Increasing safety

- **Selection 2 of 2..8** (`SelValue` = 7..13):
  The input parameter `SelValue` defines the relevant input parameters of the process values to be considered for the determination of PV.

SelValue	Relevant input parameter	Irrelevant input parameter
7	PV1..PV2	PV3..PV8
8	PV1..PV3	PV4..PV8
9	PV1..PV4	PV5..PV8
10	PV1..PV5	PV6..PV8
11	PV1..PV6	PV7..PV8
12	PV1..PV7	PV8
13	PV1..PV8	-

The block determines whether two of the relevant input parameters have the same level of signal status priority and whether they deviate from one another by no more than the setting at input parameter `PlDiff`. Only then does the `PV` output parameter also have this signal status. The analog value of `PV` is set to the value of the input parameter with the lowest index of the values determined.
If all input parameters deviate from one another by more than `PlDiff`, the signal status of `PV` is set to "Bad, device-related". The analog value of `PV`is set to the value of the `PV1` input parameter.
If the least two input parameters do not deviate from one another by more than `PlDiff`, but have a different signal state, the value with the signal status of the second highest priority from these input parameters is written to the `PV` output parameter.
Display of process parameters with "bad" signal status
When the relevant input parameters (`PV1`, `PV2`, .., `PV8`) are not plausible or have a "bad" signal status (`16#00` or `16#28`), the corresponding output parameters (`PV1Bad`, `PV2Bad`..`PV8Bad`) are set. The corresponding output parameters of the irrelevant input parameters are always reset.
Mean value, maximum and minimum
The `PV` values are plausible if the absolute values of the deviation are less than or equal to the `PlDiff`.
If the absolute deviation is less than or equal to `PlDiff`, the mean value, maximum and minimum are calculated as follows:

1. Determination of the mean from `PV1.. PV`n
   `n` depends on `SelValue`
2. Determination of the deviations `ABS`(`PVi` – Mean value)
3. If all `PV` are plausible, then the mean, maximum and minimum are determined, as described in section "Increasing availability `SelValue = 0..6`".

4. If one or more `PV` are not plausible, repeat the calculation of the mean starting with 1. without PV input with the highest deviation. If there is more than one PV input with equally high deviation, only the first PV input is not used for further calculation.
5. If only 2 `PV` inputs remain and they are plausible, then the mean, maximum and minimum are determined, as described in section "Increasing availability `SelValue = 0`". When they are not plausible, the mean value, maximum and minimum are determined as follows:

```
PV_Max.Value: Keep old value
PV_Min.Value: Keep old value
PV_Mean.Value: Keep old value
PV_Max.ST = 16#00;
PV_Min.ST = 16#00;
PV_Mean.ST = 16#00;
```

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for blocks with configurable status prioritization (Page 114).

13.4.4 MuxAn08 error handling

Error handling of MuxAn08

Parameter assignment errors are handled as follows:

- If the input parameter is `SelValue` < 0, the parameter is automatically set to `SelValue` = 0.
- If the input parameter is `SelValue` > 13, the parameter is automatically set to `SelValue` = 13 .
- If the input parameter is `SelPrio` < 0, the parameter is automatically set to `SelPrio` = 0 .
- If the input parameter is `SelPrio` > 7, the parameter is automatically set to `SelPrio` = 7 .

An error number is not output in any of these cases.

13.4.5 MuxAn08 messaging

Messaging

This block does not offer messaging.

13.4.6 MuxAn08 I/Os

I/Os of MuxAn08

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
PV1	Process value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV2	Process value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV3	Process value 3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV4	Process value 4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV5	Process value 5	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV6	Process value 6	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV7	Process value 7	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV8	Process value 8	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PlDiff	Plausibility check value	REAL	0.0
SelPrio*	Setting of the prioritization for forming the best signal status	INT	6
SelValue*	Selection criterion for the search	INT	0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
PV	Process value determined	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
PV_Mean	Process value mean	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Max	Process value maximum	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_Min	Process value minimum	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV1Bad	Process value PV1 is not plausible or bad (16#00 or 16#28)	BOOL	0
PV2Bad	Process value PV2 is not plausible or bad (16#00 or 16#28)	BOOL	0
PV3Bad	Process value PV3 is not plausible or bad (16#00 or 16#28)	BOOL	0
PV4Bad	Process value PV4 is not plausible or bad (16#00 or 16#28)	BOOL	0
PV5Bad	Process value PV5 is not plausible or bad (16#00 or 16#28)	BOOL	0
PV6Bad	Process value PV6 is not plausible or bad (16#00 or 16#28)	BOOL	0
PV7Bad	Process value PV is not plausible or bad (16#00 or 16#28)	BOOL	0
PV8Bad	Process value PV8 is not plausible or bad (16#00 or 16#28)	BOOL	0

13.4.7 MuxAn08 block diagram

MuxAn08 block diagram

A block diagram is not provided for this block.

13.5 RateLim - Signal ramp

13.5.1 Description of RateLim

Object name (type + number) and family

Type + number: FB 1882

Family: LogicAn

Area of application for RateLim

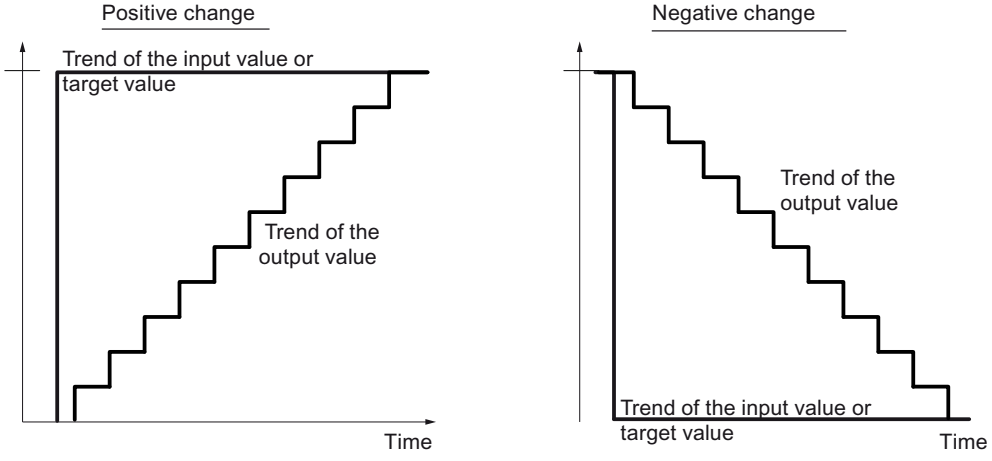
The block is used for the following applications:

- Limitation of the slope of an analog signal
- Ramped approach to a target value

How it works

Slope limit: Starting from the current output value, the new output value is calculated in such a way that it does not exceed a specified positive or negative slope.

Ramp function: The current output value can be ramped up to a target value. The ramp slope can be configured by a time period or by a positive or negative ramp slope.



The input value is forwarded directly when the gradient and ramp function are switched off (`RmpOn.Value = 0`, `RateOn.Value = 0`): `Out.Value = In.Value`.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB3x). The block is then installed automatically in startup OB (OB100).

Further addressing is not required.

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

RateLim functions (Page 1917)

RateLim messaging (Page 1920)

RateLim I/Os (Page 1921)

RateLim block diagram (Page 1923)

RateLim error handling (Page 1920)

RateLim modes (Page 1917)

13.5.2 RateLim modes

RateLim operating modes

This block does not have any modes.

See also

RateLim block diagram (Page 1923)

RateLim I/Os (Page 1921)

RateLim messaging (Page 1920)

RateLim error handling (Page 1920)

RateLim functions (Page 1917)

Description of RateLim (Page 1916)

13.5.3 RateLim functions

Functions of RateLim

The functions for this block are listed below.

Limitation of the slope of the analog signal `In.Value`

The block calculates the gradient of the input signal over time and compares it with the `UpRaLim` (positive change) `DnRaLim` (negative change) limits. See the table below.

- If the slope exceeds the amount of the respective limit (`UpRaLim` or `DnRaLim`), the output `Out` is corrected only by the valid limit and the corresponding limit display `UpRaAct` = 1 or `DnRaAct` = 1 is set.
- If the gradient is within the valid range, the input value is transferred (`In` = `Out`) and `UpRaAct` = 1 or `DnRaAct` = 1 are reset.
- If the corresponding limit is 0 (`UpRaLim` with positive gradient or `DnRaLim` with negative gradient), then the input value `In` is written directly to the output `Out`.

RateOn	ΔIn / Δt	Meaning	Output `Out`	UpRaAct	DnRaAct
1	< |`DnRaLim`|	Input value dropping too fast	`Out` - (|`DnRaLim`| · `SampleTime`) * Time factor	0	1
1	`|DnRaLim|` to `UpRaLim`	Change speed `In` is permitted	`In`	0	0
1	> |`UpRaLim`|	Input value `In` rising too fast	`Out` + (`|UpRaLim|` · `SampleTime`) * Time factor	1	0
0	-	`RateOn` is deactivated	`In`	0	0

In this case, the time factor is formed from the `TimeFactor` parameter.

`TimeFactor`	Time factor
0	1
1	1/60
2	1/(60*60)

Ramp function

Starting from the current output value `Out.Value`, the output value can be ramped to the target value `RmpTarget` when the ramp function is switched on or the target value is changed.

You can use the `RmpModTime` input parameter to specify whether the ramp is defined by time or gradients.

- If you select time (`RmpModTime = 1`): The gradient of the ramp is calculated automatically by the block so that after the start (`RmpOn.Value = 0 -> 1`) or a change of the target value (`RmpTarget.Value`), the target value is reached after the configured time (`RmpTime`). The unit of the ramp time (`RmpTime`) depends on `TimeFactor`.
- If you select gradients (`RmpModTime = 0`): The ramp slope corresponds to the configured rates of change `UpRaLim` (positive) or `DnRaLim` (negative).

When the output value has reached the target, the output value remains at the target value as long as the ramp function remains enabled.

Switching the ramp function and slope limiting function on or off

The ramp function is switched off with the `RmpOn = 0` input; the slope limiting function is switched off with the `RateOn` = 0 input. If both are switched off, the `In` input value is written directly to the `Out` output. Limits are no longer monitored. If both functions are switched on, the ramp function has priority.

Stopping the caculation of the ramp output value

Set `Hold` = 1 to interrupt the calculation of the ramp output value. The output value `Out` is held and the output parameters `UpRaAct` and `DnRaAct` are set to 0. The status of `Out`, `UpRaAct`, and `DnRaAct` are realized.

Set `Hold` = 0 to continue the calculation of the ramp output value. The calculation is continued with the last output value.

Time base of the gradient limits

The gradient limits are set at the `UpRaLim` and `DnRaLim` parameters depending on the `TimeFactor`.

`TimeFactor = 0`: Unit of the gradient limiting is Unit/Second

`TimeFactor = 1`: Unit of the gradient limiting is Unit/Minute

`TimeFactor = 2`: Unit of the gradient limiting is Unit/Hour

Forming the signal status for blocks

The signal status of `Out` is formed from:

```
RmpOn =1: Out.ST := RmpTarget.ST
```

Otherwise: `Out.ST := In.ST`

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) .

The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
8	Unit for the rate of change (Page 146)

See also

Description of RateLim (Page 1916)

RateLim messaging (Page 1920)

RateLim I/Os (Page 1921)

RateLim block diagram (Page 1923)

RateLim modes (Page 1917)

RateLim error handling (Page 1920)

13.5.4 RateLim error handling

Error handling of RateLim

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
30	The value of `In` can no longer be displayed in the REAL number field.
43	Incorrect time unit set at `TimeFactor`.

See also

RateLim block diagram (Page 1923)

RateLim I/Os (Page 1921)

RateLim messaging (Page 1920)

RateLim modes (Page 1917)

Description of RateLim (Page 1916)

RateLim functions (Page 1917)

13.5.5 RateLim messaging

Messaging

This block does not offer messaging.

See also

Description of RateLim (Page 1916)
RateLim functions (Page 1917)
RateLim I/Os (Page 1921)
RateLim block diagram (Page 1923)
RateLim error handling (Page 1920)
RateLim modes (Page 1917)

13.5.6 RateLim I/Os

I/Os of RateLim

Input parameters

Parameter	Description	Type	Default
`DnRaLim`	Maximum negative change in the output value in units/s or %/s	REAL	3.0
`EN`	1 = Called block will be processed	BOOL	1
`Feature`	I/O for additional functions (Page 1917)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`Hold`	Hold calculation	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80
`In*`	Analog input value	STRUCT • Value: REAL • ST:BYTE	- • 0.0 • 16#80
`InScale`	Scaling of the measuring range as a structure	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
`RateOn`	1 = Switch on gradient function	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80
`RmpModTime`	1 = Use time (`RmpTime`) for the ramp function 0 = Use gradients (`UpRaLim`, `DnRaLim`) for the ramp function	BOOL	0
`RmpOn`	1 = Switch on ramp function for target value `RmpTarget`	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
`RmpTarget`	Target value for the ramp function	STRUCT • Value: REAL • ST:BYTE	- • 0 • 16#80
`RmpTime*`	Duration [unit depends on `TimeFactor`] for ramp function from the current `Out` to `RmpTarget`	REAL	0.0
`SampleTime`	Sampling time [s] (assigned automatically)	REAL	0.1
`TimeFactor`	Time unit: 0 = Seconds 1 = Minutes 2 = Hours	INT	0
`UpRaLim`	Maximum positive change in the output value in units/s or %/s	REAL	3.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
`DnRaAct`	1 = Negative change in the output value	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80
`ENO`	1 = Block algorithm completed without errors	BOOL	0
`ErrorNum`	Output of pending error number. For error numbers that can be output by this block, see RateLim error handling (Page 1920)	INT	-1
`Out`	Output value	STRUCT • Value: REAL • ST:BYTE	- • 0.0 • 16#80
`UpRaAct`	1 = Positive change in the output value	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80

See also

Description of RateLim (Page 1916)
RateLim messaging (Page 1920)
RateLim block diagram (Page 1923)
RateLim modes (Page 1917)

13.5.7 RateLim block diagram

RateLim block diagram

A block diagram is not provided for this block.

See also

RateLim I/Os (Page 1921)

RateLim messaging (Page 1920)

RateLim error handling (Page 1920)

RateLim functions (Page 1917)

RateLim modes (Page 1917)

Description of RateLim (Page 1916)

13.6 RedAn02 - 1 out of 2 selection for redundant analog values

13.6.1 Description of RedAn02

Object name (type + number) and family

Type + number: FC 385

Family: LogicAn

Area of application for RedAn02

The block is used for the following applications:

- 1 out of 2 selection for redundant analog values

How it works

The block selects from two input values the one with the best signal status and outputs it at the output `Out`. In addition, the outputs `SimAct`, `Uncertain` and `LossRed` are set according to the signal status.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

RedAn02 modes (Page 1925)

RedAn02 functions (Page 1925)

RedAn02 error handling (Page 1926)

RedAn02 messaging (Page 1926)

RedAn02 I/Os (Page 1927)

RedAn02 block diagram (Page 1928)

13.6.2 RedAn02 modes

RedAn02 operating modes

This block does not have any modes.

See also

Description of RedAn02 (Page 1924)
RedAn02 functions (Page 1925)
RedAn02 error handling (Page 1926)
RedAn02 messaging (Page 1926)
RedAn02 I/Os (Page 1927)
RedAn02 block diagram (Page 1928)

13.6.3 RedAn02 functions

Functions of RedAn02

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status of redundancy blocks (Page 112).

See also

Description of RedAn02 (Page 1924)
RedAn02 modes (Page 1925)
RedAn02 error handling (Page 1926)
RedAn02 messaging (Page 1926)
RedAn02 I/Os (Page 1927)
RedAn02 block diagram (Page 1928)

13.6.4 RedAn02 error handling

RedAn02 error handling

The block does not report any errors.

See also

Description of RedAn02 (Page 1924)
RedAn02 modes (Page 1925)
RedAn02 functions (Page 1925)
RedAn02 messaging (Page 1926)
RedAn02 I/Os (Page 1927)
RedAn02 block diagram (Page 1928)

13.6.5 RedAn02 messaging

Messaging

This block does not offer messaging.

See also

Description of RedAn02 (Page 1924)
RedAn02 modes (Page 1925)
RedAn02 functions (Page 1925)
RedAn02 error handling (Page 1926)
RedAn02 block diagram (Page 1928)
RedAn02 I/Os (Page 1927)

13.6.6 RedAn02 I/Os

RedAn02 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Analog input value 1	STRUCT • Value: REAL • ST:BYTE	- • 0.0 • 16#80
In2	Analog input value 2	STRUCT • Value: REAL • ST:BYTE	- • 0.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
LossRed	1= Redundancy loss at one of the inputs	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80
Out	Output of the process value with the better signal status	STRUCT • Value: REAL • ST:BYTE	- • 0.0 • 16#80
SimAct	1 = one input value has the Simulation status	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80
Uncertain	1 = one input value has the "uncertain" status	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80

See also

Description of RedAn02 (Page 1924)
RedAn02 modes (Page 1925)
RedAn02 functions (Page 1925)
RedAn02 error handling (Page 1926)
RedAn02 messaging (Page 1926)
RedAn02 block diagram (Page 1928)

13.6.7 RedAn02 block diagram

RedAn02 block diagram

A block diagram is not provided for this block.

See also

Description of RedAn02 (Page 1924)
RedAn02 modes (Page 1925)
RedAn02 functions (Page 1925)
RedAn02 messaging (Page 1926)
RedAn02 I/Os (Page 1927)
RedAn02 error handling (Page 1926)

13.7 SelA02In - Output of two analog values

13.7.1 Description of SelA02In

Object name (type + number)

Type + number: FB 1886

Family: LogicAn

Area of application for SelA02In

The block is used for the following applications:

- Selecting one of two analog values and switching it through to the output.

How it works

Depending on the setting of parameter `SelMode` the block selects one of the two input parameters `In1` or `In2` and writes its value to the output parameter `Out`.

This is displayed at the `In2Selected` output parameter.

Configuration

Use the CFC editor to install the block in any OB.

For the SelA02In block, the Advanced Process Library contains templates for process tag types as examples and there is a example project (APL_Example_xx, xx designates the language variant) with an application scenario for this block, which describes how the block works.

Examples of process tag types:

- Source chart for GainSched function block (gain scheduling) (Page 2321)
- Override control (Page 2322)
- Override control with PIDConR (OverrideR) (Page 2324)

Application scenario in the example project:

- Process simulation including noise generator (ProcSimC; ProcSimS) (Page 2347)

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

SelA02In block diagram (Page 1933)
SelA02In I/Os (Page 1932)
SelA02In messaging (Page 1932)
SelA02In error handling (Page 1931)
SelA02In functions (Page 1930)
SelA02In modes (Page 1930)

13.7.2 SelA02In modes

SelA02In modes

This block does not have any modes.

See also

SelA02In block diagram (Page 1933)
SelA02In I/Os (Page 1932)
SelA02In messaging (Page 1932)
SelA02In error handling (Page 1931)
SelA02In functions (Page 1930)
Description of SelA02In (Page 1929)

13.7.3 SelA02In functions

Functions of SelA02In

The functions for this block are listed below.

Select input parameter

The `SelMode` parameter can affect the selection as follows:

- `SelMode` ≤ 0: The selection depends on the parameter `Sel_In2`
  - `Sel_In2` = 0: Input parameter `In1` is written with its signal status to output parameter `Out`.
  - `Sel_In2` = 1: Input parameter `In2` is written with its signal status to output parameter `Out`.
- `SelMode` = 1: The input parameter with the lower value (`In1` or `In2`) is written with its signal status to the output parameter `Out`.
- `SelMode` ≥ 2: The input parameter with the higher value (`In1` or `In2`) is written with its signal status to the output parameter `Out`.

The signal status of `Sel_In2` is output at the `In2Selected` output parameter.

See also

SelA02In block diagram (Page 1933)

SelA02In I/Os (Page 1932)

SelA02In messaging (Page 1932)

SelA02In error handling (Page 1931)

SelA02In modes (Page 1930)

Description of SelA02In (Page 1929)

13.7.4 SelA02In error handling

SelA02In error handling

The block does not report any errors.

See also

SelA02In block diagram (Page 1933)

SelA02In I/Os (Page 1932)

SelA02In messaging (Page 1932)

SelA02In functions (Page 1930)

SelA02In modes (Page 1930)

Description of SelA02In (Page 1929)

13.7.5 SelA02In messaging

Messaging

This block does not offer messaging.

See also

SelA02In block diagram (Page 1933)
SelA02In I/Os (Page 1932)
SelA02In error handling (Page 1931)
SelA02In functions (Page 1930)
SelA02In modes (Page 1930)
Description of SelA02In (Page 1929)

13.7.6 SelA02In I/Os

SelA02In I/Os

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`In1`	Analog process value 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`In2`	Analog process value 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`SelMode`	Selection of the function: 0 = Select with `Sel_In2` 1 = Minimum 2 = Maximum	INT	0
`Sel_In2`	Selecting the input parameter: 0 = `In1` 1 = `In2`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Output parameters

Parameter	Description	Type	Default
`ENO`	1 = Block algorithm completed without errors	BOOL	0
`In2Selected`	Selected input parameter: 0 = `In1` 1 = `In2`	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`Out`	Analog process value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

SelA02In block diagram (Page 1933)
SelA02In messaging (Page 1932)
SelA02In error handling (Page 1931)
SelA02In functions (Page 1930)
SelA02In modes (Page 1930)
Description of SelA02In (Page 1929)

13.7.7 SelA02In block diagram

SelA02In block diagram

A block diagram is not provided for this block.

See also

SelA02In I/Os (Page 1932)
SelA02In messaging (Page 1932)
SelA02In error handling (Page 1931)
SelA02In functions (Page 1930)
SelA02In modes (Page 1930)
Description of SelA02In (Page 1929)

13.8 SelA16In - Output of 16 analog values

13.8.1 Description of SelA16In

Object name (type + number) and family

Type + number: FB 1888

Family: LogicAn

Area of application for SelA16In

The block is used for the following applications:

- Selecting one of 16 analog values and switching it through to the output.

How it works

The block writes the value of one of the input parameters `In01` through `In16` to the output parameter `Out`. The selection is performed via the `SelInt` input parameter.

The signal status of the selected input parameter is written to the signal status of the output parameter `Out`. There is no additional signal status evaluation.

The unit `InxUnit` of the selected input parameter `Inx` (x = 01 ...16) is written to the output parameter `OutUnit`.

The analog input 1 (`In01`) is operable only if `OS_Perm.Bit22` is set to 1.

It has an upper limit associated with the parameter `In01HiLim` and a lower limit associated with the parameter `In01LoLim`.

If `Feature` Bit 7 is set to 1 and the permission is enabled (`OS_Perm.Bit22` = 1), the analog input 1 (`In01`) can be reserved for the operator. If `Feature` Bit 7 is set to 1, the analog input 1 (`In01`) is always visible independent of `Feature` Bit 5.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

You can find a description for each parameter in section SelA16In I/Os (Page 1940).

Status bit	Parameter
0 - 2	Not used
3	`OosAct.Value`
4	`OosLi.Value`
5	Not used
6	`OnAct.Value`
7 - 11	Not used
12 – 27	Used to highlight the green line in the standard view
28	1 = With internal/external selection from the operator (`Feature` Bit 6)
29	1 = `SelExtAct`
30 - 31	Not used

Status word allocation for `Status2` parameter

Status bit	Parameter
0	1 = Display input 1
...	...
15	1 = Display input 16
16 - 31	Not used

See also

SelA16In block diagram (Page 1944)

SelA16In messaging (Page 1939)

SelA16In error handling (Page 1939)

SelA16In functions (Page 1936)

SelA16In modes (Page 1935)

13.8.2 SelA16In modes

SelA16In operating modes

The block can be operated using the following modes:

- On (Page 71)
- Out of service (Page 71)

"On"

You can find general information about the "On" mode in the section On (Page 71).

"Out of service"

You can find general information about the "Out of service" mode in the section Out of service (Page 71).

See also

SelA16In block diagram (Page 1944)

SelA16In I/Os (Page 1940)

SelA16In messaging (Page 1939)

SelA16In error handling (Page 1939)

SelA16In functions (Page 1936)

Description of SelA16In (Page 1934)

13.8.3 SelA16In functions

Functions of SelA16In

The functions for this block are listed below.

Opening additional faceplates

This block provides the standard function Opening additional faceplates (Page 203).

Operator control permissions

This block provides the standard function Operator control permissions (Page 248).

The block has the following permissions for the `OS_Perm` parameter:

Bit	Function
0	Not used
1	1 =Operator can switch to "On" mode
2	Not used
3	1 =Operator can switch to "Out of service" mode
4	1 = Operator can set the input 1
5	1 = Operator can set the input 2

Bit	Function
6	1 = Operator can set the input 3
7	1 = Operator can set the input 4
8	1 = Operator can set the input 5
9	1 = Operator can set the input 6
10	1 = Operator can set the input 7
11	1 = Operator can set the input 8
12	1 = Operator can set the input 9
13	1 = Operator can set the input 10
14	1 = Operator can set the input 11
15	1 = Operator can set the input 12
16	1 = Operator can set the input 13
17	1 = Operator can set the input 14
18	1 = Operator can set the input 15
19	1 = Operator can set the input 16
20	1 = Operator can switch to external selection
21	1 = Operator can switch to internal selection
22	1 = Operator can change the value of input 1
23 - 31	Not used

Note

If you interconnect a parameter that is also listed in `OS_Perm` as a parameter, you have to reset the corresponding `OS_Perm` bit.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status of analog logic blocks (Page 111).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `In1.ST`

etc. to

- `In16.ST`

Selecting a unit of measure

This block provides the standard function Selecting a unit of measure (Page 207).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) section. The following functionality is available for this block at the relevant bits:

Bit	Function
1	Reaction to the out of service mode (Page 176)
2	Resetting the commands for changing the mode (Page 160)
4	Setting switch or button mode (Page 166)
5	Display only input values that are interconnected in the faceplate (Page 156)
6	External/internal selection specification (Page 179)
7	Analog input 1 is reserved for the operator (Page 182)
24	Enabling local operator authorization (Page 157)

Cascading of SelA16In

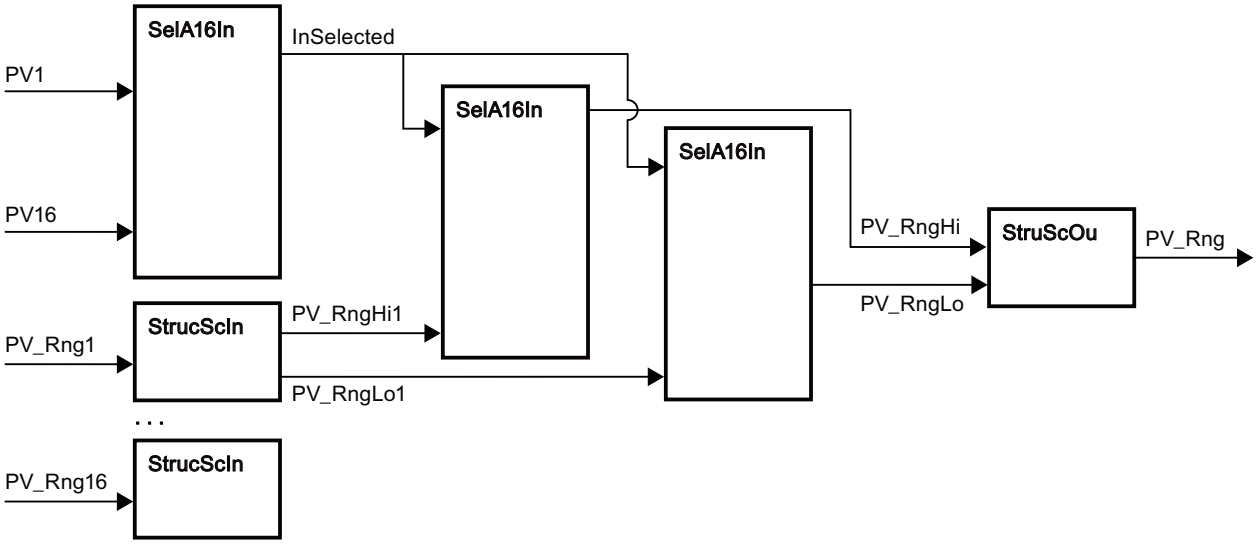


Figure 13-1 Process value area can be realized using conversion blocks and cascaded SelA16In blocks.

See also

SelA16In block diagram (Page 1944)

SelA16In I/Os (Page 1940)

SelA16In messaging (Page 1939)

SelA16In error handling (Page 1939)

SelA16In modes (Page 1935)

Description of SelA16In (Page 1934)

13.8.4 SelA16In error handling

Error handling of SelA16In

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
18	1 > SelExt > 16

See also

SelA16In block diagram (Page 1944)

SelA16In I/Os (Page 1940)

SelA16In messaging (Page 1939)

SelA16In functions (Page 1936)

SelA16In modes (Page 1935)

Description of SelA16In (Page 1934)

13.8.5 SelA16In messaging

Messaging

This block does not offer messaging.

See also

SelA16In block diagram (Page 1944)

SelA16In I/Os (Page 1940)

SelA16In error handling (Page 1939)

SelA16In functions (Page 1936)

SelA16In modes (Page 1935)

Description of SelA16In (Page 1934)

13.8.6 SelA16In I/Os

I/Os of SelA16In

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`Feature`	I/O for additional functions (Page 1936)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
`In01*`	Analog input parameter 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
`In01HiLim`	High limit for the parameter `In01`	REAL	100.0
`In01LoLim`	Low limit for the parameter `In01`	REAL	0.0
`In02`	Analog input parameter 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
`In03`	Analog input parameter 3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
`In04`	Analog input parameter 4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
`In05`	Analog input parameter 5	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
`In06`	Analog input parameter 6	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
`In07`	Analog input parameter 7	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF

Parameter	Description	Type	Default
In08	Analog input parameter 8	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
In09	Analog input parameter 9	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
In10	Analog input parameter 10	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
In11	Analog input parameter 11	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
In12	Analog input parameter 12	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
In13	Analog input parameter 13	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
In14	Analog input parameter 14	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
In15	Analog input parameter 15	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
In16	Analog input parameter 16	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#FF
In01Unit	Unit for parameter In01	INT	1001
In02Unit	Unit for parameter In02	INT	1001
In03Unit	Unit for parameter In03	INT	1001
In04Unit	Unit for parameter In04	INT	1001
In05Unit	Unit for parameter In05	INT	1001
In06Unit	Unit for parameter In06	INT	1001
In07Unit	Unit for parameter In07	INT	1001
In08Unit	Unit for parameter In08	INT	1001
In09Unit	Unit for parameter In09	INT	1001
In10Unit	Unit for parameter In10	INT	1001
In11Unit	Unit for parameter In11	INT	1001
In12Unit	Unit for parameter In12	INT	1001
In13Unit	Unit for parameter In13	INT	1001
In14Unit	Unit for parameter In14	INT	1001

Parameter	Description	Type	Default
In15Unit	Unit for parameter In15	INT	1001
In16Unit	Unit for parameter In16	INT	1001
LiOp	• With Feature.Bit6 = 0: Selection of the input parameter via: – 0 = Operator (SelInt) – 1 = Interconnection or SFC (SelExt) • With Feature.Bit6 = 1: Internal/external selection via: – 0 = Operator (SelIntOp or SelExtOp) – 1 = Interconnection or SFC (SelIntLi or SelExtLi)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OnOp*	1 = "On" mode via operator	BOOL	0
OosLi	1 = "Out of service", via interconnection or SFC (0-1 edge transition)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosOp*	1 = "Out of service", via OS operator	BOOL	0
OpSt_In	Input parameter for local operating permission, connected with the Out output parameter of the upstream block, OpStations (Page 399)	DWORD	16#00000000
OS_Perm	I/O for operator control permissions (Page 1936)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 1 • 1 • 1
SelExt	External selection of the input parameter: 1 = In01 selected 16 = In16 selected Selection outside the range 1...16 is not possible	INT	1
SelFp1	Call a block saved in this parameter as an additional faceplate (Page 203) in the standard view	ANY	
SelFp2	Call a block saved in this parameter as an additional faceplate (Page 203) in the preview	ANY	
SelInt*	Selecting the input parameter: 1 = In01 selected 16 = In16 selected You cannot select anything outside the range of 1 to 16.	INT	1
SelExtLi	Linkable input for external selection: 1 = External selection Active only with Feature.Bit6 = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SelExtOp	Operator input for external selection: 1 = External selection Active only with Feature.Bit6 = 1	BOOL	0

Parameter	Description	Type	Default
SelIntLi	Linkable input for internal selection: 1 = Internal selection Active only with Feature.Bit6 = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SelIntOp	Operator Input for Internal Selection: 1 = Internal selection Active only with Feature.Bit6 = 1	BOOL	0
UserStatus	Freely assignable bits for use in PCS 7 OS	BYTE	16#00

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of current error number. For error numbers that can be output by this block, see SelA16In error handling (Page 1939).	INT	-1
InSelected	Selected input parameter: 1 = In01 selected 16 = In16 selected	STRUCT • Value: INT • ST: BYTE	- • 1 • 16#80
OnAct	1 = "On" mode enabled	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
OosAct	1 = Block is "Out of service"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpSt_Out	Value of the OpSt_In input parameter, for feedforwarding to other blocks. Bit 31 of this parameter is used by Feature bit 24	DWORD	16#00000000
OS_PermLog	Display of OS_Perm with settings changed by the block algorithm	DWORD	16#FFFFFFFF
OS_PermOut	Display of OS_Perm	DWORD	16#FFFFFFFF
Out	Output	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
OutUnit	Unit for parameter Out	INT	1001
SelExtAct	0 = Internal selection is active 1 = External selection is active	BOOL	0
ST_Worst	Worst signal status	BYTE	16#80
Status	Status word (Page 1934)	DWORD	16#00000000
Status2	Status word (Page 1934)	DWORD	16#00000000

See also

SelA16In block diagram (Page 1944)
SelA16In messaging (Page 1939)
SelA16In modes (Page 1935)

13.8.7 SelA16In block diagram

SelA16In block diagram

A block diagram is not provided for this block.

See also

SelA16In I/Os (Page 1940)
SelA16In messaging (Page 1939)
SelA16In error handling (Page 1939)
SelA16In functions (Page 1936)
SelA16In modes (Page 1935)
Description of SelA16In (Page 1934)

13.8.8 Operator control and monitoring

13.8.8.1 SelA16In views

Views of the SelA16In block

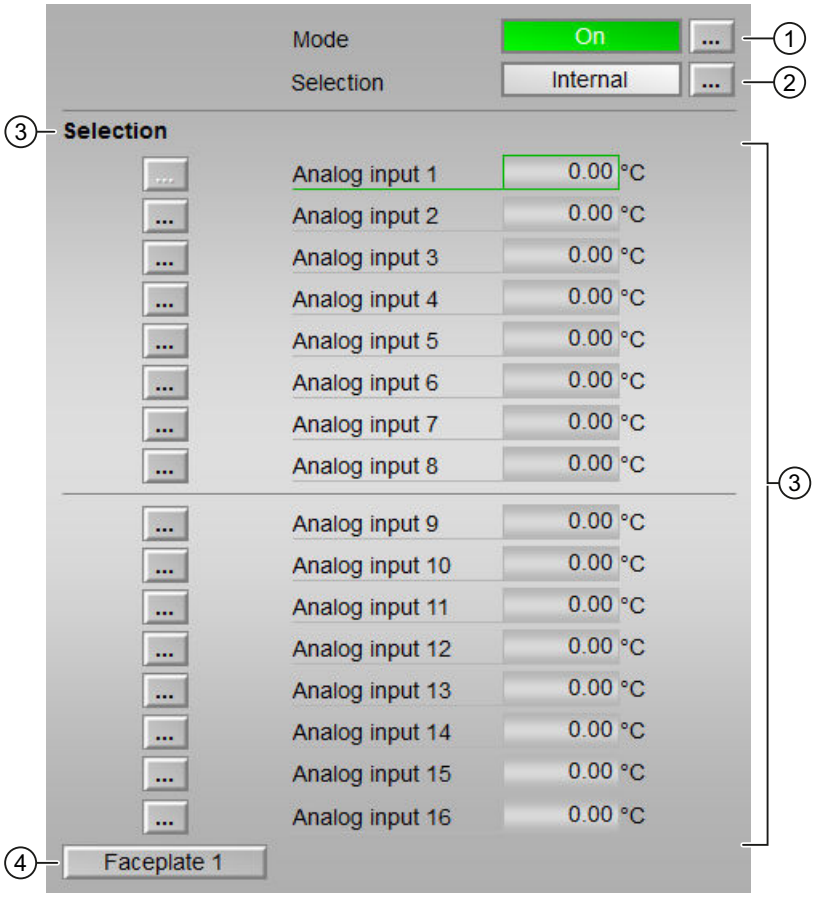
The block SelA16In provides the following views:

- SelA16In standard view (Page 1945)
- SelA16In preview (Page 1946)
- Memo view (Page 298)
- Block icon for SelA16In (Page 1947)

Refer to the sections Structure of the faceplate (Page 242) and Block icon structure (Page 226) for general information on the faceplate and block icon.

13.8.8.2 SelA16In standard view

SelA16In standard view



(1) Displaying and switching the operating mode

This area provides information on the currently valid operating mode. The following operating modes can be shown here:

- On (Page 71)
- Out of service (Page 71)

Refer to the Switching operating states and operating modes (Page 251) section for information on switching the operating mode.

(2) Displaying and switching the selection

This area provides information on the current selection. The following selection can be made:

- External: Selecting values externally (program)
- Internal: Selecting values internally (operator)

(3) Switching analog values

This area shows you the analog values connected in the ES for this block.

You can find additional information on this in Switching operating states and operating modes (Page 251).

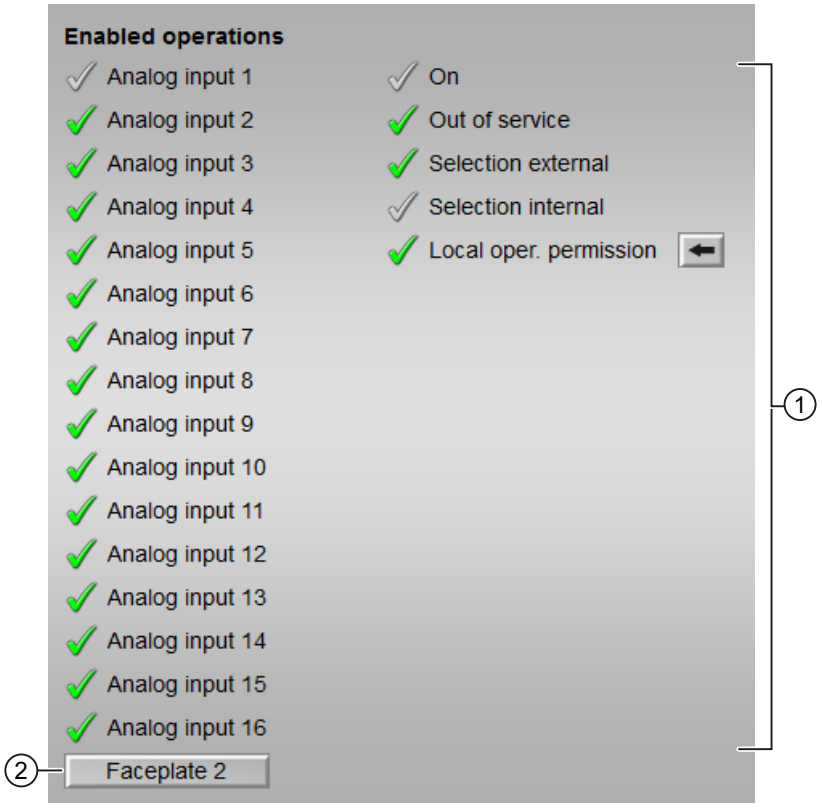
(4) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the Opening additional faceplates (Page 203) section.

13.8.8.3 SelA16In preview

Preview of SelA16In



(1) Enabled operations

This area shows all operations for which special operator control permissions are assigned. They depend on the configuration in the engineering system (ES) that applies to this block.

Symbols for enabled operations:

- **Green check mark**: the OS operator can control this parameter
- **Gray check mark**: the OS operator cannot control this parameter at this time due to the process
- **Red cross**: the OS operator cannot control this parameter due to the configured AS operator control permissions (`OS_Perm` or `OS1Perm`)

The following enabled operations are shown here:

- Analog input 1 to 16: You can switch to this analog input.
- "On": You can switch to "On" operating mode.
- "Out of service": You can switch to "Out of service" operating mode.
- "Selection external": You can select values externally (program).
- "Selection internal": You can select values internally (operator).
- "Local operating permission": Use the ← button to switch to the standard view of the OpStations block. Additional information is available in the section Operator control permissions (Page 248).

(2) Navigation button for switching to the standard view of any faceplate

Use this navigation button to reach the standard view of a block configured in the engineering system. The visibility of this navigation button depends on the configuration in the engineering system (ES).

You can find additional information on this in the Opening additional faceplates (Page 203) section.

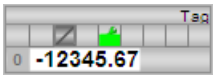
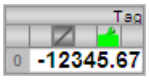
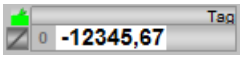
13.8.8.4 Block icon for SelA16In

Block icons for SelA16In

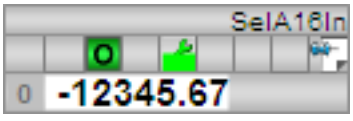
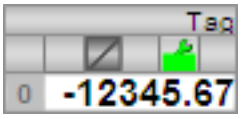
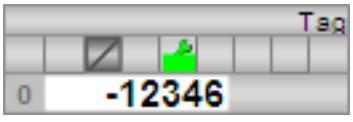
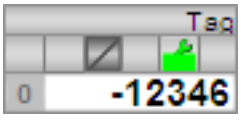
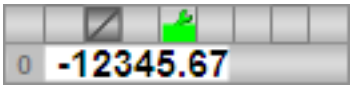
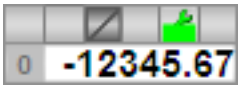
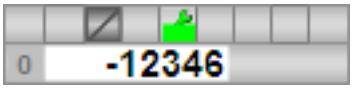
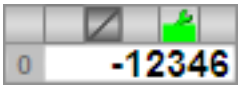
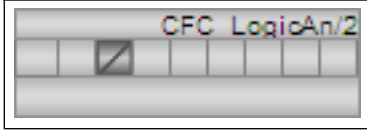
A variety of block icons are available with the following functions:

- Process tag type
- Operating modes
- Signal status, release for maintenance
- Memo display
- Display of the selected analog value

The block icons from template @TemplateAPLV8.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	
	2	
	3	Block icon in the full display

The block icons from template @TemplateAPLV7.PDL:

Icons	Selection of the block icon in CFC	Special features
	1	Block icon in the full display
	2	
	3	
	4	
	5	
	6	
	7	
	8	
	-	Block icon in "Out of service" mode (example with type 1 block icon)

Note

The display of the unit in the types 1/3/5/7 depends on the analog value used, although the unit can only be displayed correctly when the units of the inputs have been configured with a code (see "Selecting a unit of measure (Page 207)"). Units configured with the S7_unit attribute are not supported.

Additional information on the block icon and the control options in the block icon is available in the following sections:

- Configuring the block icons (Page 233)
- Block icon structure (Page 226)
- Operation via the block icon (Page 234).

Digital logic blocks 14

14.1 And04 - Forming an AND signal from 4 binary input signals

14.1.1 Description of And04

Object name (type + number) and family

Type + number: FC 355

Family: LogicDi

Area of application for And04

The block is used for the following applications:

- Forming an AND-output signal from four binary input values

How it works

Four input parameters are combined via the AND function ("and-ing") into one output value `Out`.

You can use this block to e.g. start or stop a device if all incoming signals are the same.

You can find additional information on forming the signal status under Forming and outputting the signal status of digital logic blocks (Page 110)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

And04 block diagram (Page 1955)

And04 I/Os (Page 1954)

And04 messaging (Page 1953)
And04 error handling (Page 1953)
And04 functions (Page 1952)
And04 modes (Page 1952)

14.1.2 And04 modes

And04 modes

This block does not have any modes.

See also

And04 block diagram (Page 1955)
And04 I/Os (Page 1954)
And04 messaging (Page 1953)
And04 error handling (Page 1953)
And04 functions (Page 1952)
Description of And04 (Page 1951)

14.1.3 And04 functions

Functions of And04

This block does not have any other functions.

See also

And04 block diagram (Page 1955)
And04 I/Os (Page 1954)
And04 messaging (Page 1953)
And04 error handling (Page 1953)
And04 modes (Page 1952)
Description of And04 (Page 1951)

14.1.4 And04 error handling

And04 error handling

The block does not report any errors.

See also

And04 block diagram (Page 1955)
And04 I/Os (Page 1954)
And04 messaging (Page 1953)
And04 functions (Page 1952)
And04 modes (Page 1952)
Description of And04 (Page 1951)

14.1.5 And04 messaging

Messaging

This block does not offer messaging.

See also

And04 block diagram (Page 1955)
And04 I/Os (Page 1954)
And04 error handling (Page 1953)
And04 functions (Page 1952)
And04 modes (Page 1952)
Description of And04 (Page 1951)

14.1.6 And04 I/Os

And04 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Input 1	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In2	Input 2	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In3	Input 3	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In4	Input 4	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

And04 block diagram (Page 1955)
And04 messaging (Page 1953)
And04 error handling (Page 1953)
And04 functions (Page 1952)
And04 modes (Page 1952)
Description of And04 (Page 1951)

14.1.7 And04 block diagram

And04 block diagram

A block diagram is not provided for this block.

See also

And04 I/Os (Page 1954)
And04 messaging (Page 1953)
And04 error handling (Page 1953)
And04 functions (Page 1952)
And04 modes (Page 1952)
Description of And04 (Page 1951)

14.2 And08 - Forming an AND signal from 8 binary input signals

14.2.1 Description of And08

Object name (type + number) and family

Type + number: FC 356

Family: LogicDi

Area of application for And08

The block is used for the following applications:

- Forming an AND- output signal from eight binary input values

How it works

Eight input parameters are combined via the AND function ("and-ing") into one output value `Out`.

You can use this block to e.g. start or stop a device if all incoming signals are the same.

You can find additional information on forming the signal status under Forming and outputting the signal status of digital logic blocks (Page 110)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

And08 block diagram (Page 1960)

And08 I/Os (Page 1958)

And08 messaging (Page 1958)

And08 error handling (Page 1957)

And08 functions (Page 1957)

And08 modes (Page 1957)

14.2.2 And08 modes

And08 modes

This block does not have any modes.

See also

And08 block diagram (Page 1960)
And08 I/Os (Page 1958)
And08 messaging (Page 1958)
And08 error handling (Page 1957)
And08 functions (Page 1957)
Description of And08 (Page 1956)

14.2.3 And08 functions

Functions of And08

This block does not have any other functions.

See also

And08 block diagram (Page 1960)
And08 I/Os (Page 1958)
And08 messaging (Page 1958)
And08 error handling (Page 1957)
And08 modes (Page 1957)
Description of And08 (Page 1956)

14.2.4 And08 error handling

And08 error handling

The block does not report any errors.

See also

And08 block diagram (Page 1960)
And08 I/Os (Page 1958)
And08 messaging (Page 1958)
And08 functions (Page 1957)
And08 modes (Page 1957)
Description of And08 (Page 1956)

14.2.5 And08 messaging

Messaging

This block does not offer messaging.

See also

And08 block diagram (Page 1960)
And08 I/Os (Page 1958)
And08 error handling (Page 1957)
And08 functions (Page 1957)
And08 modes (Page 1957)
Description of And08 (Page 1956)

14.2.6 And08 I/Os

And08 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Input 1	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In2	Input 2	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80

Parameter	Description	Type	Default
In3	Input 3	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In4	Input 4	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In5	Input 5	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In6	Input 6	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In7	Input 7	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
In8	Input 8	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

And08 block diagram (Page 1960)
And08 messaging (Page 1958)
And08 error handling (Page 1957)
And08 functions (Page 1957)
And08 modes (Page 1957)
Description of And08 (Page 1956)

14.2.7 And08 block diagram

And08 block diagram

A block diagram is not provided for this block.

See also

And08 I/Os (Page 1958)
And08 messaging (Page 1958)
And08 error handling (Page 1957)
And08 functions (Page 1957)
And08 modes (Page 1957)
Description of And08 (Page 1956)

14.3 FlipFlop - preparation of a bistabile flip-flop

14.3.1 Description of FlipFlop

Object name (type + number) and family

Type + number: FC 389

Family: LogicDi

Area of application for FlipFlop

The block is used for the following applications:

- Preparation of a bistabile flip-flop

How it works

The block provides the function of a bistabile flip-flop, whereby the Mode input parameter can be used to select between SR- (`Mode = 0`) and RS-FlipFlop (`Mode = 1`).

The flip-flop provides the two control inputs for setting and resetting:

- `SetLi`: Set
- `RstLi`: Reset

With a positive edge at the input parameter for setting `SetLi`, the output parameter `Out` is set to 1. Simultaneously, the output parameter `InvOut` is reset.

With a positive edge at the input parameter for resetting `RstLi`, the output parameter `Out` is reset. Simultaneously, the output parameter `InvOut` is set.

If the two input parameters are`SetLi` and `RstLi = 0`, then the block retains its state. In this case, nothing changes at the output parameters.

How the block operates as SR-FlipFlop (`Mode` = 0)

The input parameter for setting `SetLi` has priority over the input parameter for resetting `RstLi`.

Truth table:

RstLi	SetLi	Out	InvOut
0	0	Last value	Last value
0	1	1	0
1	0	0	1
1	1	1	0

How the block operates as RS-FlipFlop (`Mode` = 1)

The input parameter for resetting `RstLi` has priority over the input parameter for setting `SetLi`.

Truth table:

RstLi	SetLi	Out	InvOut
0	0	Last value	Last value
0	1	1	0
1	0	0	1
1	1	0	1

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

A startup behavior can be reached with a connection of the input parameter `Init` to the output parameter `InitOut` of the Trigger block. For more information, refer to Startup characteristics over Trigger block (Page 69).

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

FlipFlop modes (Page 1962)
FlipFlop functions (Page 1963)
FlipFlop error handling (Page 1964)
FlipFlop messaging (Page 1965)
FlipFlop I/Os (Page 1965)
FlipFlop block diagram (Page 1966)

14.3.2 FlipFlop modes

FlipFlop operating modes

This block does not have any modes.

See also

Description of FlipFlop (Page 1961)
FlipFlop functions (Page 1963)
FlipFlop error handling (Page 1964)
FlipFlop messaging (Page 1965)
FlipFlop I/Os (Page 1965)
FlipFlop block diagram (Page 1966)

14.3.3 FlipFlop functions

Functions of FlipFlop

The block provides the following functions:

Forming the signal status for blocks – SR-FlipFlop

The signal status is formed as follows:

RstLi	SetLi	Out.ST	InvOut.ST
0	0	Last signal status	Last signal status
0	1	= SetLi.ST	= SetLi.ST
1	0	= RstLi.ST	= RstLi.ST
1	1	= SetLi.ST	= SetLi.ST

Forming the signal status for blocks - RS-FlipFlop

The signal status is formed as follows:

RstLi	SetLi	Out.ST	InvOut.ST
0	0	Last signal status	Last signal status
0	1	= SetLi.ST	= SetLi.ST
1	0	= RstLi.ST	= RstLi.ST
1	1	= RstLi.ST	= RstLi.ST

Initialization

With the input parameter `Init`, the output parameters `Out` and `InvOut` can be initialized:

- `Init` = 0 (default): Normal flip-flop behavior
  If both input parameters `SetLi` and `RstLi` are reset, the output parameters `Out` and `InvOut` will remain unchanged.
- `Init` = 1: Outputs will be initialized as follow:

Mode	Out	InvOut	Out.ST and InvOut.ST
0	0	1	`SetLi.ST`
1	1	0	`RstLi.ST`

To initialize the output parameters `Out` and `InvOut` after a startup, connect the input parameter `Init` with the output parameter `InitOut` of the Trigger block. For more information, refer to Startup characteristics over Trigger block (Page 69).

See also

Description of FlipFlop (Page 1961)

FlipFlop modes (Page 1962)

FlipFlop error handling (Page 1964)

FlipFlop messaging (Page 1965)

FlipFlop I/Os (Page 1965)

FlipFlop block diagram (Page 1966)

14.3.4 FlipFlop error handling

FlipFlop error handling

The block does not report any errors.

See also

Description of FlipFlop (Page 1961)

FlipFlop modes (Page 1962)

FlipFlop functions (Page 1963)

FlipFlop messaging (Page 1965)

FlipFlop I/Os (Page 1965)

FlipFlop block diagram (Page 1966)

14.3.5 FlipFlop messaging

Messaging

This block does not offer messaging.

See also

Description of FlipFlop (Page 1961)
FlipFlop modes (Page 1962)
FlipFlop functions (Page 1963)
FlipFlop error handling (Page 1964)
FlipFlop I/Os (Page 1965)
FlipFlop block diagram (Page 1966)

14.3.6 FlipFlop I/Os

FlipFlop I/Os

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`Init`	Initialize function block	BOOL	0
`Mode`	Specifying how the block works: • 0 = SR-FlipFlop • 1 = RS-FlipFlop	BOOL	0
`RstLi`	1 = Reset via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SetLi`	1 = Set via interconnection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
InvOut	Inverted output signal	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Description of FlipFlop (Page 1961)
FlipFlop modes (Page 1962)
FlipFlop functions (Page 1963)
FlipFlop error handling (Page 1964)
FlipFlop messaging (Page 1965)
FlipFlop block diagram (Page 1966)

14.3.7 FlipFlop block diagram

FlipFlop block diagram

A block diagram is not provided for this block.

See also

Description of FlipFlop (Page 1961)
FlipFlop modes (Page 1962)
FlipFlop functions (Page 1963)
FlipFlop error handling (Page 1964)
FlipFlop messaging (Page 1965)
FlipFlop I/Os (Page 1965)

14.4 Or04 - Forming an OR signal from 4 binary input signals

14.4.1 Description of Or04

Object name (type + number) and family

Type + number: FC 364

Family: LogicDi

Area of application for Or04

The block is used for the following applications:

- Forming an OR-output signal from four binary input values

How it works

Four input parameters are combined via the OR-function ("or-ing") into one output value `Out`.

You can find additional information on forming the signal status under Forming and outputting the signal status of digital logic blocks (Page 110)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

For the Or04 block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Cascade control with control loop monitoring through ConPerMon (CascadeControl) (Page 2316)
- Cascade control with PIDConR (CascadeR) (Page 2319)
- Dosing (Dose_Lean) (Page 2332)
- PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303)
- PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301)
- Model-based predictive control (ModPreCon) (Page 2325)
- Override control (Page 2322)
- Override control with PIDConR (OverrideR) (Page 2324)
- PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298)
- PIDConR with safety logic and control loop monitoring (PIDConR_ConPerMon) (Page 2299)

- Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312)
- Ratio control with PIDConR (RatioR) (Page 2315)
- PID controller with Smith predictor (SmithPredictorControl) (Page 2306)
- Step controller with assigned actuator block and position feedback (StepControlActor) (Page 2308)
- Valve (Valve_Lean) (Page 2341)
- Dosing with PA/FF devices (Dose_Lean_Fb) (Page 2333)
- PID controller for PA/FF devices (PIDControl_Lean_Fb) (Page 2297)
- Control valve (VlvAnL) (Page 2344)
- Control valve for PA/FF devices (ValveAnalog_Fb) (Page 2345)
- Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl) (Page 2309)

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

Or04 block diagram (Page 1971)

Or04 I/Os (Page 1970)

Or04 messaging (Page 1970)

Or04 error handling (Page 1969)

Or04 functions (Page 1969)

Or04 modes (Page 1968)

14.4.2 Or04 modes

Or04 modes

This block does not have any modes.

See also

Or04 block diagram (Page 1971)
Or04 I/Os (Page 1970)
Or04 messaging (Page 1970)
Or04 error handling (Page 1969)
Or04 functions (Page 1969)
Description of Or04 (Page 1967)

14.4.3 Or04 functions

Functions of Or04

This block does not have any other functions.

See also

Or04 block diagram (Page 1971)
Or04 I/Os (Page 1970)
Or04 messaging (Page 1970)
Or04 error handling (Page 1969)
Or04 modes (Page 1968)
Description of Or04 (Page 1967)

14.4.4 Or04 error handling

Or04 error handling

The block does not report any errors.

See also

Or04 block diagram (Page 1971)
Or04 I/Os (Page 1970)
Or04 messaging (Page 1970)
Or04 functions (Page 1969)
Or04 modes (Page 1968)
Description of Or04 (Page 1967)

14.4.5 Or04 messaging

Messaging

This block does not offer messaging.

See also

Or04 block diagram (Page 1971)
Or04 I/Os (Page 1970)
Or04 error handling (Page 1969)
Or04 functions (Page 1969)
Or04 modes (Page 1968)
Description of Or04 (Page 1967)

14.4.6 Or04 I/Os

Or04 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Value 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In2	Value 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In3	Value 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In4	Value 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Or04 block diagram (Page 1971)
Or04 messaging (Page 1970)
Or04 error handling (Page 1969)
Or04 functions (Page 1969)
Or04 modes (Page 1968)
Description of Or04 (Page 1967)

14.4.7 Or04 block diagram

Or04 block diagram

A block diagram is not provided for this block.

See also

Or04 I/Os (Page 1970)
Or04 messaging (Page 1970)
Or04 error handling (Page 1969)
Or04 functions (Page 1969)
Or04 modes (Page 1968)
Description of Or04 (Page 1967)

14.5 Or08 - Forming an OR signal from 8 binary input signals

14.5.1 Description of Or08

Object name (type + number) and family

Type + number: FC 365

Family: LogicDi

Area of application for Or08

The block is used for the following applications:

- Forming an OR- output signal from eight binary input values

How it works

Eight input parameters are combined via the OR-function ("or-ing") into one output value `Out`.

You can find additional information on forming the signal status under Forming and outputting the signal status of digital logic blocks (Page 110)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

For the Or08 block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Monitoring eight digital process tags (Digital8Monitoring) (Page 2329)
- Step controller with assigned actuator block and position feedback (StepControlActor) (Page 2308)
- Step controller with direct access to the actuator and without position feedback (StepControlDirect) (Page 2307)
- Two-speed motor (Motor2Speed) (Page 2336)
- Reversing motor (MotorReversible) (Page 2337)
- Reversible motor with controllable speed (MotorSpeedControlled) (Page 2338)
- Two-way valve (Valve2Way) (Page 2342)
- Motor valve (ValveMotor) (Page 2343)

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

Or08 block diagram (Page 1976)
Or08 I/Os (Page 1975)
Or08 messaging (Page 1974)
Or08 error handling (Page 1974)
Or08 functions (Page 1973)
Or08 modes (Page 1973)

14.5.2 Or08 modes

Or08 modes

This block does not have any operating modes.

See also

Or08 block diagram (Page 1976)
Or08 I/Os (Page 1975)
Or08 messaging (Page 1974)
Or08 error handling (Page 1974)
Or08 functions (Page 1973)
Description of Or08 (Page 1972)

14.5.3 Or08 functions

Functions of Or08

This block does not have any other functions.

See also

Or08 block diagram (Page 1976)
Or08 I/Os (Page 1975)

Or08 messaging (Page 1974)
Or08 error handling (Page 1974)
Or08 modes (Page 1973)
Description of Or08 (Page 1972)

14.5.4 Or08 error handling

Or08 error handling

The block does not report any errors.

See also

Or08 block diagram (Page 1976)
Or08 I/Os (Page 1975)
Or08 messaging (Page 1974)
Or08 functions (Page 1973)
Or08 modes (Page 1973)
Description of Or08 (Page 1972)

14.5.5 Or08 messaging

Messaging

This block does not offer messaging.

See also

Or08 block diagram (Page 1976)
Or08 I/Os (Page 1975)
Or08 error handling (Page 1974)
Or08 functions (Page 1973)
Or08 modes (Page 1973)
Description of Or08 (Page 1972)

14.5.6 Or08 I/Os

Or08 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Value 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In2	Value 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In3	Value 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In4	Value 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In5	Value 5	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In6	Value 6	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In7	Value 7	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In8	Value 8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Or08 block diagram (Page 1976)
Or08 messaging (Page 1974)
Or08 error handling (Page 1974)
Or08 functions (Page 1973)
Or08 modes (Page 1973)
Description of Or08 (Page 1972)

14.5.7 Or08 block diagram

Or08 block diagram

A block diagram is not provided for this block.

See also

Or08 I/Os (Page 1975)
Or08 messaging (Page 1974)
Or08 error handling (Page 1974)
Or08 functions (Page 1973)
Or08 modes (Page 1973)
Description of Or08 (Page 1972)

14.6 Not01 - Inversion of an input signal

14.6.1 Description of Not01

Object name (type + number) and family

Type + number: FC 382

Family: LogicDi

Area of application for Not01

The block is used for the following applications:

- Inversion of an input signal

How it works

The block inverts the binary signal available at the `In` input parameter and writes the result to its output parameter `Out`.

The signal status is passed from the input directly to the output.

You can find additional information on forming the signal status under Forming and outputting the signal status of digital logic blocks (Page 110)

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

For the Not01 block, the Advanced Process Library contains a template for process tag types as an example with an application scenario for this block.

Example of process tag types:

- Step controller with assigned actuator block and position feedback (StepControlActor) (Page 2308)
- Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312)

See also

Not01 block diagram (Page 1980)

Not01 I/Os (Page 1979)

Not01 messaging (Page 1979)

Not01 error handling (Page 1978)

Not01 functions (Page 1978)

Not01 modes (Page 1978)

14.6.2 Not01 modes

Not01 operating modes

This block does not have any operating modes.

See also

Not01 block diagram (Page 1980)
Not01 I/Os (Page 1979)
Not01 messaging (Page 1979)
Not01 error handling (Page 1978)
Not01 functions (Page 1978)
Description of Not01 (Page 1977)

14.6.3 Not01 functions

Functions of Not01

There are no other functions for this block.

See also

Not01 block diagram (Page 1980)
Not01 I/Os (Page 1979)
Not01 messaging (Page 1979)
Not01 error handling (Page 1978)
Not01 modes (Page 1978)
Description of Not01 (Page 1977)

14.6.4 Not01 error handling

Not01 error handling

The block does not report any errors.

See also

Not01 block diagram (Page 1980)
Not01 I/Os (Page 1979)
Not01 messaging (Page 1979)
Not01 functions (Page 1978)
Not01 modes (Page 1978)
Description of Not01 (Page 1977)

14.6.5 Not01 messaging

Messaging

This block does not offer messaging.

See also

Not01 block diagram (Page 1980)
Not01 I/Os (Page 1979)
Not01 error handling (Page 1978)
Not01 functions (Page 1978)
Not01 modes (Page 1978)
Description of Not01 (Page 1977)

14.6.6 Not01 I/Os

Not01 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In	Input value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Not01 block diagram (Page 1980)
Not01 messaging (Page 1979)
Not01 error handling (Page 1978)
Not01 functions (Page 1978)
Not01 modes (Page 1978)
Description of Not01 (Page 1977)

14.6.7 Not01 block diagram

Not01 block diagram

A block diagram is not provided for this block.

See also

Not01 I/Os (Page 1979)
Not01 messaging (Page 1979)
Not01 error handling (Page 1978)
Not01 functions (Page 1978)
Not01 modes (Page 1978)
Description of Not01 (Page 1977)

14.7 RedDi02 - 1 out of 2 selection for redundant digital values

14.7.1 Description of RedDi02

Object name (type + number) and family

Type + number: FC 386

Family: LogicDi

Area of application for RedDi02

The block is used for the following applications:

- 1 out of 2 selection for redundant digital values

How it works

The block selects from two input values the one with the best signal status and outputs it at the output `Out`. In addition, the outputs `SimAct`, `Uncertain` and `LossRed` are set according to the signal status.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

RedDi02 modes (Page 1982)

RedDi02 functions (Page 1982)

RedDi02 error handling (Page 1983)

RedDi02 messaging (Page 1983)

RedDi02 I/Os (Page 1984)

RedDi02 block diagram (Page 1985)

14.7.2 RedDi02 modes

RedDi02 operating modes

This block does not have any operating modes.

See also

Description of RedDi02 (Page 1981)
RedDi02 functions (Page 1982)
RedDi02 error handling (Page 1983)
RedDi02 messaging (Page 1983)
RedDi02 I/Os (Page 1984)
RedDi02 block diagram (Page 1985)

14.7.3 RedDi02 functions

Functions of RedDi02

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status of redundancy blocks (Page 112).

See also

Description of RedDi02 (Page 1981)
RedDi02 modes (Page 1982)
RedDi02 error handling (Page 1983)
RedDi02 messaging (Page 1983)
RedDi02 I/Os (Page 1984)
RedDi02 block diagram (Page 1985)

14.7.4 RedDi02 error handling

RedDi02 error handling

The block does not report any errors.

See also

Description of RedDi02 (Page 1981)
RedDi02 modes (Page 1982)
RedDi02 functions (Page 1982)
RedDi02 messaging (Page 1983)
RedDi02 I/Os (Page 1984)
RedDi02 block diagram (Page 1985)

14.7.5 RedDi02 messaging

Messaging

This block does not offer messaging.

See also

Description of RedDi02 (Page 1981)
RedDi02 modes (Page 1982)
RedDi02 functions (Page 1982)
RedDi02 error handling (Page 1983)
RedDi02 I/Os (Page 1984)
RedDi02 block diagram (Page 1985)

14.7.6 RedDi02 I/Os

I/Os of RedDi02

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Digital input value 1	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80
In2	Digital input value 2	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
LossRed	1= Redundancy loss at one of the inputs	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80
Out	Output of the process value with the better signal status	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80
SimAct	1 = one input value has the Simulation status	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80
Uncertain	1 = one input value has the "uncertain" status	STRUCT • Value: BOOL • ST:BYTE	- • 0 • 16#80

See also

Description of RedDi02 (Page 1981)
RedDi02 modes (Page 1982)
RedDi02 functions (Page 1982)
RedDi02 error handling (Page 1983)
RedDi02 messaging (Page 1983)
RedDi02 block diagram (Page 1985)

14.7.7 RedDi02 block diagram

RedDi02 block diagram

A block diagram is not provided for this block.

See also

Description of RedDi02 (Page 1981)
RedDi02 modes (Page 1982)
RedDi02 functions (Page 1982)
RedDi02 error handling (Page 1983)
RedDi02 messaging (Page 1983)
RedDi02 I/Os (Page 1984)

14.8 SelD02In - Output of one of two digital signals

14.8.1 Description of SelD02In

Object name (type + number) and family

Type + number: FC 391

Family: LogicDi

Area of application for SelD02In

The block is used for the following applications:

- Selection from two digital values

How it works

Depending on the value of the input `Sel_In2`, the block switches the value of the input `In1` or the input `In2` to the output `Out`.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

SelD02In modes (Page 1987)

SelD02In functions (Page 1987)

SelD02In error handling (Page 1988)

SelD02In messaging (Page 1988)

SelD02In I/Os (Page 1989)

SelD02In block diagram (Page 1990)

14.8.2 SelD02In modes

SelD02In operating modes

This block does not have any operating modes.

See also

Description of SelD02In (Page 1986)
SelD02In functions (Page 1987)
SelD02In error handling (Page 1988)
SelD02In messaging (Page 1988)
SelD02In I/Os (Page 1989)
SelD02In block diagram (Page 1990)

14.8.3 SelD02In functions

Functions of SelD02In

The functions for this block are listed below.

Select input parameter

You can use the parameter `Sel_In2` to specify whether the input parameter `In1` or `In2` is to be output at the output parameter:

- `Sel_In2` = 0: Input parameter `In1` is written with its signal status to output parameter `Out`.
- `Sel_In2` = 1: Input parameter `In2` is written with its signal status to output parameter `Out`.

Display of selected value

The output parameter `In2Selected` indicates which of the two input parameters has just been output:

- `In2Selected` = 0: At the output parameter `Out`, the value of the input parameter `In1` is output.
- `In2Selected` = 1: At the output parameter `Out`, the value of the input parameter `In2` is output.

The signal status of `Sel_In2` is output at the `In2Selected` output parameter.

See also

Description of SelD02In (Page 1986)
SelD02In modes (Page 1987)
SelD02In error handling (Page 1988)
SelD02In messaging (Page 1988)
SelD02In I/Os (Page 1989)
SelD02In block diagram (Page 1990)

14.8.4 SelD02In error handling

SelD02In error handling

The block does not report any errors.

See also

Description of SelD02In (Page 1986)
SelD02In modes (Page 1987)
SelD02In functions (Page 1987)
SelD02In messaging (Page 1988)
SelD02In I/Os (Page 1989)
SelD02In block diagram (Page 1990)

14.8.5 SelD02In messaging

Messaging

This block does not offer messaging.

See also

Description of SelD02In (Page 1986)
SelD02In modes (Page 1987)
SelD02In functions (Page 1987)
SelD02In error handling (Page 1988)
SelD02In I/Os (Page 1989)
SelD02In block diagram (Page 1990)

14.8.6 SelD02In I/Os

SelD02In I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Input 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In2	Input 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Sel_In2	Selecting the input parameter: 0 = In1 1 = In2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
In2Selected	Selected input parameter: 0 = In1 1 = In2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Description of SelD02In (Page 1986)
SelD02In modes (Page 1987)
SelD02In functions (Page 1987)
SelD02In error handling (Page 1988)
SelD02In messaging (Page 1988)
SelD02In block diagram (Page 1990)

14.8.7 SelD02In block diagram

SelD02In block diagram

A block diagram is not provided for this block.

See also

Description of SelD02In (Page 1986)
SelD02In modes (Page 1987)
SelD02In functions (Page 1987)
SelD02In error handling (Page 1988)
SelD02In messaging (Page 1988)
SelD02In I/Os (Page 1989)

14.9 StrctCom - Structure composer for 32-bit structures

14.9.1 Description of StrctCom

Object name (type + number) and family

Type + number: FC 393

Family: LogicDi

Area of application for StrctCom

The block forms a structure type output with 32 binary structural elements from 32 binary input signals. The structure corresponds, for example, to the OS_Perm inputs of the technological blocks and can be interconnected to them.

How it works

The block is used for create a structure of 32 input bits.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

StrctCom block diagram (Page 1994)

StrctCom I/Os (Page 1993)

StrctCom messaging (Page 1993)

StrctCom error handling (Page 1993)

StrctCom functions (Page 1992)

StrctCom modes (Page 1992)

Forming and outputting the signal status of digital logic blocks (Page 110)

14.9.2 StrctCom modes

StrctCom operating modes

This block does not have any operating modes.

See also

StrctCom block diagram (Page 1994)
StrctCom I/Os (Page 1993)
StrctCom messaging (Page 1993)
StrctCom error handling (Page 1993)
StrctCom functions (Page 1992)
Description of StrctCom (Page 1991)

14.9.3 StrctCom functions

Functions of StrctCom

The functions for this block are listed below.

Any 32 bits at the input are assigned to the structure in the output.

```
Out.Bit0 = Bit0
Out.Bit1 = Bit1
  .....
Out.Bit31 = Bit31
```

See also

StrctCom block diagram (Page 1994)
StrctCom I/Os (Page 1993)
StrctCom messaging (Page 1993)
StrctCom error handling (Page 1993)
StrctCom modes (Page 1992)
Description of StrctCom (Page 1991)

14.9.4 StrctCom error handling

Error handling of StrctCom

The block does not report any errors.

See also

StrctCom block diagram (Page 1994)
StrctCom I/Os (Page 1993)
StrctCom messaging (Page 1993)
StrctCom functions (Page 1992)
StrctCom modes (Page 1992)
Description of StrctCom (Page 1991)

14.9.5 StrctCom messaging

Messaging

This block does not offer messaging.

See also

StrctCom block diagram (Page 1994)
StrctCom I/Os (Page 1993)
StrctCom error handling (Page 1993)
StrctCom functions (Page 1992)
StrctCom modes (Page 1992)
Description of StrctCom (Page 1991)

14.9.6 StrctCom I/Os

I/Os of StrctCom

Input parameters

Parameter	Description	Type	Default
`Bit0...Bit31`	Binary input value Bit0...Bit 31	BOOL	0
`EN`	1 = Called block will be processed	BOOL	1
`IN`	Binary input value • Bit 0 .. 31	 • BOOL	

Output parameters

Parameter	Description	Type	Default
`ENO`	1 = Block algorithm completed without errors	BOOL	0
`Out`	Structure of 0...31 binary signals	STRUCT • Bit0: BOOL • ... • Bit31: BOOL	- • 0

See also

StrctCom block diagram (Page 1994)
StrctCom messaging (Page 1993)
StrctCom error handling (Page 1993)
StrctCom functions (Page 1992)
StrctCom modes (Page 1992)
Description of StrctCom (Page 1991)

14.9.7 StrctCom block diagram

Block diagram of StrctCom

A block diagram is not provided for this block.

See also

StrctCom I/Os (Page 1993)
StrctCom messaging (Page 1993)
StrctCom error handling (Page 1993)
StrctCom functions (Page 1992)

StrctCom modes (Page 1992)
Description of StrctCom (Page 1991)

14.10 StrctDeC - Structure decomposer for 32-bit structures

14.10.1 Description of StrctDeC

Object name (type + number) and family

Type + number: FC 396

Family: LogicDi

Area of application for StrctDeC

The block forms a DWORD type output and 32 single binary outputs from a structure type input with 32 binary structural elements. The input structure corresponds, for example, to the `OS_Perm` inputs of the technological blocks. If you use an `OS_Perm` input in a CFC source chart, you can use this block to transform the structured input in separate binary values inside the CFC source chart. For more information, refer to the Source chart for GainSched function block (gain scheduling) (Page 2321).

How it works

The block is used to create a DWORD or single binary outputs from the structure of 32 input bits.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

StrctDeC modes (Page 1997)

StrctDeC functions (Page 1997)

StrctDeC error handling (Page 1998)

StrctDeC messaging (Page 1998)

StrctDeC I/Os (Page 1998)

StrctDeC block diagram (Page 1999)

14.10.2 StrctDeC modes

StrctDeC operating modes

This block does not have any operating modes.

See also

Description of StrctDeC (Page 1996)
StrctDeC functions (Page 1997)
StrctDeC error handling (Page 1998)
StrctDeC messaging (Page 1998)
StrctDeC I/Os (Page 1998)
StrctDeC block diagram (Page 1999)

14.10.3 StrctDeC functions

Functions of StrctDeC

The functions for this block are listed below.

The input structure `In` is assigned to a DWORD output and to 32 single binary outputs.

```
OutDW = Structure In
Bit0 = In.Bit0
Bit1 = In.Bit1
...
Bit31 = In.Bit31
```

See also

Description of StrctDeC (Page 1996)
StrctDeC modes (Page 1997)
StrctDeC error handling (Page 1998)
StrctDeC messaging (Page 1998)
StrctDeC I/Os (Page 1998)
StrctDeC block diagram (Page 1999)

14.10.4 StrctDeC error handling

Error handling of StrctDeC

The block does not report any errors.

See also

Description of StrctDeC (Page 1996)
StrctDeC modes (Page 1997)
StrctDeC functions (Page 1997)
StrctDeC messaging (Page 1998)
StrctDeC I/Os (Page 1998)
StrctDeC block diagram (Page 1999)

14.10.5 StrctDeC messaging

Messaging

This block does not offer messaging.

See also

Description of StrctDeC (Page 1996)
StrctDeC modes (Page 1997)
StrctDeC functions (Page 1997)
StrctDeC error handling (Page 1998)
StrctDeC I/Os (Page 1998)
StrctDeC block diagram (Page 1999)

14.10.6 StrctDeC I/Os

I/Os of StrctDeC

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In	Structure of 0...31 binary signals Bit 0...31	STRUCT • Bit0: BOOL • ... • Bit31: BOOL	- • 0 • 0 • 0

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
OutDW	DWORD output value	DWORD	16#00000000
Bit0	Binary output value	Bit0: BOOL	0
...	...	...	...
Bit31	Binary output value	Bit0: BOOL	0

See also

Description of StrctDeC (Page 1996)
StrctDeC modes (Page 1997)
StrctDeC functions (Page 1997)
StrctDeC error handling (Page 1998)
StrctDeC messaging (Page 1998)
StrctDeC block diagram (Page 1999)

14.10.7 StrctDeC block diagram

Block diagram of StrctDeC

A block diagram is not provided for this block.

See also

Description of StrctDeC (Page 1996)
StrctDeC modes (Page 1997)
StrctDeC functions (Page 1997)
StrctDeC error handling (Page 1998)

StrctDeC messaging (Page 1998)
StrctDeC I/Os (Page 1998)

14.11 Trigger - Detection of rising and falling edges

14.11.1 Description of Trigger

Object name (type + number) and family

Type + number: FB 1821

Family: LogicDi

Area of application for Trigger

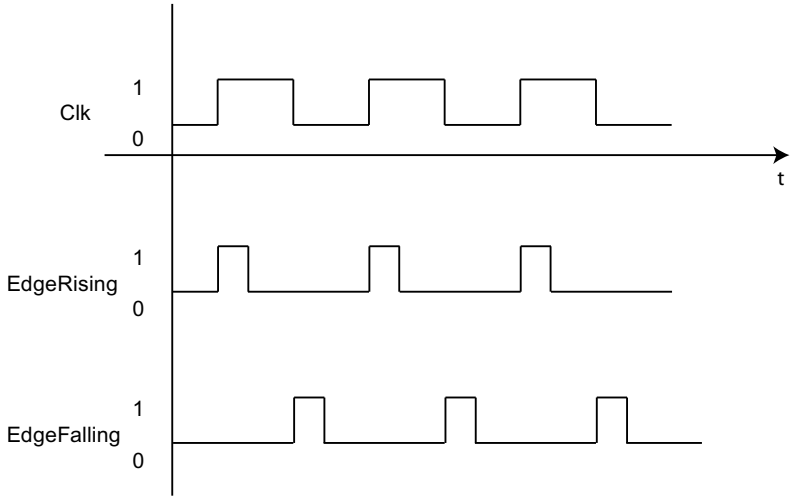
The block is used for the following applications:

- Detection of rising and falling edges

How it works

This block checks the input variable Clk for occurrences of rising and falling edges, and signals them at the outputs `EdgeRising` and `EdgeFalling` .

Output `EdgeRising` is set to 1 for a block cycle at a rising edge. Output `EdgeFalling` is set to 1 for a block cycle at a falling edge.



Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

Edge detection is suppressed at startup or initial startup.

With the input parameter `InitOnRestart`, you can influence the startup behavior of the output parameters `InitOut`, `EdgeRising`, and `EdgeFalling`:

- `InitOnRestart` = 0 (Default):
  The output parameters `EdgeRising` and `EdgeFalling` remain unchanged and the output parameter `InitOut` will reset to false.
- `InitOnRestart` = 1:
  The output parameters `EdgeRising` and `EdgeFalling` will reset to false and the output parameter `InitOut` will set to true. In the second cycle of OB30 to OB38, the output parameter `InitOut` will be reset. You can connect this output parameter with the input parameter `Init` of the block Limit, TimerP, or FlipFlop to initialize these blocks. For more information, refer to Startup characteristics over Trigger block (Page 69).

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Trigger modes (Page 2002)
Trigger functions (Page 2003)
Trigger error handling (Page 2003)
Trigger messaging (Page 2004)
Trigger I/Os (Page 2004)
Trigger block diagram (Page 2005)

14.11.2 Trigger modes

Trigger operating modes

This block does not have any operating modes.

See also

Description of Trigger (Page 2001)
Trigger functions (Page 2003)
Trigger error handling (Page 2003)
Trigger messaging (Page 2004)
Trigger I/Os (Page 2004)
Trigger block diagram (Page 2005)

14.11.3 Trigger functions

Functions of Trigger

This block provides the function:

Forming the signal status for blocks

The `Clk.ST` status is transmitted with each cycle to `EdgeRising.ST` and `EdgeFalling.ST`.

Trigger forms the signal status as follows:

- `EdgeRising.ST` := `Clk.ST`
- `EdgeFalling.ST` := `Clk.ST`

See also

Description of Trigger (Page 2001)
Trigger modes (Page 2002)
Trigger error handling (Page 2003)
Trigger messaging (Page 2004)
Trigger I/Os (Page 2004)
Trigger block diagram (Page 2005)

14.11.4 Trigger error handling

Error handling of Trigger

The block does not report any errors.

See also

Description of Trigger (Page 2001)
Trigger modes (Page 2002)
Trigger functions (Page 2003)
Trigger messaging (Page 2004)
Trigger I/Os (Page 2004)
Trigger block diagram (Page 2005)

14.11.5 Trigger messaging

Messaging

This block does not offer messaging.

See also

Description of Trigger (Page 2001)
Trigger modes (Page 2002)
Trigger functions (Page 2003)
Trigger error handling (Page 2003)
Trigger I/Os (Page 2004)
Trigger block diagram (Page 2005)

14.11.6 Trigger I/Os

SelD02In I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Clk	Input	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
InitOnRestart	1 = Initialization on warm restart	BOOL	0

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
EdgeRising	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
EdgeFalling	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
InitOut	Recognition of warm restart (OB100)	BOOL	0

See also

Description of Trigger (Page 2001)

Trigger modes (Page 2002)

Trigger functions (Page 2003)

Trigger error handling (Page 2003)

Trigger messaging (Page 2004)

Trigger block diagram (Page 2005)

14.11.7 Trigger block diagram

Trigger block diagram

A block diagram is not provided for this block.

See also

Description of Trigger (Page 2001)

Trigger modes (Page 2002)

Trigger functions (Page 2003)

Trigger error handling (Page 2003)

Trigger messaging (Page 2004)

Trigger I/Os (Page 2004)

14.12 XOr04 - EXCLUSIVE OR logic operation

14.12.1 Description of XOr04

Object name (type + number) and family

Type + number: FC 388

Family: LogicDi

Area of application for XOr04

The block is used for the following applications:

- EXCLUSIVE OR logic operation with up to 4 inputs

How it works

The block performs a logic operation with up to four input parameters `In1` to `In4`. The output parameter `Out` then becomes exactly 1, if there is a 1 at an uneven number of inputs and 0 at the rest of them.

Truth table

	In1	In2	In3	In4	Out
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	0

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

XOr04 modes (Page 2007)
XOr04 functions (Page 2007)
XOr04 error handling (Page 2008)
XOr04 messaging (Page 2008)
XOr04 I/Os (Page 2009)
XOr04 block diagram (Page 2010)

14.12.2 XOr04 modes

XOr04 operating modes

This block does not have any operating modes.

See also

Description of XOr04 (Page 2006)
XOr04 functions (Page 2007)
XOr04 error handling (Page 2008)
XOr04 messaging (Page 2008)
XOr04 I/Os (Page 2009)
XOr04 block diagram (Page 2010)

14.12.3 XOr04 functions

Functions of XOr04

The functions for this block are listed below.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status of digital logic blocks (Page 110).

See also

Description of XOr04 (Page 2006)

XOr04 modes (Page 2007)

XOr04 error handling (Page 2008)

XOr04 messaging (Page 2008)

XOr04 I/Os (Page 2009)

XOr04 block diagram (Page 2010)

14.12.4 XOr04 error handling

XOr04 error handling

The block does not report any errors.

See also

Description of XOr04 (Page 2006)

XOr04 modes (Page 2007)

XOr04 functions (Page 2007)

XOr04 messaging (Page 2008)

XOr04 I/Os (Page 2009)

XOr04 block diagram (Page 2010)

14.12.5 XOr04 messaging

Messaging

This block does not offer messaging.

See also

Description of XOr04 (Page 2006)

XOr04 modes (Page 2007)

XOr04 functions (Page 2007)

XOr04 error handling (Page 2008)
XOr04 I/Os (Page 2009)
XOr04 block diagram (Page 2010)

14.12.6 XOr04 I/Os

XOr04 I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Input 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In2	Input 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In3	Input 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
In4	Input 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Output	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Description of XOr04 (Page 2006)
XOr04 modes (Page 2007)
XOr04 functions (Page 2007)
XOr04 error handling (Page 2008)

XOr04 messaging (Page 2008)
XOr04 block diagram (Page 2010)

14.12.7 XOr04 block diagram

XOr04 block diagram

A block diagram is not provided for this block.

See also

Description of XOr04 (Page 2006)
XOr04 modes (Page 2007)
XOr04 functions (Page 2007)
XOr04 error handling (Page 2008)
XOr04 messaging (Page 2008)
XOr04 I/Os (Page 2009)

Generator blocks 15

15.1 NoiseGen - Generating signal noise

15.1.1 Description of NoiseGen

Object name (type + number) and family

Object name: FB 1863

Family: Genrator

Area of application for NoiseGen

The block is used for the following applications:

- Noise generator

The block can only be used for the simulation in the example project.

How it works

This block serves to generate signal noise. It is used for demonstration purposes in the example project so that the simulated signals react naturally.

You can employ it for simulation examples, demonstrations, trade fair models etc. It is never needed in real plants because real measuring devices always supply signals with noise.

There is an example project for the NoiseGen block (APL_Example_xx, xx refers to the language variant) with an application scenario for this block, which explains how the block works.

Application scenario in the example project:

- Process simulation including noise generator (ProcSimC; ProcSimS) (Page 2347)

See also

NoiseGen I/Os (Page 2012)

15.1.2 NoiseGen I/Os

NoiseGen I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Enable	1 = generate noise signals	BOOL	1
Offset	Mean temporal value of the Noise output signal	REAL	20.0
Restart*	1 = block restart	BOOL	1
StdDev	Standard deviation of the noise signal	REAL	1.0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Noise	Generated noise signal	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of NoiseGen (Page 2011)

Channel blocks

16

16.1 Information on using channel blocks

Information on using channel blocks

- The descriptions of the channel blocks specify the OBs in which the blocks are installed. Please note that not all OBs listed will be generated for all CPUs. You can find additional information in the online help of the particular OB.
- If the driver generator uses the channel blocks of the PCS 7 libraries, you require firmware version V3.1 or higher on the CPU.
- The CFC function "Generate module drivers" interconnects and configures the required I/Os automatically. The function is called and executed if hardware modifications are detected when compiling the program, for example.

Signal-processing blocks

Various types of channel blocks are provided in this PCS 7 library for processing signals from inputs and outputs:

1. **Standard channel blocks**
   This applies to the following blocks:
   - Pcs7AnIn
   - Pcs7AnOu
   - Pcs7DiIn
   - Pcs7DiOu
   - Pcs7DiIT

   These blocks are used only for processing the signals of S7-300/400 SM modules. Use these standard blocks if you want to optimize memory and runtime utilization and do not need to process any PA devices.

2. **FF/PA channel blocks**
   This applies to the following blocks:
   - FbAnIn
   - FbAnOu
   - FbAnTot
   - FbDiIn
   - FbDiOu

   These blocks are designed especially for use with PA field devices and the PROFIBUS 3.0 Class A and B or with FF field devices. In particular, you should use these blocks if you want to make use of the special features of these devices. In contrast to standard channel blocks, PA channel blocks not only process the signal itself but also all variables, according to the desired device configuration selected in the hardware configuration.

16.2 FbAnIn - Analog input channel block for field devices

16.2.1 Description of FbAnIn

Object name (type + number) and family

Type + number: FB 1813

Family: Channel

Area of application for FbAnIn

The block is used for the following applications:

- Signal processing (cyclic service) in accordance with "Transmitter" PROFIBUS PA profile of an analog input value:
  - Of a PA field device in accordance with PROFIBUS 3.0 class A and B
  - Of an auxiliary variable of a HART field device
  - Of an FF field device
- Processing of the redundant input channels.

How it works

Block FbAnIn cyclically reads the process value and the signal status of the field device from the process image (partition). The process value is available as a physical variable. The signal status contains information about the status of the field device.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Connect the symbol generated in HW Config (symbol table) for the input channel with the `PV` input parameter.

In case of redundant I/O configuration, use symbol of the channel of the master module.

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter `Mode` is interconnected with the corresponding output parameter `OMODE_xx` of the MOD_PAL0, FF_MOD32- or MOD_PAX0 block.
- The in/out parameter `DataXchg` is interconnected with the corresponding output parameter `DXCHG_xx` of the MOD_PAL0, FF_MOD32 - or MOD_PAX0 block.

- The symbol for the signal status of the analog input channel is interconnected to input `PV_ST`.
- The `MS` parameter is interconnected to the `O_MS` output parameter of the diagnostics block.

Note

If you are not using the CFC function "Generate module drivers" you must set the in/out parameter `Mode` manually. Refer to the Mode settings for field devices (Page 2191) chapter for more on this.

In case of redundancy, connection with both quality bits and connection with the redundant slave channel:

- The in parameter `PVSlv` is interconnected with the symbol of the slave channel.
- The in parameter `PV_ST` is interconnected with the symbol of the quality byte of the master channel.
- The in parameter `PVSlv_ST` is interconnected with the symbol of the quality byte of the slave channel.

For the FbAnIn block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Monitoring of an analog process tag for PA/FF devices (AnalogMonitoring_Fb) (Page 2331)
- Dosing with PA/FF devices (Dose_Lean_Fb) (Page 2333)
- PID controller for PA/FF devices (PIDControl_Lean_Fb) (Page 2297)

Quality code (ET200 SP HA modules)

The quality code is read from the process image and has two states: “Good” or “Bad”.

Redundancy

The block monitors the values of the redundant signals (master and slave) with quality code and:

- the value of the signal with “Good” quality code along with its quality code is transferred to the process when one of the signals delivers “Good” quality code.
- the value of the master signal along with its quality code is transferred to the process when both the signals deliver “Bad” quality code.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

FbAnIn block diagram (Page 2023)
FbAnIn I/Os (Page 2021)
FbAnIn messaging (Page 2020)
FbAnIn error handling (Page 2019)
FbAnIn functions (Page 2017)
FbAnIn modes (Page 2017)

16.2.2 FbAnIn modes

FbAnIn modes

This block does not have any operating modes.

See also

FbAnIn block diagram (Page 2023)
FbAnIn I/Os (Page 2021)
FbAnIn messaging (Page 2020)
FbAnIn error handling (Page 2019)
FbAnIn functions (Page 2017)
Description of FbAnIn (Page 2015)

16.2.3 FbAnIn functions

Functions of FbAnIn

The functions for this block are listed below.

Obtaining the standard value

The analog value of the process image (partition) is output as a standard value at the `PV_Li` output parameter.

Holding the last value if raw value is invalid

If the block is to hold its last valid value when the analog value is invalid or in the initialization phase of the device, you must activate this function at the `Feature` Bit Outputting last valid value if raw value is invalid (Page 153).

Output substitute value if raw value is invalid

If the block is to output a substitute value (`SubsPV`) when the analog value is invalid or in the initialization phase of the device, you must activate this function at the `Feature` Bit Output substitute value if raw value is invalid (Page 148).

Outputting an invalid value if analog value is invalid

If the block is to output an invalid value (`PV_Li` = `PV`), you must activate this function at the `Feature` Bit Output invalid raw value (Page 173).

This function is preset.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67)

Signal status for Fb channel blocks

The block provides the standard function Forming and outputting the signal status for channel blocks for field devices (Page 118).

The signal status of process value `PV_Li` is generated from internal events such as channel errors, higher-level errors, or simulations, as well as from the signal status `PV_ST`, which comes directly from the device.

The signal status `PV_ST` can accept values of 16#00 - 16#FF.

The block recognizes a higher-level error, for example, failure of a DP/PA link, via the `Mode` input parameter.

- If the high byte is `Mode` = 16#80, then the values in the process image (partition) are valid.
- If the high byte `Mode` = 16#40 (value status = higher-level error, `ModErr` = 1), then the analog value is treated as invalid.

The measuring type set in the low word of the `Mode` input parameter will be ignored.

The bit combinations of signal status `PV_ST` are output as output parameters (BOOL values). These conform to the bit combinations specified in PROFIBUS 3.0 "General Requirements".

For FF field devices only: The values of the signal status `PV_ST` = 16#84 - 16#87 and 16#90 – 16#93 are evaluated by the link in the same way as signal status 16#80 – 16#83.

If the signal status `PV_ST` = 16#80 and a process value `PV` with the value 16#7FFFFFFF (invalid) are transferred from the FF field device, the block will deal with the signal status 16#00 (invalid) from the FF field device.

Simulating signals

The block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
28	Output invalid raw value (Page 173)
29	Output substitute value if raw value is invalid (Page 148)
30	Outputting last valid value if raw value is invalid (Page 153)

Sign-of-life monitoring

If an input value whose signal status is `16#80` (good) remains constant for a programmable time (monitoring time), the input value is detected as faulty and the outputs `Bad = 1` and `FrzVal = 1` are set.

The monitoring time is set at the `FrznTmIn` input parameter in seconds. With `FrznTmIn = 0` or `FrznEn = 0` (default setting), the sign-of-life monitoring is deactivated, any pending errors are reset.

The input value is considered as faulty as long as it seen as constant. The monitoring time is restarted each time the input value is changed.

See also

FbAnIn block diagram (Page 2023)

FbAnIn I/Os (Page 2021)

FbAnIn messaging (Page 2020)

FbAnIn error handling (Page 2019)

FbAnIn modes (Page 2017)

Description of FbAnIn (Page 2015)

16.2.4 FbAnIn error handling

Error handling of FbAnIn

Please refer to the section Error handling (Page 119) in the basic instructions.

The following errors can be displayed for this block:

- Channel error
- Higher-level error

- Invalid measuring range
- Frozen input value (sign-of-life monitoring)

Channel error

At the output parameter `Bad`, channel errors are displayed with 1. The channel error is generated from the signal status `PV_ST`.

Higher-level error / invalid measuring range

A higher-level error is displayed at the output parameters `ModErr` and `Bad` with 1 if the signal status at `High Word` of the input parameter `Mode` accepts the value 16#40.

A higher-level error is also present when an incorrect measuring type is entered in `Low Word` of the input parameter `Mode`.

In addition, at the output parameter `PV_Li` of the signal status, either 16#00 (in the event of an error) or 16#60 (for manipulated value (for example, substitute value, simulation, last valid value) is output.

See also

FbAnIn block diagram (Page 2023)
FbAnIn I/Os (Page 2021)
FbAnIn messaging (Page 2020)
FbAnIn functions (Page 2017)
FbAnIn modes (Page 2017)
Description of FbAnIn (Page 2015)

16.2.5 FbAnIn messaging

Messaging

This block does not offer messaging.

See also

FbAnIn block diagram (Page 2023)
FbAnIn I/Os (Page 2021)
FbAnIn error handling (Page 2019)
FbAnIn functions (Page 2017)
FbAnIn modes (Page 2017)
Description of FbAnIn (Page 2015)

16.2.6 FbAnIn I/Os

I/Os of FbAnIn

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 2017)	STRUCT • Bit 0: BOOL • ... • Bit 28: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0
FlutEn	1 = Flutter suppression activated	BOOL	0
FlutTmIn*	Flutter time [s]	INT	0
FrznEn	1 = Sign-of-life monitoring activated	BOOL	0
FrznTmIn	Monitoring time in [s]	REAL	0.0
MS	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ms_Ext	External maintenance status	DWORD	16#00000000
PV	Process value (analog value)	REAL	0.0
PVSlv	Process value of the slave	REAL	0.0
PVSlv_ST	State of the slave process value	BYTE	16#00
PV_ST	Signal status for the process value	BYTE	16#80
PV_Unit	Unit of measure for process value	INT	1001
SampleTime	Sampling time in [s]	REAL	0.1
Scale	Scaling of the process value as a structure	STRUCT • High: REAL • Low:REAL	- • 100.0 • 0.0
SimOn	1 = Simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV	Process value used for SimOn = 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SubsPV	Substitute value	REAL	0.0
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see FbAnIn error handling (Page 2019)	INT	-1
FrznVal	Frozen process value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Li	Standard value (physical variable)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_LiUnit	Unit of the process value	INT	0
RemTime	Remaining monitoring time [s]	REAL	0.0
SampleTime	Sampling time [s]	REAL	0.1
ScaleOut	Scaling of the process value as a structure	STRUCT • High: REAL • Low:REAL	- • 100.0 • 0.0
SimAct	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

FbAnIn block diagram (Page 2023)
FbAnIn messaging (Page 2020)
FbAnIn modes (Page 2017)
Description of FbAnIn (Page 2015)

16.2.7 FbAnIn block diagram

FbAnIn block diagram

A block diagram is not provided for this block.

See also

FbAnIn I/Os (Page 2021)
FbAnIn messaging (Page 2020)
FbAnIn error handling (Page 2019)
FbAnIn functions (Page 2017)
FbAnIn modes (Page 2017)
Description of FbAnIn (Page 2015)

16.3 FbAnOu - Analog output channel block for field devices

16.3.1 Description of FbAnOu

Object name (type + number) and family

Type + number: FB 1814

Family: Channel

Area of application for FbAnOu

The block is used for the following applications:

Signal processing (cyclic service) in accordance with "Actuator" PROFIBUS PA profile of an analog input value:

- Of a PA field device in accordance with PROFIBUS 3.0 class A and B
- Of an FF field device

How it works

Block FbAnOu cyclically reads the process values and the signal status of the field device from the process image (partition). The process values are available as physical variables. The signal status contains information about the status of the field device.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

You must interconnect each used signal of the block with the symbols configured in HW Config or in the symbol table according to your user data configuration:

PA device connection	Data type	I/O
`Rbk`	REAL	Input
`RCasOut`	REAL	Input
`PosD`	BYTE	Input
`SP`	REAL	Output
`RCasIn`	REAL	Output

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter `Mode` is interconnected with the corresponding output parameter `OMode_xx` of the MOD_PAL0, FF_MOD32- or MOD_PAX0 block.
- The in/out parameter `DataXchg` is interconnected with the corresponding output parameter `DataXchg_xx` of the MOD_PAL0, FF_MOD32- or MOD_PAX0 block.

- The `MS` parameter is interconnected to the `O_MS` output parameter of the diagnostics block.
- The corresponding signal status is symbolically interconnected depending on your user data configuration:

I/O field device	Data type	I/O
`RbkST`	BYTE	Input
`RCasOutST`	BYTE	Input
`PosD_ST`	BYTE	Input
`SP_ST`	BYTE	Output
`RCasIn_ST`	BYTE	Output
`CbkBy0`	BYTE	Input
`CbkBy1`	BYTE	Input
`CbkBy2`	BYTE	Input

Note

If you are not using the CFC function "Generate module drivers" you must set the in/out parameter `Mode` manually. Refer to the Mode settings for field devices (Page 2191) chapter for more on this.

For the FbAnOu block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- PID controller for PA/FF devices (PIDControl_Lean_Fb) (Page 2297)
- Control valve for PA/FF devices (ValveAnalog_Fb) (Page 2345)

Startup characteristics

The block is executed once in OB100 at system start. The output and in/out parameters are calculated.

Quality code (ET200 SP HA modules)

The quality code is read from the process image and has two states: “Good” or “Bad”.

Redundancy

The block monitors the values of the redundant signals (master and slave) with quality code and the value of the signal with “Good” quality code along with its quality code is transferred to the process.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

FbAnOu block diagram (Page 2033)
FbAnOu I/Os (Page 2029)
FbAnOu error handling (Page 2028)
FbAnOu functions (Page 2026)
FbAnOu modes (Page 2026)
FbAnOu messaging (Page 2028)

16.3.2 FbAnOu modes

FbAnOu modes

This block does not have any modes.

16.3.3 FbAnOu functions

Functions of FbAnOu

The functions for this block are listed below.

Obtaining the standard value

The signals of the FF field device are read from the process image (partition) of the inputs and written to the process image (partition) of the outputs. The controlled variable `Rbk` and discrete position feedback`PosD`, are read, as well as the active reference variable `RCasOut`, together with its associated signal status `RbkST`, `PosD_ST` and `RCasOutST` are read and written to the output parameters `SP` and `RCasIn`, with the associated signal status `RbkST` and `RCasInST`.

Additional detailed device information (`Cbk0` - `Cbk2`) can be read as an option. The device information is available per bit at the block output.

The signal status `RbkST` or `RCasOutST` that comes directly from the device can have values from 16#00 – 16#FF.

The values of the signal status `PosD_ST` and `RbkST` of 16#84 - 16#87and 16#90 – 16#93of the FF field device are evaluated like the signal status 16#80 – 16#83.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67)

Signal status for Fb channel blocks

The block provides the standard function Forming and outputting the signal status for channel blocks for field devices (Page 118).

The signal status of the process values (`RCasInLi` or `RbkLi`) is generated from internal events such as channel errors, higher-level errors, or simulations, as well as from the signal status `RbkST` or `RCasInST`, which comes directly from the device.

Value	Meaning
16#80	Valid value
16#60	Manipulated value (for example, substitute value, simulation, last valid value)
16#28	Bad, process related
16#68	Uncertain, device related
16#78	Uncertain, process related
16#A4	Maintenance request present
16#00	Invalid value

Simulating signals

The block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
30	Outputting a de-energized value for block-external simulation (Page 147)

See also

FbAnOu block diagram (Page 2033)

FbAnOu I/Os (Page 2029)

FbAnOu error handling (Page 2028)

FbAnOu modes (Page 2026)

Description of FbAnOu (Page 2024)

FbAnOu messaging (Page 2028)

16.3.4 FbAnOu error handling

Error handling of FbAnOu

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Channel error
- Higher-level error
- Invalid measuring range

Channel error

At the output parameter `Bad`, channel errors are displayed with 1.

The channel error is generated from the signal states `RbkST`, `RCasOutST`, and `PosD_ST`. In addition to this, a channel error (`Bad` = 1) also occurs if the signal status has a valid value (`RbkST` or `RCasOutST` is 16#80) and the present position of actuator `Rbk` or `RCasOut` has the value 16#7FFFFFFF (invalid).

Higher-level error / invalid measuring range

A higher-level error is displayed at the output parameters `ModErr` and `Bad` with 1 if the signal status at `High Word` of the input parameter `Mode` accepts the value 16#40.

A higher-level error is also present when an incorrect measuring type is entered in `Low Word` of the input parameter `Mode`.

16.3.5 FbAnOu messaging

Messaging

This block does not offer messaging.

See also

Description of FbAnOu (Page 2024)

FbAnOu modes (Page 2026)

FbAnOu functions (Page 2026)

FbAnOu error handling (Page 2028)

FbAnOu I/Os (Page 2029)

FbAnOu block diagram (Page 2033)

16.3.6 FbAnOu I/Os

I/Os of FbAnOu

Input parameters

Parameter	Description	Type	Default
CbkBy0	Additional information on the actuator status	BYTE	16#00
CbkBy1	Additional information on the actuator status	BYTE	16#00
CbkBy2	Additional information on the actuator status	BYTE	16#00
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 2026)	STRUCT • Bit 0: BOOL • ... • Bit 30: BOOL • Bit 31: BOOL	- • 0 • 0 • 1 • 0
FlutEn	1 = Flutter suppression activated	BOOL	0
FlutTmIn*	Flutter time [s]	INT	0
MS	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ms_Ext	External maintenance status	DWORD	16#00000000
PosD	Discrete position feedback (current position) of the valve: 0 = Not initialized 1 = Closed 2 = Opened 3 = Intermediate	BYTE	16#00
PosD_ST	Signal status of PosD	BYTE	16#80
PVSlv_ST	State of the slave process value	BYTE	16#00
RCasInLi	Setpoint for the remote cascade operating mode	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RCasOut	Setpoint from function block in device	REAL	0.0
RCasOutST	Signal status of RCasOut	BYTE	16#80
Rbk	Current position of actuator (actual value)	REAL	0.0
RbkST	Signal status of Rbk	BYTE	16#80
Scale	Scaling of the process value as structure for display	STRUCT • High: REAL • Low:REAL	- • 100.0 • 0.0

Parameter	Description	Type	Default
SP_Li	Setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_LiUnit	Unit of measure for setpoint	INT	1342
SimOn	1 = Simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPosD	Discrete position feedback for (output parameter PosD_Li), that is used for SimOn = 1.	BYTE	16#00
SimRCasInLi	Setpoint for the remote cascade operating mode RCasInLi, used for SimOn = 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimRbk	Current position of the valve (actual value) Rbk, used for SimOn = 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimSP_Li	Setpoint SP_Li, used for SimOn = 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk0	1 = Field device in safe position	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Cbk1	1 = Request for "local operation"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk2	1 = Device is operated locally	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk3	1 = Emergency operation is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk4	1 = Deviation in direction of movement	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk5	1 = Stop reached (actuator fully open)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk6	1 = Stop reached (actuator fully closed)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk7	1 = Runtime overshoot	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk8	1 = Actuator is opening	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk9	1 = Actuator is closing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk10	1 = Alarm generated by any change to the static data (FB and TB)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk11	1 = Simulation mode	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk12	Not used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk13	1 = Internal control loop interrupted	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Cbk14	1 = Closed-loop control inactive	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk15	1 = Self-test is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk16	1 = Stroke integral exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk17	1 = Additional input is active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see FbAnOu error handling (Page 2028)	INT	-1
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PosD_Li	Discrete position feedback (current position) of the valve: 0 = Not initialized 1 = Closed 2 = Opened 3 = Intermediate	BYTE	16#00
PosD_LiST	Signal status PosD_Li	BYTE	16#00
PosDCloseLi	1=feedback Close	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PosDOpenLi	1=feedback Open	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RCasIn	Setpoint for the remote cascade operating mode	REAL	0.0
RCasInST	Signal status of RCasIn	BYTE	16#00

Parameter	Description	Type	Default
RCasOutLi	Setpoint from function block in device	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
RbkLi	Current position of actuator (actual value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ScaleOut	Scaling of the process value as structure for display	STRUCT • High: REAL • Low:REAL	- • 100.0 • 0.0
SimAct	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP	Setpoint	REAL	0.0
SP_ST	Signal status of setpoint	BYTE	16#00
SP_Slv	Output setpoint of the slave device	REAL	0.0
SP_Unit	Unit of the process value (SP_LiUnit)	INT	0

16.3.7 FbAnOu block diagram

FbAnOu block diagram

A block diagram is not provided for this block.

16.4 FbDiIn - Digital input channel block for field devices

16.4.1 Description of FbDiIn

Object name (type + number) and family

Type + number: FB 1815

Family: Channel

Area of application for FbDiIn

The block is used for the following applications:

Signal processing of digital input values (discrete input) of a field device (cyclic service in accordance with PROFIBUS PA):

- Of a PA field device in accordance with PROFIBUS 3.0 class A and B
- Of an FF field device

How it works

Block FbDiIn cyclically reads the process values and the signal status of the field device from the process image (partition). The process values are grouped in one byte. The signal status contains information about the status of the field device.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter Mode is interconnected with the corresponding output parameter `OMode_xx` of the MOD_PAL0, FF_MOD32- or MOD_PAX0 block.
- The in/out parameter `DataXchg` is interconnected with the corresponding output parameter `DataXchg_xx` of the MOD_PAL0, FF_MOD32- or MOD_PAX0 block.
- The symbol for the signal status of the analog input channel is interconnected to input `PV_ST`.
- The `MS` parameter is interconnected to the `O_MS` output parameter of the diagnostics block.

Note

If you are not using the CFC function "Generate module drivers" you must set the in/out parameter `Mode` manually. Refer to the Mode settings for field devices (Page 2191) chapter for more on this.

For the FbDiIn block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Monitoring a digital process tag for PA/FF devices (DigitalMonitoring_Fb) (Page 2328)

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

FbDiIn block diagram (Page 2042)
FbDiIn I/Os (Page 2039)
FbDiIn messaging (Page 2038)
FbDiIn error handling (Page 2038)
FbDiIn functions (Page 2036)
FbDiIn modes (Page 2035)

16.4.2 FbDiIn modes

FbDiIn modes

This block does not have any modes.

See also

FbDiIn block diagram (Page 2042)
FbDiIn I/Os (Page 2039)
FbDiIn messaging (Page 2038)
FbDiIn error handling (Page 2038)
FbDiIn functions (Page 2036)
Description of FbDiIn (Page 2034)

16.4.3 FbDiIn functions

Functions of FbDiIn

The functions for this block are listed below.

Obtaining the standard value

The digital values (WORD format) of the process image (partition) are output at the `PV_Li0` - `PV_Li7` output parameters.

The signal status `PV_ST` that comes directly from the device can have values from 16#00 – 16#FF.

The values of the signal status `PV_ST` of 16#84 - 16#87 and 16#90 – 16#93 of the FF field device are evaluated in the same way as 16#80 – 16#83.

Holding the last value if raw value is invalid

If the block is to hold its last valid value when the digital value is invalid or in the initialization phase of the device, you must activate this function at the `Feature` Bit Outputting last valid value if raw value is invalid (Page 153).

Output substitute value if raw value is invalid

If the block is to output a substitute value (`SubsPV`) when the digital value is invalid or in the initialization phase of the device, you must activate this function at the `Feature` Bit Output substitute value if raw value is invalid (Page 148).

Output of invalid value if raw value is invalid

If the block is to output an invalid value (`PV_Li` = `PV`), you must activate this function at the `Feature` Bit Output invalid raw value (Page 173).

This function is preset.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67)

Signal status for Fb channel blocks

The block provides the standard function Forming and outputting the signal status for channel blocks for field devices (Page 118).

The signal status of the process value `PV_Li0` - `PV_Li7` is generated from internal events such as channel errors, higher-level errors, or simulations, as well as from the signal status `PV_ST`, which comes directly from the device.

Value	Meaning
16#80	Valid value
16#60	Manipulated value (for example, substitute value, simulation, last valid value)
16#28	Bad, process related
16#68	Uncertain, device related
16#78	Uncertain, process related
16#A4	Maintenance request present
16#00	Invalid value

Simulating signals

The block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
28	Output invalid raw value (Page 173)
29	Output substitute value if raw value is invalid (Page 148)
30	Outputting last valid value if raw value is invalid (Page 153)

See also

FbDiIn block diagram (Page 2042)

FbDiIn I/Os (Page 2039)

FbDiIn messaging (Page 2038)

FbDiIn error handling (Page 2038)

FbDiIn modes (Page 2035)

Description of FbDiIn (Page 2034)

16.4.4 FbDiIn error handling

Error handling of FbDiIn

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Channel error
- Higher-level error
- Invalid measuring range

Channel error

At the output parameter `Bad`, channel errors are displayed with 1. The channel error is generated from the signal status `PV_ST`.

Higher-level error / invalid measuring range

A higher-level error is displayed at the output parameters `ModErr` and `Bad` with 1 if the signal status at `High Word` of the input parameter `Mode` accepts the value 16#40.

A higher-level error is also present when an incorrect measuring type is entered in `Low Word` of the input parameter `Mode`.

In addition, at the output parameter `PV_LiX` of the signal status, either 16#00 (in the event of an error) or 16#60 (for manipulated value (for example, substitute value, simulation, last valid value) is output. (`PV_LiX`: X = 0 ... 7)

See also

FbDiIn block diagram (Page 2042)

FbDiIn I/Os (Page 2039)

FbDiIn messaging (Page 2038)

FbDiIn functions (Page 2036)

FbDiIn modes (Page 2035)

Description of FbDiIn (Page 2034)

16.4.5 FbDiIn messaging

Messaging

This block does not offer messaging.

See also

FbDiIn block diagram (Page 2042)
FbDiIn I/Os (Page 2039)
FbDiIn error handling (Page 2038)
FbDiIn functions (Page 2036)
FbDiIn modes (Page 2035)
Description of FbDiIn (Page 2034)

16.4.6 FbDiIn I/Os

FbDiIn I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 2036)	STRUCT • Bit 0: BOOL • ... • Bit 28: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0
FlutEn	1 = Flutter suppression activated	BOOL	0
FlutTmIn*	Flutter time [s]	INT	0
MS	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Ext	External maintenance status	DWORD	16#00000000
PV	Process value (digital value)	BYTE	16#00
PV_ST	Signal status for the process value	BYTE	16#80
SimOn	1 = Simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV0	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV1	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
SimPV2	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV3	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV4	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV5	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV6	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV7	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SubsPV	Substitute value	BYTE	16#00
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see FbDiIn error handling (Page 2038)	INT	-1

Parameter	Description	Type	Default
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Li0	Process value 0	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Li1	Process value 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Li2	Process value 2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Li3	Process value 3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Li4	Process value 4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Li5	Process value 5	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Li6	Process value 6	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Li7	Process value 7	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimAct	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

FbDiIn block diagram (Page 2042)
FbDiIn messaging (Page 2038)
FbDiIn modes (Page 2035)
Description of FbDiIn (Page 2034)

16.4.7 FbDiIn block diagram

FbDiIn block diagram

A block diagram is not provided for this block.

See also

FbDiIn I/Os (Page 2039)
FbDiIn messaging (Page 2038)
FbDiIn error handling (Page 2038)
FbDiIn modes (Page 2035)
FbDiIn functions (Page 2036)
Description of FbDiIn (Page 2034)

16.5 FbDiOu - Digital output channel block for field devices

16.5.1 Description of FbDiOu

Object name (type + number) and family

Type + number: FB 1816

Family: Channel

Area of application for FbDiOu

The block is used for the following applications:

Signal processing of max 8 digital input/output values of a field device (cyclic service in accordance with PROFIBUS PA):

- Of a PA field device in accordance with PROFIBUS 3.0 class A and B
- Of an FF field device

How it works

Block FbDiOu cyclically reads the process values and the signal status of the field device from the process image (partition). The eight process values are grouped in one byte for each of the input parameters `SP_Li` and `RCasInLi`. The signal status contains information about the status of the field device.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

You must interconnect each used signal of the block with the symbols configured in HW Config or in the symbol table according to your user data configuration:

I/O field device	Data type	I/O
`Rbk`	BYTE	Input
`RCasOut`	BYTE	Input
`SP`	BYTE	Output
`RCasIn`	BYTE	Output

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter Mode is interconnected with the corresponding output parameter `OMode_xx` of the MOD_PAL0, FF_MOD32- or MOD_PAX0 block.
- The in/out parameter `DataXchg` is interconnected with the corresponding output parameter `DataXchg_xx` of the MOD_PAL0, FF_MOD32- or MOD_PAX0 block.

- The `MS` parameter is interconnected to the `O_MS` output parameter of the diagnostics block.
- The corresponding signal status is symbolically interconnected depending on your user data configuration:

I/O field device	Data type	I/O
`RbkST`	BYTE	Input
`RCasOutST`	BYTE	Input
`SP_ST`	BYTE	Output
`RCasInST`	BYTE	Output
`CbkBy0`	BYTE	Input
`CbkBy1`	BYTE	Input
`CbkBy2`	BYTE	Input

Note

If you are not using the CFC function "Generate module drivers" you must set the `Mode` in/out parameter manually. Refer to the Mode settings for field devices (Page 2191) section for more on this.

For the FbDiOu block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Dosing with PA/FF devices (Dose_Lean_Fb) (Page 2333)

Startup characteristics

The block will run through one time in OB100 at system start. The output and in/out parameters are calculated.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

FbDiOu block diagram (Page 2052)

FbDiOu I/Os (Page 2048)

FbDiOu messaging (Page 2047)

FbDiOu error handling (Page 2046)

FbDiOu functions (Page 2045)

FbDiOu modes (Page 2045)

16.5.2 FbDiOu modes

FbDiOu modes

This block does not have any modes.

See also

FbDiOu block diagram (Page 2052)
FbDiOu I/Os (Page 2048)
FbDiOu messaging (Page 2047)
FbDiOu error handling (Page 2046)
FbDiOu functions (Page 2045)
Description of FbDiOu (Page 2043)

16.5.3 FbDiOu functions

Functions of FbDiOu

The functions for this block are listed below.

Obtaining the standard value

The signals of the FF field device are read from the process image (partition) of the inputs and written to the process image (partition) of the outputs. The input parameter `Rbk` and the active reference variable `RCasOut`, together with its associated signal status `RbkST` and `RCasOutST` are read and written to the output parameters `SP` and `RCasIn` with the associated signal status `RbkST` and `RCasOutSt`. Additional detailed device information (`CbkBy0` - `CbkBy2`) can be read as an option. The device information is available per bit at the block output.

The signal status `RbkST` or `RCasOutST` that comes directly from the device can have values from 16#00 – 16#FF.

The values of the signal status `RbkST` and `RCasOutST` of 16#84 - 16#87and 16#90 – 16#93of the FF field device are evaluated by the link as 16#80 – 16#83.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67)

Signal status for Fb channel blocks

The block provides the standard function Forming and outputting the signal status for channel blocks for field devices (Page 118).

The signal status of the process values (`RCasIn` or `RbkLi`) is generated from internal events such as channel errors, higher-level errors, or simulations, as well as from the signal status `RbkLiST` or `RCasInST`, which comes directly from the device.

Value	Meaning
16#80	Valid value
16#60	Manipulated value (for example, substitute value, simulation, last valid value)
16#28	Bad, process related
16#68	Uncertain, device related
16#78	Uncertain, process related
16#A4	Maintenance request present
16#00	Invalid value

Simulating signals

The block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
10	Condition monitoring information at the channel blocks (Page 144)
30	Outputting a de-energized value for block-external simulation (Page 147)

See also

FbDiOu block diagram (Page 2052)

FbDiOu I/Os (Page 2048)

FbDiOu messaging (Page 2047)

FbDiOu error handling (Page 2046)

FbDiOu modes (Page 2045)

Description of FbDiOu (Page 2043)

16.5.4 FbDiOu error handling

Error handling of FbDiOu

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Channel error
- Higher-level error
- Invalid measuring range

Channel error

At the output parameter `Bad`, channel errors are displayed with 1. The channel error is generated from the signal states `RbkST` and `RCasOutSt`.

Higher-level error / invalid measuring range

A higher-level error is displayed at the output parameters `ModErr` and `Bad` with 1 if the signal status at `High Word` of the input parameter `Mode` accepts the value 16#40.

A higher-level error is also present when an incorrect measuring type is entered in `Low Word` of the input parameter `Mode`.

See also

FbDiOu block diagram (Page 2052)
FbDiOu I/Os (Page 2048)
FbDiOu messaging (Page 2047)
FbDiOu functions (Page 2045)
FbDiOu modes (Page 2045)
Description of FbDiOu (Page 2043)

16.5.5 FbDiOu messaging

Messaging

This block does not offer messaging.

See also

FbDiOu block diagram (Page 2052)
FbDiOu I/Os (Page 2048)
FbDiOu error handling (Page 2046)
FbDiOu functions (Page 2045)
FbDiOu modes (Page 2045)
Description of FbDiOu (Page 2043)

16.5.6 FbDiOu I/Os

I/Os of FbDiOu

Input parameters

Parameter	Description	Type	Default
CbkBy0	Additional information on the actuator status	BYTE	16#00
CbkBy1	Additional information on the actuator status	BYTE	16#00
CbkBy2	Additional information on the actuator status	BYTE	16#00
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 2045)	STRUCT • Bit 0: BOOL • ... • Bit 30: BOOL • Bit 31: BOOL	- • 0 • 0 • 0 • 0
FlutEn	1 = Flutter suppression activated	BOOL	0
FlutTmIn*	Flutter time [s]	INT	0
MS	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Ext	External maintenance status	DWORD	16#00000000
RCasInLi	Setpoint for the remote cascade operating mode	BYTE	16#00
RCasInLiST	Signal status for the setpoint value (RCasInLi)	BYTE	16#80
Rbk	Current position of actuator (actual value)	BYTE	16#00
RbkST	Signal status of Rbk	BYTE	16#80
RCasOut	Setpoint from function block in device	BYTE	16#00
RCasOutST	Signal status of RCasOut	BYTE	16#80
SP_Li	Setpoint	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimOn	1 = Simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimRCasInLi	Setpoint for the remote cascade operating mode RCasInLi, used for SimOn = 1	BYTE	16#00
SimRbk	0 = Actuator is closed 1 = Actuator is open	BOOL	0
SimSP_Li	1 = Setpoint SP_Li, used for SimOn = 1	BOOL	0
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk0	1 = Field device in safe position active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk1	1 = Request for "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk2	1 = Field device in "manual mode"	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk3	1 = Emergency override active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk4	1 = Current position differs from expected position	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk5	1 = Valve connection break	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk6	1 = Indicates a short-circuit at the valve connection	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk7	1 = Not used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk8	1 = Actuator is opening	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Cbk9	1 = Actuator is closing	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk10	1 = Alarm generated by any change to the static data (FB and TB)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk11	1 = Simulation of process values is enabled	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk12	1 = Not used	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk13	1 = Internal control loop interrupted	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk14	1 = Valve not active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk15	1 = Device under self test	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk16	1 = Valve travel limit has been exceeded	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk17	1 = Limit of break time exceeded when changing from OPEN to CLOSE	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk18	1 = Limit of break time exceeded when changing from CLOSE to OPEN	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk19	1 = Error occurred in the internal cycle test	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk20	1 = Timeout during the transition from OPEN to CLOSE	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk21	1 = Timeout during the transition from CLOSE to OPEN	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Cbk22	1 = Valve blocked mechanically	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Cbk23	1 = Zero point not reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see FbDiOu error handling (Page 2046)	INT	-1
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RCasOutLi	Setpoint from function block in device	BYTE	16#00
RCasOutLiST	Signal status for setpoint from function block in device (RCasOutLi)	BYTE	16#00
RCasIn	Setpoint for the remote cascade operating mode	BYTE	16#00
RCasInST	Signal status of RCasIn	BYTE	16#00
RbkCloseLi	1 = Actuator closed feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RbkLi	Current position of actuator (actual value)	BYTE	16#00
RbkLiST	Signal status of current position of actuator (RbkLi)	BYTE	16#00
RbkOpenLi	1 = Actuator open feedback signal	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimAct	1 = The block is in simulation	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SP	Setpoint	BYTE	16#00
SP_ST	Signal status of setpoint	BYTE	16#00

See also

FbDiOu block diagram (Page 2052)
FbDiOu messaging (Page 2047)
FbDiOu modes (Page 2045)
Description of FbDiOu (Page 2043)

16.5.7 FbDiOu block diagram

FbDiOu block diagram

A block diagram is not provided for this block.

See also

FbDiOu I/Os (Page 2048)
FbDiOu messaging (Page 2047)
FbDiOu error handling (Page 2046)
FbDiOu functions (Page 2045)
FbDiOu modes (Page 2045)
Description of FbDiOu (Page 2043)

16.6 FbDrive - Channel block for compact drives

16.6.1 Description of FbDrive

Object name (type + number) and family

Type + number: FB 1905

Family: Channel

Area of application of FbDrive

The block is used for the following applications:

- Integration of compact drives in PCS 7

How it works

The FbDrive block integrates any compact drives that meet the following conditions and are detected by the system:

- Message frame type "1" with 2 input words and 2 output words
- Message frame type "20" with 6 input words and 2 output words

Configuration

The interconnection is made symbolic to the first input word. All other interconnections are updated automatically using a wizard. The input and output word must have the same beginning in HW Config.

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

The block is also installed automatically in the startup OB (OB100).

Connect the symbol generated in HW Config (symbol table) for the input channel with the `PZDIn1` input parameter.

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The Mode in/out parameter is interconnected to the corresponding `OMODE_xx` output parameter of the MOD_DRV block.
- The in/out parameter `DataXchg` is interconnected to the corresponding `DXCHG_xx` output parameter of the MOD_DRV block.
- With the "1" message frame type, the `PZDIn1` and `PZDIn2` inputs are interconnected to the `PZDOutx` outputs, with the "20" message frame type the `PZDIn3` - `PZDIn6` inputs are also interconnected.
- The `MS` parameter is interconnected to the `O_MS` output parameter of the diagnostics block.

Note

"Configuration of frequency converters with the APL channel block "FbDrive" in SIMATIC PCS 7 (https://support.industry.siemens.com/cs/document/64181993/configuration-of-frequency-converters-with-the-apl-channel-block-%E2%80%9Efbdrive%E2%80%9D-in-simatic-pcs-7?dti=0&lc=en-WW)"

- The use of different channel block types for the same device type of drive or inverter is not allowed.

Startup characteristics

The block has startup characteristics.

16.6.2 FbDrive modes

Operating modes of FbDrive

This block does not have any modes.

16.6.3 FbDrive functions

Functions of FbDrive

The functions for this block are listed below.

Reading messages

You can specify the format of messages using the `Feature` bit Reading messages (Page 146).

Transmission of messages

If the block is to send messages to the upstream diagnostics block, you must activate this function at the `Feature` bit Transmission of status information of devices (Page 174).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the Feature parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
5	Setting the scaling for the process values (Page 147)
6	Failure handling (Page 136)
10	Condition monitoring information at the channel blocks (Page 144)
28	Reading messages (Page 146)
29	Transmission of status information of devices (Page 174)
30	Outputting a de-energized value for block-external simulation (Page 147)

Forming the control word

When `STW1`(control word) is created and `Stw1ST` is simultaneously at `16#80`, the Boolean control inputs listed below have no effect and the control word `STW1` is set at the process value output `PZDOUT1`.

When the `Stw1ST` signal status has a bad value, the data is taken from the Boolean control inputs.

`On`	Control word 1 bit 0
`NoCoastSt`	Control word 1 bit 1
`NoQuickSt`	Control word 1 bit 2
`EnOp`	Control word 1 bit 3
`EnRampGen`	Control word 1 bit 4
`UnfreeRamp`	Control word 1 bit 5
`EnSp`	Control word 1 bit 6
`Ackn`	Control word 1 bit 7
`Jogg1`	Control word 1 bit 8
`Jogg2`	Control word 1 bit 9
`Local`	Control word 1 bit 10
`InvSP`	Control word 1 bit 11
`Ctrl12`	Control word 1 bit 12
`Ctrl13`	Control word 1 bit 13
`Ctrl14`	Control word 1 bit 14
`Ctrl15`	Control word 1 bit 15

Setting the limits for the process value or for the calculation limits

The limits for the process value or the calculation limits are set by the SP_LiScale input.

Calculation:

SpeedLi.Value := (PZD2In*((SP_LiScale.HIGH-SP_LiScale.LOW)/16384.0))
+SP_LiScale.LOW;

16.6.4 FbDrive error handling

Error handling of FbDrive

Error handling of all the blocks is described in chapter Error handling (Page 119) in the basic instructions.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The ErrorNum I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
1	Invalid process value (output parameter Bad = 1)

16.6.5 FbDrive messaging

Messaging of FbDrive

This block does not offer messaging.

16.6.6 FbDrive status word

Forming the status word

The `ZSW1` (status word), or the Boolean status outputs, are set via the process value input PZDIN1.

Status word 1	Bit 0	`RdyOn`
Status word 1	Bit 1	`RdyOp`
Status word 1	Bit 2	`OpEn`
Status word 1	Bit 3	`Fault`
Status word 1	Bit 4	`NoOff2`
Status word 1	Bit 5	`NoOff3`
Status word 1	Bit 6	`SwOn`
Status word 1	Bit 7	`Warning`
Status word 1	Bit 8	`SpeedErr`
Status word 1	Bit 9	`CtrlReq`

Status word 1	Bit 10	F_N_Reach
Status word 1	Bit 11	Zsw1_11
Status word 1	Bit 12	Zsw1_12
Status word 1	Bit 13	Zsw1_13
Status word 1	Bit 14	Zsw1_14
Status word 1	Bit 15	Zsw1_15

16.6.7 FbDrive I/Os

I/Os of FbDrive

Input parameters

Parameter	Description	Type	Default
Ackn	OR operation with control word 1, bit 7	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl12	Control signal 12	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl13	Control signal 13	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl14	Control signal 14	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl15	Control signal 15	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EN	1 = Called block will be processed	BOOL	1
EnOp	Control word 1 bit 3 1= Operator control active 0 = Operator control deactivated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EnRampGen	Control word 1 bit 4 1 = Ramp generation active 0 = Ramp generation reset	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EnSp	Control word 1 bit 6 1 = Setpoint active 0 = Setpoint deactivated	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
Feature	I/O for additional functions	STRUCT • Bit 0: BOOL • ... • Bit 29 • ... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0
FlutEn	1 = Flutter suppression on	BOOL	0
FlutTmIn*	Flutter time: If a certain number of status changes occur within this time, the flutter is suppressed	INT	0
InvSp	Setpoint inverted	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Jogg1	Control word 1 bit 8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Jogg2	Control word 1 bit 9	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Local	Control word 1 bit 10 1 = Control via AS 0 = No control via AS	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS*	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Ext	Maintenance status external	DWORD	16#00000000
NoCoastSt	Control word 1 bit 1 1 = Do not de-energize and coast down drive 0 = De-energized and coast down drive	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NoQuickSt	Control word 1 bit 2 1 = No rapid stop 0 = Rapid stop	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PZDIn1	Input word 1 - ZSW1	WORD	16#0000
PZDIn2	Input word 2 - NIST_A_GLATT/ FIST_GLATT	WORD	16#0000
PZDIn3	Input word 3 - IAIST_GLATT	WORD	16#0000
PZDIn4	Input word 4 - ITIST_GLATT/ MIST_GLATT	WORD	16#0000
PZDIn5	Input word 5 - PIST_GLATT	WORD	16#0000
PZDIn6	Input word 6 - MsgNamur	WORD	16#0000

Parameter	Description	Type	Default
PZDIn2Unit	Unit of measure for process value	INT	1342
PZDIn2Scale	Scaling of the process value as a structure	STRUCT • HIGH: REAL • LOW: REAL	- • 100.0 • 0.0
PZDIn3Unit	Unit of measure for process value	INT	1342
PZDIn3Scale	Scaling of the process value as a structure	STRUCT • HIGH: REAL • LOW: REAL	- • 100.0 • 0.0
PZDIn4Unit	Unit of measure for process value	INT	1342
PZDIn4Scale	Scaling of the process value as a structure	STRUCT • HIGH: REAL • LOW: REAL	- • 100.0 • 0.0
PZDIn5Unit	Unit of measure for process value	INT	1342
PZDIn5Scale	Scaling of the process value as a structure	STRUCT • HIGH: REAL • LOW: REAL	- • 100.0 • 0.0
PZDIn6Unit	Unit of measure for process value	INT	1342
PZDIn6Scale	Scaling of the process value as a structure	STRUCT • HIGH: REAL • LOW: REAL	- • 100.0 • 0.0
SP_Li	Speed setpoint	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SP_LiScale	Scaling of the process value as a structure (device data)	STRUCT • HIGH: REAL • LOW: REAL	- • 50.0 • 0.0
Stw1	Control word	WORD	16#0000
Stw1ST	Status control word	BYTE	16#00
On	Control word 1 bit 0 1 = ON 0 = OFF	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Telegram	Control object	INT	1
TextRef	Text reference	WORD	16#0000
UnfreeRamp	Control word 1 bit 5 1 = Ramp generation can be changed 0 = Ramp generation cannot be changed	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CtrlReq	1 = Control request	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CurrentLi	Normal value PZD 3	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
CurrentScale	Scaling PZD 3	STRUCT • HIGH: REAL • LOW: REAL	- • 100.0 • 0.0
CurrentUnit	Unit of measurement PZD 3	INT	0
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number	INT	-1
F_N_Reach	1 = f or n reached or exceeded 0 = f or n not reached	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Fault	1 = Error is present	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FreeLi	Normal value PZD 6	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
FreeScale	Scaling PZD 6	STRUCT • HIGH: REAL • LOW: REAL	- • 100.0 • 0.0
FreeUnit	Unit of measurement PZD 6	INT	0

Parameter	Description	Type	Default
MsgNamur	PZD 6 in VIK-NAMUR mode	WORD	16#0000
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NoOff2	1 = De-energize and coast down not active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
NoOff3	1 = Rapid stop not active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpEn	1 = Enable setpoint operation	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Power1Li	Normal value PZD 4	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Power1Scale	Scaling PZD 4	STRUCT • HIGH: REAL • LOW: REAL	- • 100.0 • 0.0
Power1Unit	Unit of measurement PZD 4	INT	0
Power2Li	Normal value PZD 5	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
Power2Scale	Scaling PZD 5	STRUCT • HIGH: REAL • LOW: REAL	- • 100.0 • 0.0
Power2Unit	Unit of measurement PZD 5	INT	0
PZDOut1	Output word 1 - STW1	WORD	16#0000
PZDOut2	Output word 2 - NSOLL_A/ FSOLL	WORD	16#0000
RdyOn	1 = Ready to switch on 0 = Not ready to switch on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
RdyOp	1 = Ready for operation 0 = Not ready for operation	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SpeedErr	1 = Error speed within the tolerance range 0= Error speed outside the tolerance range	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SpeedLi	Normal value PZD 2	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SpeedScale	Scaling PZD 2	STRUCT • HIGH: REAL • LOW: REAL	- • 100.0 • 0.0
SpeedUnit	Unit of measurement PZD 2	INT	%
SwOn	1 = Cannot switch on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Warning	1 = Warning active 0 = No warning active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Zsw1	Status word	WORD	0
Zsw1_11	Device-specific	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Zsw1_12	Device-specific	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Zsw1_13	Device-specific	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Zsw1_14	Device-specific	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Zsw1_15	Device-specific	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

16.6.8 FbDrive block diagram

Block diagram of FbDrive

A block diagram is not provided for this block.

16.7 FbEnMe - Channel block for ET 200SP Energy Meter

16.7.1 Description of FbEnMe

Object name (type + number) and family

Type + Nubmer: FB 1908

Family: Channel

Area of application for FbEnMe

The block is used for the following application:

- Integration of energy meter in ET 200SP with firmware V3.1 in PCS7.

How it works

The block FbEnMe integrates the above mentioned module if the following conditions are fulfilled:

- In the HW Config, user selects the dataset 253 or 252 or 251 or 250 or 159.
- Module version 32 bytes input data/12 bytes output data.

Configuration

The interconnection is made symbolic to the first input word. All other interconnections are updated automatically using a wizard.

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

The block is also installed automatically in the startup OB (OB100).

Connect the symbol generated in HW Config (symbol table) for the input channel with the `PZDIn1` input parameter.

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameters `Mode`, `DataXchg`, `DataXchg1`, and `MS_Xchg` of the FbEnMe block are interconnected with the corresponding output parameters `OMODE_00`, `DXCHG_00`, `DXCHG1_00`, and `MS_XCHG_00` of the MOD_ENME block.
- The `PZD1` to `PZD32` input parameters and the `PZDOut1` to `PZDOut12` output parameters are interconnected with the corresponding symbols in the process image.
- The `MS` parameter is interconnected to the `O_MS` output parameter of the MOD_ENME diagnostic block.

Startup characteristics

The block has startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

FbEnMe modes (Page 2065)
FbEnMe functions (Page 2065)
FbEnMe error handling (Page 2067)
FbEnMe messaging (Page 2067)
FbEnMe I/Os (Page 2068)
FbEnMe block diagram (Page 2074)

16.7.2 FbEnMe modes

FbEnMe operating modes

This block does not have any modes.

See also

Description of FbEnMe (Page 2064)
FbEnMe functions (Page 2065)
FbEnMe error handling (Page 2067)
FbEnMe messaging (Page 2067)
FbEnMe I/Os (Page 2068)
FbEnMe block diagram (Page 2074)

16.7.3 FbEnMe functions

Functions of FbEnMe

The functions of this block are listed below.

Transmission of messages

If the block is to send messages to the upstream diagnostics block, you must activate this function at the `Feature` bit Transmission of status information of devices (Page 174).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130). The following functionalities are available for this block at the relevant bits:

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
29	Transmission of status information of devices (Page 174)
30	Outputting a de-energized value for block-external simulation (Page 147)

Forming the control word

When `STW1` (control word) is created and `Stw1ST` is simultaneously set at 16#80, the Boolean control inputs listed below have no effect and the control word `STW1` is set at the process value output `PZDOUT2`.

When the `Stw1ST` signal status has a bad value, the data is taken from the Boolean control inputs.

`PZDOut1`	Variant ID from the input word `PZDIn1`
`PZDOut2` Bit 6	`Opn_EnCnt`
`PZDOut2` Bit 7	`Rst_EnCnt`

Energy meter gate:

This function has to be activated in parameter 128 first before you can use the gate. Please refer to hardware manual. If the function is activated, the parameter “Opn_EnCnt” switches the gate to open or close.

"Reset on request":

With a high edge of the parameter `Rst_EnCnt` of this channel block, the values are adopted from energy meter data record 143.

Note

The data record 143 of the energy meter will not be written by the channel block FbEnMe to the device. This has to be set by the operator manually.

See also

Description of FbEnMe (Page 2064)

FbEnMe modes (Page 2065)

FbEnMe error handling (Page 2067)

FbEnMe messaging (Page 2067)

FbEnMe I/Os (Page 2068)

FbEnMe block diagram (Page 2074)

16.7.4 FbEnMe error handling

Error handling of FbEnMe

Error handling of all the blocks is described in chapter Error handling (Page 119) in the basic instructions.

The following error can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` parameter can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.

See also

Description of FbEnMe (Page 2064)

FbEnMe modes (Page 2065)

FbEnMe functions (Page 2065)

FbEnMe messaging (Page 2067)

FbEnMe I/Os (Page 2068)

FbEnMe block diagram (Page 2074)

16.7.5 FbEnMe messaging

Messaging

This block does not offer messaging.

See also

Description of FbEnMe (Page 2064)

FbEnMe modes (Page 2065)

FbEnMe functions (Page 2065)

FbEnMe error handling (Page 2067)

FbEnMe I/Os (Page 2068)

FbEnMe block diagram (Page 2074)

16.7.6 FbEnMe I/Os

I/Os of FbEnMe

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
PZDIn1	Input byte 0	BYTE	16#00
...	...	...	...
PZDIn32	Input byte 31	BYTE	16#00
MS_Release	Release for maintenance	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
MS	Maintenance state	DWORD	16#00000000
MS_Ext	External maintenance status	DWORD	16#00000000
TextRef	Text 1 reference external messages	WORD	16#0000
FlutEn	1 = Flutter suppression on	BOOL	0
FlutTmIn	Flutter suppression time	INT	0
Stw1	Control word	BYTE	16#00
Stw1ST	Control word status	BYTE	16#00
Rst_EnCnt	Reset energy meter	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
Opn_EnCnt	Open energy meter gate	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
Feature	I/O for additional functions (Page 2065)	STRUCT	-
		• Bit 0: BOOL	• 0
		• ...	• 0
		• Bit 29: BOOL	• 1
		• Bit 30: BOOL	• 1
		• Bit 31: BOOL	• 0

In/Out parameters

Parameter	Description	Type	Default
Mode0	Quality and mode - Phase 1	DWORD	16#00000000
Mode1	Quality and mode - Phase 2	DWORD	16#00000000
Mode2	Quality and mode - Phase 3	DWORD	16#00000000

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
PZDOut1	Output byte 0	BYTE	16#00
...	...	...	...
PZDOut12	Output byte 11	BYTE	16#00
Bad	1 = Process value is not valid	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
ErrorNum	Error number	INT	-1
ModErr	1 = Device/module is faulty	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
Variant	User data variant	INT	0
VarError	Configured user data variant is not supported	BOOL	1
CurL1Li	Current L1	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
CurL1LiScale	Scale factor of current L1	STRUCT	-
		• High: REAL	• 65.535
		• Low: REAL	• 0.0
CurL1LiUnit	Unit of current L1	INT	1209
CurL2Li	Current L2	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
CurL2LiScale	Scale factor of current L2	STRUCT	-
		• High: REAL	• 65.535
		• Low: REAL	• 0.0
CurL2LiUnit	Unit of current L2	INT	1209
CurL3Li	Current L3	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80

Parameter	Description	Type	Default
CurL3LiScale	Scale factor of current L3	STRUCT	-
		• High: REAL	• 65.535
		• Low: REAL	• 0.0
CurL3LiUnit	Unit of current L3	INT	1209
PL1Li	Power L1	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PL1LiScale	Scale factor of power L1	STRUCT	-
		• High: REAL	• 27648.0
		• Low: REAL	• -27648.0
PL1LiUnit	Unit of power L1	INT	0
PL2Li	Power L2	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PL2LiScale	Scale factor of power L2	STRUCT	-
		• High: REAL	• 27648.0
		• Low: REAL	• -27648.0
PL2LiUnit	Unit of power L2	INT	0
PL3Li	Power L3	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PL3LiScale	Scale factor of power L3	STRUCT	-
		• High: REAL	• 27648.0
		• Low: REAL	• -27648.0
PL3LiUnit	Unit of power L3	INT	0
PTotLi	Power L1L2L3	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PTotLiScale	Scale factor of power L1L2L3	STRUCT	-
		• High: REAL	• 27648.0
		• Low: REAL	• -27648.0
PTotLiUnit	Engineering units of power L1L2L3	INT	0
ETotLi	Energy L1L2L3	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
ETotLiScale	Scale factor of energy L1L2L3	STRUCT	-
		• High: REAL	• 4294967295.0
		• Low: REAL	• 0.0
ETotLiUnit	Unit of energy L1L2L3	INT	0

Parameter	Description	Type	Default
PFaL1Li	Power factor L1	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PFaL1LiScale	Scale factor of power factor L1	STRUCT	-
		• High: REAL	• 1.0
		• Low: REAL	• 0.0
PFaL1LiUnit	Unit of power factor L1	INT	0
PFaL2Li	Power factor L2	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PFaL2LiScale	Scale factor of power factor L2	STRUCT	-
		• High: REAL	• 1.0
		• Low: REAL	• 0.0
PFaL2LiUnit	Engineering units of power factor L2	INT	0
PFaL3Li	Power factor L3	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PFaL3LiScale	Scale factor of power factor L3	STRUCT	-
		• High: REAL	• 1.0
		• Low: REAL	• 0.0
PFaL3LiUnit	Unit of power factor L3	INT	0
PFaTotLi	Power factor L1L2L3	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PFaTotLiScale	Scale factor of power factor L1L2L3	STRUCT	-
		• High: REAL	• 1.0
		• Low: REAL	• 0.0
PFaTotLiUnit	Unit of power factor L1L2L3	INT	0
VL1Li	Voltage L1	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
VL1LiScale	Scale factor of voltage L1	STRUCT	-
		• High: REAL	• 300.0
		• Low: REAL	• 0.0
VL1LiUnit	Engineering units of voltage L1	INT	1240
VL2Li	Voltage L2	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
VL2LiScale	Scale factor of voltage L2	STRUCT	-
		• High: REAL	• 300.0
		• Low: REAL	• 0.0

Parameter	Description	Type	Default
VL2LiUnit	Unit of voltage L2	INT	1240
VL3Li	Voltage L3	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
VL3LiScale	Scale factor of voltage L3	STRUCT	-
		• High: REAL	• 300.0
		• Low: REAL	• 0.0
VL3LiUnit	Unit of voltage L3	INT	1240
VL1L2Li	Voltage L1L2	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
VL1L2LiScale	Scale factor of voltage L1L2	STRUCT	-
		• High: REAL	• 600.0
		• Low: REAL	• 0.0
VL1L2LiUnit	Unit of voltage L1L2	INT	1240
VL2L3Li	Voltage L2L3	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
VL2L3LiScale	Scale factor of voltage L2L3	STRUCT	-
		• High: REAL	• 600.0
		• Low: REAL	• 0.0
VL2L3LiUnit	Unit of voltage L2L3	INT	1240
VL3L1Li	Voltage L3L1	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
VL3L1LiScale	Scale factor of voltage L3L1	STRUCT	-
		• High: REAL	• 600.0
		• Low: REAL	• 0.0
VL3L1LiUnit	Unit of voltage L3L1	INT	1240
AcPwrLi	Active power L1	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
AcPwrLiScale	Scale factor of active power L1	STRUCT	-
		• High: REAL	• 27648.0
		• Low: REAL	• -27648.0
AcPwrLiUnit	Unit of active power L1	INT	1186
RePwrLi	Reactive power L1	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80

Parameter	Description	Type	Default
RePwrLiScale	Scale factor of reactive power L1	STRUCT • High: REAL • Low: REAL	- • 27648.0 • -27648.0
RePwrLiUnit	Engineering units of reactive power L1	INT	0
ApPwrLi	Apparent power L1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ApPwrLiScale	Scale factor of apparent power L1	STRUCT • High: REAL • Low: REAL	- • 27648.0 • -27648.0
ApPwrLiUnit	Unit of apparent power L1	INT	0
AcErgLi	Active energy L1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
AcErgLiScale	Scale factor of active energy L1	STRUCT • High: REAL • Low: REAL	- • 4294967295.0 • 0.0
AcErgLiUnit	Unit of active energy L1	INT	1175
ReErgLi	Reactive energy L1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ReErgLiScale	Scale factor of reactive energy L1	STRUCT • High: REAL • Low: REAL	- • 4294967295.0 • 0.0
ReErgLiUnit	Unit of reactive energy L1	INT	0
ApErgLi	Apparent energy L1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ApErgLiScale	Scale factor of apparent energy L1	STRUCT • High: REAL • Low: REAL	- • 4294967295.0 • 0.0
ApErgLiUnit	Unit of apparent energy L1	INT	0
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OMS_Ext	Reserved	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000

See also

Description of FbEnMe (Page 2064)
FbEnMe modes (Page 2065)
FbEnMe error handling (Page 2067)
FbEnMe messaging (Page 2067)
FbEnMe block diagram (Page 2074)

16.7.7 FbEnMe block diagram

Block diagram of FbEnMe

A block diagram is not provided for this block.

See also

Description of FbEnMe (Page 2064)
FbEnMe modes (Page 2065)
FbEnMe functions (Page 2065)
FbEnMe error handling (Page 2067)
FbEnMe messaging (Page 2067)
FbEnMe I/Os (Page 2068)

16.8 FbSwtMMS - Channel block for MM starter

16.8.1 Description of FbSwtMMS

Object name (type + number) and family

Type + number: FB 1907

Family: Channel

Area of application of FbSwtMMS

The block is used for the following applications:

- Signal processing of motor management starter with profile type 1.

NOTICE
Integration of the block into technological charts Make sure when installing the block into the technological charts that the device is a switching device of slave family 2 (switching device). Make sure that the device is installed according to profile type 1 for motor managed starters of the LVSG; otherwise, the block will not work correctly. The block is automatically connected with the process image according to the specifications of profile type 1.

How it works

The FbSwtMMS block integrates a motor management starter from any switch or starter object.

Configuration

The interconnection is made symbolic to the first input or output word. All other interconnections are updated automatically using a wizard. The input and output word must have the same beginning in HW Config.

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

The block is also installed automatically in the startup OB (OB100).

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The `MODE` input is interconnected to the relevant `OMODE` output of the MOD_SWT block.
- The `PZDIn2` input and the `PZDOut1` output are interconnected using the icons of the compact drives.
- The `DataXchg` input is interconnected to the relevant `DXCHG_00` output of the MOD_SWT block.
- The `MS` input is interconnected to the relevant `O_MS` output of the MOD_SWT block.

Note

"Configuration of direct starters with the APL channel block "FbSwtMMS" in SIMATIC PCS 7 (https://support.industry.siemens.com/cs/document/64182525/configuration-of-direct-starters-with-the-apl-channel-block-%E2%80%9Efbswtmms%E2%80%9D-in-simatic-pcs-7?dti=0&lc=en-DE)"

Startup characteristics

The block has startup characteristics.

16.8.2 FbSwtMMS modes

Operating modes of FbSwtMMS

This block does not have any modes.

16.8.3 FbSwtMMS functions

Functions of FbSwtMMS

The functions for this block are listed below.

Transmission of messages

If the block is to send messages to the upstream diagnostics block, you must activate this function at the `Feature` bit Transmission of status information of devices (Page 174).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the Feature parameter in the chapter Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
29	Transmission of status information of devices (Page 174)
30	Outputting a de-energized value for block-external simulation (Page 147)

Forming the control word

When STW1(control word) is created and Stw1ST is at 16#80, the Boolean control inputs listed below have no effect and the control word STW1 is set at the process value output PZDOUT1.

When the Stw1ST signal status has a bad value, the data is taken from the Boolean control inputs.

Rev	Control word 1 bit 0
Off	Control word 1 bit 1
Fwd	Control word 1 bit 2
StartTest	Control word 1 bit 3
StartEmerg	Control word 1 bit 4
Auto	Control word 1 bit 5
ResetTrip	Control word 1 bit 6
Ctrl7	Control word 1 bit 7
Ctrl8	Control word 1 bit 8
Ctrl9	Control word 1 bit 9
Ctrl10	Control word 1 bit 10
Ctrl11	Control word 1 bit 11
ManSpec1	Control word 1 bit 12
ManSpec2	Control word 1 bit 13
ManSpec3	Control word 1 bit 14
ManSpec4	Control word 1 bit 15

16.8.4 FbSwtMMS error handling

Error handling of FbSwtMMS

Error handling of all the blocks is described in chapter Error handling (Page 119) in the basic instructions.

The following errors can be displayed for this block:

- Error numbers

Note

If an error occurs during switching, all outputs are written as before to the field device. The inputs of the field device are displayed and the status with the value 16#00 is formed according to the MS.

Note

The outputs are locked during Drive, only the Local and Reset are transmitted to the device. The inputs react as with FbSwtMMS.

Overview of error numbers

The ErrorNum I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not processed.
0	There is no error.
1	Invalid process value (output parameter Bad = 1)

16.8.5 FbSwtMMS messaging

Messaging

This block does not offer messaging.

16.8.6 FbSwtMMS status word

Forming the status word

The `ZSW1` (status word), or the Boolean status outputs, is set via the process value input `PZDIN1`.

Status word 1	Bit 0	`RdyOn`
Status word 1	Bit 1	`RdyOp`
Status word 1	Bit 2	`OpEn`
Status word 1	Bit 3	`Fault`
Status word 1	Bit 4	`NoOff2`
Status word 1	Bit 5	`NoOff3`
Status word 1	Bit 6	`SwOn`
Status word 1	Bit 7	`Warning`
Status word 1	Bit 8	`SpeedErr`
Status word 1	Bit 9	`CtrlReq`
Status word 1	Bit 10	`F_N_Reach`
Status word 1	Bit 11	`Zsw1_11`
Status word 1	Bit 12	`Zsw1_12`
Status word 1	Bit 13	`Zsw1_13`
Status word 1	Bit 14	`Zsw1_14`
Status word 1	Bit 15	`Zsw1_15`

16.8.7 FbSwtMMS I/Os

I/Os of FbSwtMMS

Input parameters

Parameter	Description	Type	Default
Auto	Control command for remote control I bit 0.5	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl7	Control signal 7	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl8	Control signal 8	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl9	Control signal 9	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl10	Control signal 10	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ctrl11	Control signal 11	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions	STRUCT • Bit 0: BOOL • ... • Bit 29 • ... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0
FlutEn	1 = Flutter suppression activated	BOOL	0
FlutTmln*	Flutter time: If a certain number of state changes occur within this time, fluttering is suppressed.	INT	0
Fwd	Control forward I bit 0.2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
IScale	Scaling for measured current	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

Parameter	Description	Type	Default
IUnit	Unit of measurement for measured current	INT	1342
ManSpec1	Device-specific command, functionality in accordance with manufacturer specifications 1 I bit 1.4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManSpec2	Device-specific command, functionality in accordance with manufacturer specifications 2 I bit 1.5	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManSpec3	Device-specific command, functionality in accordance with manufacturer specifications 3 I bit 1.6	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManSpec4	Device-specific command, functionality in accordance with manufacturer specifications 4 I bit 1.7	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS*	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Ext	Maintenance status external	DWORD	16#00000000
Off	Control Off I bit 0.1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PZDIn1	Connectable input word 1	WORD	16#0000
PZDIn2	Connectable input word 2	WORD	16#0000
ResetTrip	Undo last operation I bit 0.6	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Rev	Control backward I bit 0.0	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StartEmerg	Start command is issued despite a pending internal error I bit 0.4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StartTest	Start command for an internal self-test I bit 0.3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Stw1	Control word	WORD	16#0000
Swt1ST	Status control word	BYTE	16#00
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrAct	Status information internal device fault Q bit 0.6	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum	Output of pending error number	INT	-1
FdkAuto	Feedback information for the remote control status Q bit 0.5	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FdkFwd	Feedback information forward Q bit 0.2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FdkOff	Feedback information Off Q bit 0.1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
FdkRev	Feedback information backward Q bit 0.0	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Imax	Value for highest measured current	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
IScaleOut	Scaling for measured current	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

Parameter	Description	Type	Default
IUnitOut	Unit of measurement for measured current	INT	0
LockTmAct	Status information, the drive is temporarily locked Q bit 0.4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManSpcO1	Device-specific feedback, functionality in accordance with manufacturer specifications 1 Q bit 1.4	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManSpcO2	Device-specific feedback, functionality in accordance with manufacturer specifications 2 Q bit 1.5	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManSpcO3	Device-specific feedback, functionality in accordance with manufacturer specifications 3 Q bit 1.6	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManSpcO4	Device-specific feedback, functionality in accordance with manufacturer specifications 4 Q bit 1.7	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManSpcO5	Device-specific feedback, functionality in accordance with manufacturer specifications 5 Q bit 1.2	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ManSpcO6	Device-specific feedback, functionality in accordance with manufacturer specifications 6 Q bit 1.3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OverIAct	Status information internal warning overload active Q bit 0.3	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PZDOut1	Connectable output word	WORD	16#0000

Parameter	Description	Type	Default
Status8	Status output 8 Q bit 1.0	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Status9	Status output 9 Q bit 1.1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
WarnAct	Status information internal maintenance active Q bit 0.7	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Zsw1	Status word	WORD	16#0000

16.8.8 FbSwtMMS block diagram

Block diagram of FbSwtMMS

A block diagram is not provided for this block.

16.9 Pcs7AnIn - Analog input channel block

16.9.1 Description of Pcs7AnIn

Object name (type + number) and family

Type + number: FB 1869

Family: Channel

Area of application for Pcs7AnIn

The block is used for the following applications:

- Signal processing of an analog input value of S7-300/400 SM analog input modules including ET 200M Ex-io module in °C.
- Processing of the redundant input channels.

How it works

The block cyclically processes all channel-specific signal functions of an analog input module.

It reads a raw analog value from the process image (partition) and converts it to its physical value or calculates a percentage value based on this raw value. Use the status at input parameter `Mode` to define the format of the raw value and how it is processed.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

The block is also installed automatically in the startup OB (OB100).

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter `Mode` is interconnected to the corresponding `OMode_xx` output parameter of the MOD block.
- The in/out parameter `DataXchg` is interconnected to the corresponding `DataXchg_xx` output parameter of the MOD block.
- The `MS` parameter is interconnected to the `O_MS` output parameter of the diagnostics block.

In case of redundancy, connection with both quality bits and connection with the redundant slave channel:

- The in parameter `PV_InSlv` is interconnected with the symbol of the slave channel.
- The in parameter `ProImQB` is interconnected with the symbol of the quality bit of the master channel.
- The in parameter `ProImQBSlv` is interconnected with the symbol of the quality bit of the slave channel.

Connect the symbol generated in HW Config (symbol table) for the input channel with the `PV_In` input parameter.

In case of redundant I/O configuration, use symbol of the channel of the master module.

Note

If you are not using the CFC function "Generate module drivers" you must set the `Mode` in/out parameter manually. Refer to the Mode Settings for SM Modules (Page 2180) section for more on this.

For the Pcs7AnIn block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Monitoring an analog process tag (AnalogMonitoring) (Page 2330)
- Cascade control with control loop monitoring through ConPerMon (CascadeControl) (Page 2316)
- Cascade control with PIDConR (CascadeR) (Page 2319)
- PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303)
- PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301)
- Override control (Page 2322)
- Override control with PIDConR (OverrideR) (Page 2324)
- PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298)
- PIDConR with safety logic and control loop monitoring (PIDConR_ConPerMon) (Page 2299)
- Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312)
- Ratio control with PIDConR (RatioR) (Page 2315)
- PID controller with Smith predictor (SmithPredictorControl) (Page 2306)
- Step controller with assigned actuator block and position feedback (StepControlActor) (Page 2308)
- Step controller with direct access to the actuator and without position feedback (StepControlDirect) (Page 2307)
- Model-based predictive control (ModPreCon) (Page 2325)
- Reversible motor with controllable speed (MotorSpeedControlled) (Page 2338)
- Dosing (Dose_Lean) (Page 2332)
- Control valve (VlvAnL) (Page 2344)
- Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl) (Page 2309)

Startup characteristics

The accept value delay is started when `CountLim` ≠ 0 .

Quality code (ET200 SP HA modules)

The quality code is read from the process image and has two states: “Good” or “Bad”.

Redundancy

The block monitors the values of the redundant signals (master and slave) with quality code and the value of the signal with “Good” quality code along with its quality code is transferred to the process.

In case both the signals deliver “Bad” quality code, the substitute value with “Bad” quality code is transferred to the process.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Pcs7AnIn block diagram (Page 2094)
Pcs7AnIn I/Os (Page 2092)
Pcs7AnIn messaging (Page 2091)
Pcs7AnIn error handling (Page 2090)
Pcs7AnIn functions (Page 2087)
Pcs7AnIn modes (Page 2086)

16.9.2 Pcs7AnIn modes

Pcs7AnIn modes

This block does not have any modes.

See also

Pcs7AnIn block diagram (Page 2094)
Pcs7AnIn I/Os (Page 2092)
Pcs7AnIn messaging (Page 2091)
Pcs7AnIn error handling (Page 2090)

Pcs7AnIn functions (Page 2087)
Description of Pcs7AnIn (Page 2084)

16.9.3 Pcs7AnIn functions

Functions of Pcs7AnIn

The functions for this block are listed below.

Checking the raw value

The nominal range sets the range for converting analog signals into digital values (raw values), depending on the measuring type and the range of the analog input module. The nominal range is defined in the hardware configuration and is automatically saved in the in/out parameter when the block driver `Mode` is created.

This includes an overshoot/undershoot range within which an analog signal can still be converted to a digital value. This range is defined in relation to the nominal range (roughly 18.5%). Outside this range an overflow or underflow occurs and output parameter `Bad` = 1 is set.

- Output parameter `PV_LoAct` = 1 is set if the value is outside the nominal low range.
- Output parameter `PV_HiAct` = 1 is set if the value is outside the nominal high range.

NAMUR limit checking (only with modules of 4 to 20 mA)

In "Life Zero" monitoring, the process signal is invalid (`Bad` = 1), if the measured current is less than 3.6 mA or greater than 21 mA (defined by NAMUR).

The NAMUR limits are set as fixed defaults for limit monitoring. You can define other limits by setting input parameter `NamurOff` = 1 and setting corresponding new limits in [mA] at the `HighLimit` and `LowLimit` input parameters. If the active limits are exceeded or undershot (`PV_HiAct` or `PV_LoAct` = 1), `Bad` = 1 is set for a "Life Zero" analog signal.

Note

The limits that can be selected must lie within the overshoot and undershoot range of the module. Values outside the NAMUR range are also possible, if the module does not automatically limit the measured values.

Obtaining the standard value

The standard value (a physical quantity) is obtained from the raw value using parameters `Scale` and `Mode`. Set two scale values on the structured parameter `Scale`.

- High scale value (`Scale.High`)
- Low scale value (`Scale.Low`)

If you are not using the CFC function "Generate module drivers" you must set the Mode in/out parameter manually. Refer to the Mode Settings for SM Modules (Page 2180) section for more on this.

The settings of the parameter Scale are copied to the output parameterScaleOut. The output parameter can be interconnected to a corresponding input parameter of a technologic block (e.g. PV_OpScale).

The standard value is obtained using a linear characteristic. Scale.Low is the lowest physical value that the process variable can take and Scale.High is the highest.

If Scale.Low = 0 and Scale.High = 100 a percentage is obtained.

Special cases when obtaining the standard value using the Scale parameter:

- If you set Scale.High = Scale.Low, you obtain the analog input module's electrical input signal (e.g., mA) according to the Mode parameter setting.
- If the raw value is already a physical quantity (for example, RTD or TC), set Scale.Low = 0 and Scale.High = 1. This will cause the raw value to be output unchanged, as a physical quantity.
- When using measuring type PTC (**P**ositive **T**emperature **C**oefficient binary evaluation of resistance thermometers), the analog value contains an encoded binary signal. The PV_Out output provides the following information:
  - If the measured resistance is within the normal range, PV_Out = 0.0
  - If the measured resistance is within the prewarning range, PV_Out = 4.0
  - If the measured resistance is in the operating range, PV_Out = 1.0
    This is only true if the input parameters are Scale.Low = 0 and Scale.High = 1 . During simulation or if the substitute value is output, you can only set the input parameters SimPV_In and SubsPV_In to 0.0 or 1.0.
- With the measuring type "External or internal comparison of thermocouple values", the raw value is adapted to the physical variable ± 80 mV range in S7 300 modules. You have to determine the temperature using the corresponding conversion tables in the module manual. The physical equivalent in [mV] is returned by the module as a raw value. Set Scale to ± 80 mV.

Holding the last value if raw value is invalid

If the block is to hold the most recent valid value when the raw value is invalid, you must activate this function at the Feature Bit Outputting last valid value if raw value is invalid (Page 153).

You can also influence this function via the input parameter DeltaVal.

- DeltaVal ≤ 0: the last value is retained and is not influenced
- DeltaVal > 0: the last or the next to last value is output

If you set the parameter DeltaVal > 0, the last PV_Out(k - 1) or next to last PV_Out(k - 2) valid output value is output (PV_Out(k) is the current value, k is the current time).

At parameter DeltaVal you can preset a permitted process value change (PV_Out) between two calls.

You have the following options:

- For invalid raw values and DeltaVal > 0:
  - If |PV_Out(k - 1) - PV_Out(k - 2)| > DeltaVal, then PV_Out = PV_Out(k - 2) (last but one valid output value is output)
  - If |PV_Out(k) - PV_Out(k - 1)| ≤ DeltaVal, then PV_Out = PV_Out(k - 1) (last valid output value is output)
- For valid raw values and DeltaVal > 0:
  - |PV_Out(k) - PV_Out(k - 1)| > DeltaVal, so for one cycle PV_Out = PV_Out(k - 1) is output, i.e. DeltaVal is used to limit the change made to the valid raw value. In addition, the signal status at the output parameter PV_Out is set to 16#60 and the output parameter is set to Bad = 0 .
    The value of DeltaVal should be selected with due care. If the value is too low, the quality code may flutter between16#80 and 16#60, regardless whether or not the raw value is OK.

Output substitute value if raw value is invalid

If the block is to output a substitute value (SubsPV_In) when the raw value is invalid, you must activate this function at the Feature Bit Output substitute value if raw value is invalid (Page 148).

Output of invalid value if raw value is invalid

If the block is to output an invalid value (PV_Out = PV_In), you must activate this function at the Feature Bit Output invalid raw value (Page 173).

This function is preset.

Value application delay

After a restart, or if the output parameter Bad changes its value from 1 to 0 the signal status and the value of output parameter PV_Out are not updated until the number of cycles for delayed application of the value (input parameter CountLim) have elapsed. During the value application delay, the signal status at the output parameter is PV_Out = 16#00 and Bad = 1. The last value is retained during the value application delay.

If CountLim = 0 , the function is deactivated.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67)

Signal status for PCS7 channel blocks

The block provides the standard function Forming and outputting the signal status for PCS 7 channel blocks (Page 118).

Simulating signals

The block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
5	Activate LowCutOff (Page 154)
10	Condition monitoring information at the channel blocks (Page 144)
28	Output invalid raw value (Page 173)
29	Output substitute value if raw value is invalid (Page 148)
30	Outputting last valid value if raw value is invalid (Page 153)

Sign-of-life monitoring

If an input value whose signal status is `16#80` (good) remains constant for a programmable time (monitoring time), the input value is detected as faulty and the outputs `Bad = 1` and `FrzVal = 1` are set. The signal status is set to `PV_Out.ST = 16#00`.

The monitoring time is set at the `FrznTmIn` input parameter in seconds. With `FrznTmIn = 0` or `FrznEn = 0` (default setting), the sign-of-life monitoring is deactivated, any pending errors are reset.

The input value is considered as faulty as long as it seen as constant. The monitoring time is restarted each time the input value is changed.

See also

Pcs7AnIn block diagram (Page 2094)

Pcs7AnIn I/Os (Page 2092)

Pcs7AnIn messaging (Page 2091)

Pcs7AnIn error handling (Page 2090)

Pcs7AnIn modes (Page 2086)

Description of Pcs7AnIn (Page 2084)

16.9.4 Pcs7AnIn error handling

Error handling of Pcs7AnIn

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Channel error
- Higher-level error
- Invalid measuring range
- Frozen input value (sign-of-life monitoring)

Channel error

At the output parameter `Bad`, channel errors are displayed with 1. Channel errors can be detected using the raw value, the NAMUR check or the sign-of-life monitoring.

`PV_LoAct` or `PV_HiAct` remain set to = 1 if a channel error occurs due to the module diagnoses "Undershoot or overshoot of the measurement range“.

Higher-level error / invalid measuring range

A higher-level error is output (output parameter `ModErr` = 1 and `Bad` = 1) if either:

- the signal status in the `High Word` of input parameter `Mode` takes the value 16#40, or
- there is an invalid measuring type in the `Low Word` of the input parameter `Mode`.

See also

Pcs7AnIn block diagram (Page 2094)
Pcs7AnIn I/Os (Page 2092)
Pcs7AnIn messaging (Page 2091)
Pcs7AnIn functions (Page 2087)
Pcs7AnIn modes (Page 2086)
Description of Pcs7AnIn (Page 2084)

16.9.5 Pcs7AnIn messaging

Messaging

This block does not offer messaging.

See also

Pcs7AnIn block diagram (Page 2094)
Pcs7AnIn I/Os (Page 2092)
Pcs7AnIn error handling (Page 2090)
Pcs7AnIn functions (Page 2087)

Pcs7AnIn modes (Page 2086)

Description of Pcs7AnIn (Page 2084)

16.9.6 Pcs7AnIn I/Os

I/Os of Pcs7AnIn

Input parameters

Parameter	Description	Type	Default
`CountLim`	Startup counter limit	INT	0
`DeltaVal`	Delta value (`PV_In` - Last valid value)	REAL	0.0
`EN`	1 = Called block will be processed	BOOL	1
`Feature`	I/O for additional functions (Page 2087)	STRUCT • Bit 0: BOOL • ... • Bit 28: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0
`FlutEn`	1 = Flutter suppression activated	BOOL	0
`FlutTmIn*`	Flutter time [s]	INT	0
`FrznEn`	1 = Sign-of-life monitoring activated	BOOL	0
`FrznTmIn`	Monitoring time in [s]	REAL	0.0
`HighLimit`	High limit used if NAMUR check (`NamurOff` = 1) is deactivated	REAL	21.5
`LowLimit`	Low limit used if NAMUR check (`NamurOff` = 1) is deactivated	REAL	3.3
`LowCutOff`	Input value Lowcutoff	REAL	0.0
`MS`	Maintenance status	DWORD	16#00000000
`MS_Release`	Release for maintenance (interconnected with `MS_Release` of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`MS_Ext`	External maintenance status	DWORD	16#00000000
`NamurOff`	1= NAMUR limit verification user-defined	BOOL	0
`PV_In`	Process value (raw value)	WORD	16#0000
`PV_InUnit`	Unit of measure for process value	INT	1001
`PV_InSlv`	Input value of the slave channel	WORD	16#0000
`ProImQB`	Quality bit from the process image	BOOL	0
`ProImQBSlv`	Quality bit from the process image of the slave channel	BOOL	0
`SampleTime`	Sampling time in [s]	REAL	0.1

Parameter	Description	Type	Default
Scale	Scaling of the process value as a structure	STRUCT • High: REAL • Low:REAL	- • 100.0 • 0.0
SimOn	1 = Simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV_In	Process value used for SimOn = 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SubsPV_In	Substitute value	REAL	0.0
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
FrznVal	Frozen process value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see Pcs7AnIn error handling (Page 2090)	INT	-1
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000

Parameter	Description	Type	Default
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_HiAct	1 = Overshoot of process value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_LcoAct	Limited to within range limits Lowcutoff	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_LoAct	1 = Undershoot of process value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	Standard value (physical variable)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_OutUnit	Unit of the process value	INT	0
RemTime	Remaining monitoring time [s]	REAL	0.0
SampleTime	Sampling time [s]	REAL	0.1
ScaleOut	Scaling of the process value for display	STRUCT • High: REAL • Low:REAL	- • 100.0 • 0.0
SimAct	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Pcs7AnIn block diagram (Page 2094)
Pcs7AnIn messaging (Page 2091)
Pcs7AnIn modes (Page 2086)
Description of Pcs7AnIn (Page 2084)

16.9.7 Pcs7AnIn block diagram

Pcs7AnIn block diagram

A block diagram is not provided for this block.

See also

Pcs7AnIn I/Os (Page 2092)
Pcs7AnIn messaging (Page 2091)
Pcs7AnIn error handling (Page 2090)
Pcs7AnIn functions (Page 2087)
Pcs7AnIn modes (Page 2086)
Description of Pcs7AnIn (Page 2084)

16.10 Pcs7AnOu - Analog output channel block

16.10.1 Description of Pcs7AnOu

Object name (type + number) and family

Type + number: FB 1870

Family: Channel

Area of application for Pcs7AnOu

The block is used for the following applications:

- Signal processing of an analog output value from S7-300/400 SM analog output groups
- Processing of the redundant output channels

How it works

The block outputs the process value as analog raw value for a process image (partition). Use the Mode in/out parameter to define how the raw value is to be obtained.

The current raw value is always output to the process image (partition).

Normally, the process value is written to both redundant channels.

If both redundant modules are working fine but cannot communicate with each other, the channel block Pcs7AnOu reduces the process value to 50%. And therefore, the two redundant channels use double the reduced process value 2 x 50% = 100% and not 2 x 100% = 200% which is incorrect.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter `Mode` is interconnected to the corresponding `OMode_xx` output parameter of the MOD block.
- The in/out parameter `DataXchg` is interconnected to the corresponding `DataXchg_xx` output parameter of the MOD block.
- The `MS` parameter is interconnected to the `O_MS` output parameter of the diagnostics block.
- The `Feature` Bit 0 (Set startup characteristics (Page 137)) is set with a default automatically when the module driver is generated.

In case of redundancy, connection with both quality bits and connection with the redundant slave channel:

- The out parameter `PV_OutSlv` is interconnected with the symbol of the slave channel.
- The in parameter `ProImQB` is interconnected with the symbol of the quality bit of the master channel.
- The in parameter `ProImQBSlv` is interconnected with the symbol of the quality bit of the slave channel.

Connect the symbol generated in HW Config (symbol table) for the output channel with the `PV_Out` output parameter.

In case of redundant I/O configuration, use symbol of the channel of the master module.

The templates of the Advanced Process Library contain an example of an Pcs7AnOu application.

Note

If you are not using the CFC function "Generate module drivers" you must set the in/out parameter `Mode` manually. Refer to the Mode Settings for SM Modules (Page 2180) chapter for more on this.

For the Pcs7AnOu block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Cascade control with control loop monitoring through ConPerMon (CascadeControl) (Page 2316)
- Cascade control with PIDConR (CascadeR) (Page 2319)
- PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303)
- PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301)
- Model-based predictive control (ModPreCon) (Page 2325)
- Reversible motor with controllable speed (MotorSpeedControlled) (Page 2338)
- Override control (Page 2322)
- Override control with PIDConR (OverrideR) (Page 2324)
- PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298)
- PIDConR with safety logic and control loop monitoring (PIDConR_ConPerMon) (Page 2299)
- Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312)
- Ratio control with PIDConR (RatioR) (Page 2315)
- PID controller with Smith predictor (SmithPredictorControl) (Page 2306)

- Control valve (VlvAnL) (Page 2344)
- Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl) (Page 2309)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Quality code (ET200 SP HA modules)

The quality code is read from the process image and has two states: “Good” or “Bad”.

Redundancy

The block monitors the values of the redundant signals (master and slave) with quality code and delivers 100% of the application program value to both the signals:

- when one of the signals delivers “Good” quality code, or
- when both signals deliver “Good” quality code and there is no “IO Redundancy Warning” (`Mode` <> 16#x5xxxxx) reported

In case both the signals deliver “Good” quality code and there is “IO Redundancy Warning” (`Mode` = 16#x5xxxxx) reported then 50% of the application program value is delivered to the process.

In case both the signals deliver “Bad” quality code, the last process value is retained.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Pcs7AnOu block diagram (Page 2105)

Pcs7AnOu I/Os (Page 2102)

Pcs7AnOu messaging (Page 2102)

Pcs7AnOu error handling (Page 2101)

Pcs7AnOu modes (Page 2099)

Pcs7AnOu functions (Page 2099)

16.10.2 Pcs7AnOu modes

Pcs7AnOu modes

This block does not have any modes.

See also

Pcs7AnOu block diagram (Page 2105)
Pcs7AnOu I/Os (Page 2102)
Pcs7AnOu messaging (Page 2102)
Pcs7AnOu error handling (Page 2101)
Description of Pcs7AnOu (Page 2096)
Pcs7AnOu functions (Page 2099)

16.10.3 Pcs7AnOu functions

Functions of Pcs7AnOu

The functions for this block are listed below.

Forming an I/O value

The peripheral value `PV_Out` is formed from:

- the scale value (input parameter `Scale`)
- the process value (input parameter `PV_In`)
- the measuring type (in/out parameter `Mode`)

Example of measuring type 4 ... 20 mA

If this measuring type is to be used, you must set the Mode parameter with 16#203 accordingly. In the measuring type, the peripheral value for 4mA is output for `PV_In` = `Scale.Low` and the peripheral value for 20 mA is output for `PV_In` = `Scale.High`.

The block writes the input parameter `Scale` directly to the output parameter `ScaleOut` and interconnects it directly to a technologic block. This can be, for example, the input parameter `MV_Opscale` of a control block.

Limiting the process or peripheral value

The peripheral value can be limited in two different ways:

- Limited to within range limits
- Limited to scale values

Limited to within range limits (physical limits of the module): If you want to restrict the peripheral value (`PV_Out`), you must activate this function via the `ScaleOff` = 1 parameter.

The peripheral value is now limited to the following range limits:

- Top: 16#7EFF (32511 dec.)
- Bottom (unipolar): 0 or
- Bottom, unipolar (4 - 20 mA; 1 - 5 V): 16#E500 (-6912 dec.)
- Bottom (bipolar): 16#8100 (-32512 dec.)

If the limits are undershot or overshot `PV_HiAct` = 1 (top) or `PV_LowAct` = 1 (bottom) is displayed at the output parameters. The signal status of the `PV_ChnST` output parameter is set to 16#78 .

Limited to scale values: If you want to restrict the peripheral value (`PV_Out`) to the scale values, you must activate this function via the `ScaleOff` = 0 parameter. You define the high and low scale limits in the `Scale` parameter. If one of the limits is violated, the limit you have entered is output at the `PV_Out` output parameter. This is displayed at the `PV_HiAct` or `PV_LoAct` = 1 output parameter. The signal status of the `PV_ChnST` output parameter is set to 16#78.

Simulating signals

The block provides the standard function Simulating signals (Page 58).

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67)

Forming the signal status for PCS7 channel blocks

The block provides the standard function Forming and outputting the signal status for PCS 7 channel blocks (Page 118).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) section. The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
10	Condition monitoring information at the channel blocks (Page 144)
30	Outputting a de-energized value for block-external simulation (Page 147)

See also

Pcs7AnOu block diagram (Page 2105)

Pcs7AnOu I/Os (Page 2102)

Pcs7AnOu messaging (Page 2102)

Pcs7AnOu error handling (Page 2101)

Pcs7AnOu modes (Page 2099)

Description of Pcs7AnOu (Page 2096)

16.10.4 Pcs7AnOu error handling

Error handling of Pcs7AnOu

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Channel error
- Higher-level error
- Invalid measuring range

Channel error

At the output parameter `Bad`, channel errors are displayed with 1. They can be detected using the raw value or the NAMUR check.

`PV_LoAct` or `PV_HiAct` remain set to = 1 if a channel error occurs due to the module diagnoses "Undershoot or overshoot of the measurement range“.

Higher-level error / invalid measuring range

A higher-level error is output (output parameter `ModErr` = 1 and `Bad` = 1) if either:

- the signal status in the `High Word` of input parameter `Mode` takes the value 16#40, or
- there is an invalid measuring type in the `Low Word` of the input parameter `Mode`.

See also

Pcs7AnOu block diagram (Page 2105)

Pcs7AnOu I/Os (Page 2102)

Pcs7AnOu messaging (Page 2102)

Pcs7AnOu modes (Page 2099)

Description of Pcs7AnOu (Page 2096)

Pcs7AnOu functions (Page 2099)

16.10.5 Pcs7AnOu messaging

Messaging

This block does not offer messaging.

See also

Pcs7AnOu block diagram (Page 2105)

Pcs7AnOu I/Os (Page 2102)

Pcs7AnOu error handling (Page 2101)

Pcs7AnOu modes (Page 2099)

Description of Pcs7AnOu (Page 2096)

Pcs7AnOu functions (Page 2099)

16.10.6 Pcs7AnOu I/Os

Pcs7AnOu I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 2099)	STRUCT • Bit 0: BOOL • ... • Bit 30: BOOL • Bit 31: BOOL	- • 0 • 0 • 1 • 0
FlutEn	1 = Flutter suppression activated	BOOL	0
FlutTmIn*	Flutter time [s]	INT	0
MS	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Ext	External maintenance status	DWORD	16#00000000

Parameter	Description	Type	Default
PV_In	Process value (raw value)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_InUnit	Unit of measure for process value	INT	1342
ProImQB	Quality bit from the process image	BOOL	0
ProImQBSlv	Quality bit from the process image of the slave channel	BOOL	0
Scale	Scaling of the process value as a structure	STRUCT • High: REAL • Low:REAL	- • 100.0 • 0.0
ScaleOff	0 = Limitation to scale limits (Scale) active 1 = Limitation to within range limits (physical limits of the module) active	BOOL	0
SimOn	1 = Simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV_In	Process value used for SimOn = 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
StartVal	Starting value that is used on starting the block if Feature bit 0 = 1.	REAL	0.0
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Reserved for future error numbers	INT	-1

Parameter	Description	Type	Default
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_ChnST	Signal status of the output channel and value of PV_Out	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_OutSlv	Output value of the slave channel	WORD	16#0000
PV_HiAct	1 = Overshoot of process value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_LoAct	1 = Undershoot of process value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	Process value	WORD	16#0000
PV_OutUnit	Unit of the process value	INT	0
ScaleOut	Scaling of the process value as a structure	STRUCT • High: REAL • Low:REAL	- • 100.0 • 0.0
SimAct	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Pcs7AnOu block diagram (Page 2105)
Pcs7AnOu messaging (Page 2102)
Pcs7AnOu error handling (Page 2101)
Pcs7AnOu modes (Page 2099)
Description of Pcs7AnOu (Page 2096)

16.10.7 Pcs7AnOu block diagram

Pcs7AnOu block diagram

A block diagram is not provided for this block.

See also

Pcs7AnOu I/Os (Page 2102)
Pcs7AnOu messaging (Page 2102)
Pcs7AnOu error handling (Page 2101)
Pcs7AnOu modes (Page 2099)
Description of Pcs7AnOu (Page 2096)
Pcs7AnOu functions (Page 2099)

16.11 FbAnTot - Analog totalizer channel block for field devices

16.11.1 Description of FbAnTot

Object name (type + number) and family

Type + number: FB 1817

Family: Channel

Area of application for FbAnTot

The FbAnTot block processes the cyclic parameters of the "Totalizer" PA profile of a PA field device according to PROFIBUS 3.0 class A and B.

How it works

The FbAnTot block reads the process value (PV) with the status byte (PV_ST) of the PROFIBUS field device (structure corresponds to the totalizer of the PA profiles) cyclically from the process image (partition). The process value is available as a physical variable. The status byte (PV_ST) contains information about the state of the PROFIBUS field device.

In addition to the status byte and to improve interconnectability, additional important detail information is provided at the output interface as structured variables or in the structures. These conform to the bit combinations specified in PROFIBUS 3.0 "General Requirements".

The PV_Set_Li and PV_Mode_Li input variables can be used to write to the process image (partition) as an option. For this, the values of input variables are transferred to the PV_Set and PV_Mode outputs, which are interconnected to the process image.

The block detects a higher-level error, for example, failure of a DP/PA link, via the Mode input parameter.

- If the high byte is Mode = 16#80, then the values in the process image (partition) are valid.
- If the high byte is Mode = 16#40 (value status = higher-level error, ModErr = TRUE), then the analog value is treated as invalid.

Configuration

The block is installed in the CFC editor in a cyclic interrupt OB (OB30 to OB38). The block is installed automatically in the startup OB (OB100).

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The symbol for the quality code of the analog input channel is interconnected with the PV_ST input parameter and other selected options.
  The in/out parameter Mode is interconnected to the corresponding OMODE_xx output parameter of the MOD block.
- The MS parameter is interconnected to the O_MS output parameter of the diagnostics block.
- The in/out parameter DataXchg is interconnected to the corresponding DataXchg output parameter of the MOD block.

Startup characteristics

The block does not have any startup characteristics.

16.11.2 FbAnTot modes

FbAnTot modes

This block does not have any operating modes.

16.11.3 FbAnTot functions

FbAnTot functions

The functions for this block are listed below.

Obtaining the standard value

The analog value of the process image (partition) is output as a standard value at the PV_Li output parameter.

Holding the last value if raw value is invalid

If the block is to hold its last valid value when the analog value is invalid or in the initialization phase of the device, you must activate this function at the Feature Bit Outputting last valid value if raw value is invalid (Page 153).

Output substitute value if raw value is invalid

If the block is to output a substitute value (SubsPV) when the analog value is invalid or in the initialization phase of the device, you must activate this function at the Feature Bit Output substitute value if raw value is invalid (Page 148).

Outputting an invalid value if analog value is invalid

If the block is to output an invalid value (PV_Li = PV), you must activate this function at the Feature Bit Output invalid raw value (Page 173). This function is preset.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67)

Signal status for Fb channel blocks

The block provides the standard function Forming and outputting the signal status for channel blocks for field devices (Page 118).

The signal status of process value PV_Li is generated from internal events such as channel errors, higher-level errors, or simulations, as well as from the signal status PV_ST, which comes directly from the device.

The signal status PV_ST can accept values of 16#00 - 16#FF.

The block recognizes a higher-level error, for example, failure of a DP/PA link, via the Mode input parameter.

- If the high byte is Mode = 16#80, then the values in the process image (partition) are valid.
- If the high byte Mode = 16#40 (value status = higher-level error, ModErr = 1), then the analog value is treated as invalid.

The measuring type set in the low word of the Mode input parameter will be ignored.

The bit combinations of signal status PV_ST are output as output parameters (BOOL values). These conform to the bit combinations specified in PROFIBUS 3.0 "General Requirements".

For FF field devices only: The values of the signal status PV_ST = 16#84 - 16#87 and 16#90 – 16#93 are evaluated by the link in the same way as signal status 16#80 – 16#83.

If the signal status PV_ST = 16#80 and a process value PV with the value 16#7FFFFFFF (invalid) are transferred from the FF field device, the block will deal with the signal status 16#00 (invalid) from the FF field device.

Simulating signals

The block provides the standard function Simulating signals (Page 58).

Configurable reactions using the Feature parameter

You can find an overview of all reactions provided by the Feature parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
28	Output invalid raw value (Page 173)

Bit	Function
29	Output substitute value if raw value is invalid (Page 148)
30	Outputting last valid value if raw value is invalid (Page 153)

Sign-of-life monitoring

If an input value whose signal status is `16#80` (good) remains constant for a programmable time (monitoring time), the input value is detected as faulty and the output parameters `Bad = 1` and `FrzVal = 1` are set.

The monitoring time is set at the `FrznTmIn` input parameter in seconds. With `FrznTmIn = 0` or `FrznEn = 0` (default setting), the sign-of-life monitoring is deactivated, any pending errors are reset.

The input value is considered as faulty as long as it seen as constant. The monitoring time is restarted each time the input value is changed.

16.11.4 FbAnTot error handling

Error handling of FbAnTot

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error. Bit is set during startup

See also

Error handling (Page 119)

16.11.5 FbAnTot messaging

Messaging

This block does not offer messaging.

16.11.6 FbAnTot I/Os

I/Os of FbAnTot

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	0
`Feature`	I/O for additional functions (Page 2107)	STRUCT • Bit 0: BOOL • ... • Bit 28: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0
`FlutEn`	1 = Flutter suppression activated	BOOL	0
`FlutTmIn*`	Flutter time [s]	INT	0
`FrznEn`	1 = Sign-of-life monitoring activated	BOOL	0
`FrznTmIn`	Monitoring time in [s]	REAL	0.0
`MS`	Maintenance status	DWORD	16#00000000
`MS_Release`	Release for maintenance (interconnected with `MS_Release` of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`MS_Ext`	External maintenance status	DWORD	16#00000000
`PV`	Process value (analog value)	REAL	0.0
`PV_Mode_Li`	Mode totalizer 0 = Balanced, 1 = pos_only, 2 = neg_only, 3 = hold	BYTE	16#00
`PV_Set_Li`	Algorithm 0 = total, 1 = reset assign 0 , 2 = Default assign `PRESET_TOT`	BYTE	16#00
`PV_ST`	Signal status for the process value	BYTE	16#80
`PV_Unit`	Unit of measure for process value	INT	1001
`SampleTime`	Sampling time in [s]	REAL	0.1
`Scale`	Scaling of the process value as a structure	STRUCT • High: REAL • Low:REAL	- • 100.0 • 0.0
`SimOn`	1 = Simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SimPV`	Process value used for `SimOn` = 1	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
`SubsPV`	Substitute value	REAL	0.0
`TextRef`	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see FbAnTot error handling (Page 2109)	INT	-1
FrznVal	Frozen process value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Li	Standard value (physical variable)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV_LiUnit	Unit of the process value	INT	0
PV_Mode	Adder in operation	BYTE	16#00
PV_Set	Add up algorithmically	BYTE	16#00
RemMonTm	Remaining monitoring time	REAL	0.0

Parameter	Description	Type	Default
`ScaleOut`	Scaling of the process value as a structure	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
`SimAct`	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

16.11.7 FbAnTot block diagram

FbAnTot block diagram

A block diagram is not provided for this block.

16.12 Pcs7DiIn - Digital input channel block

16.12.1 Description of Pcs7DiIn

Object name (type + number) and family

Type + number: FB 1871

Family: Channel

Area of application for Pcs7DiIn

The block is used for the following applications:

- Signal processing of a digital input value from S7-300/400 SM digital input groups
- Processing of the redundant input channels

How it works

The block cyclically processes all channel-specific signal functions of a digital input module.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

The block is also installed automatically in the startup OB (OB100).

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter `Mode` is interconnected to the corresponding `OMode_xx` output parameter of the MOD block.
- The in/out parameter `DataXchg` is interconnected to the corresponding `DataXchg_xx` output parameter of the MOD block.
- The `MS` parameter is interconnected to the `O_MS` output parameter of the diagnostics block.

In case of redundancy, connection with both quality bits and connection with the redundant slave channel:

- The in parameter `PV_InSlv` is interconnected with the symbol of the slave channel.
- The in parameter `ProImQB` is interconnected with the symbol of the quality bit of the master channel.
- The in parameter `ProImQBSlv` is interconnected with the symbol of the quality bit of the slave channel.

Connect the symbol generated in HW Config (symbol table) for the input channel with the `PV_In` input parameter.

In case of redundant I/O configuration, use symbol of the channel of the master module.

Note

If you are not using the CFC function "Generate module drivers" you must set the in/out parameter `Mode` manually. Refer to the Mode Settings for SM Modules (Page 2180) chapter for more on this.

If the process image (partition) also contains the value status (status bit) of the digital input channel, the corresponding symbol is connected with the input `ProImQB` and the input `SelQB` = 1 is set.

For the Pcs7DiIn block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Monitoring eight digital process tags (Digital8Monitoring) (Page 2329)
- Monitoring of a digital process tag (DigitalMonitoring) (Page 2327)
- Reversible motor with controllable speed (MotorSpeedControlled) (Page 2338)
- Step controller with assigned actuator block and position feedback (StepControlActor) (Page 2308)
- Step controller with direct access to the actuator and without position feedback (StepControlDirect) (Page 2307)
- Two-speed motor (Motor2Speed) (Page 2336)
- Reversing motor (MotorReversible) (Page 2337)
- Valve (Valve_Lean) (Page 2341)
- Two-way valve (Valve2Way) (Page 2342)
- Motor valve (ValveMotor) (Page 2343)
- Control valve (VlvAnL) (Page 2344)

Startup characteristics

The block has startup characteristics.

Quality code (ET200 SP HA modules)

The quality code is read from the process image and has two states: “Good” or “Bad”.

Redundancy

The block monitors the values of the redundant signals (master and slave) with quality code and:

- When one of the signals delivers “Good” quality code then the value of the signal with “Good” quality code along with its quality code is transferred to the process.
- When both the signals deliver “Good” quality code then the value of the master signal along with its quality code is transferred to the process.

In case both the signals deliver “Bad” quality code then either the substitution value or the last value depending on the selection of the user or neither when both are not selected is delivered to the process.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Pcs7DiIn block diagram (Page 2120)
Pcs7DiIn I/Os (Page 2118)
Pcs7DiIn messaging (Page 2118)
Pcs7DiIn error handling (Page 2117)
Pcs7DiIn functions (Page 2116)
Pcs7DiIn modes (Page 2115)

16.12.2 Pcs7DiIn modes

Pcs7DiIn modes

This block does not have any modes.

See also

Pcs7DiIn block diagram (Page 2120)
Pcs7DiIn I/Os (Page 2118)
Pcs7DiIn messaging (Page 2118)
Pcs7DiIn error handling (Page 2117)
Pcs7DiIn functions (Page 2116)
Description of Pcs7DiIn (Page 2113)

16.12.3 Pcs7DiIn functions

Functions of Pcs7DiIn

The functions for this block are listed below.

Obtaining the standard value

The digital value of the process image (partition) is output at output parameter `PV_Out` with the signal status 16#80.

If you are not using the CFC function "Generate module drivers" you must set the in/out parameter `Mode` manually for further settings. Refer to the Mode Settings for SM Modules (Page 2180) section for more on this.

Holding the last value if raw value is invalid

If the block is to hold the most recent valid value when the raw value is invalid, you must activate this function at the `Feature` Bit Outputting last valid value if raw value is invalid (Page 153).

Output substitute value if raw value is invalid

If the block is to output a substitute value (`SubsPV_In`) when the raw value is invalid, you must activate this function at the `Feature` Bit Output substitute value if raw value is invalid (Page 148).

Output of invalid value if raw value is invalid

If the block is to output an invalid value (`PV_Out` = `PV_In`), you must activate this function at the `Feature` Bit Output invalid raw value (Page 173).

This function is pre-selected.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67)

Signal status for PCS7 channel blocks

The block provides the standard function Forming and outputting the signal status for PCS 7 channel blocks (Page 118).

Simulating signals

The block provides the standard function Simulating signals (Page 58).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) section. The following functionality is available for this block at the relevant bits:

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
28	Output invalid raw value (Page 173)
29	Output substitute value if raw value is invalid (Page 148)
30	Outputting last valid value if raw value is invalid (Page 153)

See also

Pcs7DiIn block diagram (Page 2120)

Pcs7DiIn I/Os (Page 2118)

Pcs7DiIn messaging (Page 2118)

Pcs7DiIn error handling (Page 2117)

Pcs7DiIn modes (Page 2115)

Description of Pcs7DiIn (Page 2113)

16.12.4 Pcs7DiIn error handling

Error handling of Pcs7DiIn

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Channel error
- Higher-level error
- Invalid measuring range

Channel error

At the output parameter `Bad`, channel errors are displayed with 1. They can be detected using the raw value or the NAMUR check.

Higher-level error / invalid measuring range

A higher-level error is output (output parameter `ModErr` = 1) if either:

- the signal status in the `High Word` of input parameter `Mode` takes the value 16#40, or
- there is an invalid measuring type in the `Low Word` of the input parameter `Mode`.

See also

Pcs7DiIn block diagram (Page 2120)
Pcs7DiIn I/Os (Page 2118)
Pcs7DiIn messaging (Page 2118)
Pcs7DiIn functions (Page 2116)
Pcs7DiIn modes (Page 2115)
Description of Pcs7DiIn (Page 2113)

16.12.5 Pcs7DiIn messaging

Messaging

This block does not offer messaging.

See also

Pcs7DiIn block diagram (Page 2120)
Pcs7DiIn I/Os (Page 2118)
Pcs7DiIn error handling (Page 2117)
Pcs7DiIn functions (Page 2116)
Pcs7DiIn modes (Page 2115)
Description of Pcs7DiIn (Page 2113)

16.12.6 Pcs7DiIn I/Os

I/Os of Pcs7DiIn

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 2116)	STRUCT • Bit 0: BOOL • ... • Bit 28: BOOL • .... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0
FlutEn	1 = Flutter suppression activated	BOOL	0

Parameter	Description	Type	Default
FlutTmIn*	Flutter time [s]	INT	0
MS	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Ext	External maintenance status	DWORD	16#00000000
ProImQB	Quality information from the process image	BOOL	0
PV_In	Process value (raw value)	BOOL	0
PV_InSlv	Input value of the slave channel	BOOL	0
ProImQB	Quality bit from the process image	BOOL	0
ProImQBSlv	Quality bit from the process image of the slave channel	BOOL	0
SelQB	1 = Use the quality bit from the process image	BOOL	0
SimOn	1 = Simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV_In	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SubsPV_In	Substitute value	BOOL	0
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see Pcs7DiIn error handling (Page 2117)	INT	-1

Parameter	Description	Type	Default
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	Standard value (physical variable)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimAct	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Pcs7DiIn block diagram (Page 2120)
Pcs7DiIn messaging (Page 2118)
Pcs7DiIn modes (Page 2115)
Description of Pcs7DiIn (Page 2113)

16.12.7 Pcs7DiIn block diagram

Pcs7DiIn block diagram

A block diagram is not provided for this block.

See also

Pcs7DiIn I/Os (Page 2118)
Pcs7DiIn messaging (Page 2118)
Pcs7DiIn error handling (Page 2117)
Pcs7DiIn functions (Page 2116)
Pcs7DiIn modes (Page 2115)
Description of Pcs7DiIn (Page 2113)

16.13 Pcs7DiIT - Digital input channel block with time stamp

16.13.1 Description of Pcs7DiIT

Object name (type + number) and family

Type + number: FB 1872

Family: Channel

Area of application for Pcs7DiIT

The block is used for the following applications:

- Signal processing of a digital input value from S7-300/400 SM digital input modules with time stamp.
- Processing of the redundant input channels.

How it works

The cyclic block processes all channel-specific signal functions of a digital input module with configured time stamp.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

The block is also installed automatically in the startup OB (OB100).

Connect the symbol generated in HW Config (symbol table) for the input channel with the `PV_In` input parameter.

In case of redundant I/O configuration, use symbol of the channel of the master module.

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter `Mode` is interconnected to the corresponding `OMode_xx` output parameter of the MOD block.
- The in/out parameter `DataXchg` is interconnected to the corresponding `DataXchg_xx` output parameter of the MOD block.
- The `MS` parameter is interconnected to the `O_MS` output parameter of the diagnostics block.
- If the process image (partition) also contains the value status (status bit) of the digital input channel, the corresponding symbol is connected with input `ProImQB` and the input `SelQB` = 1 is set.

- The `TS_In` parameter is interconnected to the `TS_XX` output parameter of the IMDRV_TS or IM_TS_PN block.
- The in/out parameter `TS_C` is interconnected to the corresponding `TS_C_xx` output parameter of the IMDRV_TS or IM_TS_PN block.

Note

If you are not using the CFC function "Generate module drivers" you must set the in/out parameter `Mode` manually. Refer to the Mode Settings for SM Modules (Page 2180) chapter for more on this.

In case of redundancy, connection with both quality bits and connection with the redundant slave channel:

- The in parameter `PV_InSlv` is interconnected with the symbol of the slave channel.
- The in parameter `ProImQB` is interconnected with the symbol of the quality bit of the master channel.
- The in parameter `ProImQBSlv` is interconnected with the symbol of the quality bit of the slave channel.

If the time stamp for the input channel is deactivated, the aforementioned connections to the block IMDRV_TS or IM_TS_PN are not used. The block can be used as a simple channel block. When the charts are compiled, a warning is issued as the module driver is generated.

Refer to the section on configuring the channel blocks for information on configuration.

Startup characteristics

The block has startup characteristics.

Quality code (ET200 SP HA modules)

The quality code is read from the process image and has two states: “Good” or “Bad”.

Redundancy

The block monitors the values of the redundant signals (master and slave) with quality code and:

- When one of the signals delivers “Good” quality code then the value of the signal with “Good” quality code along with its quality code is transferred to the process.
- When both the signals deliver “Good” quality code then the value of the master signal along with its quality code is transferred to the process.

In case both the signals deliver “Bad” quality code then either the substitution value or the last value depending on the selection of the user or neither when both are not selected is delivered to the process.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Pcs7DiIT block diagram (Page 2129)

Pcs7DiIT I/Os (Page 2126)

Pcs7DiIT messaging (Page 2126)

Pcs7DiIT error handling (Page 2125)

Pcs7DiIT functions (Page 2123)

Pcs7DiIT modes (Page 2123)

16.13.2 Pcs7DiIT modes

Pcs7DiIT modes

This block does not have any modes.

See also

Pcs7DiIT block diagram (Page 2129)

Pcs7DiIT I/Os (Page 2126)

Pcs7DiIT messaging (Page 2126)

Pcs7DiIT error handling (Page 2125)

Pcs7DiIT functions (Page 2123)

Description of Pcs7DiIT (Page 2121)

16.13.3 Pcs7DiIT functions

Functions of Pcs7DiIT

The functions for this block are listed below.

Obtaining the standard value

The digital value of the process image (partition) is output at output parameter `PV_Out` with the signal status 16#80 .

If you are not using the CFC function "Generate module drivers" you must set the in/out parameter `Mode` manually for further settings. Refer to the section Mode Settings for SM Modules (Page 2180).

Holding the last value if raw value is invalid

If the block is to hold the most recent valid value when the raw value is invalid, you must activate this function at the `Feature` Bit Outputting last valid value if raw value is invalid (Page 153).

Output substitute value if raw value is invalid

If the block is to output a substitute value (`SubsPV_In`) when the raw value is invalid, you must activate this function at the `Feature` Bit Output substitute value if raw value is invalid (Page 148).

Output of invalid value if raw value is invalid

If the block is to output an invalid value (`PV_Out` = `PV_In`), you must activate this function at the `Feature` Bit Output invalid raw value (Page 173).

This function is pre-selected.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67)

Signal status for PCS7 channel blocks

The block provides the standard function Forming and outputting the signal status for PCS 7 channel blocks (Page 118).

Simulating signals

The block provides the standard function Simulating signals (Page 58).

Time stamp

The block provides the standard function Time stamp (Page 201).

Interconnect the signal with the time stamp from the I/O devices with input parameter `TS_In`.

Interconnect input parameter `TS_In` with the channel-specific output parameter `TS_Oxx` of block IMDRV_TS or IM_TS_PN.

Interconnect in/out parameter `TS_C` with the channel-specific output parameter `TS_Cxx` of block IMDRV_TS or IM_TS_PN. This happens automatically when the CFC function "Generate module drivers" is used.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the Feature parameter in the section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
28	Output invalid raw value (Page 173)
29	Output substitute value if raw value is invalid (Page 148)
30	Outputting last valid value if raw value is invalid (Page 153)

See also

Pcs7DiIT block diagram (Page 2129)

Pcs7DiIT I/Os (Page 2126)

Pcs7DiIT messaging (Page 2126)

Pcs7DiIT error handling (Page 2125)

Pcs7DiIT modes (Page 2123)

Description of Pcs7DiIT (Page 2121)

16.13.4 Pcs7DiIT error handling

Error handling of Pcs7DiIT

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Channel error
- Higher-level error
- Invalid measuring range

Channel error

At the output parameter `Bad`, channel errors are displayed with 1. The channel error is generated from the signal status `PV_ST`.

Higher-level error / invalid measuring range

A higher-level error is displayed at the output parameters `ModErr` and `Bad` with 1 if the signal status at `High Word` of the input parameter `Mode` accepts the value 16#40.

A higher-level error is also present when an incorrect measuring type is entered in `Low Word` of the input parameter `Mode`.

See also

Pcs7DiIT block diagram (Page 2129)
Pcs7DiIT I/Os (Page 2126)
Pcs7DiIT messaging (Page 2126)
Pcs7DiIT functions (Page 2123)
Pcs7DiIT modes (Page 2123)
Description of Pcs7DiIT (Page 2121)

16.13.5 Pcs7DiIT messaging

Messaging

This block does not offer messaging.

See also

Pcs7DiIT block diagram (Page 2129)
Pcs7DiIT I/Os (Page 2126)
Pcs7DiIT error handling (Page 2125)
Pcs7DiIT functions (Page 2123)
Pcs7DiIT modes (Page 2123)
Description of Pcs7DiIT (Page 2121)

16.13.6 Pcs7DiIT I/Os

I/Os of Pcs7DiIT

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`Feature`	I/O for additional functions (Page 2123)	STRUCT • Bit 0: BOOL • ... • Bit 28: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0
`FlutEn`	1 = Flutter suppression activated	BOOL	0

Parameter	Description	Type	Default
FlutTmIn*	Flutter time [s]	INT	0
MS	Maintenance status	DWORD	16#00000000
MS_Ext	External maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_In	Process value (raw value)	BOOL	0
PV_InSlv	Input value of the slave channel	BOOL	0
ProImQB	Quality bit from the process image	BOOL	0
ProImQBSlv	Quality bit from the process image of the slave channel	BOOL	0
SimOn	1 = Simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV_In	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SelQB	1 = Use the quality bit from the process image	BOOL	0
SubsPV_In	Substitute value	BOOL	0
TextRef	Text reference	WORD	16#0000
TS_In	Time stamp of IMDRV_TS; The time is in ISP format	STRUCT • MsgSig: BOOL • TriInf: BOOL • HdSh: BOOL • ST: BYTE • TS0: DWORD • TS1: DWORD	- • 0 • 0 • 0 • 16#80 • 16#00000000 • 16#00000000
TS_In_PN	Time stamp from IM_TS_PN The time is in ISP format	STRUCT • MsgSig: BOOL • TriInf: BOOL • HdSh: BOOL • ST: BYTE • TS0: DWORD • TS1: DWORD • TS2: DWORD	- • 0 • 0 • 0 • 16#80 • 16#00000000 • 16#00000000 • 16#00000000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
TS_C	Time-stamp communication	BYTE	16#00

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Reserved for future error numbers	INT	-1
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	1 = Standard value (physical variable)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimAct	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
TS_Out	Time stamp	STRUCT • MsgSig: BOOL • TriInf: BOOL • HdSh: BOOL • ST: BYTE • TS0: DWORD • TS1: DWORD • Link: BOOL	- • 0 • 0 • 0 • 16#80 • 16#00000000 • 16#00000000 • 0

See also

Pcs7DiIT block diagram (Page 2129)
Pcs7DiIT messaging (Page 2126)
Pcs7DiIT error handling (Page 2125)
Pcs7DiIT modes (Page 2123)
Description of Pcs7DiIT (Page 2121)

16.13.7 Pcs7DiIT block diagram

Pcs7DiIT block diagram

A block diagram is not provided for this block.

See also

Pcs7DiIT I/Os (Page 2126)
Pcs7DiIT messaging (Page 2126)
Pcs7DiIT error handling (Page 2125)
Pcs7DiIT functions (Page 2123)
Pcs7DiIT modes (Page 2123)
Description of Pcs7DiIT (Page 2121)

16.14 Pcs7DiOu - Digital output channel block

16.14.1 Description of Pcs7DiOu

Object name (type + number) and family

Type + number: FB 1873

Family: Channel

Area of application for Pcs7DiOu

The block is used for the following applications:

- Signal processing of a digital output value from S7-300/400 SM digital output groups.
- Processing of the redundant output channels.

How it works

The cyclic block processes all channel-specific signal functions of a digital output module.

Normally, the process value is written to both redundant channels.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

The block is also installed automatically in the startup OB (OB100).

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter `Mode` is interconnected to the corresponding `OMode_xx` output parameter of the MOD block.
- The in/out parameter `DataXchg` is interconnected to the corresponding `DataXchg_xx` output parameter of the MOD block.
- The `MS` parameter is interconnected to the `O_MS` output parameter of the diagnostics block.
- The `Feature` Bit 0 (Set startup characteristics (Page 137)) is set with a default automatically when the module driver is generated.

In case of redundancy, connection with both quality bits and connection with the redundant slave channel:

- The out parameter `PV_OutSlv` is interconnected with the symbol of the slave channel.
- The in parameter `ProImQB` is interconnected with the symbol of the quality bit of the master channel.
- The in parameter `ProImQBSlv` is interconnected with the symbol of the quality bit of the slave channel.

Connect the symbol generated in HW Config (symbol table) for the output channel with the `PV_Out` output parameter.

In case of redundant I/O configuration, use symbol of the channel of the master module.

The templates of the Advanced Process Library contain an example of an Pcs7DiOu application.

Note

If you are not using the CFC function "Generate module drivers" you must set the in/out parameter `Mode` manually. Refer to the Mode Settings for SM Modules (Page 2180) chapter for more on this.

For the Pcs7DiOu block, the Advanced Process Library contains templates for process tag types as examples with various application scenarios for this block.

Examples of process tag types:

- Dosing (Dose_Lean) (Page 2332)
- Two-speed motor (Motor2Speed) (Page 2336)
- Reversing motor (MotorReversible) (Page 2337)
- Reversible motor with controllable speed (MotorSpeedControlled) (Page 2338)
- Step controller with assigned actuator block and position feedback (StepControlActor) (Page 2308)
- Step controller with direct access to the actuator and without position feedback (StepControlDirect) (Page 2307)
- Valve (Valve_Lean) (Page 2341)
- Two-way valve (Valve2Way) (Page 2342)
- Motor valve (ValveMotor) (Page 2343)

Startup characteristics

Use the `Feature` Bit Set startup characteristics (Page 137) to define the startup characteristics of this block.

Quality code (ET200 SP HA modules)

The quality code is read from the process image and has two states: “Good” or “Bad”.

Redundancy

The block monitors the values of the redundant signals (master and slave) with quality code and the application program value is deliverd to both the signals when one or both of the signals deliver(s) “Good” quality code along with the quality code is transferred to the process.

In case both the signals deliver “Bad” quality code, the application program value is delivered to the process with “Bad” quality code.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Pcs7DiOu block diagram (Page 2136)
Pcs7DiOu I/Os (Page 2134)
Pcs7DiOu messaging (Page 2134)
Pcs7DiOu error handling (Page 2133)
Pcs7DiOu functions (Page 2132)
Pcs7DiOu modes (Page 2132)

16.14.2 Pcs7DiOu modes

Pcs7DiOu modes

The block does not have any operating modes.

See also

Pcs7DiOu block diagram (Page 2136)
Pcs7DiOu I/Os (Page 2134)
Pcs7DiOu messaging (Page 2134)
Pcs7DiOu error handling (Page 2133)
Pcs7DiOu functions (Page 2132)
Description of Pcs7DiOu (Page 2130)

16.14.3 Pcs7DiOu functions

Functions of Pcs7DiOu

The functions for this block are listed below.

Forming a peripheral value

The digital value is written to the process image (partition). The signal status of the process value (`PV_ChnST`) is set to "good" (16#80).

Simulating signals

The block provides the standard function Simulating signals (Page 58).

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67)

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in section Configurable functions using the Feature I/O (Page 130). The following functionality is available for this block at the relevant bits:

Bit	Function
0	Set startup characteristics (Page 137)
10	Condition monitoring information at the channel blocks (Page 144)
30	Outputting a de-energized value for block-external simulation (Page 147)

See also

Pcs7DiOu block diagram (Page 2136)

Pcs7DiOu I/Os (Page 2134)

Pcs7DiOu messaging (Page 2134)

Pcs7DiOu error handling (Page 2133)

Pcs7DiOu modes (Page 2132)

Description of Pcs7DiOu (Page 2130)

16.14.4 Pcs7DiOu error handling

Error handling of Pcs7DiOu

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Channel error
- Higher-level error
- Invalid measuring range

Channel error

At the output parameter `Bad`, channel errors are displayed with 1. The channel error is generated from the signal status `PV_ST`.

Higher-level error / invalid measuring range

A higher-level error is output (output parameter `ModErr` = 1) if either:

- the signal status in the `High Word` of input parameter `Mode` takes the value 16#40, or
- there is an invalid measuring type in the `Low Word` of the input parameter `Mode`.

See also

Pcs7DiOu block diagram (Page 2136)
Pcs7DiOu I/Os (Page 2134)
Pcs7DiOu messaging (Page 2134)
Pcs7DiOu functions (Page 2132)
Pcs7DiOu modes (Page 2132)
Description of Pcs7DiOu (Page 2130)

16.14.5 Pcs7DiOu messaging

Messaging

This block does not offer messaging.

See also

Pcs7DiOu block diagram (Page 2136)
Pcs7DiOu I/Os (Page 2134)
Pcs7DiOu error handling (Page 2133)
Pcs7DiOu functions (Page 2132)
Pcs7DiOu modes (Page 2132)
Description of Pcs7DiOu (Page 2130)

16.14.6 Pcs7DiOu I/Os

I/Os of Pcs7DiOu

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional functions (Page 2132)	STRUCT • Bit 0: BOOL • ... • Bit 30: BOOL • Bit 31: BOOL	- • 0 • 0 • 1 • 0
FlutEn	1 = Flutter suppression activated	BOOL	0
FlutTmIn*	Flutter time [s]	INT	0
MS	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance (interconnected with MS_Release of the technology block)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Ext	External maintenance status	DWORD	16#00000000
PV_In	Process value (raw value)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ProImQB	Quality bit from the process image	BOOL	0
ProImQBSlv	Quality bit from the process image of the slave channel	BOOL	0
SimOn	1 = Simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV_In	Process value used for SimOn = 1	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
StartVal	Starting value that is used on starting the block if Feature bit 0 = 1.	BOOL	0
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Description	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
Mode	Value status and measuring type	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
Bad	1 = Process value is not valid	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number. For error numbers that can be output by this block, see Pcs7DiOu error handling (Page 2133)	INT	-1
ModErr	1 = Device/module is faulty	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	1 = Field device is undergoing maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_ChnST	Signal status of the output channel and value of PV_Out	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	Process value	BOOL	0
PV_OutSlv	Output value of the slave channel	BOOL	0
SimAct	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

Pcs7DiOu block diagram (Page 2136)

Pcs7DiOu messaging (Page 2134)

Pcs7DiOu modes (Page 2132)

Description of Pcs7DiOu (Page 2130)

16.14.7 Pcs7DiOu block diagram

Pcs7DiOu block diagram

A block diagram is not provided for this block.

See also

Pcs7DiOu I/Os (Page 2134)
Pcs7DiOu messaging (Page 2134)
Pcs7DiOu error handling (Page 2133)
Pcs7DiOu functions (Page 2132)
Pcs7DiOu modes (Page 2132)
Description of Pcs7DiOu (Page 2130)

16.15 Pcs7Cnt1 - Controlling and reading FM 350 modules

16.15.1 Description of Pcs7Cnt1

Object name (type + number)

Type + number: FB 1833

Family: Channel

Area of application

The Pcs7Cnt1 block is used for controlling and reading count or measured values of an FM 350-1 or FM 350-2 module.

How it works

"FM 350" refers to the FM 350-1 and FM 350-2 modules in the following.

- The block only communicates through the process image for the FM 350-1. The data are written and read continuously.
- The control and status information and selected count and measured values are contained in the process image for the FM 350-2. The remaining count and measured values can be read via data records.
  In HW Config (User_Type1 and User_Type2) you define how the count or measured values will be saved in the process image. The `LoadPv1` and `CmpVx` parameters are loaded from the FM_CNT block to the FM 350-2 using data records. The writing of the parameters is first triggered in the subsequent cycle of the FM_CNT block.

If an FM 350-2 module is being used, the block writes the `LoadPV1` (load count value immediately) or `CmpVx` (comparison value) parameters to the module (x = channel number) via data records. If the parameter `LoadDir` = 1 is set in the block, it writes `LoadPV1`. If `LoadPre` = 1 is set, it writes in preparation `LoadPV1`. The `CmpVx` parameter is written after every change.

The `Mode` input indicates in what format the count and/or measured value is available in the process image. If the high word of the input parameter `Mode` = 16#40xxxx (value status = higher-level error, `ModErr`= 1), the count or measured value is treated as invalid.

The measured values are written to the corresponding outputs, `PV1`, `PV1_Li` as well as `PV2` and `PV2_Li`; a differential value between the old value and new value is formed for the `PV1` via the last cycle and sent at the `PV1CycLi` output. If the read values are in order, the status of the outputs is set to 16#80.

The units for PV1_UnitLi, ScalePV1_Li, PV2_UnitLi and ScalePV2_Li are set with the extension "_IN" via the inputs of the same name.

Note

Status CmpVal0(comparator 1), CmpVal1 (comparator 2), ZeroSt (zero crossing), OFlow (overflow) and UFlow (underflow) are automatically acknowledged. They are active for at least one cycle.

The measured value is output as a numeric value by the FM 350. Additional information on this is available in the manual for the module.

The block also supplies the counted pulses of PV1(PV1_Li) at the PV1CycLi output with each block call.

The LoadPV1 input parameter is the relevant value which is passed to the module. If, however, LoadPV1_Li is interconnected or the status of LoadPV1_Li is 16#80 , the value is written to LoadPV1 and applied.

Note

The display in PCS 7 is limited by the data type DINT. However, the settings in the module are permitted unsigned up to 32 bits.

Configuration

When the CFC function "**Generate module drivers**" is used, the following occurs automatically:

- The Laddr and Channel inputs are configured.
- The Mode input is interconnected with the OMODEx output of the FM_CNT block.
- The FM_DATA structure is interconnected with the structure of the same name of the FM_CNT block.
- The DataXchg input is interconnected to the relevant DXCHG_00 output of the MOD_D1 block.
- The MS input is interconnected to the relevant O_MS output of the MOD_D1 block.

The block is installed in the CFC editor in a cyclic interrupt OB (OB30 to OB38) .

Startup characteristics

The block does not have any startup characteristics.

Status word allocation

This block does not have the Status parameter.

See also

Pcs7Cnt1 I/Os (Page 2143)

16.15.2 Pcs7Cnt1 modes

Pcs7Cnt1 operating modes

This block does not have any modes.

16.15.3 Pcs7Cnt1 functions

Functions of Pcs7Cnt1

The functions for this block are listed below.

Addressing

1. Create icons for the required count or measured values in the icon table, in accordance with the base address of the FM 350 module. Please note the following:

- FM 350-1: Count or measured value is always in the process image:
  - Select "ED base address" of the module (e.g. ED512) as the address.
- FM 350-2: Count or measured value of the desired channel is in the process image:
  - In "HW Config FM 350-2 configure counter" you can specify where count or measured values will be stored in the process image. Depending on the configuration of User_Type1 or User_Type2, you must select EW for WORD or ED for DWORD. The address is calculated according to the following table:

Count or measured value is defined as:	Measured or count value is in User_Type1:	Measured or count value is in User_Type2:
DWORD or LOW WORD	FM 350-2 base address + 8 bytes	FM 350-2 base address + 12 bytes
HIGH WORD	FM 350-2 base address + 10 bytes	FM 350-2 base address + 14 bytes
Example: The desired count value of channel 2 is in User_Type2 in the high word. The address is calculated for a base address of 512 as: Address = EW 526.		

- FM 350-2: Count or measured value of the desired channel is not in the process image:
  - Select as address: Input word "Base address of the module + channel number" interconnected (e.g. base address = 512, channel number = 5; EW517).

2. Connect the input `Connect` in the CFC chart with the previously created icon via "Interconnection to address...".

Count and measured values that are not in the process image of the FM 350-2 are read out of the module cyclically as a data record, if the inputs `PV1_EN` or `PV2_EN` = 1. Both inputs should be set to 0 for performance reasons, if count or measured value are not needed for the

channel in the user program. This prevents count or measured values from being read via data records, if they are not in the process image.

Note

Even if the `PV1_EN` or `PV2_EN` inputs are not set, a read can be performed via data records if the `PV1_EN` or `PV2_EN` are set inputs are set at a different instance of Pcs7Cnt1 (other channel) of the associated FM350-2.

Simulation

The block provides the standard function Simulating signals (Page 58)

With `SimOn = 1` , the simulation values `SimPV1` and `SimPV2` are written to the outputs `PV1`, `PV1_Li` as well as `PV2` and `PV2_Li` . The status of the outputs is set to 16#60 and `Bad = 1` in the process.

Simulation takes highest priority.

If the block is in simulation state, `SimAct = 1` is set.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
28	Output invalid raw value (Page 173)
29	Output substitute value if raw value is invalid (Page 148)
30	Outputting last valid value if raw value is invalid (Page 153)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for channel blocks for field devices (Page 118).

Value	Meaning
16#80	Valid value
16#60	Manipulated value (for example, substitute value, simulation, last valid value)
16#00	Invalid value

The worst signal status ST_Worst for the block is formed from the following parameters:

- LoadDir.ST
- LoadPre.ST
- LoadPV1_Li.ST
- CntRun.ST
- CntDir.ST
- PV1_Li.ST
- PV2_Li.ST

16.15.4 Pcs7Cnt1 error handling

Error handling of Pcs7Cnt1

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Higher-level error

Overview of error numbers

The ErrorNum I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
21	Pulse counter overflow

Higher-level error

A higher-level error is displayed at the output parameters ModErr and Bad with 1 if the signal status in the high word of the input parameter Mode assumes the value 16#40.

In addition, at the output parameter PV_Li of the signal status, either 16#00 (in the event of an error) or 16#60 (for manipulated value (for example, substitute value, simulation, last valid value) is output.

16.15.5 Pcs7Cnt1 messaging

Messaging

This block does not offer messaging.

16.15.6 Pcs7Cnt1 I/Os

I/Os of Pcs7Cnt1

Input parameters

Parameter	Meaning	Type	Default
Channel	Channel FM 350	INT	0
CmpV0	New comparison value 0	DINT	0
CmpV1	New comparison value 1	DINT	0
CmpV2	New comparison value 2	DINT	0
CmpV3	New comparison value 3	DINT	0
Connect	Connection value (WORD or DWORD)	ANY	
CtrlDO0	1 = Enable digital output DO	BOOL	1
CtrlDO1	1 = Enable digital output DO1 (FM 350-1 or FM 350-2 only, dosing mode)	BOOL	1
CtrlDO2	1 = Enable digital output DO2 (FM 350-2 only, dosing mode)	BOOL	1
CtrlDO3	1 = Enable digital output DO3 (FM 350-2 only, dosing mode)	BOOL	1
EnSetDn	1 = enable for setting in reverse direction	BOOL	1
EnSetUp	1 = Enable for setting in forward direction	BOOL	1
`Feature`	I/O for additional functions (Page 2140)	STRUCT • Bit 0: BOOL • ... • Bit 28: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0
FlutEn	1 = Flutter suppression active	BOOL	0
FlutTmIn*	Flutter suppression time	INT	0
Laddr	Logical address FM 350	INT	0
LoadPv1*	Load counter	DINT	0
LoadPV1_Li	Load counter structured	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
LoadDir*	1 = Load counter directly	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Meaning	Type	Default
LoadPre*	1 = Load prepared counter	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Ext	External maintenance status	DWORD	16#00000000
PV1En	1 = Counted value used	BOOL	1
PV2En	1 = Measured value used	BOOL	1
PV1_Unit	Unit of the current value	INT	0
PV2_Unit	Unit of the measured value	INT	0
RstOpErr	1 = Reset operator error	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstSync*	1 = Reset synchronization	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ScalePV1	Range of the current count value	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
ScalePV2	Range of current measured value	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
SetDO0	1 = Open DO0	BOOL	1
SetDO1	1 = Open DO1 (FM 350-1 or FM 350-2 only, dosing mode)	BOOL	1
SetDO2	1 = Open DO2 (FM 350-2 only, dosing mode)	BOOL	1
SetDO3	1 = Open DO3 (FM 350-2 only, dosing mode)	BOOL	1
SimPV1	Simulation count value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimPv2	Simulation measured value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimON	1 = simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Stopgate	1 = General GATE stop	BOOL	0
StopGate	1 = Stop gate	BOOL	1
SubsPV1	Count substitute value	DINT	0
SubsPV2	Measured substitute value	DINT	0

Parameter	Meaning	Type	Default
SubsOn	1 = substitute value active	BOOL	0
SwGateEn	1 = Enable SW-TOR	BOOL	0
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Meaning	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000
Mode	Operating mode	DWORD	16#00000000

Output parameters

Parameter	Meaning	Type	Default
Bad	1 = invalid values	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CmpValL0	1 = comparison value 1	BOOL	0
CmpVal1	1 = comparison value 2	BOOL	0
CmpVal2	1 = comparison value 3	BOOL	0
CmpVal3	1 = comparison value 4	BOOL	0
CntDir	Direction counter	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CntRun	1 = status counter working	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CntSync	1 = status counter synchronized	BOOL	0
ErrorNum	Parameter error	INT	-1
IntGate	1 = status of internal TOR	BOOL	0
ModErr	1 = higher-level error	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV1	Current load or LATCH value/current measured value	DINT	0

Parameter	Meaning	Type	Default
PV1CycLi	Pulses per cycle	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PV1_Li	Current count value structured	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PV1_UnitLi	Unit of the current count value	INT	0
PV2	Current measured value	DINT	0
PV2_Li	Measured value structured	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#80
PV2_UnitLi	Unit of the measured value	INT	0
NewLatch	1 = new LATCH value (in clock-synchronous mode only)	BOOL	0
OpErr	1 = operator error	BOOL	0
OosAct	Field device out of service, maintenance	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
OFlow	1 = status overflow	BOOL	0
RstSync	1 = Reset synchronization	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
ScalePV1_Li	Range of current counter values	STRUCT	-
		• High: REAL	• 100.0
		• Low: REAL	• 0.0
ScalePV2_Li	Range of the measured values	STRUCT	-
		• High: REAL	• 100.0
		• Low: REAL	• 0.0
SetDi	1 = status digital input DI set	BOOL	0
SimAct	1 = simulation values	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
StartDI	1 = digital input DI start	BOOL	0
StCmpV0	1 = Saved status of comparator 1	BOOL	0
StCmpV1	1 = Saved status of comparator 2	BOOL	0
StCmpV2	1 = Saved status of comparator 3;	BOOL	0
StCmpV3	1 = Saved status of comparator 4;	BOOL	0
StopDI	1 = status digital input DI stop	BOOL	0
ST_Worst	Worst signal status	BYTE	16#80
SwGate	1 = Software gate	BOOL	0
UFlow	1 = status underflow	BOOL	0
ZeroSt	1 = status zero crossing	BOOL	0

See also

Pcs7Cnt1 functions (Page 2140)

16.15.7 Pcs7Cnt1 block diagram

Block diagram of Pcs7Cnt1

A block diagram is not provided for this block.

16.16 Pcs7Cnt2 - Control and read an 8-DI_NAMUR module of the ET 200iSP

16.16.1 Description of Pcs7Cnt2

Object name (type + number)

Type + number: FB 1834

Family: Channel

Area of application

The Pcs7Cnt2 block is used to control and read a count or frequency value from an 8-DI NAMUR module of the ET 200iSP. The block supports the following configurations of the module:

- 2 counters or 1 counter cascaded
- 2 frequency measurements

How it works

Depending on the mode setting of the module in HW Config, the user data of the module are stored in the process image. The Pcs7Cnt2 block differentiates between the following modes:

MODE (low word)	Operating mode	HW Config setting: "Config-uration"	HW Config setting: “Channel (0 to 1) mode”
1	Counter (16 bits) without control function by means of digital signals	(Channel 0 to 1): COUNT(Channel 2 to 7): DI	Periodic or normal count functions (up or down counter)
2	Counter (32 bits) without control function by means of digital signals	(Channel 0 to 1): COUNT (Channel 2 to 7): DI	Cascade function (channel 0 only) (down counter)
3	Counter (16 bits) with control function by means of digital signals	(Channel 0 to 1): COUNT (Channel 2 to 7): CONTROL	Periodic or normal count functions (up or down counter)
4	Counter (32 bits) with control function by means of digital signals	(Channel 0 to 1): COUNT (Channel 2 to 7): CONTROL	Cascade function (channel 0 only) (down counter)
5	Frequency (16 bits)	(Channel 0 to 1): TRACE (Channel 2 to 7): DI	-

The driver generator sets the module mode configured in HW Config at the `MODE` input of the `MOD_D1` block on the appropriate channel of the module. The `Mode` input indicates in what format the count or frequency value is available in the process image. If the high word of the input parameter `MODE = 16#40xxxx` (value status = higher-level error, `ModErr = 1`), the count or frequency value is treated as invalid.

Depending on the mode, either two independent counters (16 bits) or one counter (32 bits) exist in the process image. The `Channel` input specifies the module counter for which the block is responsible.

The counter functions can be controlled by signals that can be influenced both over the digital inputs of the module or over the user data of the process image.

Note

Please note that the signals of the digital inputs are ORed with the equivalent signals from the PIO in the module.

The following signals are available:

Block input	Module	Meaning
-	Z1	Counter pulse counter 1
-	Z2	Counter pulse counter 2
`StopGate (Channel = 0)`	TOR1	With the active GATE signal, an active count operation can be interrupted. The GATE = "1" signal stops the count operation despite pending count pulses. At the same time, the assigned output is deactivated if it was active. This state remains until the GATE signal is set to "0". The output is brought to the previous state and the count operation is continued. The GATE signal is subordinate to the RSO and RSC signals, in other words, the RSO and RSC signals have the effect described above regardless of an active GATE signal.
`StopGate (Channel = 1)`	TOR2	See description of "GATE 1"
`RstPV1 (Channel = 0)`	RSZ1	The rising edge of the RSC signal sets the count of the assigned channel as follows: • When counting up (normal counter function), back to zero • When counting down (periodic counter function and cascade function), to the defined setpoint When counting down (periodic counter function and cascade function), any output that is set is also reset.
`RstPV1 (Channel = 1)`	RSZ2	See description of "RSC1"
`RstDO (Channel = 0)`	RSA1	On the rising edge of the RSO signal, the assigned output can be reset. The count is not influenced by setting RSO.
`Rst_DO (CHANNEL = 1)`	RSA2	See description of "RSO2"

The in/out parameters `RstPV1` and `RstDO` are always reset to zero. After resetting, the earliest point at which a renewed reset will be possible is in the next cycle but one (rising edge).

The count value or frequency value and their states are stored in the process image as shown below and are indicated at the following block outputs:

Bytes	Bit	Input signal	Block output	Meaning
0, 1	0-15	Proc. value counter 1	`PV1_Li`	16-bit counter 1 or 32-bit counter (bytes 0 to 3) or frequency value 1
2, 3	0-15	Proc. value counter 2		16-bit counter 2 (only with 16-bit counter 1) or frequency value 2
4	0	A 1	`ZeroSt`	Zero crossing counter 1
	1	A2		Zero crossing counter 2

Bytes	Bit	Input signal	Block output	Meaning
	2	TOR 1	PV1	Status gate 1
	3	TOR 2		Status gate 2
	4	RSZ1	SimAct	Status reset counter 1
	5	RSZ2		Status reset counter 2
	6	RSA1	RstDO_Out	Status reset outputs counter 1
	7	RSA2		Status reset outputs counter 2

The LoadPV1 parameter is always written to the process image. Depending on the mode set with HW Config, it is either the 16-bit or 32-bit setpoint (down counter) or the count limit (up counter).

Depending on the mode setting, only the following integer values of LoadPV1 or LoadPV1_Li are transferred to the module based on the status at LoadPV1_Li:

- 16-bit counter: 0 to 65 535
- 32-bit counter: 0 to 2 080 374 784

If the value for LoadPV1 or LoadPV1_Li is outside of these limits, the last valid value of LoadPV1 or LoadPV1_Li is retained in the module and OpErr = 1 is set.

The block also supplies the counted pulses of PV1(PV1_Li) at the PV1CycLi output with each block call.

The PV1_UnitLi and ScalePV1_Li outputs are written with the inputs of the same name PV1_Unit and ScalePV1.

The LoadPV1 input parameter is the relevant value which is passed to the module. If, however, LoadPV1_Li is interconnected or the status of LoadPV1_Li is 16#80 , this value is written to LoadPV1 and applied.

Note

Make sure that no values with zero are loaded with the "Down counter" counter function.

Configuration

When the CFC function "**Generate module drivers**" is used, the following occurs automatically:

- The Laddr, Laddr1, Channel inputs are configured.
- The Mode input is interconnected with the OMODEx output of the MOD_D1 block.
- The DataXchg input is interconnected to the relevant DXCHG_00 output of the MOD_D1 block.
- The MS input is interconnected to the relevant O_MS output of the MOD_D1 block.

Note

In HW Config, it is possible to assign only digital signals DI2 to DI7 of the module (HW Config channel 2 to 7 = DI) instead of the control signals `TOR 1` to `RSA2`. In this configuration of the DI NAMUR module, the states of outputs `PV1`, `Zero`, `SimAct` and `RstDo_Out` are based on the inputs of the block.

When using the digital control signals `TOR 1` to `RSA2` of the module (HW Config channel 2 to 7 = `CONTRO`L), conflicts with the block digital signals may arise depending on the signal state. In this case, the digital signals do not take effect. If you want control over the block, you must not assign the control signals in HW Config.

Example:

Module	Block	Has the effect	
`TOR 1 = 1`	`StopGate = 0`		TOR on
`TOR 1 = 0`	`StopGate = 1`		TOR on
`TOR 1 = 0`	`StopGate = 0`		TOR off

Startup characteristics

The block does not have any startup characteristics.

Status word allocation

This block does not have the `Status` parameter.

See also

Pcs7Cnt2 I/Os (Page 2154)

16.16.2 Pcs7Cnt2 modes

Operating modes of Pcs7Cnt2

This block does not have any modes.

16.16.3 Pcs7Cnt2 functions

Functions of Pcs7Cnt2

The functions for this block are listed below.

Addressing

You must connect the icon (from the icon table) for the count or frequency value with the `Connect` input parameter.

Simulation

The block provides the standard function Simulating signals (Page 58).

With `SimOn = 1` , the simulation values `SimPV1` and `SimPV2` are written to the outputs `PV1`, `PV1_Li` as well as `PV2` and `PV2_Li` . The status of the outputs is set to 16#60 and `Bad = 1` in the process.

Simulation takes highest priority.

If the block is in simulation state, `SimAct = 1` is set.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

You need to assign parameters for three Feature bits for the response to an invalid raw value for the channel blocks.

If more than one of these Feature bits are set (=1), the following priority applies:

- Output invalid raw value (Feature bit 28 = highest priority)
- Output substitute value if raw value is invalid (Feature bit 29)
- Output the last valid value if raw value is invalid (Feature bit 30, lowest priority)

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
28	Output invalid raw value (Page 173)
29	Output substitute value if raw value is invalid (Page 148)
30	Outputting last valid value if raw value is invalid (Page 153)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for channel blocks for field devices (Page 118).

Value	Meaning
16#80	Valid value
16#60	Manipulated value (for example, substitute value, simulation, last valid value)
16#00	Invalid value

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `LoadPV1_Li.ST`
- `PV1_Li.ST`
- `RstDO.ST`
- `RstPV1.ST`

Redundancy

The higher-level MOD_D1 block evaluates the redundancy of DP master systems operating in an H system.

16.16.4 Pcs7Cnt2 error handling

Error handling of Pcs7Cnt2

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers
- Higher-level error

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
21	Pulse counter overflow

Higher-level error

A higher-level error is displayed at the output parameters `ModErr` and `Bad` with 1 if the signal status in the high word of the input parameter `Mode` assumes the value 16#40.

In addition, at the output parameter `PV_Li` of the signal status, either 16#00 (in the event of an error) or 16#60 (for manipulated value (for example, substitute value, simulation, last valid value) is output.

16.16.5 Pcs7Cnt2 messaging

Messaging

This block does not offer messaging.

16.16.6 Pcs7Cnt2 I/Os

I/Os of CH_CNT2

Input parameters

Parameter	Meaning	Type	Default
Channel	Channel 8 DI NAMUR (outputs)	INT	0
Connect	Connects values	ANY	
`Feature`	I/O for additional functions (Page 2151)	STRUCT	-
		• Bit 0: BOOL	• 0
		• ...	• 0
		• Bit 28: BOOL	• 1
		• ...	• 0
		• Bit 31: BOOL	• 0
FlutTmIn*	Flutter suppression time	INT	0
Laddr	Logical input address	INT	0
Laddr1	Logical output address	INT	0
LoadPV1*	Counter load value	DINT	0
LoadPV1_Li	Counter load value structured	STRUCT	-
		• Value: REAL	• 0.0
		• ST: BYTE	• 16#00
MS	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance	STRUCT	-
		• Value: BOOL	• 0
		• ST: BYTE	• 16#80
MS_Ext	External maintenance status	DWORD	16#00000000

Parameter	Meaning	Type	Default
PV1_Unit	Unit of the current value	INT	0
RstDO*	1 = Reset digital outputs	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RstPV1*	1 = Reset counter	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ScalePV1	Range of the current count value	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
SimOn	1 = simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV1	Simulation value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
StopGate	1 = Stop gate	BOOL	1
SubsPV1	Substitute value	DINT	0
TextRef	Text reference	WORD	16#0000

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Meaning	Type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
Mode	Operating mode	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000

Output parameters

Parameter	Meaning	Type	Default
Bad	1 = invalid values	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ErrorNum	Parameter error	INT	-1
Gate	1 = TOR on	BOOL	0
ModErr	1 = higher-level error	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Meaning	Type	Default
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OosAct	Field device out of service, maintenance is taking place	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OpErr	1 = operator error	BOOL	0
PV1	Current count value/frequency value	DINT	0
PV1CycLi	Pulses per cycle	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV1_Li	Current count value structured	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV1_UnitLi	Unit of the current count value	INT	
RstDO_Out	1 = Reset digital outputs	BOOL	0
RstPV1_Out	1 = Reset counter	BOOL	0
ScalePV1_Li	Range of the current count value	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
SimAct	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal value	BYTE	16#80
ZeroSt	1 = status zero crossing	BOOL	0

See also

Pcs7Cnt2 functions (Page 2151)

16.16.7 Pcs7Cnt2 block diagram

Block diagram of Pcs7Cnt2

A block diagram is not provided for this block.

16.17 Pcs7Cnt3 - Control and read the 1 COUNT 24V/100kHz module for count mode

16.17.1 Description of Pcs7Cnt3

Object name (type + number)

Type + number: FB 1835

Family: Channel

Area of application

The block is used to control and read count, measured and latch values of the "1 COUNT 24V/100kHz" module (as of 6ES7 138-4DA04-0AB0) for the count and measurement modes.

How it works

The block communicates via the process image. The data are written and read continuously.

In `Mode` 1- 3, the `LoadVal` parameter is sent to the module when a positive edge is detected either at the `LoadPre` in/out parameter (= load counter in preparation) or `LoadDir` (= load counter immediately). The parameters `CmpV1` (comparison value 1) and `CmpV2` (comparison value 2) are transferred to the module when they are changed and during startup.

In `Mode` 4- 6, the parameters `UFlowLi`(low limit) and `OFlowLi` (high limit) are transferred to the module when they are changed and during startup.

Depending on the mode setting of the module in HW Config, the user data of the module is stored in the process image. The block distinguishes between the following modes:

MODE (LowWord)	Operating mode	Description
1	Continuous counting	The 1Count24V/100kHz counts continuously from the load value in this mode
2	One-time counting	In this mode, 1Count24V/100kHz counts once, depending on the configured main counting direction
3	Periodic counting	In this mode, 1Count24V/100kHz counts periodically, depending on the configured main counting direction
4	Frequency measurement	1Count24V/100kHz determines the frequency of the pulse sequence at the input
5	Speed measurement	1Count24V/100kHz determines the speed of the device connected to the input
6	Period duration measurement	1Count24V/100kHz determines the pulse length of the pulse sequence at the input

The Mode input indicates the format in which the count and latch value or the count and measured value is available in the process image. If the high byte of input/output parameter is Mode = 16#40 (value status = higher-level error, ModErr = 1), the count, measured and latch values are treated as invalid.

The states PV1_sync (synchronization), CmpVal0 (comparator 0), CmpVal1 (comparator 1), OFlow (overflow), Uflow (underflow) and ZeroSt (zero crossing) are acknowledged automatically by the block. They are active for at least one cycle.

The block also supplies the counted pulses of PV1(PV1_Li) at the PV1CycLi output with each block call.

The units for PV1_UnitLi, ScalePV1_Li, PV2_UnitLi and ScalePV2_Li are set with the PV1_Unit, ScalePV1, PV2_Unit and ScalePV2 inputs of the same name.

The LoadPV1 input parameter is the relevant value which is passed to the module. If, however, LoadPV1_Li is interconnected or the status of LoadPV1_Li is 16#80 , the value is written to LoadPV1 and applied.

Calling OBs

OB100 and cyclic OB (recommendation 100 ms) in which the data will be received and sent.

Configuration

Connect the symbol (from the symbol table) for the count value with the Connect input parameter. You need to enter the symbol (Symbol column) in the symbol table and the supplement the row in the Address column with the ED base address of the module (e.g. ED512). When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The Laddr input is configured.
- The Mode input is interconnected to the OMODE_00output of the MOD_D1 block.
- The DataXchg input is interconnected to the relevant DXCHG_00 output of the MOD_D1 block.
- The MS input is interconnected to the relevant O_MS output of the MOD_D1 block.

The block is also installed automatically in the startup OB (OB100).

Startup characteristics

In Mode 1-3, the parameters CmpV1 (comparison value 1) and CmpV2 (comparison value 2) are transferred to the module during startup.

In Mode 4-6, the parameters UflowLi (low limit) and OFlowLi (high limit) are transferred to the module during startup .

Status word allocation

This block does not have the Status parameter.

See also

Pcs7Cnt3 I/Os (Page 2161)

16.17.2 Pcs7Cnt3 modes

Operating modes of Pcs7Cnt3

This block does not have any modes.

16.17.3 Pcs7Cnt3 functions

Functions of Pcs7Cnt3

The functions for this block are listed below.

Addressing

Connect the symbol (from the symbol table) for the count value with the `Connect` input parameter.

You need to enter the symbol (Symbol column) in the symbol table and the supplement the row in the Address column with the ED base address of the module (e.g. ED512).

Simulation

The block provides the standard function Simulating signals (Page 58)

With `SimOn = 1` , the simulation values `SimPV1` and `SimPV2` are written to the outputs `PV1`, `PV1_Li` as well as `PV2` and `PV2_Li` . The status of the outputs is set to 16#60 and `Bad = 1` in the process.

Simulation takes highest priority.

If the block is in simulation state, `SimAct = 1` is set.

Flutter suppression

This block provides the standard function Flutter suppression for channel blocks (Page 67).

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) . The following functionality is available for this block at the relevant bits:

Bit	Function
10	Condition monitoring information at the channel blocks (Page 144)
28	Output invalid raw value (Page 173)
29	Output substitute value if raw value is invalid (Page 148)
30	Outputting last valid value if raw value is invalid (Page 153)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for channel blocks for field devices (Page 118).

Value	Meaning
16#80	Valid value
16#60	Manipulated value (for example, substitute value, simulation, last valid value)
16#00	Invalid value

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `LoadPV1_Li.ST`
- `PV1_Li.ST`
- `PV2_Li.ST`

16.17.4 Pcs7Cnt3 error handling

Error handling of Pcs7Cnt3

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Higher-level error
- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; the block is not executed
0	There is no error.
21	Pulse counter overflow

Higher-level error

A higher-level error is displayed at the output parameters `ModErr` and `Bad` with 1 if the signal status in the high word of the input parameter `Mode` assumes the value 16#40.

In addition, at the output parameter `PV_Li` of the signal status, either 16#00 (in the event of an error) or 16#60 (for manipulated value (for example, substitute value, simulation, last valid value) is output.

16.17.5 Pcs7Cnt3 messaging

Messaging

This block does not offer messaging.

16.17.6 Pcs7Cnt3 I/Os

I/Os of Pcs7Cnt3

Input parameters

Parameter	Meaning	Type	Default
AckTrip	1 = Error detected	BOOL	0
CmpV1	Comparison value 1	DINT	0
CmpV2	Comparison value 2	DINT	0
Connect	Connection value	DWORD	16#00000000
CtrlDO1	1=Enable DO1	BOOL	1
CtrlDO2	1=Enable DO2	BOOL	1
CtrlSyncEn	1=Enable synchronization	BOOL	0

Parameter	Meaning	Type	Default
Feature	I/O for additional functions (Page 2159)	STRUCT • Bit 0: BOOL • ... • Bit 28: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 1 • 0 • 0
FlutEn	1 = Flutter suppression active	BOOL	0
FlutTmIn*	Flutter suppression time	INT	0
Laddr	Logic address of the module	INT	0
LoadPV1*	Load counter	DINT	0
LoadPV1_Li	Load counter structured	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#00
LoadDir	1 = Load counter directly	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
LoadPre	1 = Load prepared counter	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS	Maintenance status	DWORD	16#00000000
MS_Release	Release for maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Ext	External maintenance status	DWORD	16#00000000
PV1_Unit	Unit of the current value	INT	0
PV2_Unit	Unit of the measured value	INT	0
OFlow*	High limit	DINT	0
OFlowLi	High limit structured	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
ScalePV1	Range of the current count value	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
ScalePV2	Range of current measured value	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
SetDO1	1= Open DO1	BOOL	0
SetDO2	1= Open DO2	BOOL	0
SimPV1	Simulation count value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Meaning	Type	Default
SimPV2	Simulation measured value/simulation latch value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
SimOn	1= simulation on	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SubsPV1	Count substitute value	DINT	0
SubsPV2	Measured/latch substitute value	DINT	0
SwGateEn	1= enable software gate	BOOL	0
TextRef	Text reference	WORD	16#0000
UFlow*	Low limit	DINT	0
UFlowLi	Low limit structured	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

* Values can be written back to these inputs during processing of the block by the block algorithm.

In/out parameters

Parameter	Meaning	Data type	Default
DataXchg	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
DataXchg1	Internal data exchange channel to the diagnostics block	DWORD	16#00000000
Mode	Operating mode	DWORD	16#00000000
MS_Xchg	Exchange of the maintenance status	DWORD	16#00000000

Output parameters

Parameter	Meaning	Data type	Default
Bad	1 = invalid process value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
CmpVal1	1 = status of comparator 1	BOOL	0
CmpVal2	1 = status of comparator 2	BOOL	0
DO1St	1 = status DO1	BOOL	0
DO1Err	1= short circuit / wire break / overtemperature	BOOL	0
DO2St	1 = status DO2	BOOL	0
ErrorNum	Error message	INT	-1
IntGate	1 = status of internal gate	BOOL	0
LoadErr	1= error in load function	BOOL	0

Parameter	Meaning	Data type	Default
ModErr	1= higher-level error	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Dev	Maintenance state	DWORD	16#00000000
MS_Req	1 = Maintenance request from the diagnostic area	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
OFlowSt	1 = high count value	BOOL	0
OosAct	Field device out of service, maintenance is taking place	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ParaErr	1= parameter assignment error	BOOL	0
PV1	Current count value	DINT	0
PV1CycLi	Pulses per cycle	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV1_Down	1 = status direction down	BOOL	0
PV1_Li	Current count value structured	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV1_UnitLi	Unit of the current count value	INT	0
PV1_Up	1 = status direction up	BOOL	0
PV1_Sync	1 = Count synchronous	BOOL	0
PV2	Measured value/latch value	DINT	0
PV2_Li	Structured measured value/latch value	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PV2_UnitLi	Unit of the measured value	INT	0
ScalePV1_Li	Range of the current count value	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
ScalePV2_Li	Range of current measured values/latch values	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
SetDI	1 = Set digital input	BOOL	0
SimAct	1= simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status	BYTE	16#80
UFlowSt	1 = low count value	BOOL	0

Parameter	Meaning	Data type	Default
V24Err	1= short circuit sensor power supply	BOOL	0
ZeroSt	1= status zero crossing	BOOL	0

See also

Pcs7Cnt3 functions (Page 2159)

16.17.7 Pcs7Cnt3 block diagram

Block diagram of Pcs7Cnt3

A block diagram is not provided for this block.

16.18 Pcs7HaAI - HART variable channel block for AI-HART modules

16.18.1 Description of Pcs7HaAI

Object name (type + number) and family

Type + number: FC 1931

Family: Channel

Area of application for Pcs7HaAI

The block is used for the following application:

Signal processing (cyclic service) of an analog input value:

- Of an auxiliary variable of a HART field device

How it works

Pcs7HaAI block cyclically reads the process value and the signal status of the field device from the process image (partition). The process value is available as a physical variable. The signal status contains information about the status of the field device.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Connect the symbol generated in HW Config (symbol table) for the input channel with the `PV` input parameter.

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter `MultiHART` is interconnected with the corresponding output parameter `MULTI_HART_xx` of the MHA_CO or OR_MHA_CO block.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Pcs7HaAI modes (Page 2167)

Pcs7HaAI functions (Page 2167)

Pcs7HaAI error handling (Page 2168)
Pcs7HaAI messaging (Page 2168)
Pcs7HaAI I/Os (Page 2169)
Pcs7HaAI block diagram (Page 2171)

16.18.2 Pcs7HaAI modes

Pcs7HaAI operating modes

This block does not have any operating modes.

See also

Description of Pcs7HaAI (Page 2166)
Pcs7HaAI functions (Page 2167)
Pcs7HaAI error handling (Page 2168)
Pcs7HaAI messaging (Page 2168)
Pcs7HaAI I/Os (Page 2169)
Pcs7HaAI block diagram (Page 2171)

16.18.3 Pcs7HaAI functions

Functions of Pcs7HaAI

The functions for this block are listed below.

Obtaining the standard value

The HART variable value of the process image (partition) is outputted as a standard value at the `HV_OutP`/`HV_OutS`/`HV_OutT`/`HV_OutQ` output parameters.

See also

Description of Pcs7HaAI (Page 2166)
Pcs7HaAI modes (Page 2167)
Pcs7HaAI error handling (Page 2168)
Pcs7HaAI messaging (Page 2168)
Pcs7HaAI I/Os (Page 2169)
Pcs7HaAI block diagram (Page 2171)

16.18.4 Pcs7HaAI error handling

Error handling of Pcs7HaAI

The following errors can be displayed for this block:

- Invalid value

Invalid value

The invalid values are displayed with the value 1 at the output parameters `HV_OutPBad`/`HV_OutSBad`/`HV_OutTBad`/`HV_OutQBad`. The invalid value error is generated from the signal status HV_PriQC/HV_SecQC/HV_TerQC/HV_QuaQC of the `MultiHART` in/out parameter when the signal status is 16#00 (initial value from S7- System) or 16#37 (initial value from analog module after module start) respectively.

See also

Description of Pcs7HaAI (Page 2166)
Pcs7HaAI modes (Page 2167)
Pcs7HaAI functions (Page 2167)
Pcs7HaAI messaging (Page 2168)
Pcs7HaAI I/Os (Page 2169)
Pcs7HaAI block diagram (Page 2171)

16.18.5 Pcs7HaAI messaging

Messaging

This block does not offer messaging.

See also

Description of Pcs7HaAI (Page 2166)
Pcs7HaAI modes (Page 2167)
Pcs7HaAI functions (Page 2167)
Pcs7HaAI error handling (Page 2168)
Pcs7HaAI I/Os (Page 2169)
Pcs7HaAI block diagram (Page 2171)

16.18.6 Pcs7HaAI I/Os

I/Os of Pcs7HaAI

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional features	STRUCT • Bit0: BOOL • ... • Bit28: BOOL • ... • Bit31: BOOL	- • 0 • 0 • 1 • 0 • 0
HV_PriEn	1= Read primary HART variable	BOOL	0
HV_SecEn	1= Read secondary HART variable	BOOL	0
HV_TerEn	1= Read tertiary HART variable	BOOL	0
HV_QuaEn	1= Read quaternary HART variable	BOOL	0
PV	Process value (analog value)	WORD	16#0000
HV_UnitP	Unit of primary HART variable	INT	1001
ScaleP	Range of primary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_UnitS	Unit of secondary HART variable	INT	1001
ScaleS	Range of secondary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_UnitT	Unit of tertiary HART variable	INT	1001
ScaleT	Range of tertiary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_UnitQ	Unit of quaternary HART variable	INT	1001
ScaleQ	Range of quaternary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

In/out parameters

Parameter	Description	Type	Default
MultiHART	Internal data exchange channel to the diagnostics block	STRUCT • HV_PriEn: BOOL • HV_SecEn: BOOL • HV_TerEn: BOOL • HV_QuaEn: BOOL • HV_PriQC: BYTE • HV_SecQC: BYTE • HV_TerQC: BYTE • HV_QuaQC: BYTE • Command: BYTE • Ack: BYTE • HV_PriVal: REAL • HV_SecVal: REAL • HV_TerVal: REAL • HV_QuaVal: REAL	- • 0 • 0 • 0 • 0 • 16#00 • 16#00 • 16#00 • 16#00 • 16#00 • 16#00 • 0.0 • 0.0 • 0.0 • 0.0

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number	INT	-1
HV_OutP	Primary HART variable with status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
HV_OutS	Secondary HART variable with status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
HV_OutT	Tertiary HART variable with status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
HV_OutQ	Quaternary HART variable with status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
HV_OutPBad	1 = Bad primary HART variable	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
HV_OutSBad	1 = Bad secondary HART variable	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
HV_OutTBad	1 = Bad tertiary HART variable	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
HV_OutQBad	1 = Bad quaternary HART variable	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
HV_OUniP	Unit of primary HART variable	INT	0
ScaleOuP	Range of primary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_OUniS	Unit of secondary HART variable	INT	0
ScaleOuS	Range of secondary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_OUniT	Unit of tertiary HART variable	INT	0
ScaleOuT	Range of tertiary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_OUniQ	Unit of quaternary HART variable	INT	0
ScaleOuQ	Range of quaternary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

See also

Description of Pcs7HaAI (Page 2166)
Pcs7HaAI modes (Page 2167)
Pcs7HaAI functions (Page 2167)
Pcs7HaAI error handling (Page 2168)
Pcs7HaAI messaging (Page 2168)
Pcs7HaAI block diagram (Page 2171)

16.18.7 Pcs7HaAI block diagram

Block diagram of Pcs7HaAI

A block diagram is not provided for this block.

See also

Description of Pcs7HaAI (Page 2166)
Pcs7HaAI modes (Page 2167)
Pcs7HaAI functions (Page 2167)
Pcs7HaAI error handling (Page 2168)
Pcs7HaAI messaging (Page 2168)
Pcs7HaAI I/Os (Page 2169)

16.19 Pcs7HaAO - HART variable channel block for AO-HART modules

16.19.1 Description of Pcs7HaAO

Object name (type + number) and family

Type + number: FC 1932

Family: Channel

Area of application for Pcs7HaAO

The block is used for the following application:

Signal processing (cyclic service) of an analog input value:

- Of an auxiliary variable of a HART output field device

How it works

Pcs7HaAO block cyclically reads the process value and the signal status of the field device from the process image (partition). The process value is available as a physical variable. The signal status contains information about the status of the field device.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Connect the symbol generated in HW Config (symbol table) for the output channel with the `PV_Out` output parameter.

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The in/out parameter `MultiHART` is interconnected with the corresponding output parameter `MULTI_HART_xx` of the MHA_CO or OR_MHA_CO block.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

Pcs7HaAO modes (Page 2174)

Pcs7HaAO functions (Page 2174)

16.19.2 Pcs7HaAO modes

Pcs7HaAO operating modes

This block does not have any operating modes.

See also

16.19.3 Pcs7HaAO functions

Functions of Pcs7HaAO

The functions for this block are listed below.

Obtaining the standard value

The HART variable value of the process image (partition) is outputted as a standard value at the `HV_OutP`/`HV_OutS`/`HV_OutT`/`HV_OutQ` output parameters.

See also

16.19.4 Pcs7HaAO error handling

Error handling of Pcs7HaAO

The following errors can be displayed for this block:

- Invalid value

Invalid value

The invalid values are displayed with the value 1 at the output parameters `HV_OutPBad`/ `HV_OutSBad`/`HV_OutTBad`/`HV_OutQBad`. The invalid value error is generated from the signal status HV_PriQC/HV_SecQC/HV_TerQC/HV_QuaQC of the `MultiHART` in/out parameter when the signal status is 16#00 (initial value from S7- System) or 16#37 (initial value from analog module after module start) respectively.

See also

Description of Pcs7HaAO (Page 2173)
Pcs7HaAO modes (Page 2174)
Pcs7HaAO functions (Page 2174)
Pcs7HaAO messaging (Page 2175)
Pcs7HaAO I/Os (Page 2176)
Pcs7HaAO block diagram (Page 2178)

16.19.5 Pcs7HaAO messaging

Messaging

This block does not offer messaging.

See also

Description of Pcs7HaAO (Page 2173)
Pcs7HaAO modes (Page 2174)
Pcs7HaAO functions (Page 2174)
Pcs7HaAO error handling (Page 2175)
Pcs7HaAO I/Os (Page 2176)
Pcs7HaAO block diagram (Page 2178)

16.19.6 Pcs7HaAO I/Os

I/Os of Pcs7HaAO

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Feature	I/O for additional features	STRUCT • Bit0: BOOL • ... • Bit28: BOOL • ... • Bit31: BOOL	- • 0 • 0 • 1 • 0 • 0
HV_PriEn	1= Read primary HART variable	BOOL	0
HV_SecEn	1= Read secondary HART variable	BOOL	0
HV_TerEn	1= Read tertiary HART variable	BOOL	0
HV_QuaEn	1= Read quaternary HART variable	BOOL	0
HV_UnitP	Unit of primary HART variable	INT	1001
ScaleP	Range of primary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_UnitS	Unit of secondary HART variable	INT	1001
ScaleS	Range of secondary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_UnitT	Unit of tertiary HART variable	INT	1001
ScaleT	Range of tertiary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_UnitQ	Unit of quaternary HART variable	INT	1001
ScaleQ	Range of quaternary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

In/out parameters

Parameter	Description	Type	Default
MultiHART	Internal data exchange channel to the diagnostics block	STRUCT • HV_PriEn: BOOL • HV_SecEn: BOOL • HV_TerEn: BOOL • HV_QuaEn: BOOL • HV_PriQC: BYTE • HV_SecQC: BYTE • HV_TerQC: BYTE • HV_QuaQC: BYTE • Command: BYTE • Ack: BYTE • HV_PriVal: REAL • HV_SecVal: REAL • HV_TerVal: REAL • HV_QuaVal: REAL	- • 0 • 0 • 0 • 0 • 16#00 • 16#00 • 16#00 • 16#00 • 16#00 • 16#00 • 0.0 • 0.0 • 0.0 • 0.0

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ErrorNum	Output of pending error number	INT	-1
HV_OutP	Primary HART variable with status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
HV_OutS	Secondary HART variable with status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
HV_OutT	Tertiary HART variable with status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
HV_OutQ	Quaternary HART variable with status	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
HV_OutPBad	1 = Bad primary HART variable	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
HV_OutSBad	1 = Bad secondary HART variable	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Parameter	Description	Type	Default
HV_OutTBad	1 = Bad tertiary HART variable	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
HV_OutQBad	1 = Bad quaternary HART variable	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
PV_Out	Process value	WORD	16#0000
HV_OUniP	Unit of primary HART variable	INT	0
ScaleOuP	Range of primary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_OUniS	Unit of secondary HART variable	INT	0
ScaleOuS	Range of secondary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_OUniT	Unit of tertiary HART variable	INT	0
ScaleOuT	Range of tertiary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0
HV_OUniQ	Unit of quaternary HART variable	INT	0
ScaleOuQ	Range of quaternary HART variable	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

See also

Description of Pcs7HaAO (Page 2173)
Pcs7HaAO modes (Page 2174)
Pcs7HaAO functions (Page 2174)
Pcs7HaAO error handling (Page 2175)
Pcs7HaAO messaging (Page 2175)
Pcs7HaAO block diagram (Page 2178)

16.19.7 Pcs7HaAO block diagram

Block diagram of Pcs7HaAO

A block diagram is not provided for this block.

See also

Description of Pcs7HaAO (Page 2173)
Pcs7HaAO modes (Page 2174)
Pcs7HaAO functions (Page 2174)
Pcs7HaAO error handling (Page 2175)
Pcs7HaAO messaging (Page 2175)
Pcs7HaAO I/Os (Page 2176)

16.20 Annex for channel blocks

16.20.1 Mode Settings for SM Modules

Mode Structure

The structure and meaning of the input Mode of data type DWORD are shown below:

Byte 3:	16#80: Value status "valid value" 16#00: Value status "invalid value" 16#40: Value status "invalid value"	 (Channel error) (Higher-level error)
Byte 2:	16#01: Restart (OB 100) has been carried out 16#02: Measuring range high limit violated 16#04: Measuring range low limit violated 16#xy: Variant identification with multiple mode assignment (see below)	 (Channel-error diagnostics) (Channel-error diagnostics)
Byte 1, 0 (low word):	Measuring range coding (see below)	

Example:

16#80010203 = value status "valid value", restart has been carried out, current 4 mA to 20mA

Mode: Celsius and Fahrenheit

Variant	x	y	Unit of measurement
0	0, 1 (depending on the other variant IDs)	0,1,2,4	Celsius
1	8, 9 (depending on the other variant IDs)	0,1,2,4	Fahrenheit

Mode: 16#090C and variant

Variant	x	y	Measuring range
0	0	0,1,2,4	Ni 120 standard range
1	1	0,1,2,4	KTY84/110

Mode: 16#090D and variant

Variant	x	y	Measuring range
0	0	0,1,2,4	Ni 120 climate range
1	1	0,1,2,4	KTY84/130

Mode: 16#07 (Coding A) and variant

Variant	x	y	Measuring range
0	0	0,1,2,4	HART interface
1	1	0,1,2,4	Thermocouple, dynamic reference temperature

MODE: 16#08, 16#0A, 16#0B, 16#0D, 16#0E (coding A) and variant

Table 16-1 PCS 7 as of V8.1: MODE parameter variant

Variant	x	y	Measuring range
0	0	0,1,2,4	S7-300 module
1	3	0,1,2,4	S7-300, ET 200M Ex I/O module

Table 16-2 PCS 7 V7.x: MODE parameter variant

Variant	mode	Measuring range
0	16#08, 16#0A, 16#0B, 16#0D, 16#0E	S7-300 module
1	16#18, 16#1A, 16#1B, 16#1D, 16#1E	S7-300, ET 200M Ex I/O module

MODE: 1oo1-/1oo2 evaluation for F modules

Variant	x	y	Measuring range
0	0	0,1,2,4	1oo1 (1v1) evaluation
1	4	0,1,2,4	1oo2 (2v2) evaluation

Measuring range coding of the analog input modules

Depending on the measuring range coding of the analog input modules, the parameter `Mode` (low word = measuring range coding) corresponding to the channel must be specified in accordance with the table. When thermocouples are used there are various options for combining the measurement type (coding A) with the measuring range (coding B). In this case, the low word of parameter `Mode` must be calculated according to the following formula and the result entered at the Mode input parameter as an INTEGER value:

Measuring range coding = 256 · Code A + Code B

Please note: The table shows codes **A** and **B** in binary format, and the measuring range coding as a hexadecimal number as the result.

Measuring type	Code (A)	Measuring range	Code(B)	Measuring range coding (256*A+B)
Disabled				16#0000
Voltage	2#0001	± 25 mV	2#1010	16#010A
		± 50mV	2#1011	16#010B
		± 80 mV	2#0001	16#0101
		± 250 mV	2#0010	16#0102
		± 500 mV	2#0011	16#0103
		± 1 V	2#0100	16#0104
		± 2,5 V	2#0101	16#0105
		± 5 V	2#0110	16#0106
		1 to 5 V	2#0111	16#0107
		0 to 10V	2#1000	16#0108
		± 10 V	2#1001	16#0109
		± 100 mV	2#1100	16#010C
4-wire measuring transducer	2#0010	± 3.2 mA	2#0000	16#0200
		± 5 mA	2#0101	16#0205
		± 10 mA	2#0001	16#0201
		0 to 20 mA	2#0010	16#0202
		4 to 20 mA	2#0011	16#0203
		± 20 mA	2#0100	16#0204
HART interface	2#0111	4 to 20 mA (variant 0)	2#1100	16#070C
2-wire measuring transducer	2#0011	0 to 20 mA	2#0010	16#0302
		4 to 20 mA	2#0011	16#0303
		± 20 mA	2#0100	16#0304
		0 to 10 mA	2#0110	16#0306
Resistor 4-wire connection	2#0100	48 Ω	2#0000	16#0400
		150 Ω	2#0010	16#0402
		300 Ω	2#0100	16#0404
		600 Ω	2#0110	16#0406
		1000 Ω	2#0111	16#040E
		3000 Ω	2#0111	16#0407
		6000 Ω	2#1000	16#0408
		PTC	2#1111	16#040F

Measuring type	Code (A)	Measuring range	Code(B)	Measuring range coding (256*A+B)
Resistor 3-wire connection	2#0101	48 Ω	2#0000	16#0500
		150 Ω	2#0010	16#0502
		300 Ω	2#0100	16#0504
		600 Ω	2#0110	16#0506
		1000 Ω	2#0111	16#050E
		3000 Ω	2#0111	16#0507
		6000 Ω	2#1000	16#0508
		PTC	2#1111	16#050F
Resistor 2-wire connection	2#0110	48 Ω	2#0000	16#0600
		150 Ω	2#0010	16#0602
		300 Ω	2#0100	16#0604
		600 Ω	2#0110	16#0606
		1000 Ω	2#0111	16#060E
		3000 Ω	2#0111	16#0607
		6000 Ω	2#1000	16#0608
		PTC	2#1111	16#060F

Measuring type	Code (A)	Measuring range	Code(B)	Measuring range coding (256*A+B)
Thermal resistance, linear, 4-wire connection	2#1000	Pt 100 climate range	2#0000	16#0800
		Pt 200 climate range	2#0111	16#0807
		Pt 500 climate range	2#1000	16#0808
		Pt 1000 climate range	2#1001	16#0809
		Ni 100 climate range	2#0001	16#0801
		Ni 1000 / LG-Ni 1000 climatic range	2#1010	16#080A
		Pt 100 standard range	2#0010	16#0802
		Pt 200 standard range	2#0011	16#0803
		Pt 500 standard range	2#0100	16#0804
		Pt 1000 standard range	2#0101	16#0805
		Ni 100 standard range	2#1011	16#080B
		Ni 1000 / LG-Ni 1000 standard range	2#0110	16#0806
		Ni 120 standard range	2#1100	16#080C
		Ni 120 climate range	2#1101	16#080D
		Cu 10 climate range	2#1110	16#080E
		Cu 10 standard range	2#1111	16#080F
		Ni 200 standard range	2#10000	16#0810
		Ni 200 climate range	2#10001	16#0811
		Ni 500 standard range	2#10010	16#0812
		Ni 500 climate range	2#10011	16#0813
		Pt 10 GOST climatic	2#10100	16#0814
		Pt 10 GOST standard (TC = 3910)	2#10101	16#0815
		Pt 50 GOST climatic	2#10110	16#0816
		Pt 50 GOST standard (TC = 3910)	2#10111	16#0817
		Pt 100 GOST climatic	2#11000	16#0818
		Pt 100 GOST standard (TC = 3910)	2#11001	16#0819
		Pt 500 GOST climatic	2#11010	16#081A
		Pt 500 GOST standard (TC = 3910)	2#11011	16#081B
		Cu 10 GOST climatic	2#11100	16#081C
		Cu 10 GOST standard (TC = 426)	2#11101	16#081D
		Cu 50 GOST climatic	2#11110	16#081E
		Cu 50 GOST standard (TC = 426)	2#11111	16#081F
		Cu 100 GOST climatic	2#100000	16#0820
		Cu 100 GOST standard (TC = 426)	2#100001	16#0821
		Ni 100 GOST climatic	2#100010	16#0822

Measuring type	Code (A)	Measuring range	Code(B)	Measuring range coding (256*A+B)
		Ni 100 GOST standard	2#100011	16#0823
		Pt 10 GOST standard (TC = 3850)	2#1010101	16#0855
		Pt 50 GOST standard (TC = 3850)	2#1010111	16#0857
		Pt 100 GOST standard (TC = 3850)	2#1011001	16#0859
		Pt 500 GOST standard (TC = 3850)	2#1011011	16#085B
		Cu 10 GOST standard (TC= 428)	2#10011101	16#089D
		Cu 50 GOST standard (TC= 428)	2#10011111	16#089F
		Cu 100 GOST standard (TC= 428)	2#10100001	16#08A1

Measuring type	Code (A)	Measuring range	Code(B)	Measuring range coding (256*A+B)
Thermal resistance, linear, 3-wire connection	2#1001	Pt 100 climate range	2#0000	16#0900
		Pt 200 climate range	2#0111	16#0907
		Pt 500 climate range	2#1000	16#0908
		Pt 1000 climate range	2#1001	16#0909
		Ni 100 climate range	2#0001	16#0901
		Ni 1000 / LG-Ni 1000 climatic range	2#1010	16#090A
		Pt 100 standard range	2#0010	16#0902
		Pt 200 standard range	2#0011	16#0903
		Pt 500 standard range	2#0100	16#0904
		Pt 1000 standard range	2#0101	16#0905
		Ni 100 standard range	2#1011	16#090B
		Ni 1000 / LG-Ni 1000 standard range	2#0110	16#0906
		Ni 120 standard range (variant 0) KTY83/110 (variant 1)	2#1100	16#090C
		Ni 120 climate range (variant 0) KTY84/130 (variant 1)	2#1101	16#090D
		Cu 10 climate range	2#1110	16#090E
		Cu 10 standard range	2#1111	16#090F
		Ni 200 standard range	2#10000	16#0910
		Ni 200 climate range	2#10001	16#0911
		Ni 500 standard range	2#10010	16#0912
		Ni 500 climate range	2#10011	16#0913
		Pt 10 GOST climatic	2#10100	16#0914
		Pt 10 GOST standard (TC = 3910)	2#10101	16#0915
		Pt 50 GOST climatic	2#10110	16#0916
		Pt 50 GOST standard (TC = 3910)	2#10111	16#0917
		Pt 100 GOST climatic	2#11000	16#0918
		Pt 100 GOST standard (TC = 3910)	2#11001	16#0919
		Pt 500 GOST climatic	2#11010	16#091A
		Pt 500 GOST standard (TC = 3910)	2#11011	16#091B
		Cu 10 GOST climatic	2#11100	16#091C
		Cu 10 GOST standard (TC = 426)	2#11101	16#091D
		Cu 50 GOST climatic	2#11110	16#091E
		Cu 50 GOST standard (TC = 426)	2#11111	16#091F

Measuring type	Code (A)	Measuring range	Code(B)	Measuring range coding (256*A+B)
		Cu 100 GOST climatic	2#100000	16#0920
		Cu 100 GOST standard (TC = 426)	2#100001	16#0921
		Ni 100 GOST climatic	2#100010	16#0922
		Ni 100 GOST standard	2#100011	16#0923
		Pt 10 GOST standard (TC = 3850)	2#1010101	16#0955
		Pt 50 GOST standard (TC = 3850)	2#1010111	16#0957
		Pt 100 GOST standard (TC = 3850)	2#1011001	16#0959
		Pt 500 GOST standard (TC = 3850)	2#1011011	16#095B
		Cu 10 GOST standard (TC= 428)	2#10011101	16#099D
		Cu 50 GOST standard (TC= 428)	2#10011111	16#099F
		Cu 100 GOST standard (TC= 428)	2#10100001	16#09A1
Thermal resistance, linear, 2-wire connection	2#1111	Pt 100 climate range	2#0000	16#0F00
		Pt 200 climate range	2#0111	16#0F07
		Pt 500 climate range	2#1000	16#0F08
		Pt 1000 climate range	2#1001	16#0F09
		Ni 100 climate range	2#0001	16#0F01
		Ni 1000 / LG-Ni 1000 climatic range	2#1010	16#0F0A
		Pt 100 standard range	2#0010	16#0F02
		Pt 200 standard range	2#0011	16#0F03
		Pt 500 standard range	2#0100	16#0F04
		Pt 1000 standard range	2#0101	16#0F05
		Ni 100 standard range	2#1011	16#0F0B
		Ni 1000 / LG-Ni 1000 standard range	2#0110	16#0F06
		Ni 120 standard range	2#1100	16#0F0C
		Ni 120 climate range	2#1101	16#0F0D
		Cu 10 climate range	2#1110	16#0F0E
		Cu 10 standard range	2#1111	16#0F0F
		Ni 200 standard range	2#10000	16#0F10
		Ni 200 climate range	2#10001	16#0F11
		Ni 500 standard range	2#10010	16#0F12
		Ni 500 climate range	2#10011	16#0F13

Measuring type	Code (A)	Measuring range	Code(B)	Measuring range coding (256*A+B)
Thermocouple, linear, reference temperature 0 ºC / No reference point	2#1010	Type B [PtRh-PtRh]	2#0000	16#0A00
		Type N [NiCrSi-NiSi]	2#0001	16#0A01
		Type E [NiCr-CuNi]	2#0010	16#0A02
		Type R [PtRh-Pt]	2#0011	16#0A03
		Type S [PtRh-Pt]	2#0100	16#0A04
		Type J [Fe-CuNi IEC]	2#0101	16#0A05
		Type L [Fe-CuNi DIN]	2#0110	16#0A06
		Type T [Cu-CuNi IEC]	2#0111	16#0A07
		Type K [NiCr-Ni]	2#1000	16#0A08
		Type U [Cu-CuNi DIN]	2#1001	16#0A09
		Type C	2#1010	16#0A0A
		Type TXK/XK(L)	2#1011	16#0A0B
Thermocouple, linear, reference temperature 50 ºC	2#1011	Type B [PtRh-PtRh]	2#0000	16#0B00
		Type N [NiCrSi-NiSi]	2#0001	16#0B01
		Type E [NiCr-CuNi]	2#0010	16#0B02
		Type R [PtRh-Pt]	2#0011	16#0B03
		Type S [PtRh-Pt]	2#0100	16#0B04
		Type J [Fe-CuNi IEC]	2#0101	16#0B05
		Type L [Fe-CuNi DIN]	2#0110	16#0B06
		Type T [Cu-CuNi IEC]	2#0111	16#0B07
		Type K [NiCr-Ni]	2#1000	16#0B08
		Type U [Cu-CuNi DIN]	2#1001	16#0B09
		Type C	2#1010	16#0B0A
		Type TXK/XK(L)	2#1011	16#0B0B
Thermocouple, fixed ref. temp	2#1100	Type B [PtRh-PtRh]	2#0000	16#0C00
		Type N [NiCrSi-NiSi]	2#0001	16#0C01
		Type E [NiCr-CuNi]	2#0010	16#0C02
		Type R [PtRh-Pt]	2#0011	16#0C03
		Type S [PtRh-Pt]	2#0100	16#0C04
		Type J [Fe-CuNi IEC]	2#0101	16#0C05
		Type L [Fe-CuNi DIN]	2#0110	16#0C06
		Type T [Cu-CuNi IEC]	2#0111	16#0C07
		Type K [NiCr-Ni]	2#1000	16#0C08
		Type U [Cu-CuNi DIN]	2#1001	16#0C09
		Type C	2#1010	16#0C0A
		Type TXK/XK(L)	2#1011	16#0C0B

Measuring type	Code (A)	Measuring range	Code(B)	Measuring range coding (256*A+B)
Thermocouple, linear, internal compensation/internal reference point	2#1101	Type B [PtRh-PtRh]	2#0000	16#0D00
		Type N [NiCrSi-NiSi]	2#0001	16#0D01
		Type E [NiCr-CuNi]	2#0010	16#0D02
		Type R [PtRh-Pt]	2#0011	16#0D03
		Type S [PtRh-Pt]	2#0100	16#0D04
		Type J [Fe-CuNi IEC]	2#0101	16#0D05
		Type L [Fe-CuNi DIN]	2#0110	16#0D06
		Type T [Cu-CuNi IEC]	2#0111	16#0D07
		Type K [NiCr-Ni]	2#1000	16#0D08
		Type U [Cu-CuNi DIN]	2#1001	16#0D09
		Type C	2#1010	16#0D0A
		Type TXK/XK(L)	2#1011	16#0D0B
Thermocouple, linear, internal compensation/ reference point RTD (0)	2#1110	Type B [PtRh-PtRh]	2#0000	16#0E00
		Type N [NiCrSi-NiSi]	2#0001	16#0E01
		Type E [NiCr-CuNi]	2#0010	16#0E02
		Type R [PtRh-Pt]	2#0011	16#0E03
		Type S [PtRh-Pt]	2#0100	16#0E04
		Type J [Fe-CuNi IEC]	2#0101	16#0E05
		Type L [Fe-CuNi DIN]	2#0110	16#0E06
		Type T [Cu-CuNi IEC]	2#0111	16#0E07
		Type K [NiCr-Ni]	2#1000	16#0E08
		Type U [Cu-CuNi DIN]	2#1001	16#0E09
		Type C	2#1010	16#0E0A
		Type TXK/XK(L)	2#1011	16#0E0B
Thermocouple, dynamic ref. temp	2#0111	Type B [PtRh-PtRh] (variant 1)	2#0000	16#0700
		Type N [NiCrSi-NiSi] (variant 1)	2#0001	16#0701
		Type E [NiCr-CuNi] (variant 1)	2#0010	16#0702
		Type R [PtRh-Pt] (variant 1)	2#0011	16#0703
		Type S [PtRh-Pt] (variant 1)	2#0100	16#0704
		Type J [Fe-CuNi IEC] (variant 1)	2#0101	16#0705
		Type L [Fe-CuNi DIN] (variant 1)	2#0110	16#0706
		Type T [Cu-CuNi IEC] (variant 1)	2#0111	16#0707
		Type K [NiCr-Ni] (variant 1)	2#1000	16#0708

Effect of the temperature coefficients on the measuring range

- Setting TC = 3850 at GOST Standard Pt 10, Pt 50, Pt 100, Pt 500 sets Bit 7 within the measuring range byte (0 x 40)
- Setting TC = 428 at GOST Standard Cu 10, Cu 50, Cu 100 sets Bit 8 within the measuring range byte (0 x 80)

Measuring range coding of the analog output modules

Depending on the measuring range coding of the analog output modules, the parameter `Mode` (measuring-range coding) corresponding to the channel must be specified in accordance with the table.

Measuring type	Measuring range	`Mode`
Voltage	± 5 V	16#0106
	1 to 5 V	16#0107
	0 to 10V	16#0108
	± 10 V	16#0109
Current	0 to 10 mA	16#0206
	0 to 20 mA	16#0202
	4 to 20 mA	16#0203
	± 20 mA	16#0204
HART interface	4 to 20 mA	16#070C

Measuring-Range Coding of the Digital Input and Output Modules

There is no measuring type and no measuring range for digital input modules and digital output modules:

- `Mode` **= 16#FFFF** (for DiIn)
- `Mode` **= 16#FFFE** (for DiOu)

Measuring range coding of the controller module

There is no measuring type and no measuring range for controller modules:

- `Mode` **= 16#FFFD**

Measuring range coding of the ET 200SP Analog Input Energy Meter module

There is no measuring type and no measuring range for analoge input energy meter moduls:

- Mode = **16#200C**

16.20.2 Mode settings for field devices

Mode structure

The structure and meaning of the input Mode of data type DWORD are shown below:

Byte 3:	16#80: Value status "valid value" 16#00: Value status "invalid value" 16#40: Value status "invalid value"	 (Channel error) (Higher-level error)
Byte 2:	16#01: Restart (OB 100) has been carried out 16#xy: Variant identification with multiple mode assignment (see below)	
Byte 1, 0 (low word):	Measuring range coding (see below)	

Example:

16#80010001 = Value status "valid value", restart has been carried out, PA device with analog input

PA/FF device variant identifier

Variant	x	y	Field device
0	0	0	PA device
1	1	0	FF device

Mode settings for field devices

MODE_xx input parameters are available for a maximum of 32 slots of a PA/FF field device. Their initial value is zero (no read/write access). You must set the combination selected from the options of the PROFIBUS PA 3.0 profile at the MODE_xx input of each slot channel xx:

Block	Parameter at block ...	...and at field device	Input/output from block view (PLS view)	Mode 16#xxyy O = xx I = yy	Configuration possible with field device
Analog input (PA_AI/ FbAnIn)	PV	OUT	I	16#0001	PA + FF
Totalizer (FbAnTot)	PV		I	16#000F	PA + FF
Totalizer (FbAnTot)	PV PV_Set		I O	16#070F	PA + FF
Totalizer (FbAnTot)	PV PV_Set PV_Mode		I O O	16#080F	PA + FF
Analog output (FbAnOu)	SP	SP	O	16#0100	PA + FF

Block	Parameter at block ...	...and at field device	Input/output from block view (PLS view)	Mode 16#xxyy O = xx I = yy	Configuration possible with field device
Analog output (FbAnOu)	SP Rbk PosD	SP READBACK POS_D	O I I	16#0103	PA
Analog output (FbAnOu)	SP CbkBy0 - CbkBy2	SP CHECK_BACK	O I	16#0104	PA
Analog output (FbAnOu)	SP Rbk PosD CbkBy0 - CbkBy2	SP READBACK POS_D CHECK_BACK	O I I I	16#0105	PA
Analog output (FbAnOu)	RCasIn, RCasOut	RCAS_IN RCAS_OUT	O I	16#0206	PA
Analog output (FbAnOu)	RCasIn RCasOut CbkBy0 - CbkBy2	RCAS_IN RCAS_OUT CHECK_BACK	O I I	16#0207	PA
Analog output (FbAnOu)	SP RCasIn Rbk RCasOut PosD CbkBy0 - CbkBy2	SP RCAS_IN READBACK RCAS_OUT POS_D CHECK_BACK	O O I I I I	16#0308	PA
Binary input (FbDiIn)	PV	OUT_D	I	16#0002	PA + FF
Binary output (FbDiOu)	SP	SP_D	O	16#0400	PA + FF
Binary output (FbDiOu)	SP Rbk	SP_D READBACK_D	O I	16#0409	PA
Binary output (FbDiOu)	SP CbkBy0 - CbkBy2	SP_D CHECKBACK_D	O I	16#040A	PA
Binary output (FbDiOu)	SP Rbk CbkBy0 - CbkBy2	SP_D READBACK_D CHECK_BACK_D	O I I	16#040B	PA
Binary output (FbDiOu)	RCasIn RCasOut	RCAS_IN_D RCAS_OUT_D	O I	16#050C	PA
Binary output (FbDiOu)	RCasIn RCasOut CbkBy0 - CbkBy2	RCAS_IN_D RCAS_OUT_D CHECK_BACK_D	O I I	16#050D	PA
Binary output (FbDiOu)	SP RCasIn Rbk RCasOut CbkBy0 - CbkBy2	SP_D RCAS_IN_D READBACK_D RCAS_OUT_D CHECK_BACK_D	O O I I I	16#060E	PA

Conversion blocks 17

17.1 StrgToBy - String in byte structure (Struct of Byte)

17.1.1 Description of StrgToBy

Object name (type + number) and family

Type + number: FC 384

Family: Convert

Area of application for StrgToBy

The block converts a string with a maximum length of 32 into a byte structure (Struct of Byte). This enables strings to be interconnected via the StrgToBy block to the inputs for the associated values (`ExtValxx`) of the technological blocks.

How it works

The block uses the input string "`StringIn`", converts it to a byte structure and transfers it to the output parameter `StrctOut`".

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

StrgToBy modes (Page 2194)

StrgToBy functions (Page 2194)

StrgToBy error handling (Page 2195)

StrgToBy messaging (Page 2195)

StrgToBy I/Os (Page 2195)
StrgToBy block diagram (Page 2196)

17.1.2 StrgToBy modes

Operating modes of `StrgToBy`

This block does not have any operating modes.

See also

Description of StrgToBy (Page 2193)
StrgToBy functions (Page 2194)
StrgToBy error handling (Page 2195)
StrgToBy messaging (Page 2195)
StrgToBy I/Os (Page 2195)
StrgToBy block diagram (Page 2196)

17.1.3 StrgToBy functions

Functions of StrgToBy

The block uses the input string "`StringIn`", converts it to a byte structure and transfers it to the output parameter `StrctOut`".

See also

Description of StrgToBy (Page 2193)
StrgToBy modes (Page 2194)
StrgToBy error handling (Page 2195)
StrgToBy messaging (Page 2195)
StrgToBy I/Os (Page 2195)
StrgToBy block diagram (Page 2196)

17.1.4 StrgToBy error handling

Error handling of StrgToBy

The block does not report any errors.

See also

Description of StrgToBy (Page 2193)
StrgToBy modes (Page 2194)
StrgToBy functions (Page 2194)
StrgToBy messaging (Page 2195)
StrgToBy I/Os (Page 2195)
StrgToBy block diagram (Page 2196)

17.1.5 StrgToBy messaging

Messaging

This block does not offer messaging.

See also

Description of StrgToBy (Page 2193)
StrgToBy modes (Page 2194)
StrgToBy functions (Page 2194)
StrgToBy error handling (Page 2195)
StrgToBy I/Os (Page 2195)
StrgToBy block diagram (Page 2196)

17.1.6 StrgToBy I/Os

I/Os of `StrgToBy`

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`StringIn`	Input string	STRING[32]	''

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	1
StrctOut	Byte structure	STRUCT	-
		• Byte1:BYTE	• 16#00
		• Byte2:BYTE	• 16#00
		• Byte3:BYTE	• 16#00
		• ...	• 16#00
		• Byte32:BYTE	• 16#00

See also

Description of StrgToBy (Page 2193)
StrgToBy modes (Page 2194)
StrgToBy functions (Page 2194)
StrgToBy error handling (Page 2195)
StrgToBy messaging (Page 2195)
StrgToBy block diagram (Page 2196)

17.1.7 StrgToBy block diagram

Block diagram of StrgToBy

A block diagram is not provided for this block.

See also

Description of StrgToBy (Page 2193)
StrgToBy modes (Page 2194)
StrgToBy functions (Page 2194)
StrgToBy error handling (Page 2195)
StrgToBy messaging (Page 2195)
StrgToBy I/Os (Page 2195)

17.2 StruAnIn - Separating an analog structured variable

17.2.1 Description of StruAnIn

Object name (type + number) and family

Type + number: FC 375

Family: Convert

Area of application for StruAnIn

The block is used for the following applications:

- Separation of an analog value with a structure into a variable of the REAL data type and a signal status

How it works

The block separates an analog value with a structure interconnected to the `In` input parameter into a variable (`Out`) of the REAL data type and a (`ST`) signal status.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for parameter `Status`

This block does not have the `Status` parameter.

See also

StruAnIn block diagram (Page 2200)

StruAnIn I/Os (Page 2199)

StruAnIn messaging (Page 2199)

StruAnIn error handling (Page 2198)

StruAnIn functions (Page 2198)

StruAnIn modes (Page 2198)

17.2.2 StruAnIn modes

StruAnIn modes

This block does not have any modes.

See also

StruAnIn block diagram (Page 2200)
StruAnIn I/Os (Page 2199)
StruAnIn messaging (Page 2199)
StruAnIn error handling (Page 2198)
StruAnIn functions (Page 2198)
Description of StruAnIn (Page 2197)

17.2.3 StruAnIn functions

Functions of StruAnIn

There are no other functions for this block.

See also

StruAnIn block diagram (Page 2200)
StruAnIn I/Os (Page 2199)
StruAnIn messaging (Page 2199)
StruAnIn error handling (Page 2198)
StruAnIn modes (Page 2198)
Description of StruAnIn (Page 2197)

17.2.4 StruAnIn error handling

StruAnIn error handling

The block does not report any errors.

See also

StruAnIn block diagram (Page 2200)
StruAnIn I/Os (Page 2199)
StruAnIn messaging (Page 2199)
StruAnIn functions (Page 2198)
StruAnIn modes (Page 2198)
Description of StruAnIn (Page 2197)

17.2.5 StruAnIn messaging

Messaging

This block does not offer messaging.

See also

StruAnIn block diagram (Page 2200)
StruAnIn I/Os (Page 2199)
StruAnIn error handling (Page 2198)
StruAnIn functions (Page 2198)
StruAnIn modes (Page 2198)
Description of StruAnIn (Page 2197)

17.2.6 StruAnIn I/Os

StruAnIn I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In	Analog value with structure	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ST	Signal status	BYTE	16#80
Value	Analog value	REAL	0.0

See also

StruAnIn block diagram (Page 2200)
StruAnIn messaging (Page 2199)
StruAnIn error handling (Page 2198)
StruAnIn functions (Page 2198)
StruAnIn modes (Page 2198)
Description of StruAnIn (Page 2197)

17.2.7 StruAnIn block diagram

StruAnIn block diagram

A block diagram is not provided for this block.

See also

StruAnIn I/Os (Page 2199)
StruAnIn messaging (Page 2199)
StruAnIn error handling (Page 2198)
StruAnIn functions (Page 2198)
StruAnIn modes (Page 2198)
Description of StruAnIn (Page 2197)

17.3 StruAnOu - Creating an analog structured variable

17.3.1 Description of StruAnOu

Object name (type + number) and family

Type + number: FC 376

Family: Convert

Area of application for StruAnOu

The block is used for the following applications:

- Merging a variable of the REAL data type and a signal status into an analog process value.

How it works

The block merges an analog value (`Value`) of the REAL data type and a signal status (`ST`) to form an analog value (`Out`) with a structure.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

StruAnOu modes (Page 2202)

StruAnOu functions (Page 2202)

StruAnOu error handling (Page 2202)

StruAnOu messaging (Page 2203)

StruAnOu I/Os (Page 2203)

StruAnOu block diagram (Page 2204)

17.3.2 StruAnOu modes

StruAnOu modes

This block does not have any modes.

See also

Description of StruAnOu (Page 2201)
StruAnOu functions (Page 2202)
StruAnOu error handling (Page 2202)
StruAnOu messaging (Page 2203)
StruAnOu I/Os (Page 2203)
StruAnOu block diagram (Page 2204)

17.3.3 StruAnOu functions

Functions of StruAnOu

There are no other functions for this block.

See also

Description of StruAnOu (Page 2201)
StruAnOu modes (Page 2202)
StruAnOu error handling (Page 2202)
StruAnOu messaging (Page 2203)
StruAnOu I/Os (Page 2203)
StruAnOu block diagram (Page 2204)

17.3.4 StruAnOu error handling

StruAnOu error handling

The block does not report any errors.

See also

Description of StruAnOu (Page 2201)
StruAnOu modes (Page 2202)
StruAnOu functions (Page 2202)
StruAnOu messaging (Page 2203)
StruAnOu I/Os (Page 2203)
StruAnOu block diagram (Page 2204)

17.3.5 StruAnOu messaging

Messaging

This block does not offer messaging.

See also

Description of StruAnOu (Page 2201)
StruAnOu modes (Page 2202)
StruAnOu functions (Page 2202)
StruAnOu error handling (Page 2202)
StruAnOu I/Os (Page 2203)
StruAnOu block diagram (Page 2204)

17.3.6 StruAnOu I/Os

StruAnOu I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
ST	Signal status	BYTE	16#80
Value	Analog value	REAL	0.0

Output parameters

Parameter	Description	Type	Default
Bad	1 = (ST = 16#00 to 16#3F)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Analog value with structure	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

See also

Description of StruAnOu (Page 2201)
StruAnOu modes (Page 2202)
StruAnOu functions (Page 2202)
StruAnOu error handling (Page 2202)
StruAnOu messaging (Page 2203)
StruAnOu block diagram (Page 2204)

17.3.7 StruAnOu block diagram

StruAnOu block diagram

A block diagram is not provided for this block.

See also

Description of StruAnOu (Page 2201)
StruAnOu modes (Page 2202)
StruAnOu functions (Page 2202)
StruAnOu error handling (Page 2202)
StruAnOu messaging (Page 2203)
StruAnOu I/Os (Page 2203)

17.4 StruDiIn - Separating a digital structured variable

17.4.1 Description of StruDiIn

Object name (type + number) and family

Type + number: FC 377

Family: Convert

Area of application for StruDiIn

The block is used for the following applications:

- Separating a binary process value into a variable of the BOOL data type, a process value and signal status.

How it works

The block separates a binary process value interconnected to the `In` input parameter into a variable of the BOOL data type and a signal status.

Configuration

Use the CFC editor to install the block in any OB

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

StruDiIn block diagram (Page 2208)

StruDiIn I/Os (Page 2207)

StruDiIn messaging (Page 2207)

StruDiIn error handling (Page 2206)

StruDiIn functions (Page 2206)

StruDiIn modes (Page 2206)

17.4.2 StruDiIn modes

StruDiIn modes

This block does not have any modes.

See also

StruDiIn block diagram (Page 2208)
StruDiIn I/Os (Page 2207)
StruDiIn messaging (Page 2207)
StruDiIn error handling (Page 2206)
StruDiIn functions (Page 2206)
Description of StruDiIn (Page 2205)

17.4.3 StruDiIn functions

Functions of StruDiIn

There are no other functions for this block.

See also

StruDiIn block diagram (Page 2208)
StruDiIn I/Os (Page 2207)
StruDiIn messaging (Page 2207)
StruDiIn error handling (Page 2206)
StruDiIn modes (Page 2206)
Description of StruDiIn (Page 2205)

17.4.4 StruDiIn error handling

StruDiIn error handling

The block does not report any errors.

See also

StruDiIn block diagram (Page 2208)
StruDiIn I/Os (Page 2207)
StruDiIn messaging (Page 2207)
StruDiIn functions (Page 2206)
StruDiIn modes (Page 2206)
Description of StruDiIn (Page 2205)

17.4.5 StruDiIn messaging

Messaging

This block does not offer messaging.

See also

StruDiIn block diagram (Page 2208)
StruDiIn I/Os (Page 2207)
StruDiIn error handling (Page 2206)
StruDiIn functions (Page 2206)
StruDiIn modes (Page 2206)
Description of StruDiIn (Page 2205)

17.4.6 StruDiIn I/Os

StruDiIn I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In	Binary process value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ST	Signal status	BYTE	16#80
Value	Binary variable	BOOL	0

See also

StruDiIn block diagram (Page 2208)
StruDiIn messaging (Page 2207)
StruDiIn error handling (Page 2206)
StruDiIn functions (Page 2206)
StruDiIn modes (Page 2206)
Description of StruDiIn (Page 2205)

17.4.7 StruDiIn block diagram

StruDiIn block diagram

A block diagram is not provided for this block.

See also

StruDiIn I/Os (Page 2207)
StruDiIn messaging (Page 2207)
StruDiIn error handling (Page 2206)
StruDiIn functions (Page 2206)
StruDiIn modes (Page 2206)
Description of StruDiIn (Page 2205)

17.5 StruDiOu - Creating a digital structured variable

17.5.1 Description of StruDiOu

Object name (type + number) and family

Type + number: FC 378

Family: Convert

Area of application for StruDiOu

The block is used for the following applications:

- Merging a variable of the BOOL data type and a signal status into a binary process value.

How it works

The block merges a variable of the BOOL data type and a signal status into a binary process value.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

StruDiOu block diagram (Page 2212)

StruDiOu I/Os (Page 2211)

StruDiOu messaging (Page 2211)

StruDiOu error handling (Page 2210)

StruDiOu functions (Page 2210)

StruDiOu modes (Page 2210)

17.5.2 StruDiOu modes

StruDiOu modes

This block does not have any modes.

See also

StruDiOu block diagram (Page 2212)
StruDiOu I/Os (Page 2211)
StruDiOu messaging (Page 2211)
StruDiOu error handling (Page 2210)
StruDiOu functions (Page 2210)
Description of StruDiOu (Page 2209)

17.5.3 StruDiOu functions

Functions of StruDiOu

There are no other functions for this block.

See also

StruDiOu block diagram (Page 2212)
StruDiOu I/Os (Page 2211)
StruDiOu messaging (Page 2211)
StruDiOu error handling (Page 2210)
StruDiOu modes (Page 2210)
Description of StruDiOu (Page 2209)

17.5.4 StruDiOu error handling

StruDiOu error handling

The block does not report any errors.

See also

StruDiOu block diagram (Page 2212)
StruDiOu I/Os (Page 2211)
StruDiOu messaging (Page 2211)
StruDiOu functions (Page 2210)
StruDiOu modes (Page 2210)
Description of StruDiOu (Page 2209)

17.5.5 StruDiOu messaging

Messaging

This block does not offer messaging.

See also

StruDiOu block diagram (Page 2212)
StruDiOu I/Os (Page 2211)
StruDiOu error handling (Page 2210)
StruDiOu functions (Page 2210)
StruDiOu modes (Page 2210)
Description of StruDiOu (Page 2209)

17.5.6 StruDiOu I/Os

StruDiOu I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
ST	Signal status	BYTE	16#80
Value	Binary variable	BOOL	0

Output parameters

Parameter	Description	Type	Default
Bad	1 = (ST = 16#00 to 16#3F)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Binary process value	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

See also

StruDiOu block diagram (Page 2212)
StruDiOu messaging (Page 2211)
StruDiOu error handling (Page 2210)
StruDiOu functions (Page 2210)
StruDiOu modes (Page 2210)
Description of StruDiOu (Page 2209)

17.5.7 StruDiOu block diagram

StruDiOu block diagram

A block diagram is not provided for this block.

See also

StruDiOu I/Os (Page 2211)
StruDiOu messaging (Page 2211)
StruDiOu error handling (Page 2210)
StruDiOu functions (Page 2210)
StruDiOu modes (Page 2210)
Description of StruDiOu (Page 2209)

17.6 StruScIn - Separating a display area into two variables

17.6.1 Description of StruScIn

Object name (type + number) and family

Type + number: FC 379

Family: Convert

Area of application for StruScIn

The block is used for the following applications:

- Separation of a display area into two variables of data type REAL.

How it works

The block separates a display area interconnected to the `Scale` input parameter into two variables of the REAL. data type

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

StruScIn block diagram (Page 2216)

StruScIn I/Os (Page 2215)

StruScIn messaging (Page 2215)

StruScIn error handling (Page 2214)

StruScIn functions (Page 2214)

StruScIn modes (Page 2214)

17.6.2 StruScIn modes

StruScIn modes

This block does not have any modes.

See also

StruScIn block diagram (Page 2216)
StruScIn I/Os (Page 2215)
StruScIn messaging (Page 2215)
StruScIn error handling (Page 2214)
StruScIn functions (Page 2214)
Description of StruScIn (Page 2213)

17.6.3 StruScIn functions

Functions of StruScIn

There are no other functions for this block.

See also

StruScIn block diagram (Page 2216)
StruScIn I/Os (Page 2215)
StruScIn messaging (Page 2215)
StruScIn error handling (Page 2214)
StruScIn modes (Page 2214)
Description of StruScIn (Page 2213)

17.6.4 StruScIn error handling

StruScIn error handling

The block does not report any errors.

See also

StruScIn block diagram (Page 2216)
StruScIn I/Os (Page 2215)
StruScIn messaging (Page 2215)
StruScIn functions (Page 2214)
StruScIn modes (Page 2214)
Description of StruScIn (Page 2213)

17.6.5 StruScIn messaging

Messaging

This block does not offer messaging.

See also

StruScIn block diagram (Page 2216)
StruScIn I/Os (Page 2215)
StruScIn error handling (Page 2214)
StruScIn functions (Page 2214)
StruScIn modes (Page 2214)
Description of StruScIn (Page 2213)

17.6.6 StruScIn I/Os

StruScIn I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
Scale	Display area	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
HiScale	High limit of display area	REAL	100.0
LoScale	Low limit of display area	REAL	0.0

See also

StruScIn block diagram (Page 2216)
StruScIn messaging (Page 2215)
StruScIn error handling (Page 2214)
StruScIn functions (Page 2214)
StruScIn modes (Page 2214)
Description of StruScIn (Page 2213)

17.6.7 StruScIn block diagram

StruScIn block diagram

A block diagram is not provided for this block.

See also

StruScIn I/Os (Page 2215)
StruScIn messaging (Page 2215)
StruScIn error handling (Page 2214)
StruScIn functions (Page 2214)
StruScIn modes (Page 2214)
Description of StruScIn (Page 2213)

17.7 StruScOu - Merging two variables into a display area

17.7.1 Description of StruScOu

Object name (type + number) and family

Type + number: FC 380

Family: Convert

Area of application for StruScOu

The block is used for the following applications:

- Merging two variables of the REAL data type into a display area.

How it works

The block merges two variables of the REAL data type into a display area.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

StruScOu block diagram (Page 2220)
StruScOu I/Os (Page 2219)
StruScOu messaging (Page 2219)
StruScOu error handling (Page 2218)
StruScOu functions (Page 2218)
StruScOu modes (Page 2218)

17.7.2 StruScOu modes

StruScOu modes

This block does not have any modes.

See also

StruScOu block diagram (Page 2220)
StruScOu I/Os (Page 2219)
StruScOu messaging (Page 2219)
StruScOu error handling (Page 2218)
StruScOu functions (Page 2218)
Description of StruScOu (Page 2217)

17.7.3 StruScOu functions

Functions of StruScOu

There are no other functions for this block.

See also

StruScOu block diagram (Page 2220)
StruScOu I/Os (Page 2219)
StruScOu messaging (Page 2219)
StruScOu error handling (Page 2218)
StruScOu modes (Page 2218)
Description of StruScOu (Page 2217)

17.7.4 StruScOu error handling

StruScOu error handling

The block does not report any errors.

See also

StruScOu block diagram (Page 2220)
StruScOu I/Os (Page 2219)
StruScOu messaging (Page 2219)
StruScOu functions (Page 2218)
StruScOu modes (Page 2218)
Description of StruScOu (Page 2217)

17.7.5 StruScOu messaging

Messaging

This block does not offer messaging.

See also

StruScOu block diagram (Page 2220)
StruScOu I/Os (Page 2219)
StruScOu error handling (Page 2218)
StruScOu functions (Page 2218)
StruScOu modes (Page 2218)
Description of StruScOu (Page 2217)

17.7.6 StruScOu I/Os

StruScOu I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
HiScale	High limit of display area	REAL	100.0
LoScale	Low limit of display area	REAL	0.0

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Scale	Display area	STRUCT • High: REAL • Low: REAL	- • 100.0 • 0.0

See also

StruScOu block diagram (Page 2220)
StruScOu messaging (Page 2219)
StruScOu error handling (Page 2218)
StruScOu functions (Page 2218)
StruScOu modes (Page 2218)
Description of StruScOu (Page 2217)

17.7.7 StruScOu block diagram

StruScOu block diagram

A block diagram is not provided for this block.

See also

StruScOu I/Os (Page 2219)
StruScOu messaging (Page 2219)
StruScOu error handling (Page 2218)
StruScOu functions (Page 2218)
StruScOu modes (Page 2218)
Description of StruScOu (Page 2217)

17.8 STIn - Separating the signal status into individual binary displays

17.8.1 Description of STIn

Object name (type + number) and family

Type + number: FC 373

Family: Convert

Area of application for STIn

The block is used for the following applications:

- Separation of signal status into individual binary displays

How it works

The block separates a signal status interconnected to the input parameter into individual binary displays.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

STIn block diagram (Page 2224)

STIn I/Os (Page 2223)

STIn messaging (Page 2223)

STIn error handling (Page 2222)

STIn functions (Page 2222)

STIn modes (Page 2222)

17.8.2 STIn modes

STIn modes

This block does not have any modes.

See also

STIn block diagram (Page 2224)
STIn I/Os (Page 2223)
STIn messaging (Page 2223)
STIn error handling (Page 2222)
STIn functions (Page 2222)
Description of STIn (Page 2221)

17.8.3 STIn functions

Functions of STIn

There are no other functions for this block.

See also

STIn block diagram (Page 2224)
STIn I/Os (Page 2223)
STIn messaging (Page 2223)
STIn error handling (Page 2222)
STIn modes (Page 2222)
Description of STIn (Page 2221)

17.8.4 STIn error handling

STIn error handling

The block does not report any errors.

See also

STIn block diagram (Page 2224)
STIn I/Os (Page 2223)
STIn messaging (Page 2223)
STIn functions (Page 2222)
STIn modes (Page 2222)
Description of STIn (Page 2221)

17.8.5 STIn messaging

Messaging

This block does not offer messaging.

See also

STIn block diagram (Page 2224)
STIn I/Os (Page 2223)
STIn error handling (Page 2222)
STIn functions (Page 2222)
STIn modes (Page 2222)
Description of STIn (Page 2221)

17.8.6 STIn I/Os

I/Os of STIn

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In	Signal status	BYTE	16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ST_00	1 = Bad, device related	BOOL	0
ST_28	1 = Bad, process related	BOOL	0
ST_60	1 = Manipulated value (for example, substitute value, simulation, last valid value)	BOOL	0
ST_68	1 = Unknown, device related	BOOL	0
ST_78	1 = Unknown, process related	BOOL	0
ST_80	1 = Good	BOOL	0
ST_A4	1 = Maintenance request	BOOL	0

See also

STIn error handling (Page 2222)
STIn block diagram (Page 2224)
Description of STIn (Page 2221)
STIn modes (Page 2222)
STIn functions (Page 2222)
STIn messaging (Page 2223)

17.8.7 STIn block diagram

STIn block diagram

A block diagram is not provided for this block.

See also

Description of STIn (Page 2221)
STIn modes (Page 2222)
STIn functions (Page 2222)
STIn error handling (Page 2222)
STIn messaging (Page 2223)
STIn I/Os (Page 2223)

17.9 STOu - Merging individual binary signals into a signal status

17.9.1 Description of STOu

Object name (type + number) and family

Type + number: FC 374

Family: Convert

Area of application for STOu

The block is used for the following applications:

- Merging individual binary signals into a signal status

How it works

The block merges individual binary signals into a `Out` signal status.

If several binary signals are set, the one with the highest priority becomes effective, as described in the section Forming and outputting the signal status for technologic blocks (Page 108) for technologic blocks.

If no binary signal is set, the "Bad, process related" signal status is set.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

STOu block diagram (Page 2228)

STOu I/Os (Page 2227)

STOu messaging (Page 2227)

STOu error handling (Page 2226)

STOu functions (Page 2226)

STOu modes (Page 2226)

17.9.2 STOu modes

STOu modes

This block does not have any modes.

See also

STOu block diagram (Page 2228)
STOu I/Os (Page 2227)
STOu messaging (Page 2227)
STOu error handling (Page 2226)
STOu functions (Page 2226)
Description of STOu (Page 2225)

17.9.3 STOu functions

Functions of STOu

There are no other functions for this block.

See also

STOu block diagram (Page 2228)
STOu I/Os (Page 2227)
STOu messaging (Page 2227)
STOu error handling (Page 2226)
STOu modes (Page 2226)
Description of STOu (Page 2225)

17.9.4 STOu error handling

STOu error handling

The block does not report any errors.

See also

STOu block diagram (Page 2228)
STOu I/Os (Page 2227)
STOu messaging (Page 2227)
STOu functions (Page 2226)
STOu modes (Page 2226)
Description of STOu (Page 2225)

17.9.5 STOu messaging

Messaging

This block does not offer messaging.

See also

STOu block diagram (Page 2228)
STOu I/Os (Page 2227)
STOu error handling (Page 2226)
STOu functions (Page 2226)
STOu modes (Page 2226)
Description of STOu (Page 2225)

17.9.6 STOu I/Os

STOu I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
ST_00	1 = Bad, device related	BOOL	0
ST_28	1 = Bad, process related	BOOL	0
ST_60	1 = Manipulated value (for example, substitute value, simulation, last valid value)	BOOL	0
ST_68	1 = Unknown, device related	BOOL	0
ST_78	1 = Unknown, process related	BOOL	0
ST_80	1 = Good	BOOL	0
ST_A4	1 = Maintenance request	BOOL	0

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Signal status	BYTE	16#80

See also

STOu error handling (Page 2226)
STOu block diagram (Page 2228)
STOu messaging (Page 2227)
STOu functions (Page 2226)
STOu modes (Page 2226)
Description of STOu (Page 2225)

17.9.7 STOu block diagram

STOu block diagram

A block diagram is not provided for this block.

See also

STOu I/Os (Page 2227)
STOu messaging (Page 2227)
STOu error handling (Page 2226)
STOu functions (Page 2226)
STOu modes (Page 2226)
Description of STOu (Page 2225)

17.10 MSTIn - Separating the maintenance status into individual status displays

17.10.1 Description of MSTIn

Object name (type + number) and family

Type + number: FB 1858

Family: Convert

Area of application for MSTIn

The block is used for the following applications:

- Separation of the maintenance status into individual status displays

How it works

The block separates a maintenance status interconnected to the `In` input parameter into individual status displays.

If the input parameter receives information that at least one value is simulated, for example (`In` = 16#00000003), this is indicated at output parameter `MST_03` with the value 1.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

MSTIn block diagram (Page 2232)

MSTIn I/Os (Page 2231)

MSTIn messaging (Page 2231)

MSTIn error handling (Page 2230)

MSTIn functions (Page 2230)

MSTIn modes (Page 2230)

17.10.2 MSTIn modes

MSTIn modes

This block does not have any modes.

See also

MSTIn I/Os (Page 2231)
MSTIn messaging (Page 2231)
MSTIn error handling (Page 2230)
MSTIn functions (Page 2230)
MSTIn block diagram (Page 2232)
Description of MSTIn (Page 2229)

17.10.3 MSTIn functions

Functions of MSTIn

There are no other functions for this block.

See also

MSTIn block diagram (Page 2232)
MSTIn I/Os (Page 2231)
MSTIn messaging (Page 2231)
MSTIn error handling (Page 2230)
MSTIn modes (Page 2230)
Description of MSTIn (Page 2229)

17.10.4 MSTIn error handling

MSTIn error handling

The block does not report any errors.

See also

MSTIn block diagram (Page 2232)
MSTIn I/Os (Page 2231)
MSTIn messaging (Page 2231)
MSTIn functions (Page 2230)
MSTIn modes (Page 2230)
Description of MSTIn (Page 2229)

17.10.5 MSTIn messaging

Messaging

This block does not offer messaging.

See also

MSTIn block diagram (Page 2232)
MSTIn I/Os (Page 2231)
MSTIn error handling (Page 2230)
MSTIn functions (Page 2230)
MSTIn modes (Page 2230)
Description of MSTIn (Page 2229)

17.10.6 MSTIn I/Os

MSTIn I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In	Maintenance status	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
MST_00	1 = Good	BOOL	0
MST_01	1 = Passivated	BOOL	0
MST_02	1 = "Out of service"	BOOL	0
MST_03	1 = At least one process value is simulated	BOOL	0
MST_04	1 = "Local mode"	BOOL	0
MST_05	1 = Maintenance requirement	BOOL	0
MST_06	1 = Maintenance request	BOOL	0
MST_07	1 = Maintenance alarm	BOOL	0
MST_08	1 = Unknown	BOOL	0
MST_09	1 = Configuration changed	BOOL	0

See also

MSTIn block diagram (Page 2232)
MSTIn messaging (Page 2231)
MSTIn error handling (Page 2230)
MSTIn functions (Page 2230)
MSTIn modes (Page 2230)
Description of MSTIn (Page 2229)

17.10.7 MSTIn block diagram

MSTIn block diagram

A block diagram is not provided for this block.

See also

MSTIn I/Os (Page 2231)
MSTIn messaging (Page 2231)
MSTIn error handling (Page 2230)
MSTIn functions (Page 2230)
MSTIn modes (Page 2230)
Description of MSTIn (Page 2229)

17.11 MSTOu - Merging individual status displays into a maintenance status

17.11.1 Description of MSTOu

Object name (type + number) and family

Type + number: FB 1859

Family: Convert

Area of application for MSTOu

The block is used for the following applications:

- Merging individual status displays into a maintenance status

How it works

The block merges individual status displays into a maintenance status. If several status displays are set, the status display with the highest number becomes effective.

If status display `MST_03` = 1 is set, for example, this is indicated at output parameter `Out` with the value 16#00000003.

If no status display is set, the `Out` = 16#00 maintenance status is set.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

MSTOu block diagram (Page 2236)

MSTOu I/Os (Page 2235)

MSTOu messaging (Page 2235)

MSTOu error handling (Page 2234)

MSTOu functions (Page 2234)

MSTOu modes (Page 2234)

17.11.2 MSTOu modes

MSTOu modes

This block does not have any modes.

See also

MSTOu block diagram (Page 2236)
MSTOu I/Os (Page 2235)
MSTOu messaging (Page 2235)
MSTOu error handling (Page 2234)
MSTOu functions (Page 2234)
Description of MSTOu (Page 2233)

17.11.3 MSTOu functions

Functions of MSTOu

There are no other functions for this block.

See also

MSTOu block diagram (Page 2236)
MSTOu I/Os (Page 2235)
MSTOu messaging (Page 2235)
MSTOu error handling (Page 2234)
MSTOu modes (Page 2234)
Description of MSTOu (Page 2233)

17.11.4 MSTOu error handling

MSTOu error handling

The block does not report any errors.

See also

MSTOu block diagram (Page 2236)
MSTOu I/Os (Page 2235)
MSTOu messaging (Page 2235)
MSTOu functions (Page 2234)
MSTOu modes (Page 2234)
Description of MSTOu (Page 2233)

17.11.5 MSTOu messaging

Messaging

This block does not offer messaging.

See also

MSTOu block diagram (Page 2236)
MSTOu I/Os (Page 2235)
MSTOu error handling (Page 2234)
MSTOu functions (Page 2234)
MSTOu modes (Page 2234)
Description of MSTOu (Page 2233)

17.11.6 MSTOu I/Os

MSTOu I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
MST_00	1 = Good	BOOL	0
MST_01	1 = Passivated	BOOL	0
MST_02	1 = "Out of service"	BOOL	0
MST_03	1 = At least one process value is simulated	BOOL	0
MST_04	1 = "Local mode"	BOOL	0
MST_05	1 = Maintenance requirement	BOOL	0

Parameter	Description	Type	Default
MST_06	1 = Maintenance request	BOOL	0
MST_07	1 = Maintenance alarm	BOOL	0
MST_08	1 = Unknown	BOOL	0
MST_09	1 = Configuration changed	BOOL	0

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Maintenance status	DWORD	16#00000000

See also

MSTOu block diagram (Page 2236)
MSTOu messaging (Page 2235)
MSTOu error handling (Page 2234)
MSTOu functions (Page 2234)
MSTOu modes (Page 2234)
Description of MSTOu (Page 2233)

17.11.7 MSTOu block diagram

MSTOu block diagram

A block diagram is not provided for this block.

See also

MSTOu I/Os (Page 2235)
MSTOu messaging (Page 2235)
MSTOu error handling (Page 2234)
MSTOu functions (Page 2234)
MSTOu modes (Page 2234)
Description of MSTOu (Page 2233)

17.12 RealToDw - Converting REAL to DWORD

17.12.1 Description of RealToDw

Object name (type + number) and family

Type + number: FC 390

Family: Convert

Area of application for RealToDw

The block is used for the following applications:

- Converting a REAL number to a double word (DWORD)

Note

Converts the value of a REAL number into a double word (DWORD). You must use the R_DW block (CFC fundamental blocks) to convert the bit character string.

How it works

With a REAL number between 0 and 4.294967e09 at input `In`, the value is applied and output as a DWORD at the output `Out`.

Note

The floating-point numbers in STEP 7 are accurate to 6 decimal places. You can therefore specify a maximum of 6 decimal places for floating-point constants.

If `In` is outside these limits, the error number 30 is output at the `ErrorNum` output and the `Out` is set to these limits.

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for Status parameter

This block does not have the `Status` parameter.

17.12.2 RealToDw modes

Operating modes of RealToDw

This block does not have any modes.

17.12.3 ReaLToDw functions

Functions of RealToDw

There are no other functions for this block.

17.12.4 RealToDw error handling

Error handling of RealToDw

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed
0	There is no error.
30	`In` cannot be displayed in DWORD format

17.12.5 RealToDw messaging

Messaging

This block does not offer messaging.

17.12.6 RealToDw I/Os

I/Os of RealToDw

Input parameters

Parameter	Description	Type	Default
In	Analog input value	REAL	0

Output parameters

Parameter	Description	Type	Default
ErrorNum	Error number	INT	-1
Out	Converted output value	DWORD	0
ST	Status	BYTE	16#80

17.12.7 RealToDw block diagram

Block diagram of RealToDw

A block diagram is not provided for this block.

17.13 StateMap - Conversion of other signal states into APL signal states

17.13.1 Description of StateMap

Object name (type + number)

Type + number: FC 383

Family: Convert

Area of application of StateMap

This block can map other signal states in an APL signal states.

How it works

This block converts the following signal states into APL signal states.

State	ST_In	ST_Out
Valid value	16#80	16#80
Manipulated value (for example, substitute value, simulation, last valid value)	16#60	16#60
Substitute value	**16#48**	**16#60**
Last valid value	**16#44**	**16#60**
Uncertain, device related	16#68	16#68
Uncertain, process related	16#78	16#78
Unknown, device related, range violation	**16#54**	**16#68**
Maintenance request	16#A4	16#A4
Invalid value (-STOP)	16#00	16#00
Uncertain, device related	Otherwise	**16#68**

Configuration

Use the CFC editor to install the block in any OB.

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

17.13.2 StateMap modes

StateMap modes

This block does not have any operating modes.

17.13.3 StateMap functions

Functions of StateMap

There are no other functions for this block.

17.13.4 StateMap error handling

StateMap error handling

This block does not have any error handling.

17.13.5 StateMap messaging

Messaging

This block does not offer messaging.

17.13.6 StateMap I/Os

I/Os of StateMap

Input parameters

Parameter	Meaning	Data type	Default
EN	1 = Called block will be processed	BOOL	1
ST_In	Signal status input	BYTE	16#80

Output parameters

Parameter	Meaning	Data type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
ST_Out	Signal status output	BYTE	16#80

17.13.7 StateMap block diagram

StateMap block diagram

A block diagram is not provided for this block.

Maintenance blocks 18

18.1 MuxMST - Determination of the worst maintenance status

18.1.1 Description of MuxMST

Object name (type + number) and family

Type + number: FB 1861

Family: Maint

Area of application for MuxMST

The block is used for the following applications:

- Determination of the worst maintenance status (from a maximum of 10 statuses)

How it works

The block determines the worst of several maintenance states. Each status is made available to the block via interconnection to the `Inx` input parameter (x = 01 ... 10) . The states are compared and the highest value written to the `Out` output parameter.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

During startup, the `Inx` input parameters (x = 01 ... 10) and output parameter are reset to their defaults.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

See also

MuxMST block diagram (Page 2246)

MuxMST I/Os (Page 2246)

MuxMST messaging (Page 2245)

MuxMST error handling (Page 2245)
MuxMST functions (Page 2244)
MuxMST modes (Page 2244)

18.1.2 MuxMST modes

MuxMST operating modes

This block does not have any modes.

See also

MuxMST block diagram (Page 2246)
MuxMST I/Os (Page 2246)
MuxMST messaging (Page 2245)
MuxMST error handling (Page 2245)
MuxMST functions (Page 2244)
Description of MuxMST (Page 2243)

18.1.3 MuxMST functions

Functions of MuxMST

This block provides no other functions.

See also

MuxMST block diagram (Page 2246)
MuxMST I/Os (Page 2246)
MuxMST messaging (Page 2245)
MuxMST error handling (Page 2245)
MuxMST modes (Page 2244)
Description of MuxMST (Page 2243)

18.1.4 MuxMST error handling

MuxMST error handling

The block does not report any errors.

See also

MuxMST block diagram (Page 2246)
MuxMST I/Os (Page 2246)
MuxMST messaging (Page 2245)
MuxMST functions (Page 2244)
MuxMST modes (Page 2244)
Description of MuxMST (Page 2243)

18.1.5 MuxMST messaging

Messaging

This block does not offer messaging.

See also

MuxMST block diagram (Page 2246)
MuxMST I/Os (Page 2246)
MuxMST error handling (Page 2245)
MuxMST functions (Page 2244)
MuxMST modes (Page 2244)
Description of MuxMST (Page 2243)

18.1.6 MuxMST I/Os

MuxMST I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In01	Input for signal of maintenance status 0	DWORD	16#00000000
In02	Input for signal of maintenance status 1	DWORD	16#00000000
In03	Input for signal of maintenance status 2	DWORD	16#00000000
In04	Input for signal of maintenance status 3	DWORD	16#00000000
In05	Input for signal of maintenance status 4	DWORD	16#00000000
In06	Input for signal of maintenance status 5	DWORD	16#00000000
In07	Input for signal of maintenance status 6	DWORD	16#00000000
In08	Input for signal of maintenance status 7	DWORD	16#00000000
In09	Input for signal of maintenance status 8	DWORD	16#00000000
In10	Input for signal of maintenance status 9	DWORD	16#00000000

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Output signal with the worst signal status	DWORD	16#00000000

See also

MuxMST block diagram (Page 2246)
MuxMST messaging (Page 2245)
MuxMST error handling (Page 2245)
MuxMST functions (Page 2244)
MuxMST modes (Page 2244)
Description of MuxMST (Page 2243)

18.1.7 MuxMST block diagram

MuxMST block diagram

A block diagram is not provided for this block.

See also

MuxMST I/Os (Page 2246)
MuxMST messaging (Page 2245)
MuxMST error handling (Page 2245)
MuxMST functions (Page 2244)
MuxMST modes (Page 2244)
Description of MuxMST (Page 2243)

18.2 MuxST- Determination of the worst signal status

18.2.1 Description of MuxST

Object name (type + number) and family

Type + number: FB 1862

Family: Maint

Area of application for MuxST

The block is used for the following applications:

- Determination of the worst signal status (from a maximum of 10 statuses)

How it works

The block determines the worst of several signal states. Each status is made available to the block via interconnection to the `Inx` input parameter (x = 1 ... 10). The states are compared and the value with the highest priority is written to the `Out` output parameter.

Also refer to the section Forming and outputting the signal status for blocks with configurable status prioritization (Page 114) for information on priorities.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

During startup, the `Inx` input parameters (x = 1 ... 10) and output parameter are reset to their defaults.

Status word allocation for `Status1` parameter

This block does not have the `Status` parameter.

See also

MuxST block diagram (Page 2252)

MuxST I/Os (Page 2251)

MuxST messaging (Page 2250)

MuxST error handling (Page 2250)

MuxST functions (Page 2249)
MuxST modes (Page 2249)

18.2.2 MuxST modes

MuxST operating modes

This block does not have any modes.

See also

MuxST block diagram (Page 2252)
MuxST I/Os (Page 2251)
MuxST messaging (Page 2250)
MuxST error handling (Page 2250)
MuxST functions (Page 2249)
Description of MuxST (Page 2248)

18.2.3 MuxST functions

Functions of MuxST

The functions for this block are listed below.

Selecting signals for processing

Using the `SelInput` input parameter, you select the number of interconnectable input parameters to be used for processing in the block. The selected number denotes the input parameters used `In1` ... `Inn`.

If, for example, you set this input parameter to `SelInput` = 3, the input parameters `In1`, `In2` and `In3` will be used for processing.

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for blocks with configurable status prioritization (Page 114).

See also

MuxST block diagram (Page 2252)
MuxST I/Os (Page 2251)

MuxST messaging (Page 2250)
MuxST error handling (Page 2250)
MuxST modes (Page 2249)
Description of MuxST (Page 2248)

18.2.4 MuxST error handling

MuxST error handling

The block does not report any errors.

See also

MuxST block diagram (Page 2252)
MuxST I/Os (Page 2251)
MuxST messaging (Page 2250)
MuxST functions (Page 2249)
MuxST modes (Page 2249)
Description of MuxST (Page 2248)

18.2.5 MuxST messaging

Messaging

This block does not offer messaging.

See also

MuxST block diagram (Page 2252)
MuxST I/Os (Page 2251)
MuxST error handling (Page 2250)
MuxST functions (Page 2249)
MuxST modes (Page 2249)
Description of MuxST (Page 2248)

18.2.6 MuxST I/Os

MuxST I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
In1	Input for signal of signal status 0	BYTE	16#80
In2	Input for signal of signal status 1	BYTE	16#80
In3	Input for signal of signal status 2	BYTE	16#80
In4	Input for signal of signal status 3	BYTE	16#80
In5	Input for signal of signal status 4	BYTE	16#80
In6	Input for signal of signal status 5	BYTE	16#80
In7	Input for signal of signal status 6	BYTE	16#80
In8	Input for signal of signal status 7	BYTE	16#80
In9	Input for signal of signal status 8	BYTE	16#80
In10	Input for signal of signal status 9	BYTE	16#80
SelInput*	Selection of the input parameter for forming the worst signal status	INT	2
SelPrio*	Setting of the prioritization for forming the worst signal status	INT	0

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
Out	Output of the worst signal status	BYTE	16#80

See also

MuxST block diagram (Page 2252)
MuxST messaging (Page 2250)
MuxST error handling (Page 2250)
MuxST functions (Page 2249)
MuxST modes (Page 2249)
Description of MuxST (Page 2248)

18.2.7 MuxST block diagram

MuxST block diagram

A block diagram is not provided for this block.

See also

MuxST I/Os (Page 2251)

MuxST messaging (Page 2250)

MuxST error handling (Page 2250)

MuxST functions (Page 2249)

MuxST modes (Page 2249)

Description of MuxST (Page 2248)

18.3 STRep - Status display of block groups

18.3.1 Description of STRep

Object name (type + number) and family

Type + number: FB 1801

Family: MAINT

Area of application of STRep

The STRep block is used for displaying the status of a group of blocks that is designed for hiding messages automatically.

How it works

The block has 32 `STRUCT` type inputs that describe defined states. Depending on which `StateX` input is set, `QSTATE` is output at the INT output. If several `StateX` inputs are set, the most significant is output and the `MPSA` output is set.

The status of the `StateX` inputs does not influence the result at `QSTATE`.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB30 to OB38). The block is also installed automatically in the startup OB (OB100).

Startup characteristics

The block does not have any startup characteristics.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

18.3.2 STRep modes

Operating modes of STRep

This block does not have any modes.

18.3.3 STRep functions

Functions of STRep

There are no other functions for this block.

18.3.4 STRep error handling

Error handling of STRep

The block does not report any errors.

18.3.5 STRep messaging

Messaging

This block does not offer messaging.

18.3.6 STRep I/Os

I/Os of STRep

Input parameters

Parameter	Description	Type	Default
`State1 ... State32`	Process status 1 ... 32	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`QERR`	1 = Error	BOOL	1
`QSTATE`	Process status	INT	0

Output parameters

Parameter	Description	Type	Default
MPSA	1 = More than one process status is active	STRUCT • Value: BOOL • ST: BYTE	- • 1 • 16#80
QSTATE	Process status of the type INT 0...32	INT	0

18.3.7 STRep block diagram

Block diagram of STRep

A block diagram is not provided for this block.

18.4 AssetM - Process variable monitoring for violation of limits

18.4.1 Description of AssetM

Object name (type + number) and family

Type and number: FB 1840

Family: Maint

Area of application of AssetM

The block is used for the following applications:

Monitoring of up to 3 analog process variables for exceeding 3 limits respectively. It reports the maintenance status of the process variables:

- When overshooting a limit
- Via the device-based signal status or
- Via binary message inputs

How it works

The block monitors up to 3 inputs for high and low violations of three limits respectively. Upon reaching or exceeding a limit, the respective output is set and a corresponding message is generated.

The three process variables are equivalent for limit monitoring as long as the overshoot of a limit (e.g. maintenance request) is reported by one process variable. A new message will only be created again if all process variables have undershot this limit at least for one cycle.

The monitoring of individual limits can be deactivated.

In addition to limit monitoring, the signal status of the individual process variables is also analyzed. The device-specific signal status of a process value triggers the corresponding message that is also used by the limit monitoring.

Configuration

In the CFC editor, install the block in a cyclic interrupt OB (e.g. OB32). The block is also installed automatically in the startup OB (OB 100).

Startup characteristics

The messages are suppressed after startup for the number of cycles set at `RunUpCyc`.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

Binary message inputs

Message1	Bad, maintenance alarm	Message (S) requires acknowledgement
Message2	Uncertain, maintenance request	Message (F) requires acknowledgement
Message3	Good, maintenance required	Message (M) requires acknowledgement
Message4	Bad, local operation/functional check	Message (S) requires acknowledgement
Message5	Uncertain, simulation	Message (SA) does not require acknowl-edgement
Message6	Bad, device out of service	Message (SA) does not require acknowl-edgement
Message7	bad, passivated	Message (SA) does not require acknowl-edgement

For each PVx, there is an input (PVx_Rst) and an output (P_PVx_Rst) that can be used to reset a technologic block.

The statuses are created with ALARM_8P for messages requiring acknowledgment and with NOTIFY_8P for those not requiring acknowledgment. The message function can be disabled by setting MsgLock = 0. In this case, MS = 8 is set.

The detailed diagnostics is shown in the diagnostic view of the faceplate via the inputs Diag1 to Diag16.

If one of the inputs is set to 1, then the status display is shown in front of the corresponding text.

The texts for the relevant inputs Diag1 to Diag16 are entered in the parameter data of the EDD for the relevant instance (see also section PLT ID).

If one of the inputs Diag1 to Diag16 is set to 1, the explanatory text "Additional status available" is output if an internal interrupt is triggered through PV0, PV1 or PV2.

PLT-ID

Die PLT-ID is a connection parameter between a PDM object (parameter data EDD) and the faceplate in the maintenance station. The PLT-ID is linked to the PDM object.

The PDM object is generated in the SIMATIC Manager as follows:

1. Select **View > Process Device Plant View** in SIMATIC Manager.
2. In SIMATIC PDM, select a project in which you want to insert the AssetM block.
3. Select **File > New object**.
4. Right-click the inserted tag object and select **Device selection (Reassign)...** from the context menu.
5. In the tree structure **Devices > DATA_OBJECTS > CFC >**, select AssetMon and click **OK**.
6. Right-click the tag object and select **Open object** from the context menu, and then enter all necessary data in the parameter assignment screen form.
7. Select **File > Save** and close the parameter assignment screen form.

You can then assign parameters for the generated PLT-ID at the associated parameter "PLT_ID".

Note

The PLT-IDs cannot be changed or deleted individually.

Creating the maintenance status (MS)

MS depends on:

- From the signal status of the signals `PV0`, `PV1` and `PV2`
- from the binary message inputs (external `MS`)
- from the interconnectable input `MS_In` (external maintenance state)
- The interconnectable input `STATUS`. The process-related errors have no effect; only the device-based errors have an effect.

Of all these events the highest priority event will be displayed in the MS.

The 16 `Diagx` binary inputs do not have any influence on the MS; they are used only for visualization of the detail diagnostics in the diagnostics view of the faceplate.

Note

The table is valid exclusively for the MS and not for the signal status displays of the individual process values; these are created exclusively by the `ST` parameters.

The priority is similar to the `MS` coding. Therefore, the following applies: The greater the value of the `MS`, the higher the priority.

PVx.Value	PVx.ST	MS_In	Event	STATUS	Messa-gex.Value	O_MS
-	-	9	Configuration change	16#84 .. 87	-	9
-	-	8	Untested/unknown	-	-	8
PVx.Value >= PVx_AH	16#00	7	bad, maintenance alarm	16#00 .. 1B, 16#24 .. 27, 16#44 .. 4B	x = 1	7
PVx_AH > PVx.Value >= PVx_DH	16#68	6	uncertain, maintenance request	16#40 .. 43, 16#50 .. 5F, 16#64 .. 6B, 16#A8 .. AB	x = 2	6
PVx_DH > PVx.Value >= PVx_RH	16#A4	5	good, maintenance required	16#A4 .. A7	x = 3	5
-	-	4	bad, local operation/functional check	16#3C .. 3F	x = 4	4
-	16#60	3	Uncertain, manipulated value (for example, substitute value, simu-lation, last valid value)	16#60 .. 63, 16#70 .. 73	x = 5	3

PVx.Value	PVx.ST	MS_In	Event	STATUS	Messa-gex.Value	O_MS
-	-	2	bad, device out of service	16#1C .. 1F	x = 6	2
-	-	1	bad, passivated	16#23	x = 7	1
PVx_RH > PVx.Value	16#28, 16#78	0	process-related fault	16#28 .. 2B, 16#78 .. 7B, 16#A0 .. A3	-	

18.4.2 AssetM modes

Operating modes of AssetM

This block does not have any modes.

18.4.3 AssetM functions

Functions of AssetM

The functions for this block are listed below.

Configurable reactions using the `Feature` parameter

You can find an overview of all reactions provided by the `Feature` parameter in the Configurable functions using the Feature I/O (Page 130) section. The following functionality is available for this block at the relevant bits:

Bit	Function
22	Update acknowledgment and error status of the message call (Page 159)
28	Disabling operating points (Page 144)

Forming the signal status for blocks

This block provides the standard function Forming and outputting the signal status for interlock blocks (Page 115).

The worst signal status `ST_Worst` for the block is formed from the following parameters:

- `PV0.ST`
- `PV1.ST`
- `PV2.ST`

18.4.4 AssetM error handling

Error handling of AssetM

Refer to the section Error handling (Page 119) in the basic instructions for the error handling of all blocks.

The following errors can be displayed for this block:

- Error numbers

Overview of error numbers

The `ErrorNum` I/O can be used to output the following error numbers:

Error number	Meaning of the error number
-1	Predefined value when inserting the block; block is not processed
0	There is no error.

18.4.5 AssetM messages

Messaging

The following messages can be generated for this block:

- Process control fault
- Process messages

Process control fault

Message instance	Message identifier	Message class	Event
`MsgEvId1`	SIG 1	AS process control message - fault	Maintenance alarm @4W%t#AssetM_TXT@
	SIG 2	AS process control message - error	Maintenance demanded @4W%t#AssetM_TXT@
	SIG 3	Preventive maintenance - General	Maintenance required @4W%t#AssetM_TXT@
	SIG 4	Reserved	
	SIG 5	Reserved	
	SIG 6	Reserved	
	SIG 7	AS process control message - fault	Device out of service
	SIG 8	Reserved	

Process messages

Message instance	Message identifier	Message class	Event
MsgEvId2	1	Status AS	Passivated
	2	Status AS	
	3	Status AS	Manipulated value (for example, substitute value, simulation, last valid value)
	4	Status AS	Manipulated value (for example, substitute value, simulation, last valid value)
	5	Status AS	Configuration change
	6	Status AS	Process-related fault
	7	Reserved	
	8	Reserved	

Associated values

Message block ALARM 8P	Associated value	Meaning
MsgEvId1	4	Text number from AssetM_TXT

System text library AssetM

Index	Text
1	Further status available
2	No further status available

18.4.6 AssetM I/Os

I/Os of AssetM

Input parameters

I/O (parameter)	Meaning	Type	Default
Diagx	Asset detail diagnosis (x = 1 to 16)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Feature	I/O for additional functions (Page 2259)	STRUCT • Bit 0: BOOL • ... • Bit 31: BOOL	- • 0 • 0 • 0
Message1	1 = message: bad, maintenance alarm	BOOL	0
Message2	1 = message: uncertain, maintenance request	BOOL	0
Message3	1 = message: good, maintenance required	BOOL	0
Message4	1 = message: bad, local operation/functional check	BOOL	0
Message5	1 = message: uncertain, simulation	BOOL	0
Message6	1 = message: bad, out of service	BOOL	0
Message7	1 = message: passivated	BOOL	0
MS*	Maintenance status	DWORD	16#000000 00
MsgEvId1	Message event ID 1	DWORD	16#000000 24
MsgEvId2	Message event ID 2	DWORD	16#000000 25
MsgLock	1 = Suppress process messages. Refer also to the Suppressing messages using the MsgLock parameter (Page 201) chapter for more on this.	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
MS_Release	Release for maintenance	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
Ms_In	External interconnectable MS	DWORD	16#000000 00
PLT_ID	Asset ID of EDD	DWORD	16#000000 00
PV_Hyst	PV_Hysteresis for messages	REAL	5.0
PVx_AH	Limit PVx (x=0, 1, 2) maintenance alarm	REAL	100.0
PVx_AH_EN	1 = suppress monitoring of the limit PVx (x = 0, 1, 2), maintenance alarm exceeded	BOOL	0
PVx_DH	Limit PVx (x = 0, 1, 2) maintenance request	REAL	100.0
PVx_DH_EN	1 = suppress monitoring of the limit PVx (x = 0, 1, 2), maintenance request exceeded	BOOL	0
PVx_RH	Limit PVx (x = 0, 1, 2) maintenance required	REAL	100.0
PVx_RH_EN	1 = suppress monitoring of the limit PVx (x = 0, 1, 2), maintenance required exceeded	BOOL	0

I/O (parameter)	Meaning	Type	Default
PVx	Process value PVx (x = 0, 1, 2)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PVx_Rst*	1 = Reset PVx (x = 0, 1, 2)	BOOL	0
PVxUnit	Unit of the process value	INT	0
RunUpCyc	Number of run-up cycles	INT	3
Status	External status	BYTE	16#80

* Values can be written back to these inputs during processing of the block by the block algorithm.

Output parameters

I/O (parameter)	Meaning	Type	Default
ErrorNum	Program error	INT	-1
MsgAckn1	Message recognized	WORD	16#0000
MsgErrx	Message errors occurred (x =1, 2)	WORD	16#0000
MsgStatx	Message error information (x = 1, 2)	WORD	16#0000
O_MS	Maintenance status	DWORD	16#00000000
PVx_Diff	Differential value to the next expected alarm (x = 0, 1, 2)	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80
PVx_AH_AcT	1 = Limit PVx (x = 0, 1, 2), maintenance alarm exceeded	BOOL	0
PVx_DH_AcT	1 = Limit PVx (x = 0, 1, 2), maintenance required exceeded	BOOL	0
PVx_RH_AcT	1 = Limit PVx (x = 0, 1, 2), maintenance request exceeded	BOOL	0
P_PVx_Rst	1 = Reset PVx (x = 0, 1, 2)	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
ST_Worst	Worst signal status of the inputs PVx (x = 0, 1, 2)	BYTE	16#80

See also

AssetM functions (Page 2259)

18.4.7 AssetM block diagram

Block diagram of ASSETM

A block diagram is not provided for this block.

System blocks 19

19.1 AddInt64 - Addition of two 64-bit integer variables

19.1.1 Description of AddInt64

Object name (type + number) and family

Type + number: FC 353

Family: System

Area of application for AddInt64

The block is used for the following applications:

- Addition of two 64-bit integer variables

This block is used by the PIDConR block for calculations that are twice as accurate (64-bit number format).

Therefore this block will not be described in detail.

19.2 AddR64 - Addition of two 64-bit REAL variables

19.2.1 Description of AddR64

Object name (type + number) and family

Type + number: FC 354

Family: System

Area of application for AddR64

The block is used for the following applications:

- Addition of two 64-bit REAL variables

This block is used by the PIDConR block for calculations that are twice as accurate (64-bit number format).

Therefore this block will not be described in detail.

19.3 DiToInt64 - Converting from DINT to Int64

19.3.1 Description of DiToInt64

Object name (type + number) and family

Type + number: FC 357

Family: System

Area of application for DiToInt64

The block is used for the following applications:

- Converting from DINT to Int64

This block is used by the PIDConR block for calculations that are twice as accurate (64-bit number format).

Therefore this block will not be described in detail.

19.4 Int64ToDi - Converting from Int64 to DINT

19.4.1 Description of Int64ToDi

Object name (type + number) and family

Type + number: FC 359

Family: System

Area of application for Int64ToDi

The block is used for the following application:

- Converting from Int64 to DINT

This block will not be described in detail.

19.5 MemR256 - Increasing the number of internal previous values

19.5.1 Description of MemR256

Object name (type + number)

Object name (type + number) and family

Type and number: FB 1930

Family: System

Area of application

The block is used for the following applications:

- Increasing the internal previous values in the MeanTime block.

Therefore this block will not be described in detail.

19.6 NegInt64 - Negation of an Int64 variable

19.6.1 Description of NegInt64

Object name (type + number) and family

Type + number: FC 362

Family: System

Area of application for NegInt64

The block is used for the following applications:

- Negation of an Int64-variable

This block is used by the PIDConR block for calculations that are twice as accurate (64-bit number format).

Therefore this block will not be described in detail.

19.7 NegR64 - Negation of a Real64 variable

19.7.1 Description of NegR64

Object name (type + number) and family

Type + number: FC 363

Family: System

Area of application for NegR64

The block is used for the following application:

- Negation of an Real64-variable

This block will not be described in detail.

19.8 PIDCoefR - Calculation of coefficients

19.8.1 Description of PIDCoefR

Object name (type + number) and family

Type + number: FC 366

Family: System

Area of application for PIDCoefR

The block is used for the following applications:

- Calculation of coefficients of discrete-time difference equations for PIDConR from the continuous-time input parameters (e.g. `TI`, `TD`)

This block is used by the PIDConR block for calculations that are twice as accurate (64-bit number format).

Therefore this block will not be described in detail.

19.9 R64ToReal - Converting Real64 to REAL

19.9.1 Description of R64ToReal

Object name (type + number) and family

Type + number: FC 367

Family: System

Area of application for R64ToReal

The block is used for the following applications:

- Converting from Real64 to REAL (32 Bit)

This block is used by the PIDConR block for calculations that are twice as accurate (64-bit number format).

Therefore this block will not be described in detail.

19.10 RealToR64 - Converting REAL to Real64

19.10.1 Description of RealToR64

Object name (type + number) and family

Type + number: FC 368

Family: System

Area of application for RealToR64

The block is used for the following applications:

- Converting from REAL (32 Bit) to Real64

This block is used by the PIDConR block for calculations that are twice as accurate (64-bit number format).

Therefore this block will not be described in detail.

19.11 SelST16 - Output of the best or worst signal status

19.11.1 Description of SelST16

Object name (type + number) and family

Type and number: FC 369

Family: System

Area of application for SelST16

The block is used for the following applications:

- Output the best or worst signal status

This block is used by technologic blocks which edit and output a signal status.

Therefore this block will not be described in detail.

19.12 ShLeInt64 - Left shift of an Int64 variable

19.12.1 Description of ShLeInt64

Object name (type + number) and family

Type + number: FC 370

Family: System

Area of application for ShLeInt64

The block is used for the following applications:

- Left shift of a (positive) Int64- variable

This block is used by the PIDConR block for calculations that are twice as accurate (64-bit number format).

Therefore this block will not be described in detail.

19.13 ShRiInt64 - Right shift of an Int64 variable

19.13.1 Description of ShRiInt64

Object name (type + number) and family

Type + number: FC 371

Family: System

Area of application for ShRiInt64

The block is used for the following applications:

- Right shift of a (positive) Int64- variable

This block is used by the PIDConR block for calculations that are twice as accurate (64-bit number format).

Therefore this block will not be described in detail.

19.14 PIDKernR - Calculation of the manipulated variable

19.14.1 Description of PIDKernR

Object name (type + number) and family

Type and number: FB 1877

Family: System

Area of application for PIDKernR

The block is used for the following applications:

- Calculation of the manipulated variable in PIDConR

This block is used by the PIDConR block for calculations that are twice as accurate (64-bit number format).

Therefore this block will not be described in detail.

Communication blocks 20

20.1 Snd_DigVal - Send Boolean values including quality code (Snd_DigVal)

20.1.1 Description of Snd_DigVal

Object name (type + number) and family

Type + number: FB 1891

Family: COMM

Area of application for Snd_DigVal

The "Snd_DigVal" block represents a simple user interface to SFB 12 "BSEND". The "Snd_DigVal" block sends 128 DigVal structures via an MPI, PROFIBUS or Industrial Ethernet connection to another CPU. This CPU calls the function block "Rcv_DigVal" (FB 1892) of the PCS 7 Advanced Process Library to receive the data.

How it works

Data transfer is initiated by calling the block with the value 1 at the `SendEn` control input. While data is being sent, `SendAct` is set to 1.

If the job is completed without errors, `SendDone` is set to 1. `SendErr` = 1 if an error occurs. A new job is then automatically triggered with the current data until all data have been successfully sent.

If input `SendEn` = 0, incomplete sending is canceled. No further sending takes place afterward (`SendAct` = 0).

The ID parameter is the connection number specified in the connection configuration. and is applied only at the first call after a cold restart.

The parameter R_ID is a random number. The R_ID parameter must be identical at the corresponding send and receive blocks. The R_ID parameter is applied only for the first call after a warm restart.

Note

Error code or message at the data block

When configuring a block pair for data transmission from an SND_AnaVal block to an RCV_AnaVal block, make sure that the settings for ID/R-ID are unique throughout the project.

Only one Rcv block may be assigned to each Snd block or only one connection via ID/R-ID may be configured.

Make sure that you perform the engineering clean and correctly because the data may otherwise be transferred to several blocks or an incorrect block.

Configuration

The block is installed in the CFC editor in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Time characteristics

The block does not have any time response.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

20.1.2 Snd_DigVal modes

This block has no operating mode.

20.1.3 Snd_DigVal functions

Configurable reactions using the `Feature` parameter.

An overview of all the reactions that are provided by the `Feature` parameter is available in the section Configurable functions using the Feature I/O (Page 130).

The following functionality is available for this block at the relevant bits:

- Bit 29: Output substitute values if raw value is invalid

20.1.4 Snd_DigVal error handling

Error handling of the block is restricted to the error information of the lower-level SFB 12 "BSEND".

The bits of information are set at the `SendErr(ERR)` and `SendStat(STAT)` parameters.

If an error occurs, a new job is automatically triggered with the current data until all data have been successfully transferred.

20.1.5 Snd_DigVal messages

No message behavior is applicable to this block.

20.1.6 Snd_DigVal I/Os

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`ID`	ID of physical connection	WORD	16#0000
`R_ID`	ID of message frame connection	DWORD	16#00000000
`SndEn`	1 = Activate continuous sending	BOOL	1
`PV_In0 ... PV_In127`	Input_0 ... Input_127	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80

Output parameters

Parameter	Description	Type	Default
`ENO`	1 = Block algorithm completed without errors	BOOL	0
`SndAct`	1 = Command is active	BOOL	0
`SndDone`	1 = Command is executed	BOOL	0
`SndErr`	1 = Command is executed with error	BOOL	0
`SndStat`	Type of error	WORD	16#0000

20.1.7 Snd_DigVal block diagram

A block diagram is not provided for this block.

20.2 Rcv_DigVal - Receive Boolean values including quality code (Rcv_DigVal)

20.2.1 Description of Rcv_DigVal

Object name (type + number) and family

Type + number: FB 1892

Family: COMM

Area of application for Rcv_DigVal

The "Rcv_DigVal I" block represents a simple user interface to SFB 13 "BRCV". The block receives 128 DigVal structures via MPI, PROFIBUS or Industrial Ethernet connection to another CPU. This CPU must call the function block "Snd_DigVal" (FB 1891) of the PCS 7 Advanced Process Library to send the data.

In STEP 7, a uniform connection must be configured for both communication partners and transferred to the AS.

How it works

Data are entered in the data block asynchronously to user-program execution. After "`Rcv_DigVal`" has been called, instance DB data may not be processed as long as the job is in progress (`RcvNewData` = 0). If the job is completed without errors, the `RcvNewData` output is set to 1 for one cycle. In the next cycle, the receive enable is automatically sent by the FB to the operating system of the CPU.

The receive enable can take take effect prior to the arrival of the first receive job. In this case, the receive enable is stored by the operating system.

The `ID` parameter is the connection number specified in the connection configuration. and is applied only at the first call after a cold restart.

The parameter `R_ID` is a random number. The `R_ID` parameter must be identical at the corresponding send and receive blocks. The `R_ID` parameter is applied only for the first call after a warm restart.

Call the "Rcv_DigVal" block for the ID/R_ID pair in each program cycle (cyclically or also via timeout alarm). Each message frame requires two calls of "Rcv_DigVal".

Note

Error code or message at the data block

When configuring a block pair for data transmission from an SND_AnaVal block to an RCV_AnaVal block, make sure that the settings for ID/R-ID are unique throughout the project.

Only one Rcv block may be assigned to each Snd block or only one connection via ID/R-ID may be configured.

Make sure that you perform the engineering clean and correctly because the data may otherwise be transferred to several blocks or an incorrect block.

The outputs `RcvErr` (Error) and `RcvStat` (Status) show the specific error information that correspond to SFB 13 "BRCV".

If an error occurs, use an appropriate substitute strategy with `Feature` (bit 29/30).

If you want to simulate the block, enable simulation via the `SimOn` input. In this case, the values from the inputs `SimPVX` are written to the outputs `PV_Outx`. The data sent are not applied, since data reception is switched off.

Configuration

The block is installed in the CFC editor in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Time characteristics

The block does not have any time response.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

20.2.2 Rcv_DigVal modes

This block has no operating mode.

20.2.3 Rcv_DigVal functions

Configurable reactions using the `Feature` parameter

An overview of all the reactions that are provided by the `Feature` parameter is available in the section Configurable functions using the Feature I/O (Page 130).

The following functionality is available for this block at the relevant bits:

- Bit 29: Output substitute values if raw value is invalid

20.2.4 Rcv_DigVal error handling

Error handling of the "Rcv_DigVal" block is restricted to the error information of the lower-level SFB 13 "BRCV".

You can find additional information in the manual "*System Software for S7-300/400 - System and Standard Functions*".

In the manual "*System Software for S7-300/400 - System and Standard Functions*", you can also find a description of the `RcvErr` and `RcvStat` outputs

The substitute value is output when bit 29 is set in the `Feature` structure and no new data is received after the expiration of "RcvMonCyc" (number of cycles). While the "REC_MON" cycles are running, the last received values are applied to the output.

If Bit29 is not set in the `Feature` structure and an error occurs, the last valid values are always applied to the output.

20.2.5 Rcv_DigVal messages

No message behavior is applicable to this block.

20.2.6 Rcv_DigVal I/Os

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`ID`	ID of physical connection	WORD	16#0000
`R_ID`	ID of message frame connection	DWORD	16#00000000
`RcvMonCyc`	Number of cycles for receiving monitoring error	INT	3

Parameter	Description	Type	Default
SimOn	1= simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimPV0 ... SimPV127	Simulation value	BOOL	0
SubsPV1 ... SubsPV127	Substitute value	BOOL	0
Feature	I/O for additional functions (Page 2284) Reserved 1 = substitute value Reserved	STRUCT • Bit0 ... Bit28: BOOL • Bit29: BOOL • Bit30 ... Bit31: BOOL	- • 0 • 0 • 0

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
PV_Out0 ... PVOut127	Output_00 ... Output_07	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
SimAct	1= simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RcvMonErr	1 = no data received	BOOL	0
RcvNewData	1 = no data received	BOOL	0
RcvErr	1 = Command is executed with error	BOOl	0
RcvStat	Type of error	WORD	16#0000

20.2.7 Rcv_DigVal block diagram

A block diagram is not provided for this block.

20.3 Snd_AnaVal - Send analog values including quality code (SND_AnaVal)

20.3.1 Description of Snd_AnaVal

Object name (type + number) and family

Type + number: FB 1893

Family: COMM

Area of application for Snd_AnaVal

The "Snd_AnaVal" block represents a simple user interface to SFB 12 "BSEND". The "Snd_AnaVal" block sends 32 analog structures via an MPI, PROFIBUS or Industrial Ethernet connection to another CPU. The other CPU calls the function module "Rcv_AnaVal" (FB 1894) of the PCS 7 Advanced Process Library to receive the data.

How it works

Data transfer is initiated by calling the block with the value 1 at the `SendEn` control input. The data are sent according to the settings `CycMin`, `CycMax` or `PV_In_HysXX`. While data is being sent, `SendAct` is set to 1. If the job is completed without errors, `SendDone` is set to 1. `SendErr` = 1 if an error occurs. A new job is then automatically sent with the current data until all data have been successfully transferred.

The ID parameter is the connection number specified in the connection configuration. and is applied only at the first call after a cold restart.

The parameter R_ID is a random number. The R_ID parameter must be identical at the corresponding send and receive blocks. The R_ID parameter is applied only for the first call after a warm restart.

Note

Error code or message at the data block

When configuring a block pair for data transmission from an SND_AnaVal block to an RCV_AnaVal block, make sure that the settings for ID/R-ID are unique throughout the project.

Only one Rcv block may be assigned to each Snd block or only one connection via ID/R-ID may be configured.

Make sure that you perform the engineering clean and correctly because the data may otherwise be transferred to several blocks or an incorrect block.

If input `SendEn` = 0, incomplete sending is canceled. No further sending takes place afterward; `SendAct` = 0.

You can use `CycMIN` to set the number of cycles to wait before triggering the transfer of the current input data, regardless of any changes to one or more values.

You can use CycMAX to specify how many cycles after the last valid data transfer to wait before sending the current input data. Sending is executed even when no value has changed or when the changes of a REAL value are within the hysteresis PV_In_HysXX.

The default hysteresis value is PV_In_HysXX = 0. If sending is not performed after each change, then configure the PV_In_HysXX input accordingly. This involves an absolute value.

When CycMax is reached, transfer is forced after the corresponding number of cycles.

Configuration

The block is installed in the CFC editor in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Time characteristics

The block does not have any time response.

Status word allocation for Status parameter

This block does not have the Status parameter.

20.3.2 Snd_AnaVal modes

This block has no operating mode.

20.3.3 Snd_AnaVal functions

The block has no functions.

20.3.4 Snd_AnaVal error handling

Error handling of the block is restricted to the error information of the lower-level SFB 12 "BSEND".

The information is set at the outputs SendErr(ERR) and SendStat(STAT).

If an error occurs, a new job is automatically triggered with the current data until all data have been successfully transferred.

20.3.5 Snd_AnaVal messages

No message behavior is applicable to this block.

20.3.6 Snd_AnaVal I/Os

Input parameters

Parameter	Description	Type	Default
EN	1 = Called block will be processed	BOOL	1
ID	ID of physical connection	WORD	16#0000
R_ID	ID of message frame connection	DWORD	16#00000000
SndEn	1 = Activate continuous sending	BOOL	1
CylMin	Minimum waiting cycle	INT	1
CylMax	Maximum waiting cycle	INT	10
PV_In_Hys00 ... PV_In_Hys31	EDC-Hysteresis PV_InR_00 ... EDC-Hysteresis PV_InR_31	REAL	0.0
PV_In0 ... PV_In31	Input_0 ... Input_31	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Output parameters

Parameter	Description	Type	Default
ENO	1 = Block algorithm completed without errors	BOOL	0
SndAct	1 = Command is active	BOOL	0
SndDone	1 = Command is executed	BOOL	0
SndErr	1 = Command is executed with error	BOOL	0
SndStat	Type of error	WORD	16#00000

20.3.7 Snd_AnaVal block diagram

A block diagram is not provided for this block.

20.4 Rcv_AnaVal - Receive analog values including quality code (Rcv_AnaVal)

20.4.1 Description of Rcv_AnaVal

Object name (type + number) and family

Type + number: FB 1894

Family: COMM

Area of application for Rcv_AnaVal

The "Rcv_AnaVal" block represents a simple user interface to SFB 12 "BRCV". The block sends 32 analog structures via MPI, PROFIBUS or Industrial Ethernet connection to another CPU. The other CPU calls the function module "Snd_AnaVal" (FB 1893) of the PCS 7 Advanced Process Library to receive the data.

How it works

Data are entered in the data block asynchronously to user-program execution. After "Rcv_AnaVal" has been called, instance DB data may not be processed as long as the job is in progress (`RcvNewData` = 0). If the job is completed without errors, the `RcvNewData` output is set to 1 for one cycle. In the next cycle, the receive enable is automatically sent by the FB to the operating system of the CPU.

The receive enable can take take effect prior to the arrival of the first receive job. In this case, the receive enable is stored by the operating system.

The `ID` parameter is the connection number specified in the connection configuration. and is applied only at the first call after a cold restart.

The parameter `R_ID` is a random number. The `R_ID` parameter must be identical at the corresponding send and receive blocks. The `R_ID` parameter is applied only for the first call after a warm restart.

Call the "Rcv_AnaVal" block for the `ID/R_ID` pair in each program cycle (cyclically or also via timeout alarm). Each message frame requires two calls of "Rcv_AnaVal".

Note

Error code or message at the data block

When configuring a block pair for data transmission from an SND_AnaVal block to an RCV_AnaVal block, make sure that the settings for ID/R-ID are unique throughout the project.

Only one Rcv block may be assigned to each Snd block or only one connection via ID/R-ID may be configured.

Make sure that you perform the engineering clean and correctly because the data may otherwise be transferred to several blocks or an incorrect block.

The outputs `RcvErr` (error) and `RcvStat` (status) show the specific error information that correspond to SFB 13 "BRCV".

If an error occurs, use an appropriate substitute strategy with `Feature` bit 29.

If you want to simulate the block, enable simulation via the `SimOn` input. In this case, the values from the inputs `SimPVX` are written to the outputs `PV_Outx`. The data sent are not applied, since data reception is switched off.

Configuration

The block is installed in the CFC editor in a cyclic interrupt OB (OB30 to OB38).

Startup characteristics

The block does not have any startup characteristics.

Time characteristics

The block does not have any time response.

Status word allocation for `Status` parameter

This block does not have the `Status` parameter.

20.4.2 Rcv_AnaVal modes

This block has no operating mode.

20.4.3 Rcv_AnaVal functions

Configurable reactions using the Feature parameter

An overview of all the reactions that are provided by the Feature parameter is available in the section Configurable functions using the Feature I/O (Page 130).

The following functionality is available for this block at the relevant bits:

- Bit 29: Output substitute values if raw value is invalid.

20.4.4 Rcv_AnaVal error handling

Error handling of the "Rcv_AnaVal" block is restricted to the error information of the lower-level SFB 13 "BRCV". You can find additional information in the manual System Software for S7-300/400 - System and Standard Functions. There you can find a description for the `RcvErr` and `RcvStat` outputs.

The substitute value is output when Bit29 is set in the `Feature` structure and no new data is received after the expiration of `RcvMonCyc` (number of cycles). While the `RcvMonCyc` cycles are running, the last received values are applied to the output.

If Bit29 is not set in the `Feature` structure and an error occurs, the last valid values are always applied to the output.

20.4.5 Rcv_AnaVal messages

No message behavior is applicable to this block.

20.4.6 Rcv_AnaVal I/Os

Input parameters

Parameter	Description	Type	Default
`EN`	1 = Called block will be processed	BOOL	1
`ID`	ID of physical connection	WORD	16#0000
`R_ID`	ID of message frame connection	DWORD	16#00000000
`RcvMonCyc`	Number of cycles for receiving monitoring error	INT	3
`SimOn`	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
`SimPV0...SimPV31`	Simulation value	REAL	0.0
`SubsPV0...SubsPV31`	Substitute value	REAL	0.0
`Feature`	I/O for additional functions (Page 2290) Reserved 1 = substitute value Reserved	STRUCT • Bit0 ... Bit28: BOOL • Bit29: BOOL • Bit30 ... 31: BOOL	- • 0 • 0 • 0

Output parameters

Parameter	Description	Type	Default
`ENO`	1 = Block algorithm completed without errors	BOOL	0
`PV_Out0 ... PV_Out31`	Output_0 ... Output_31	STRUCT • Value: REAL • ST: BYTE	- • 0.0 • 16#80

Parameter	Description	Type	Default
SimAct	1 = Simulation active	STRUCT • Value: BOOL • ST: BYTE	- • 0 • 16#80
RcvMonErr	1 = no data received	BOOL	0
RcvNewData	1 = new data received	BOOL	0
RcvErr	1 = Command is executed with error	BOOL	0
RcvStat	Type of error	WORD	16#0000

20.4.7 Rcv_AnaVal block diagram

A block diagram is not provided for this block.

Process tag types (insertible templates) 21

21.1 Introduction to process tag types

Introduction

You can find control engineering information on the standard process tag types of the Advanced Process Library in a separate section in this help:

- PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298)
- PIDConR with safety logic and control loop monitoring (PIDConR_ConPerMon) (Page 2299)
- Step controller with direct access to the actuator and without position feedback (StepControlDirect) (Page 2307)
- Step controller with assigned actuator block and position feedback (StepControlActor) (Page 2308)
- Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl) (Page 2309)
- Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312)
- Ratio control with PIDConR (RatioR) (Page 2315)
- Cascade control with control loop monitoring through ConPerMon (CascadeControl) (Page 2316)
- Cascade control with PIDConR (CascadeR) (Page 2319)
- Cascade control with PIDStepL (CascadeStepControl) (Page 2320)

If you want to do without control loop monitoring in order to reduce license costs or CPU resources, you can use accordingly simplified process tag types, which are identifiable by the suffix "Lean" in their names (for example, PIDControlLean, CascadeControlLean, RatioControlLean) or you can delete the ConPerMon block from the CFC charts for any of the process tag types.

In addition to the process tag types mentioned above, you will also find the following in the Templates folder of the library:

- PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303)
- PID controller for PA/FF devices (PIDControl_Lean_Fb) (Page 2297)
- PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301)
- Override control (Page 2322)
- Override control with PIDConR (OverrideR) (Page 2324)

- PID controller with Smith predictor (SmithPredictorControl) (Page 2306)
- Model-based predictive control (ModPreCon) (Page 2325)

This list contains prototype process tag types for several of the complex control loop structures shown in the example project (Page 2346). These prototypes are simply examples of how such process tag types might appear. In most process plants, it is assumed that high-level control loop structures must be configured individually some in combination with lower-level control loops. This means that mass production of high-level control loop structures as instances of process tag type represents the exception.

Descriptions for the following process tag types are also available:

- Monitoring eight digital process tags (Digital8Monitoring) (Page 2329)
- Monitoring of a digital process tag (DigitalMonitoring) (Page 2327)
- Monitoring a digital process tag for PA/FF devices (DigitalMonitoring_Fb) (Page 2328)
- Monitoring an analog process tag (AnalogMonitoring) (Page 2330)
- Monitoring of an analog process tag for PA/FF devices (AnalogMonitoring_Fb) (Page 2331)
- Dosing (Dose_Lean) (Page 2332)
- Dosing with PA/FF devices (Dose_Lean_Fb) (Page 2333)
- Two-speed motor (Motor2Speed) (Page 2336)
- Reversing motor (MotorReversible) (Page 2337)
- Reversible motor with controllable speed (MotorSpeedControlled) (Page 2338)
- Motor (Motor_Lean) (Page 2334)
- Motor with an additional analog value and time-stamped signals (Motor_AV_EventTs) (Page 2339)
- Motor valve (ValveMotor) (Page 2343)
- Valve (Valve_Lean) (Page 2341)
- Two-way valve (Valve2Way) (Page 2342)
- Control valve (VlvAnL) (Page 2344)
- Control valve for PA/FF devices (ValveAnalog_Fb) (Page 2345)

Using process tag types

When using process tag types in a PCS 7 multiproject, the following procedure is recommended:

- Copy all the required function blocks from the Advanced Process Library to the block folder of the master data library of the multiproject (<Projectname>_Lib).
- Then in the plant view, copy the required process tag types from the Advanced Process Library to the "Process_tag_types" folder of the master data library of the multiproject.
- Make any customizations of the process tag types.

Generating the process tags

Process tags are the instances of the process tag types.

There are two ways of generating process tags:

1. Copy the process tag type from the master data library and insert it in the target folder in the plant hierarchy. You can then set the parameters for and interconnect the CFC chart in the CFC Editor.
2. Using the Import-Export assistant, you can then generate the required process tags of the process tag type that you then assign parameters to and interconnect based on an import file.

Notes on the "Measured value failure" application scenario for controller block process tag types

If there is no valid measured value for `PV`, the controller must not remain in automatic mode because a closed control loop no longer exists. The controller is unable to calculate a useful manipulated variable if there is no feedback of the actual process state. The controller must then be changed to manual or tracking mode. OS operators must be made aware of this situation by a corresponding message. Various reactions to the loss of measured values are conceivable depending on the application background:

1. Tracking a fixed, configured neutral position manipulated variable, for example, Valve closed, Heater off, or similar.
2. Holding the last valid manipulated variable `MV` constant to retain a steady process state as far as possible (if the process was already in such a state).
3. Change to manual mode, so that the OS operator can take over responsibility for process control.

A combination of reactions 2 and 3 is implemented in the template. The controller changes to tracking with the last valid manipulated variable as start value. The switchover is made via the input `MV_TrkOn` of the PID block. Since manual mode already has priority over tracking, the OS operator can subsequently use the command.

Caution: It is not advisable to freeze the last valid manipulated value if the signals are subject to heavy noise, or in control loops with frequent setpoint changes. In such situations, it is advisable to change to variant 1.

If a controller intervenes locally in the process via an actuator (for example, a valve with electropneumatic positioner Sipart PS) specific measures similar to those for a cascade controller must be taken:

- If there is local intervention, the controller tracks the current actuator position. In this case, the feedback signal is applied to input `MV_Trk` and an OR element is set before input `MV_TrkOn`.

Notes on analog position feedback for controller block process tag types

If there is no position feedback, delete the associated analog input channel block at the `MV_Rbk` parameter in the CFC chart. This will also make the corresponding display elements in the standard view of the faceplate invisible.

21.2 PID controller (PIDControl_Lean)

PID controller

This process tag type forms the basis for generating PID control instances for continuous processes. In addition to the actual PID block, it contains functionality that should be implemented consistently in each control loop. This functionality is, however, not implemented within the PID block itself so that users can adapt the functions to a specific project:

- An analog input channel block for the actual value `PV` plus an analog output channel block for the manipulated variable `MV`.
- Certain logic blocks that change the control loop to a safe mode if the measurement of the process value fails. This state is signaled by the signal status of `PV`.

Note

Read the information for process tag types with controller blocks in Introduction to process tag types (Page 2293).

21.3 PID controller for PA/FF devices (PIDControl_Lean_Fb)

PID controller

This process tag type forms the basis for generating PID control instances for continuous processes. In addition to the actual PID block, it contains functionality that should be implemented consistently in each control loop. This functionality is, however, not implemented within the PID block itself so that users can adapt the functions to a specific project:

- An analog input channel block for the actual value `PV` plus an analog output channel block for the manipulated variable `MV`.
- Certain logic blocks that change the control loop to a safe mode if the measurement of the process value fails. This state is signaled by the signal status of `PV`.

Note

Read the information for process tag types with controller blocks in Introduction to process tag types (Page 2293).

21.4 PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon)

PID controller with safety logic and control loop monitoring

This process tag type forms the basis for generating PID control instances for continuous processes. In addition to the actual PID block, it contains functionality that should be implemented consistently in each control loop. This functionality is, however, not implemented within the PID block itself so that users can adapt the functions to a specific project:

- An analog input channel block for the actual value `PV` and (if available) the position feedback (`Rbk`), plus an analog output channel block for the manipulated variable `M`V.
- A simple process simulation of the first order, controlled by the manipulated variable `MV` which provides simulation values to analog input `PV`. This functionality allows at least elementary function tests control loop before a real process is connected.
- Certain logic blocks that change the control loop to a safe mode if the measurement of the process value fails. This state is signaled by the signal status of `PV`.
- An additional function block for control loop monitoring that should be installed in all PID control loop.

Note

Read also the information on controller block process tag types in Introduction to process tag types (Page 2293).
You can find information on the use of control loop monitoring in ConPerMon functions (Page 557).

21.5 PIDConR with safety logic and control loop monitoring (PIDConR_ConPerMon)

PIDConR with safety logic and control loop monitoring

This process tag type corresponds to the process tag type PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298) and control loop monitoring when PIDConR is used rather than PIDConL.

21.6 PID controller with safety logic and manipulated variable ramp (PIDConR_MV_Ramp)

PID controller with safety logic and manipulated variable ramp

This process tag type is derived from the process tag type PIDConR with safety logic and control loop monitoring (PIDConR_ConPerMon) (Page 2299). But in this case, a manipulated variable ramp is realized instead of a control loop monitoring.

The degree of manipulated variable is limited outside of PIDConR with the block RateLim. The limited `MV` value is fed back to the input parameter `ExtReset` of the block PIDConR.

21.7 PID - control with operating-point-oriented parameter control (GainScheduling)

PID - control with operating-point-oriented parameter control

Many technical processes have a non-linear response due to non-linear physical, chemical or thermodynamic effects. When such a process needs to be kept in the close vicinity of a fixed operating point, the transfer response can be linearized around this operating point. A linear PID controller can be designed for this linearized transfer function. If, however, the process has a strongly non-linear response and/or operates at different operating points, no constantly good control response can be expected throughout the entire operating range. Due to the non-linearity, various gain factors or process time constants are in effect at different operating points. In keeping with this, different controller parameters will be considered to be optimum.

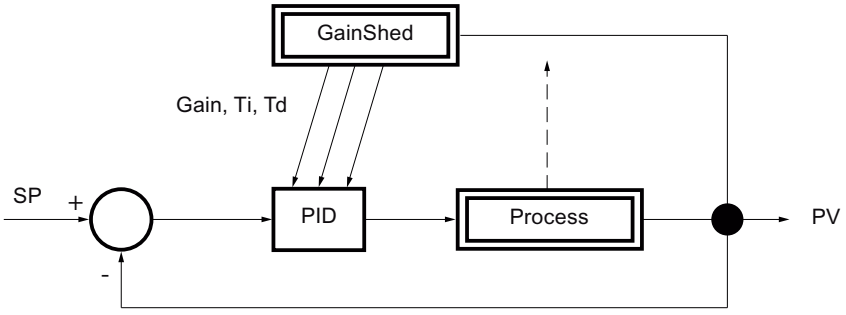


Figure 21-1 Operating-point-oriented parameter adaptation

One possible (the simplest) solution to this problem is known as gain scheduling or parameter scheduling. Using a tool such as the PCS 7 PID Tuner, various experiments are performed at different operating points, in each case with low signal amplitudes. This results in different PID parameter sets for each operating point. Up to three such parameter sets can be stored in the GainSched Function block. The suitable parameter set is selected depending on a continuously measurable variable that describes the state of the process, typically the control variable PV itself. Between the operating points for which there are exact parameter values, the values are calculated by linear interpolation of the neighboring interpolation points so that soft and bumpless transitions are possible between the operating points. The term "parameter scheduling" makes it clear that the "timetable" for adjusting the parameters is specified in advance. In contrast, an adaptive controller adapts itself automatically to the differing process response during operation.

The function block GainSched is produced from the CFC chart "fbGainSched" by compiling it as a block type. This CFC chart is supplied with the library so that the user has the option of

expanding the existing basic functionality as necessary, for example to more than three operating points.

Note

The combination of several locally optimized controllers by gain scheduling to form a non-linear controller does not necessarily represent an optimum non-linear controller for the non-linear process when considered from a mathematical point of view. This becomes clear even with benign non-linearities (that are continuous and can be differentiated) when setpoint step changes are made between different operating points. Great caution is needed with non-linearities that are discontinuous or cannot be differentiated or with non-monotonic non-linearities.

Examples of applications

- Control (especially temperature control) of batch processes, for example, batch reactors and batch columns
- pH value control
- Temperature control with phase transitions (for example, fluid/vapor form)
- Control of semi-batch plants (continuous plants with operating point changes, for example, polymerization reactors)
- Control in power plants with load changes

21.8 PID controller with dynamic feedforward control (FfwdDisturbCompensat)

PID controller with dynamic feedforward control

Feedforward can be used when a known, strong disturbance affects the process and its cause can be measured. In these cases, the following general strategy applies: "Control as much as possible (if known in advance and described by a model), control as much as necessary (the rest including the model error and immeasurable disturbances)".

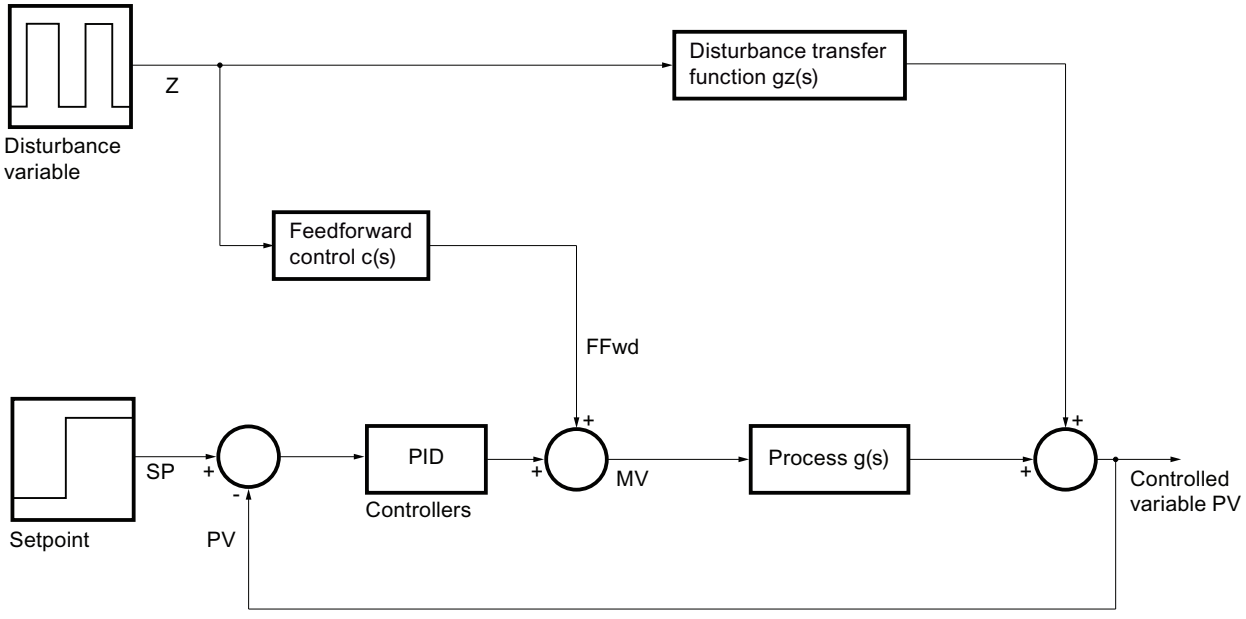


Figure 21-2 Dynamic feedforward control

The effect of a measurable disturbance can be estimated in the form of a transfer function $g_z(s) = y(s) / z(s)$ when the controller is running in manual mode so that no changes whatsoever to the controlled variable y = PV are caused by the manipulated variable and all changes can be attributed to the disturbance z(s).

The transfer function of an ideal feedforward control c(s) can be derived from the requirement that the effect of z on y should be zero for any disturbance signal z(s):

$g_z(s) \cdot z + c(s) \cdot g(s) \cdot z = (g_z(s) \cdot g(s)) \cdot z = 0$

To meet this equation, the compensation block must approximate the equation

$$c(s) = -\frac{g_z(s)}{g(s)}$$

as well as possible. For this to happen, the disturbance transfer function $g_z(s) = y(s) / z(s)$ must be known and the transfer function of the main controlled system $g(s) = y(s) / u(s)$, u = MV must be inverted. If both transfer functions can be modeled as first order with dead time

$$g(s) = \frac{k_s}{1 + t_1 s} \cdot e^{-s\theta}$$

and

$$g(s) = \frac{k_{sz}}{1 + t_{1z} s} \cdot e^{-s\theta_z}$$

and $\theta < \theta_z$ applies, the resulting compensation element must represent the transfer

$$c(s) = -\frac{k_{SZ}}{k_S} \cdot \frac{1 + t_1 s}{1 + t_{1z} s} \cdot e^{-s(\theta_Z - \theta)}$$

function.

In general, the following dynamic transfer function is required for additive feedforward control:

$$FFwd(s) = -k_c \frac{t_{cd} s + 1}{t_{cl} + 1} \cdot e^{-\theta_c s} \cdot z(s)$$

where:

$MV = MV_{PID} + FFwd$

In the example above, this function includes the following parameters:

$$k_c = \frac{k_{sz}}{k_s}, \quad t_{cd} = t_1, \quad t_{cl} = t_{1z}, \quad \theta_c = \theta_z - \theta'$$

This transfer function can be created outside the controller with a combination of elementary CFC blocks: one DT1 (Diff),, one PT1 (TimeLag) and one DeadTime block. As shown in the process tag type, the Diff block and TimeLag block are connected in parallel, and the DeadTime block as well as a multiplier are connected in series in front as a gain factor.

The input parameters k_c, t_{cd}, t_{c1} need to be configured by the user. For a static feedforward control, both time constants are set to zero.

Examples of applications

- Controlling the outlet temperature of a heat exchanger via steam pressure or heating/ cooling medium flow. Flow and inlet temperature of the medium are measurable disturbance variables.
- Fill level control in a drum steam generator using the inlet volume. The outflow is the measurable disturbance variable that is determined by the variable steam consumption in the plant.

- Temperature control in a distillation column using the reflux ration or or heating steam volume. The measurable disturbance variable is the mixture inlet.
- Temperature and concentration control in an agitated tank reactor using cooling medium flow and discharge volume. The temperature and possibly also the concentration of the inflow are measurable disturbance variables.

21.9 PID controller with Smith predictor (SmithPredictorControl)

PID control with Smith predictor

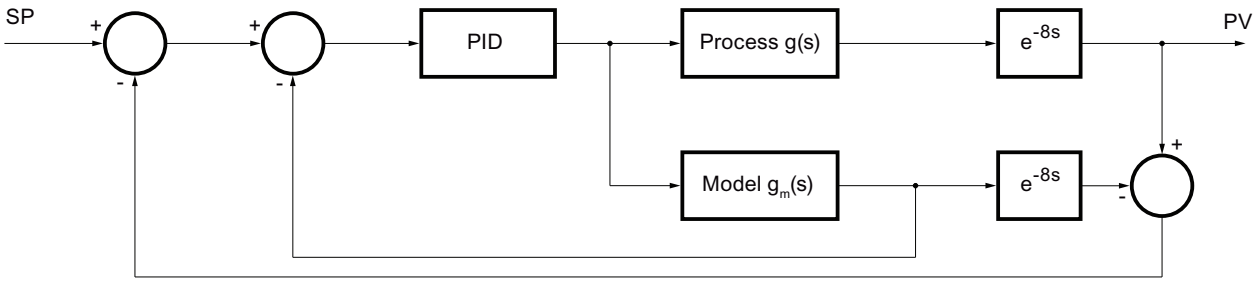


Figure 21-3 PID control with Smith predictor

In processes with large dead times (relative to the dominating time lag constant), a standard PI controller must be set very slowly and compromises must therefore be accepted in the control quality. The control quality can be significantly improved with a Smith predictor that can be derived from the IMC principle (**I**nternal **M**odel **C**ontrol) of model-based control. To achieve this, the transfer function $g_s(s) = g(s) \cdot e^{-s\theta}$ of the controlled system is split up into a part without dead time g(s) and a purely dead time slice $e^{-s\theta}$ with dead time θ. Only the controlled variable y affected by dead time can be measured in the real process. However, a virtual estimate of the controlled variable free of dead time can be taken from the process model (that will become part of the controller) and fed to the controller. This means that the controller itself can be designed for the process without a dead time slice and can therefore be set much more tightly. To compensate unknown disturbances, an estimate of the controlled variable affected by dead time is made in the model and compared with the genuine measured controlled variable. This difference is also fed back to the controller.

In terms of practical application, it must be pointed out that the performance of the Smith predictor depends largely on the model fit, in other words, the dead time must be known. The dead time must be constant or its value must be permanently adapted.

Note

To control processes with large dead times, a model predictive controller (see Description of ModPreCon (Page 676)) is also suitable in a single variable situation. It provides greater flexibility in the system modeling and is more convenient thanks to the integrated design procedures, it does however require more CPU resources.

21.10 Step controller with direct access to the actuator and without position feedback (StepControlDirect)

Step controller with direct access to the actuator and without position feedback

The output of the PIDStepL block is directly connected to the process via two digital output channel blocks. This is the simplest form of a step control. The operator can control the actuator (push-button open/close) in the faceplate of the controller. This push-button operation is preferable when operating simple actuators without position feedback since it allows bumpless switchover to auto mode. This switchover is not possible if actuators without position feedback are operated manually from any location outside the control block. For this reason, the template is designed with a PIDStepL block without position feedback.

21.11 Step controller with assigned actuator block and position feedback (StepControlActor)

Step controller with assigned actuator block and position feedback

The output of the PIDStepL block is directly interconnected with the process by way of an actuator block (for example, a motor or a valve). This additional effort is justified when special automation functions of the actuator block, for example motor current monitoring, motor protection and operation in local mode are required. The option of such combinations allows users to do without special functions for operation in local mode or actuator monitoring functions within the controller block.

If the actuator block is not in external default setpoint, or there is an interlock, motor protection or monitoring error, the actuator is incapable of accepting and executing controller commands.

In this context, the resulting structure must be interpreted as a "cascade circuit" consisting of the (primary) controller and actuator block (secondary controller) and measures similar to those for cascade control must be taken. For this purpose, the `CascaCut` output of the actuator block must be interconnected to the `TrkOn` input of the primary controller.

To ensure the bumpless switchover from local or manual mode back to cascade mode, the current actuator position must be fed back to the tracking input of the primary controller. This is, however, only possible when using actuators with position feedback. For this reason, the template is designed for operation with a PIDStepL block with position feedback.

Note

The Valve motor valve is configured for operation without motor feedback: Feature bit 12 = 1

21.12 Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl)

Split-range controller with control loop monitoring (ConPerMon)

A PID closed-loop controller can use a Split-Range block downstream of the controller output to distribute its manipulated variable to several actuators that influence the same control variable based on different physical principles and in different directions. A typical example of such an application is a temperature control, with heating via a live steam valve and cooling via a cooling water valve. The controller can request heating or cooling energy, depending on the sign of the control deviation (or more precisely of the manipulated variable). In other words, it can work with a bidirectional `MV` output (for example: -100% < `MV` < 100%), although each actuator only supports unipolar operation (for example: 0% < valve position < 100%).

The Split-Range function block contains two separate (static) characteristics for both actuators. Any significant difference between the two actuators in terms of performance (can be interpreted as different steady state gains for heating and cooling), can be compensated by setting different gradients for the characteristics, so that as far as possible, the controller is presented with a linear process response (independent of the sign).

Example: The cooling effect is not as strong as the heating effect.

If cooling is half as effective as heating, the split-range characteristic for cooling should have twice the gradient.

Note

The effective maximum values of the manipulated variables are obtained from the manipulated variable limits of the controller multiplied by the gradients of the split-range characteristics.

The cooling valve cannot go beyond fully open, in other words, the effective limitation for opening the valve is 100%. If a split range characteristic with gradient 2 is used for cooling, the low limit for the manipulated variable must be set to MV_LoLim = -50% on the controller.

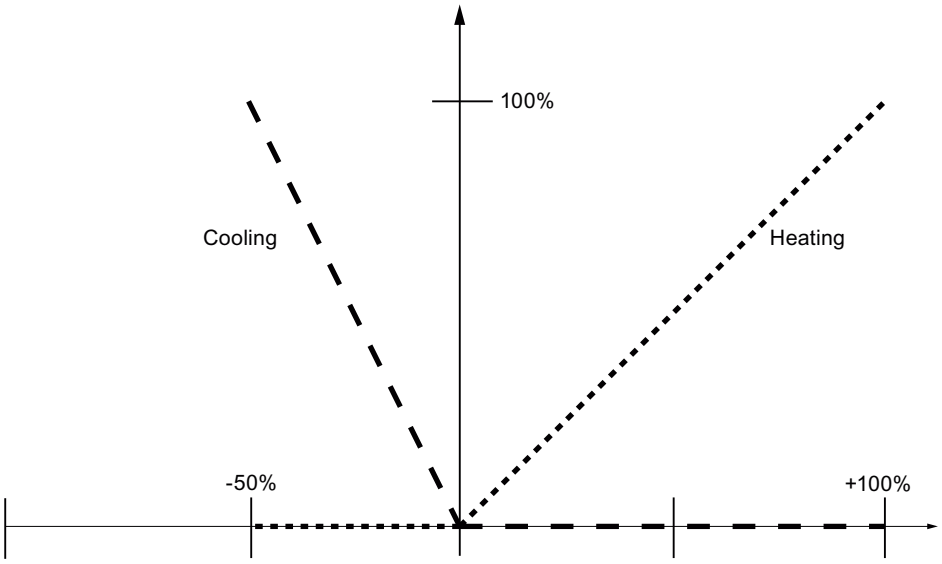


Figure 21-4 Split-range function of the example

The actual split-range function is supplied as separate SplitRange function block that is described in detail in the online help of the block.

Examples of applications:

- Control of the temperature of chemical reactors by means of steam heating valves and water cooling valves.
- Temperature control of glass furnaces or sprue channels by means of gas burners and cooling fans
- Temperature control of extruders by means of electrical heating and cooling fans
- Pressure control in a gas phase reactor by means of inlet and outlet valves
- Pressure control at a steam collecting track that supplies steam to several units by means of inlet valves from several steam generators, or the heating power of several steam generators.

21.13 Split-Range control (SplitRangeControl_Lean)

Split-range control

This process tag type is identical to the following process tag type:

Split-range controller with control loop monitoring through ConPerMon (SplitrangeControl) (Page 2309)

The ConPerMon block is not included, however.

21.14 Ratio control with control loop monitoring through ConPerMon (RatioControl)

Ratio control

Specific mixtures of several liquids or gases can be produced by means of a ratio control system consisting of several flow controllers and a Ratio block. The flow setpoint of an additive is obtained from one of these values:

1. From the current `PV` value of the main flow.
   This is the preferred alternative if the primary flow controller operates with steady-state deviation.
   or
2. From the setpoint `SP` of the main flow.
   This alternative provides a smooth, noise-free setpoint signal to the second controller, and allows for a more precise control of specified ratios in transition states where both flow control loops have approximately the same dynamic response.

It is generally advisable to use the second "setpoint-oriented" alternative for main flow controllers with I action.

Interconnect the selected reference value (actual value or setpoint of the main flow) to the In input parameter of the Ratio block.

The ratio control can be expanded by adding further components, in other words, setpoints 3 to n can be derived from `SP1` (or `PV1`) with the help of additional Ratio blocks.

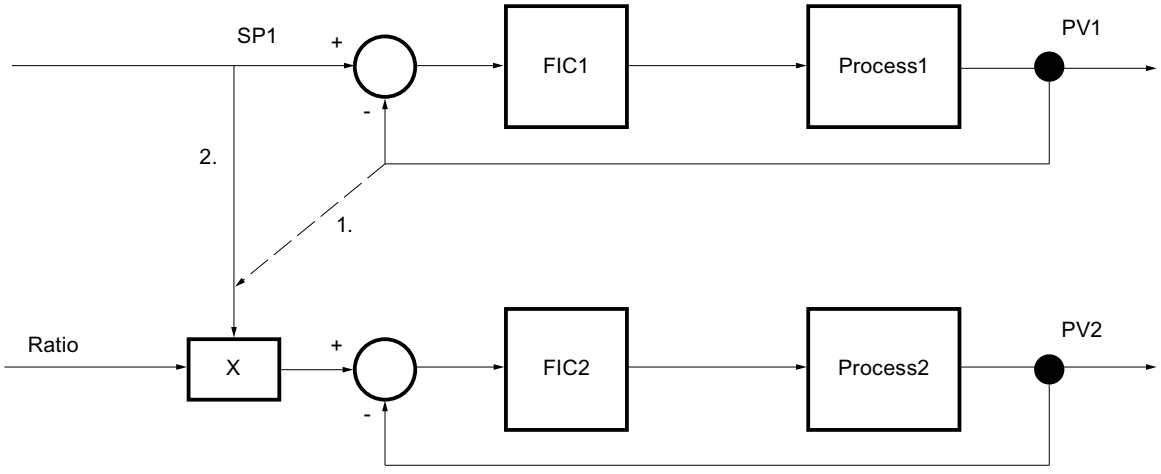


Figure 21-5 Ratio control, process value-oriented (1.) and setpoint-oriented (2.)

The main task of the Ratio block is to define the external setpoint of the secondary (added) component in accordance with

$$Out = In \cdot Ratio + Offset$$

. The actual ratio of the two flow actual values is also calculated in accordance with

$$RatioPV = \frac{SecComPV - Offset}{InPV}$$

and displayed on the control panel for checking purposes. In addition, interconnect the actual value of the main flow to the input parameter `InPV` of the Ratio block and the actual value of the flow of the secondary component to the input parameter `SecComPV`.

In both cases 1 and 2, the current value of the ratio is not controlled directly in the closed loop (no feedback-control) but rather uses feedforward-control.

Note

If the PID_Componet1 secondary controller is switched from automatic to another operating mode, the `CascaCut` output parameter is set and the Ratio block is switched to external. In this case, the external setpoint is corrected to the actual value ratio for the ratio of the control actual value to the component actual value: RatioExt = RatioPV. This results in a soft transition to automatic mode.

An additional function block for control loop monitoring must be installed, because it should always be installed in every PID control loop.

21.15 Ratio control (RatioControl_Lean)

Ratio control

This process tag type is identical to the following process tag type:

Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312)

The ConPerMon block is not included, however.

21.16 Ratio control with PIDConR (RatioR)

Ratio control with PIDConR

This process tag type corresponds to the process tag type Ratio control with control loop monitoring through ConPerMon (RatioControl) (Page 2312) when PIDConR is used rather than PIDConL.

You need to set feature bit 21 Switching operator controls for external setpoint to visible (Page 143) to 1 with a flow controller for the secondary component.

21.17 Cascade control with control loop monitoring through ConPerMon (CascadeControl)

General information on cascade control

A cascade control involves two or more PID controllers connected in series. The manipulated variable of the primary controller is interconnected to the external setpoint of the secondary controller so that both control loops are nested. The advantage of cascade control is that disturbances affecting the inner loop can be compensated much more quickly in the secondary loop than in the slower primary loop. In some situations, non-linear effects of the actuator can be compensated in the secondary loop so that a linear process response can be created for the primary loop. Cascade control is possible only when there are further measurable variables in the process in addition to the main control variable and when the internal control loop is significantly faster than the external loop.

Aspects to be clarified for the cascade control

With any cascade control, the following aspects must be carefully considered and clarified:

- The actuation range of the primary controller must match the setpoint range of the secondary controller to ensure proper operation of the anti-Windup functions of the primary controller.
- If the secondary controller is not operated in "cascade" mode (automatic mode with external setpoint) but in any other mode (for example manual or automatic mode with local setpoint) and therefore does not respond to commands of the primary controller, the primary controller must be put into "tracking" mode to prevent integration of the I action in the primary controller. The manipulated value of the primary controller tracks the process value or setpoint of the secondary controller to allow a bumpless return to cascade mode. The difference between tracking the setpoint and tracking the process value becomes apparent when the secondary controller is put into manual mode. If the process value is tracked, the response is similar to the "Track setpoint to process value in manual mode" of a simple controller.
- If the secondary controller reaches a (high/low) manipulated variable limit, the integrator of the primary controller should be blocked to prevent it going any further in this direction (up / down). The secondary controller cannot go any further in this direction anyway. This prevents any windup of the primary controller when the real actuator has already reached its physical limits while the primary controller has not yet reached its manipulated variable limits.
  Caution:
  - Swap the two bits of this interconnection if the secondary controller has a negative gain.
  - If the secondary controller then reaches a high or low limit, the integrator of the primary controller must be prevented from further integration in this direction.

Procedure

When you set up and commission the controller, you work from the "inside to the outside", in other words, you start by making the settings for the secondary controller and putting it into auto mode. You then set the primary controller parameters and change the secondary controller to cascade mode. When you set the parameters for the primary controller, remember that from its perspective the entire, inner closed control loop is the "controlled system". The parameters you set for the primary controller depend to a greater or lesser extent on the settings you made for the secondary controller. This becomes less important the greater the difference in dynamic response between the primary and secondary control loop.

Priorities

The tracking by the primary controller that is initiated by the secondary controller has lower priority than the manual mode on the primary controller. In turn, manual mode has lower priority than forced mode on the master controller as can, for example, be requested by external logic during an emergency shutdown of the plant (controller input `MV_Forced`, unlimited, activated by `MV_ForOn` with highest priority). For this reason, the PIDConL block for cascade control has an additional tracking input `MV_Trk` that is activated by `MV_TrkOn` and is subject to the normal manipulated variable limiting and has lower priority than manual mode.

Additional application examples

- Temperature control of a distillation column (primary controller) based on the reflux ratio (secondary controller at the top of the column), and heating steam flow rate (secondary controller at the column sump),
- Temperature control of a furnace using a secondary controller to control the fuel flow rate,
- Fill level control for a container using a secondary controller for the inlet and/or outlet flow.
- Position control (in drive engineering) with a secondary controller for the speed and torque.

Using secondary flow controllers

Secondary controllers are usually used for flow control to prevent detrimental effects on the primary controller resulting from changes in flow rate. The non-linearities frequently often found in a flow control element (for example a valve) are "hidden" in the secondary loop because the secondary closed loop has a linear response and non-linearities have no effect on the primary controller or its tuning.

Note

Control loop monitoring is only practical for primary controllers, as explained in the description of the ConPerMon block in the section, Functions/Cascade Control.

21.18 Cascade control (CascadeControl_Lean)

Cascade control

This process tag type is identical to the following process tag type:

Cascade control with control loop monitoring through ConPerMon (CascadeControl) (Page 2316)

The ConPerMon block is not included, however.

21.19 Cascade control with PIDConR (CascadeR)

Cascade control with PIDConR

This process tag type corresponds to a large extent to the process tag type Cascade control with control loop monitoring through ConPerMon (CascadeControl) (Page 2316) when PIDConR is used rather than PIDConL.

Special note: The current `PV_Out` output of the PID_Slave secondary controller is used as an external Reset `ExtReset` on the primary controller. Unlike cascade control, the PID_Master primary controller does not therefore have to be changed to tracking when the cascade is disconnected with all other PID controllers. There is also no need for the direction-dependent blocking of the integrator for the primary controller.

Since the primary controller is reliant on the external Reset, any control deviations in the secondary closed loop interfere with the primary controller; secondary controllers without I action are not therefore recommended for PIDConR.

You need to set the Feature bit Switching operator controls for external setpoint to visible (Page 143) to 1 with a secondary controller of the cascade.

21.20 Cascade control with PIDStepL (CascadeStepControl)

Cascade control with a step controller as a valve positioner and a downstream actuator block

The step controller is used in this template as a positioner, which regulates the position of a valve that does not have an internal positioner and is only controlled via binary inputs (open, stop, close). The motorized control valve can thus include delay behavior, i.e. that does not stop immediately at a stop command. The output of the PIDStepL block is directly connected to the process via the actuator block VlvMotL (see StepControlActor template). The PIDStepL runs as a slave controller in a cascade structure and gets his external setpoint from a master controller (flow or pressure regulator).

The PIDStepL is configured as a step controller without position feedback (WithRbk =0). The position feedback signal is applied in this example to the `PV` input and controlled to the external setpoint, which is also set by higher-level master controller (PIDConL). Master and slave controllers are connected in such a way that the following conditions are met (see CascadeControl template):

- The calculated valve position of the master controller is forwarded to the external setpoint of the slave controller.
- The master controller is switched to tracking if there is a bad process value on the master controller or with an open loop by the PIDStepL slave controller or by the VlvMotL valve block. The tracking value corresponds to the valve position at the `PV` input of the PIDStepL step controller. In case of an open loop by the VlvMotL valve block, the open and close control signals of the PIDStepL slave controller are also tracked.
- If the valve position is fully open or fully closed, the I-component of the master controller is stopped (Windup measure).

The same rules as those set in the CascadeControl template and described in the Procedure section apply for the controller setting and commissioning.

21.21 Source chart for GainSched function block (gain scheduling)

Source chart for GainSched function block

In contrast to all other function blocks, the `GainSched` block is implemented as a CFC chart and is generated with the "Compile chart as block type" function. Several application options are available in this case:

- You can use the precompiled function block `GainSched` from the library if the standard functionality is adequate for your needs.
- If you require special additional functions for gain scheduling in your application (for example more than three operating points, additional logic functions for selecting the parameters), you will need to modify the CFC source chart and compile it as a block type with a different FB number.

Internally, the `GainSched` block consists essentially of three instances of the Polygon block, one for each of the three controller parameters, Gain, TI and TD.

The polygon block itself would extrapolate linearly outside of the boundary points and thereby exceed the value range of its interpolation points. However, as this presents too high a risk for controller parameters, the controller parameters which are output are effectively limited to the table values at the boundary operating points by the automatic introduction of additional interpolation points with a horizontal end tangent outside of the specified range.

21.22 Override control

Override control

In an override control, two or more controllers share a common actuator. Depending on the current process state, a decision is made as to which controller actually has access to the actuator, in other words, the various controllers can override each other.

A typical use case is a gas pipeline with pressure and flow control using a single valve. The main aim of the control is to achieve a certain flow rate, however due to safety considerations, the pressure must be kept within certain limits. The pressure controller is therefore known as the "limiting controller" or "secondary controller".

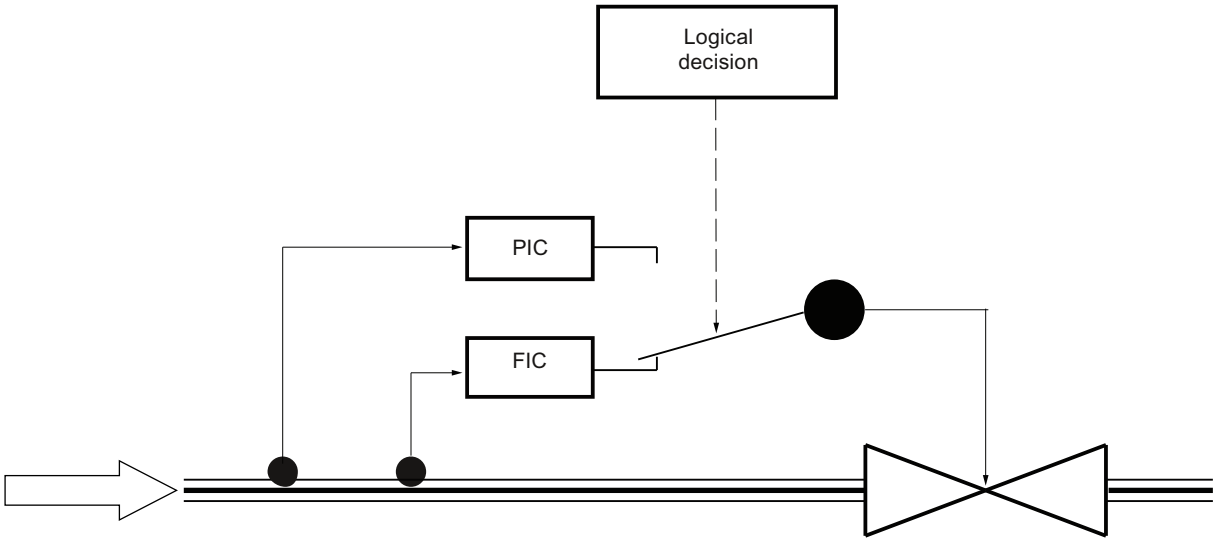


Figure 21-6 Override control with main controller FIC and limiting controller PIC

Criteria for the type of override control

The logical decision as to which controller should be active can be made based on two different criteria resulting in two different types of override controls:

1. The decision is based on a measurable process output variable, for example one of the two controlled variables. In the example above, the warning limits of the pressure controller can be used to decide whether the pressure controller should be active. The passive controller is in tracking mode to avoid Windup problems and to ensure bumpless transfer. The setpoint of the secondary controller must be somewhat lower than the switchover threshold so that the transfer can be reversed again. This type of override control is easy to understand and to implement. Its advantage is that the high and low limit of the secondary controlled variable (for example pressure) can be monitored; its disadvantage is that a limit cycle oscillation results as soon as the limiting controller needs to intervene. The secondary controller will always attempt to return its controlled variable to the safe range and to return command to the main controller (for example flow rate) so that the active and passive controllers swap over continuously. This variant is therefore only recommended when the secondary controller is seldom required and functions mainly as a safety or backup system.
2. The decision is based on a comparison of the manipulated variables of both controllers, for example the controller that demands the higher (or lower) controlled variable takes control of the actuator. In the example above, the controller that wants to open the valve further takes over control. The setpoint of the secondary controller defines the switching threshold. Both controllers run the entire time in automatic mode. To avoid Windup problems, the manipulated variable limits must be tracked in a crossover structure: When the higher (lower) manipulated variable wins, the low (high) limits of all controllers of the currently highest (lowest) manipulated variable must be corrected slightly up or down by, for example, 2% of the manipulated variable range. This means that this scheme can also be used in applications with more than two controlled variables. There is no Windup problem at the high limit because the highest manipulated variable takes over control anyway. This approach avoids the limit cycle oscillation of alternative 1 but is, in principle asymmetrical, in other words either a high or a low limit of the secondary controlled variable can be monitored but not both.
   This type of override control is described in most control textbooks, particularly in the USA. It can, however, only be used with PID algorithms that allow online manipulation of the manipulated variable limits (in PCS 7 as of V6.0).

Additional application examples

- **Steam generator:** The primary controlled variable is the steam pressure but the water level in the steam tank must be monitored so that the heating coils remain completely covered by water and the tank does not overflow. The only manipulated variable is the outlet valve.
- **Compressor:** The primary controlled variable is the throughput but the pressure must be monitored to make sure it does not exceed a safety limit. The only manipulated variable is the motor speed.
- **Steam distribution system:** Every plant involving industrial processes has a network of pipes to distribute steam at various pressures throughout the plant. The high pressure of the steam is reduced to lower levels via a valve. The primary controlled variable is the pressure at the lower-level stage, however the pressure in the high pressure piping must also be monitored to make sure that it does not exceed a safety limit.

21.23 Override control with PIDConR (OverrideR)

Override control with PIDConR

This process tag type corresponds for the most part to the process tag type Override control (Page 2322) when PIDConR is used rather than PIDConL.

Special note: The current manipulated variable output at the final controlling element (e.g. the maximum `MaxMV.Out` manipulated variable proposed by the main controller and limiting controller) is used as the external Reset`ExtReset` for both controllers. Unlike override control, the manipulated variable limits (e.g. `MV_LoLim`) of both controllers do not have to be tracked within a defined space (e.g. `MaxMV_Minus2.Out`) by the manipulated value output at the final controlling element with all other PID controllers.

21.24 Model-based predictive control (ModPreCon)

Model-based predictive control

The process tag type shows how users can expand the ModPreCon block by adding extra functions:

- External alarming with the MonAnL block
- Control quality monitoring with the ConPerMon block
- Safety logic for measure value loss.

You can find details about the ModPreCon block in Description of ModPreCon (Page 676).

Note on using the ConPerMon block in a system with multi-variable

The mathematical concept of the ConPerMon block is designed for single variable applications. If any increase of variance is detected in the channel of a multi-variable control, the ConPerMon algorithm is unable to determine whether this problem is caused by its own internal control channel or by interaction of neighboring channels. It may be helpful, however, to include a ConPerMon block for each control channel of a multi-variable system to monitor whether the control performance during operation remains within the range determined during commissioning. To achieve this, several logic operations must be performed before the `ManSuprCPI` input bit of each ConPerMon block:

- If one or several of the other channels of the multi-variable system are are not in a steady state ("root caused in this channel") due to local causes (for example a setpoint step change) that is indicated by output bit `CPI_SuRoot = 1`, the increase of variance in its own channel cannot be avoided and should not trigger a CPI warning in its own channel.
- If one or several other channels of the multi-variable system exhibit strong variations as indicated by output bit `CPI_WrnAct = 1`, the increase in variance in its own channel cannot be avoided and should not trigger a CPI warning in its own channel. This may be helpful to localize the actual cause of the problem. The channel in which excessive variations are first detected outputs the first alarm; the other channels which may only be affected by errors resulting from the first error do not generate their own alarms.

Examples of applications

- Quality control in distillation columns, for example control of the column top and sump temperature based on the reflux ratio and heating steam volume
- Temperature control of several adjacent zones of furnaces with several burners, for example, tunnel furnaces, glass smelting plants, glass sprue channels etc.
- Quality control in chemical reactors by adjusting of reaction conditions such as pressure, temperature, feed/drain etc.

- Vaporizers, for example, drum steam generators
- Mills, for example, cement mills, sieve mills: quality control (grain size) in combination with flow rate maximized, manipulated variables: sieve speed and mill feed

Note

The statistical evaluation of the service factor (time slice in automatic mode) and the time slices in the manipulated variable limits can be performed in a WinCC trend control, as in the customary case of a single variable. The binary variables from ModPreCon required for this are archived automatically. However, an appropriate trend control must be manually configured on the operator station, since the view archive in the ConPerMon block is not designed for such quantities.

21.25 Monitoring of a digital process tag (DigitalMonitoring)

Monitoring a digital process tag

This process tag type serves as a basis for monitoring a digital process tag using the MonDiL block.

The digital measurement signal is read from the I/O through the PCS7DiIn block.

The process tag type contains the required interconnections between Pcs7DiIn and MonDiL.

21.26 Monitoring a digital process tag for PA/FF devices (DigitalMonitoring_Fb)

Monitoring a digital process tag

This process tag type serves as a basis for monitoring a digital process tag using the MonDiL block.

The digital measurement signal is read from the I/O through the FbDiIn block.

The process tag type contains the required interconnections between FbDiIn and MonDiL.

21.27 Monitoring eight digital process tags (Digital8Monitoring)

Monitoring eight digital process tags

The process tag type serves as a basis for monitoring up to eight digital process tags using the MonDi08 block.

The digital measurement signals are read from the I/O through the PCS7DiIn block.

The process tag type contains the required interconnections between the eight Pcs7DiIn and MonDi08 blocks.

21.28 Monitoring an analog process tag (AnalogMonitoring)

Monitoring an analog process tag

This process tag type serves as a basis for monitoring an analog process tag using the MonAnL block.

The analog value is read from the I/O through the PCS7AnIn block.

The process tag type contains the required interconnections between Pcs7AnIn and MonAnL.

If you wish to monitor and forward the slope of the process value, you usually need to smooth the process value. You can insert a filter block such as Smooth between Pcs7AnIn and MonAnL for this purpose. The gradient of the process value can also be smoothed using the TimeLag parameter at MonAnL.

Procedure for smoothing the process value and its gradient:

1. Check the process value PV in the trend recorder and smooth it using a filter block such as Smooth only as much as is needed.

2. Check the gradient PV_Grad of the process value in the trend recorder and smooth the trend at the TimeLag parameter of MonAnL.

The sum of the time constants of Timelag from MonAnL and TimeConstant from Smooth approximately result in the length of a varying time window in which the gradient is averaged.

21.29 Monitoring of an analog process tag for PA/FF devices (AnalogMonitoring_Fb)

Monitoring an analog process tag

This process tag type serves as a basis for monitoring an analog process tag using the MonAnL block.

The analog value is read from the I/O through the FbAnIn block.

The process tag type contains the required interconnections between FbAnIn and MonAnL.

If you wish to monitor and forward the slope of the process value, you usually need to smooth the process value. You can insert a filter block such as Smooth between FbAnIn and MonAnL for this purpose. The gradient of the process value can also be smoothed using the TimeLag parameter at MonAnL.

Procedure for smoothing the process value and its gradient:

1. Check the process value PV in the trend recorder and smooth it using a filter block such as Smooth only as much as is needed.

2. Check the gradient PV_Grad of the process value in the trend recorder and smooth the trend at the TimeLag parameter of MonAnL.

The sum of the time constants of Timelag from MonAnL and TimeConstant from Smooth approximately result in the length of a varying time window in which the gradient is averaged.

21.30 Dosing (Dose_Lean)

Dosing

This process tag type serves as a basis for dosing either as "Single component dosing via flow measurement" or as "Fill/extraction weighing via dosing scale" using the DoseL block.

The analog measurement signal is read from the I/O through the PCS7AnIn block.

The interlock signals of DoseL are interconnected to the Intlk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via Pcs7DiIn.

The digital output signals are sent to the I/O through the PCS7DiOu blocks.

The process tag type contains the required interconnections between the blocks mentioned above.

21.31 Dosing with PA/FF devices (Dose_Lean_Fb)

Dosing

This process tag type serves as a basis for dosing either as "Single component dosing via flow measurement" or as "Fill/extraction weighing via dosing scale" using the DoseL block.

The analog measurement signal is read from the I/O through the FbAnIn block.

The interlock signals of DoseL are interconnected to the Intlk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via FbDiIn.

The digital output signals are sent to the I/O through the FbDiOu blocks.

The process tag type contains the required interconnections between the blocks mentioned above.

21.32 Motor (Motor_Lean)

Motor with current monitoring

This process tag type serves as a basis for controlling motors with one control signal using the MotL block.

The feedback signal of the motor is read from the I/O through the PCS7DiIn block.

The interlock signals of MotL are interconnected to the IntLk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via Pcs7DiIn.

The digital output signals are sent to the I/O through the PCS7DiOu blocks.

The process tag type contains the required interconnections between the blocks mentioned above.

21.33 Motor with PROFIdrive Drive Profile telegram 1 and 20 (Namur)

Motor with adjustable speed and two directions of rotations according to the speed control mode with standard telegram 1 and 20

The process tag type serves as a basis for control of motors according to the profile "Speed control mode with standard telegram 1 and 20 with and without Namur" with the help of blocks FbDrive and MotSpdCL.

The interlock signals of MotSpdCL are connected to the IntLk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via Pcs7DiIn.

The input/output signals of MotSpdCL are sent to the I/O through the FbDrive block.

The process tag type contains the required interconnections between the blocks mentioned above.

21.34 Two-speed motor (Motor2Speed)

Two-speed motor

This process tag type serves as a basis for controlling motors with two speeds using the MotSpdL block.

The speed feedback signals of the motor are read from the I/O through the PCS7DiIn blocks.

The interlock signals of MotSpdL are connected to the IntLk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via Pcs7DiIn.

The digital output signals are sent to the I/O through the PCS7DiOu blocks.

The process tag type contains the required interconnections between the blocks mentioned above.

21.35 Reversing motor (MotorReversible)

Reversing motor

This process tag type serves as a basis for controlling reversing motors using the MotRevL block.

The feedback signals of the motor are read from the I/O through the PCS7DiIn blocks.

The interlock signals of MotRevL are connected to the IntLk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via Pcs7DiIn.

The digital output signals are sent to the I/O through the PCS7DiOu blocks.

The process tag type contains the required interconnections between the blocks mentioned above.

21.36 Reversible motor with controllable speed (MotorSpeedControlled)

Reversible motor with controllable speed

This process tag type serves as a basis for controlling reversible motors with controllable speed using the MotSpdCL block.

The digital feedback signals of the motor are read from the I/O through the PCS7DiIn blocks.

The speed feedback of the motor is read from the I/O through the PCS7AnIn block.

The interlock signals of MotSpdCL are connected to the IntLk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via Pcs7DiIn.

The digital output signals are sent to the I/O through the PCS7DiOu blocks.

The speed control signal is output to the I/O through the PCS7AnOu block.

The process tag type contains the required interconnections between the blocks mentioned above.

21.37 Motor with an additional analog value and time-stamped signals (Motor_AV_EventTs)

Motor with an additional analog value and time-stamped signals

This process tag type shows the monitoring of an additional analog value, for example, for motor current monitoring (AV block) and additional binary signals (EventTs block) at a technologic block with MotL as an example.

By interconnecting the AV or EventTs block, messages of this block are displayed in the message view of the technologic block connected to it and can be acknowledged there.

Use of the AV block

The analog signal to be monitored can be read via the Pcs7AnIn block, for example. This has to be interconnected to the AV block. The `AV_Tech` output parameter of the AV block has to be interconnected to the `AV` input parameter of the technologic block.

Use of the EventTs block

A non-time-stamped binary signal from the I/O can be read via the Pcs7DiIn block, for example. The `PV_Out` output parameter of Pcs7DiIn has to be interconnected to the `Inx` input parameter of the EventTs block. The time stamp of this signal is generated by the EventTs block at signal change.

A time-stamped binary signal from the I/O must be read via the Pcs7DiIT block. The `TS_Out` output parameter of Pcs7DiIT has to be interconnected to the `InTSx` input parameter of the EventTs block.

The `EventTsOut` output parameter of the EventTs block has to be interconnected to the EventTsIn input parameter of the technologic block.

Additional interconnections

The interlock signals of MotL are interconnected to the IntLk02 interlock blocks. In turn, these interlock blocks are interconnected to other blocks, for example, to digital process tags via the Pcs7DiIn block.

The digital output signal is output to the I/O through the PCS7DiOu block.

The process tag type contains the required interconnections between the blocks mentioned above.

21.38 Motor according to the profile for low voltage switchgear devices with profile 1 of the MM_Starter

Reversible motor according to the manage motor starter with profile type 1

The process tag type serves as a basis for control of motors according to the profile "Manage motor starter - profile type 1" with the help of blocks FbSwtMMS and MotRevL.

The interlock signals of MotRevL are connected to the IntLk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via Pcs7DiIn.

The input/output signals of MotRevL are sent to the I/O through the FbSwrtMMS block.

The process tag type contains the required interconnections between the blocks mentioned above.

21.39 Valve (Valve_Lean)

Valve

This process tag type serves as a basis for controlling a valve with two positions (open/close) using the VlvL block.

The feedback signals of the valve are read from the I/O through the PCS7DiIn blocks.

The interlock signals of VlvL are connected to the IntLk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via Pcs7DiIn.

The digital output signals are sent to the I/O through the PCS7DiOu blocks.

The process tag type contains the required interconnections between the blocks mentioned above.

21.40 Two-way valve (Valve2Way)

Two-way valve

This serves as a basis for controlling a

- multi-way valves with up to three switching positions or
- three individual valves (valve network) to implement a 2-way valve circuit with neutral position

using the Vlv2WayL block.

The feedback signals of the valve or valves are read from the I/O through the PCS7DiIn blocks.

The interlock signals of Vlv2WayL are connected to the IntLk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via Pcs7DiIn.

The digital output signals are sent to the I/O through the PCS7DiOu blocks.

The process tag type contains the required interconnections between the blocks mentioned above.

21.41 Motor valve (ValveMotor)

Motor valve

This process tag type serves as a basis for controlling a motor valve using the VlvMotL block.

The feedback signals of the valve are read from the I/O through the PCS7DiIn blocks.

The interlock signals of VlvMotL are connected to the IntLk02 interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via Pcs7DiIn.

The digital output signals are sent to the I/O through the PCS7DiOu blocks.

The process tag type contains the required interconnections between the blocks mentioned above.

21.42 Control valve (VlvAnL)

Analog control valve

This process tag type serves as a basis for controlling an analog control valve (0 to 100%) using the `VlvAnL` block.

The feedback signals of the valve's total travelled positions (open/closed) are read by the I/Os through the `Pcs7DiIn` blocks.

The feedback signal of the current analog position is read through the `Pcs7AnIn` block.

The interlock signals of `VlvAnL` are interconnected to the `IntLk02` interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via `Pcs7DiIn`.

The analog actuating signal is output to the peripherals via the `Pcs7AnOu` blocks.

The process tag type contains the required interconnections between the blocks mentioned above.

21.43 Control valve for PA/FF devices (ValveAnalog_Fb)

Analog control valve

This process tag type serves as a basis for controlling an analog control valve (0 to 100%) using the `VlvAnL` block.

The feedback signals of the valve's total travelled positions (open/closed) and the current analog position of the value are read by the I/Os through the `FbAnOu` blocks.

The analog actuating signal is output to the peripherals via the `FbAnOu` blocks.

The process tag type contains the required interconnections between the blocks mentioned above.

The interlock signals of `VlvAnL` are interconnected to the `Intlk02` interlock blocks. These interlock blocks in turn are connected to other blocks, for example, to digital process tags via `FbDiIn`.

21.44 Example project APL_Example_xx

21.44.1 Introduction to the PCS 7 example project for Advanced Process Control

Introduction

This document relates to the PCS 7 example project, Process Control (APL_Example_xx, xx indicates the language variant) for the PCS7 Advanced Process Library.

In contrast to the process tag types (insertable templates), the example project is primarily intended for training purposes:

The main aim was to familiarize users with with the new advanced process control structures by allowing them to experiment without having to intervene in the real process. The examples provide realistic process simulation. Working with these examples helps you to understand the concept and specific structural requirements, and to assess the uses of these functions before you implement them in a real system. For this reason, the examples contain a third-order simulation model with gain, equivalence value, and measurement noise but no analog channel blocks. The process model is supplied as the "ProcSimC" CFC chart and incorporated in the examples using the chart-in-chart technique.

The following examples are provided:

- Cascade control of temperature by using the heat flow (CascadeSim) (Page 2349)
- Control loop monitoring for simulation with colored noise (ConPerMonSim) (Page 2351)
- Feedforward control to compensate a measurable disturbance variable (DisturbCompSim) (Page 2351)
- Operating point-oriented adaptation of parameters (gain scheduling) for non-linear processes (GainSchedSim) (Page 2352)
- Override control on a pipeline (OverrideSim) (Page 2353)
- Smith predictor for a dead time system (SmithPredictorSim) (Page 2353)
- Filtering of noisy measured values in a control loop (SigSmoothSim) (Page 2354)
- Predictive control of a 2x2 multi-variable controlled system (ModPreConSim) (Page 2355)
- Predictive control of a non-linear process (ModPreConNonLinSim) (Page 2355)

A template for Fuzzy Control using the *PCS 7 add-on product FuzzyControl++* can be downloaded from the Internet pages of *Siemens I&S* and is therefore not included in the demo project.

A separate tree folder ("Unit") is provided for each example in the PCS7 example project. Each folder contains a CFC chart with an interconnection example, a brief explanatory text, and an assigned OS picture with self-explanatory visualization of the process example based on a pre-configured trend recorder. A short text in the OS picture describes commissioning and presentation.

21.44.2 Process simulation including noise generator (ProcSimC; ProcSimS)

Process simulation including noise generator

Users require only a few standard blocks to create a dynamic process model that reflects the response pattern of many technological processes with adequate precision. This model is used in all example projects (APL_Example_xx). However, you can also use this model for sales presentations or to test closed-loop control functions, in other words in a project phase in which the real plant is not yet available ("virtual process", "shadow plant"). Process simulation is supplied as an open source CFC chart "ProcSimC" that users can install in other CFC charts as a nested chart (chart-in-chart). It contains three first order delay elements, a gain factor, an equivalence value `PV` (for `MV = 0`), and a noise generator for white measurement noise. An additive input is provided for (artificial) interference of the input. The Laplace transformation function described below is implemented by means of this model.

$$PV(s) = \frac{Gain}{(TimeLag1 \cdot s + 1)(TimeLag2 \cdot s + 1)(TimeLag3 \cdot s + 1)}(MV(s) + DisV(s)) + PV_0 + Noise$$

Use cases

Users can adapt this flexible model to suit the requirements of various use cases, for example:

- **Simulation of temperature control systems:** `PV0` represents the temperature without heating, for example, the ambient temperature. The value of `TimeLag1` is typically significantly higher than `TimeLag2` and `TimeLag3`. The value of the latter can also be zero. The sensor develops a typical quantization noise of 0.1° C. The gain is positive can be interpreted as the theoretical maximum temperature that can be reached at full heating power. However, in most situations this cannot be measured experimentally because many actuators are dimensioned so that the only require approximately one third of the heating power for constant operation at the operating point. The power reserve is only intended to cover operating point changes and heating up phases.
- **Simulation of pressure control systems:** If you define the valve position so that it is closed at 0% and open at 100%, the process Gain of a container pressure control system is normally a negative value because the pressure reduces (>0) when the outlet valve of the container is opened. In contrast to this, the gain of an overpressure value is positive. `PV0 > 0` is the pressure when the value is completely closed. The situation is, of course, the opposite when pressures below ambient pressure are involved, for example, in vacuum systems. Note that most valves do not return a reproducible characteristic in the region of their closed position (actuation ratio 1:20 or 1:50). The time constants for pressure control of liquids are typically fast, whereas with pressure control in gas tanks, particularly in large tanks, they are slower. The magnitude of the process gain depends largely on the physical units of pressure, for example, Bar or Pa. Pressure sensors typically develop higher measurement noise than temperature sensors.
- **Simulation of flow control systems:** If you define the valve position so that it is closed at 0% and open at 100%, the process `Gain` is usually positive, since the flow rate increases when the valve opens. `PV0 = 0` if the flow stops completely when the valve is closed, in other words, the valve closes tight. The time constants are significantly faster than in temperature controls and are usually all of the same order. The magnitude of the process gain depends largely on the physical units of flow, for example, m3/s or l/min. The measurement noise affecting flow sensors is normally higher than with temperature sensors.

To implement a dead time for process simulation, you can insert a DeadTime block before the ProcSimC input and call it in a different cyclic interrupt OB (OB3x).

Model variants

Two different model variants are supplied:

1. Continuous process simulation ProcSimC in which the MV input is an analog value, for example, heating power or valve position.
2. Process simulation ProcSimS for step controllers, with control of the actuator by two binary inputs "up"/"down" or "open"/"close". Internally, the actuator is modeled as an integrator, whereby:
   - MotorHiLim = 100%,
   - MotorLoLim = 0%
   - TI = MotorTime.

   The integrator input is derived from the binary inputs according to the following formula:

$$Integ.Input = \begin{cases} 100 & if\ Up = True \\ -100 & if\ Down = True \\ 0 & otherwise \end{cases}$$

See also

NoiseGen I/Os (Page 2012)

21.44.3 Cascade control of temperature by using the heat flow (CascadeSim)

Cascade control of a temperature by using the heat flow

This template contains simulation models for a flow and temperature controlled system and the parameters of the noise generator block (see the I/O table). You can use this model to test the mode transitions described in the Cascade control (Page 2316) section. You can also try out the properties of different parameter sets for the primary and secondary controllers. The following features are typical for this type of application:

- The controlled temperature system is slower than the controlled flow system.
- There are two time constants that are far apart.
- There is an offset corresponding to ambient temperature.
- It has less noise than the controlled flow system.

Process parameters of the example project for cascade control

ProcSimC	Gain	TimeLag1	TimeLag2	PV0	NoiceVariance
Flow control loop	8	1	1	0	0.22
Temperature control loop	0.3	8	1	20	0.1

Parameters for PID controllers --> PI cascade with fast control response

The parameters listed in the table below apply to a fast control response with low control error but with strong actuator intervention.

PID	Gain	TI	TD
TIC101	10	8.8	2.6
FIC101	0.1	1.8	0

Controller parameters for PI --> P cascade with soft controller intervention

The advantage of the parameters listed in the table below is that the controller "goes easy" on final control element (for example a valve).

PID	Gain	TI	TD
TIC101	10	6.8	0
FIC101	0.1	0	0

It is generally advisable to make the secondary controller "simpler" than the primary controller, in other words to reduce the number of different dynamic channels so that it is genuinely secondary to the primary controller.

The steady state control error in the secondary loop is not normally relevant for the application. On the other hand, the reaction time of the secondary loop is important because the time constants of the secondary closed control loop are part of the controlled system for the primary controller. If you do without I action in the secondary controller for these reasons, it is not advisable limit the setpoint ranges of the secondary controller precisely to the achievable physical range of the process value in the secondary loop, as you would not be able to use the full actuating range of the secondary controller due to the steady state deviation. You should instead set more generous setpoint limits for the secondary controller and manipulated variable limits for the primary controller. The anti-Windup measures of the primary controller are oriented on the interconnection of `IntHoldNeg` and `IntHoldPos`. If the secondary controller does not have any I action, if will not be capable of a bumpless manual-automatic changeover. You should therefore set an `MV_Offset` that approximates the typical `MV` value for the operating point of the process.

A cascade temperature control system with a secondary controller for heating and/or cooling medium flow is commonly used for

- Heat exchangers
- Reactors without a cooling jacket

21.44.4 Control loop monitoring for simulation with colored noise (ConPerMonSim)

Control loop monitoring with simulation of colored noise

The interconnection of the ConPerMon block with a PID controller can be found in the process tag type PID_Control (see PID controller with safety logic and control loop monitoring (PIDConL_ConPerMon) (Page 2298)). The example project supports you and helps to familiarize you with the concept and the potential of closed control loop monitoring. To do this, the template includes a process simulation with disturbance model. The colored noise is generated with the aid of a shape filter from a white noise signal. This produces a spectrum of disturbance signals that also contains energy components in the lower frequency ranges of the bandwidth of the closed control loop. Part of the disturbances can therefore be compensated by the PID controller while the high-frequency measurement noise cannot be corrected by any controller.

Application

After commissioning the controller and ConPerMon block, you should be able to watch the effects of the following actions that demonstrate the potential of control loop monitoring:

- Switch the controller to manual mode:
  The variance of the controlled variable will rise but the `CPI` becomes invalid because no statements can be made about the control fit unless the control loop is closed.
- Change the parameters of the process simulation, for example, change `TimeLag2` from 2s to 8s:
  This deterioration of the dynamic characteristics of the process (for example due to wear and tear) brings about a deterioration of the control quality that becomes visible in the `CPI` value long before it can be seen with the naked eye in the standard `PV` trends. If the control quality drops below a defined level, a `CPI` warning or even an alarm is generated.
- Request a setpoint step change from the controller:
  The `CPI` will become temporarily invalid because all stochastic characteristics of the control quality, such as the variance, are based on the assumption of a steady state with a constant mean value. Select the "Setpoint" view from the drop-down list box in the ConPerMon faceplate to be able to watch the deterministic characteristics such as overshoot and settling ratio. Once a steady state is achieved again at the new setpoint and the entire time window is filled with data from the steady state, the monitoring of the stochastic characteristics is reactivated automatically.

You can find detailed information on the ConPerMon block and notes on interpreting its displays in the online help on the block (Page 553).

21.44.5 Feedforward control to compensate a measurable disturbance variable (DisturbCompSim)

Feedforward control to compensate a measurable disturbance variable

The example is based on the process tag type PID controller with dynamic feedforward control (FfwdDisturbCompensat) (Page 2303) and uses the following parameter sets:

Main controlled system:

$$g(s) = \frac{2}{2s+1} e^{-1.2s}$$

Disturbance transfer function:

$$g_z(s) = \frac{1}{3s+1} e^{-1.6s}$$

- PID: Gain = 0.197
- TI= 1.9
- TD= 0

Feedforward control:

$$c(s) = -\frac{g_z(s)}{g(s)} = -\frac{1}{2} \cdot \frac{2s+1}{3s+1} e^{-0.4s}$$

The same process simulation is set up twice, one instance with disturbance feedforward and the other without (all other process and controller parameters identical). The advantages of the feedforward control can be tested in a direct comparison ("benchmark simulation", "parallel slalom ").

21.44.6 Operating point-oriented adaptation of parameters (gain scheduling) for non-linear processes (GainSchedSim)

Operating point-oriented adaptation of parameters (gain scheduling) for non-linear processes

The example is based on the process tag type PID - control with operating-point-oriented parameter control (GainScheduling) (Page 2301).

In the simulation template, the settings of the two most important process parameters are changed based on polylines depending on the operating point. The process and controller parameters for the example are shown in the following table.

Operating point	X=PV	ProcSim.Gain	ProcSim.TmLag1	ProcSim.TmLag2	Gain	TI	TD
1	20	4	5	10	0.6	14.7	3.7
2	100	3	3	10	1	8.8	2.2
3	200	2	1	10	10	4.1	1.1

The same process simulation is set up twice, one instance with gain scheduling and the other without (all other process and controller parameters are identical). The advantages of the gain scheduling can be tested in a direct comparison ("benchmark simulation", "parallel slalom").

21.44.7 Override control on a pipeline (OverrideSim)

Override control on a pipeline

The example is based on the process tag type Override control (Page 2322) and uses the following parameter sets:

Primary process (flow control):

$$g(s) = \frac{3}{(2s+1)^2}$$

Flow increases when the valve is opened and it disappears when the valve is closed.

PI flow controller: Gain= 0.33 , TI= 2.7

Secondary process (pressure control):

$$g_p(s) = \frac{-0.8}{(7s+1)(1s+1)},$$

The pressure rises when the valve is opened and is 80 bar when it is fully open.

PI pressure controller: Gain= 2.8 , TI= 4

Switching limits 15 bar < pressure < 70 bar.

21.44.8 Smith predictor for a dead time system (SmithPredictorSim)

Smith predicator for a dead time system

The example is based on the process tag type PID controller with Smith predictor (SmithPredictorControl) (Page 2306).

In the example, the same process simulation is set up twice, one instance with Smith predictor and the other without (all other process parameters are identical). The advantages of the Smith predictor can be tested in a direct comparison ("benchmark simulation", "parallel slalom").

21.44.9 Filtering of noisy measured values in a control loop (SigSmoothSim)

Filtering of noisy measured values in a control loop

The example illustrates the use of the Smooth block in a closed control loop. The block can be connected to any signal source without specialist knowledge so there is no need for a special process tag type. The simulation template is useful in testing the effects of a low-pass filter on a closed control loop by simulation. Increasing the filter time constant improves the smoothing effect but also causes a phase lag in the control loop that can have detrimental effects on the control quality and even the stability.

Parameters used

The following parameters are used in the simulation example:

Process transfer function:

$$g(s) = \frac{3}{(15s+1)(2s+1)}$$

with white noise on the output signal.

PI controller:

- Gain = 0,5
- TI = 7 s
- Sample time = 0.1s

Butterworth filter:

- TimeConstant = 3 s.

At 0.3 seconds, hardly any smoothing effect can be recognized, at 15 seconds significant deterioration of the control quality is already noticeable.

Processes with signals strongly affected by noise are a typical area of application (for example pressure sensors) and sensitive actuators (for example valves).

You can find detailed information on the Smooth block in the online help on the block (Page 1875).

21.44.10 Predictive control of a 2x2 multi-variable controlled system (ModPreConSim)

Predictive control of a 2x2 multi-variable controlled system

The example is based on the process tag type Model-based predictive control (ModPreCon) (Page 2325).

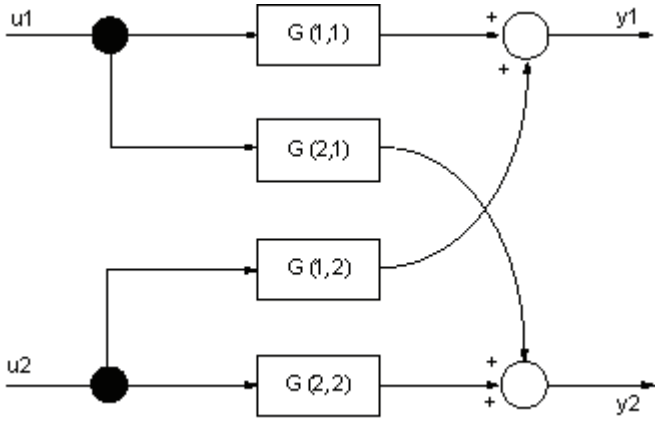


Figure 21-7 MIMO 2x2 process with a p-canonical structure

The example shows the application of the ModPreCon block for simulating a 2x2 multi-variable process consisting of the following four transfer functions:

$$\underline{G}(s) = \begin{bmatrix} G(1,1) & G(1,2) & \cdots & G(1,n_u) \\ G(2,1) & G(2,2) & \cdots & G(2,n_u) \\ \vdots & \vdots & \ddots & \vdots \\ G(n_y,1) & G(n_y,2) & \cdots & G(n_y,n_u) \end{bmatrix} = \begin{bmatrix} \dfrac{3}{(30s+1)(4s+1)} & \dfrac{1.2}{(34s+1)(14s+1)(6s+1)} \\ \dfrac{1.3}{(28s+1)(12s+1)(6s+1)} & \dfrac{4}{(26s+1)(6s+1)} \end{bmatrix}$$

where n_y = 2 = number of controlled variables, n_u = 2 = number of manipulated variables, and $G(i_y, i_u)$ the transfer function from input i_u to output i_y. This simplest of multi-variable control systems helps familiarize newcomers with the concept and application of model-based multi-variable controllers.

21.44.11 Predictive control of a non-linear process (ModPreConNonLinSim)

Predictive control of a non-linear process

The example is based on the approach used by multi-model controlling as described in the section ModPreCon functions (Page 682), Controlling linear and non-linear processes.

A multi-variable process is observed with two input variables and two output variables. The non-linear reaction of four partial transfer functions Proc511, Proc512, Proc521 and Proc522 depends on a measurable process value, in this case the controlled variable PV511.

The assumption that all non-linearities of the multi-variable process depend on the current operating point, which is defined by a single measurable variable, restricts the range of application but is reasonable in many practical applications. The approach of the presented multi-model controlling only makes sense with this assumption.

In the example it is assumed that the operating point is defined by a temperature, and the process reaction is different at high temperatures (200° C) than it is at low temperatures (20° C).

Some parameters of the third order partial transfer function

$$Proc(i, j) = \frac{CV(i)}{MV(j)} = \frac{Gain}{(TmLag1 \cdot s + 1) \cdot (TmLag2 \cdot s + 1)(TmLag3 \cdot s + 1)}$$

are set using polylines continually depending on the operating point.

The extreme values of the operating-point-oriented parameters for the example are shown in the following table:

Operating point	PV511	Proc511.Gain	Proc511.TmLag2	Proc521.Gain	Proc512.TmLag2	Proc522.Gain
1	20	4	12	0,9	19	3
2	200	2	2	1,7	9	5

The changes of the process parameters are so substantial that a single linear controller cannot achieve enough control performance over the entire operating range. The changes, however, are continual and reproducible, which is an important requirement for the multi-model approach.

All other process parameters are constant:

	Gain	TmLag1	TmLag2	TmLag3
Proc511	variable	30	variable	0
Proc512	1,2	34	variable	6
Proc521	variable	28	12	6
Proc522	variable	26	6	0

The primary controller TIC511522Low was designed using the MPC Configurator for the low operating point at 20° C, the secondary controller TIC511522High is made for the high operating point at 200° C.

The matching functions for relevance of the two controllers are polylines with four interpolation points at 0, 30, 190 and 300° C. In the range between 0 and 30° C, only the controller designed for 20° C is active; only the controller designed for 200° C is active between 190 and 300° C. The manipulated variables of both controllers overlap between 30 and 190°.

Definitions

22

22.1 Batch process

Batch process

A batch process is a process control process, which is executed according to a recipe control batch by batch, i.e. intermittently, in a continually repeating sequence, for example, dosing raw material, tempering, performing chemical reactions, cooling, discharging reactors.

22.2 Approximation

Approximation

An approximation method in the mathematical sense.

22.3 Prediction horizon

Prediction horizon

For predictive controller: Time period running from the present to the future with a defined length. A process reaction is predicted within the prediction horizon.

22.4 Trajectory

Trajectory

In physics: refers to a flight path or track.

In control engineering: course of a variable over time, described by a sequence of values in a specified time scale.

22.5 Maverick

Maverick

A maverick in a continuous physical measurement is a numerical value that changes from one sampling point to another more than would be physically plausible. In other words, the difference between two neighboring values is greater than a specified tolerance range.

22.6 Ergodic process

Ergodic process

An ergodic process in mathematical statistics is a stationary process, in which the expected value can be estimated by generating the mean value over a time period of infinite length.

22.7 Conti process

Conti process

A Conti process is a process control process, whereby raw material is fed in a continual flow, and the products are continually output.

22.8 Multivariable controller

Multivariable controller

With a multivariable controller, one manipulated variable can influence several controlled variables and one controlled variable can be influenced by several manipulated variables, as is shown in the following diagram.

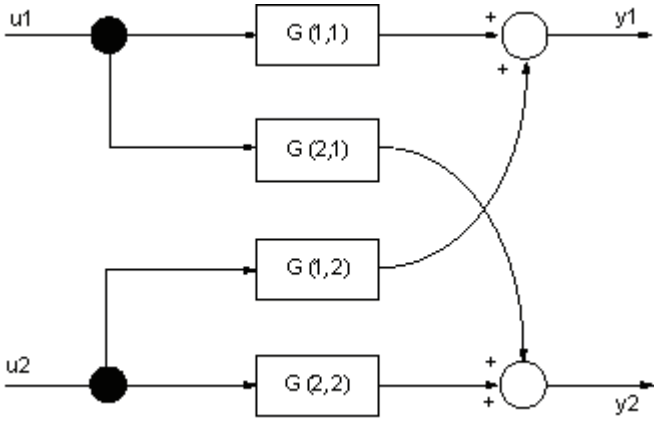


Figure 22-1 Example of a multivariable controller

When a multivariable section is automated using several individual PID controllers, the individual controllers do not take account of the interactions or links in the process. The stronger the links between the sub-sections, the harder it is to set individual controllers and the worse the control performance.

In such cases a multivariable controller offers higher control performance and simpler controller settings.

22.9 non-phase minimum

Non-phase minimum behavior mean

A phase minimum system is described by a linear, time-invariant transfer function, the frequency of which displays the smallest possible clockwise phase rotation for the given number of poles and zeros if the frequency interval is run through from negative to positive in full an infinite number of times. This means that both the transfer function and its inverse are stable.

Non-phase minimum behavior means for example that the process initially deflects downwards when the manipulated variable jumps positively before moving in a positive direction. Dead time systems are also non-phase minimal.

Index

0

2

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X