SIEMENS

SIMATIC

Process Control System PCS 7
PCS 7 Standard Library V71

Function Manual

03/2009
A5E02102227-01

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

with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken.

CAUTION

without a safety alert symbol, indicates that property damage can result if proper precautions are not taken.

NOTICE

indicates that an unintended result or situation can occur if the corresponding information is not taken into account.

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 device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel**. Within the context of the safety notes in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

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 adhered to. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of the 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
Industry Sector
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E02102227-01
℗ 11/2008

Table of contents

General Information About Block Description

1

The setup of the block description is always uniform and contains the following sections:

Header of the block description

Example: CTRL_PID: PID controller block

The header begins with the type name of the block (e.g., “"CTRL_PID"). This symbol name is entered in the symbol table and must be unique within the project.
In addition to the type name, you will also see a keyword indicating the purpose or function of the block (e.g., "PID controller block").

Object name (type + number)

FB x

The object name for the block type is made up of the type of implementation (function block = FB, function = FC) and the block number = x.

Links for displaying block I/Os

Example:

- CTRL_PID block I/Os

Click the "Block I/Os" link to display a list of block I/Os for the designated block.

Links for displaying the block icon and faceplate

If the block is intended for operator control and monitoring and a block icon and faceplate exist, the corresponding image and description can be displayed directly by clicking these links.

Example:

- CTRL_PID block icon
- CTRL_PID faceplate

Function

Here, you will find a brief description of the block function.
You will find additional information about complex blocks in the "How it works" section.

How it works

Here, you will find more detailed information, for example about the function of specific inputs, operating modes or time sequences. You must be familiar with these relationships in order to use the block effectively.

Calling OBs

Here you will find information on the organization blocks (OBs), in which the described block must be installed. If the CFC is used, the block is automatically installed in the cyclic OB (cyclic interrupt) and in the OBs listed in the block's task list (for eample in restart OB100).

CFC generates the required OBs during compilation. If you use the blocks without CFC, you will have to program these OBs and call their instance within the blocks.

Error handling

The **ENO** Boolean block output indicates the error in the CFC chart.
The value is equivalent to the **BIE** (binary result in STEP 7 STL, after completion of the block) or **OK** bit (in SCL notation) and indicates:

ENO = BIE = OK = 1 (TRUE) -> The result of the block is free of errors.

ENO = BIE = OK = 0 (FALSE) -> Invalid result or constraints (for example, input values and modes).

The FBs also return the inverted BIE at the **QERR** output of the instance DB.

QERR = NOT ENO

The error message is generated in two separate operations:

- The operating system detects a processing error (e.g. value overflow, system functions called return an error ID with BIE = 0).
  This is a system function and is not specifically mentioned in the block description.
- The block algorithm checks for functional invalidity of values and operating modes. These error events are logged in the block description.

You can evaluate the error display, for example, to generate messages or use substitute values for invalid results. You will find more information about messages in the "Message blocks" section.

Startup characteristics

The different startup behaviors are as follows:

- Initial start

  The block is called for the first time from the OB in which it has been inserted. This is usually the OB that performs the standard, process-specific operations (for example, the cyclic interrupt OB).
  The block adopts a status that conforms to its input parameters. These may be default values (additional information in "I/Os" section) or values you have already configured, for example, in CFC. The initial startup characteristics are not described separately unless the block does not conform to this rule.

- Startup

  The block is executed once during CPU startup. The block is called in the startup OB (where it is additionally installed either automatically in the ES or manually in STEP 7). In this case, the startup characteristics are described.

  Please note that the block outputs have default values and that these can take effect during the CPU startup with other blocks, if these are processed first.

  The correct startup behavior of the blocks is the responsibility of the configuration engineer.

Time response

A block assigned this function must be installed in a cyclic interrupt OB. It calculates its time constants/parameters on the basis of its sampling time (the time which elapses between two consecutive cyclic operations).

In a CFC configuration on ES, the sampling time is also determined by the segmentation of the runtime group, which ensures that the block is not executed during every OB run.

This sampling time is entered at the I/Os, in the SAMPLE_T parameter.

When configuring with CFC, this occurs automatically once the block has been inserted in the OB and the runtime group. For this reason, this input is set to invisible in CFC.

During the STEP 7 configuration, you set the time response manually.

Time response is mentioned only if the block has been assigned this feature.

Message response

A block with message response reports various events to the higher level OS. Existing parameters required for the generation of messages are documented.

Blocks without message response can be expanded with additional message blocks. A reference to the message response is found in the description of the individual message blocks.

I/Os

The I/Os of the block represent its data interface. These I/Os can be used either to transfer data to the block or to fetch results from the block.

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
U1	Addend 1	REAL	0	I	+	>0
.....						

The "I/O" table lists all I/O parameters of the block type. You can access these lists using the engineering tools. They are in alphabetical order. Elements accessible only via the block algorithm (internal variables) are not listed.
The meaning of the columns is as follows:

- **I/O**

  Name of the parameter, derived from the English, e.g. PV_IN = **P**rocess **V**ariable **IN**put (process variable, controlled variable).
  The SIMATIC naming conventions have been applied.

  The block representation in CFC as supplied is as follows:
  I/O name in **bold** characters = I/O is visible, regular = I/O is invisible.

- **Meaning**

  Function (possibly also short description)

- **Data type**

  S7 data type of the parameter (BOOL, REAL, etc.)

- **Default (default value)**

  The value of the parameter before the block runs for the first time (unless changed in the configuration)

- **Type**

  The type of access of the block algorithm to the parameter; there are inputs, outputs, and retroactive inputs:

Abbreviation	Type
I	Input. Supplies values to the block (representation in CFC: left-hand block side)
O	Output. Output value. (representation in CFC: right-hand block side)
IO	Run = input/output. Retroactive input that can be written to by the OS and written back by the block (representation in CFC: left-hand block side)

- **OCM**

  Parameters marked "+" can be adjusted and monitored via the corresponding faceplate.

- **Permissible values**

  Additional limitation within the data type's range of values

Operating and monitoring

When a faceplate exists for the AS block, links to descriptions of the corresponding faceplate and block icon are available (also with the buttons in the upper part of the topic).

See also

Block icon: CTRL_PID (Page 582)

I/Os of CTRL_PID (Page 68)

Family: COMM

2

2.1 REC_BO: Receiving 128 BOOLEAN values with BRCV

2.1.1 Description of REC_BO

Object name (type + number)

FB 208

- REC_BO block I/Os (Page 23)

Area of application

The REC_BO block represents a simple user interface to SFB 13 "BRCV".

It receives 128 BOOL values via an MPI, PROFIBUS, or Ethernet connection from another S7 CPU. This CPU needs to call the function-block type "SEND_BO" (FB 207) from the *PCS 7 Library* in order to send data. In STEP 7, a homogeneous connection must be configured for both communication partners and transferred to the AS.

Data are only available after the job has been completed, and after a 0 -> 1 signal transition at the **NDR** output.

Calling OBs

This is the cyclic-interrupt OB in which you install the block (for example, OB 35).

How it works

The internal SFB 13 "BRCV" allows 128 BOOL values to be exchanged between communication partners. Data are received by the operating system of the CPU and entered in the instance DB of the receive FB (REC_BO). Before new data can be received, the operating system must transmit an internal acknowledgment of data received.

Data are entered in the data block asynchronously to user-program execution. After "REC_BO" has been called, instance-DB data may not be computed as long as the job is in progress (**NDR** = 0). If the job is completed without error, output NDR = 1 for the duration of one cycle. In the next cycle, the FB automatically outputs the receive enable signal to the CPU operating system (**NDR** is reset to 0 as of this call).

The receive enable signal can be effective prior to the first incoming receive job. In this case, the receive enable signal is stored by the operating system.

The **ID** parameter represents the connection number specified in your connection configuration and is applied only at the first call after a cold restart.

The **R_ID** parameter is a random number (suggestion: message-frame ID); however, it must be identical at the corresponding send and receive blocks and is applied only at the first call after a cold restart.

The "REC_BO" block must be called for each **ID/R_ID** pair and in each program cycle (cyclically or also via timeout alarms). Each message frame requires two calls of "REC_BO".

The **ERR** (error) and **STAT** (status) outputs indicate specific error information relevant to SFB 13 (see "Error handling").

If an error occurs, substitute values can be output as received data.

Error handling

Error handling of the REC_BO block is restricted to the error information of the lower-level SFB 13 "BRCV". You can find additional information in the manual titled *System Software for S7-300/400 - System and Standard Functions*, which describes the **ERR** and **STAT** outputs.

If input SUBS_ON = TRUE, substitute values will be output to REC_MON (number of cycles) if receive errors occur or if new data are not received. While the REC_MON cycles are running, the last valid values are applied to the output.

If the SUBS_ON input is set to FALSE, the last valid values are always applied to the output if an error has occurred.

Startup characteristics

Not available.

Time response

Not available.

Message response

Not available.

Operating and monitoring

Not available.

2.1.2 I/Os of REC_BO

The factory setting of the block display in CFC is identified in the "I/O" column:
I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description".

I/O (parameter)	Meaning	Data type	Default	Type
ERR	1 = error (for error type see STAT)	BOOL	0	O
ID	Connection ID	WORD	0	I
NDR	1 = new data received	BOOL	0	O
QNO_REC	1 = no data received	BOOL	0	I
QSUBS_ON	1 = substitute values	BOOL	0	I
R_ID	Message-frame ID	DWORD	0	I
RD_BO_00	Received value 00	BOOL	0	O
...	...			
RD_BO_15	Received value 15	BOOL	0	O
RD_BO_16	Received value 16	BOOL	0	O
...	...			
RD_BO_127	Received value 127	BOOL	0	O
REC_MON	Reception monitoring (value = number of block calls)	INT	3	I
STAT	Error ID	WORD	0	O
SUBBO_00	Substitute value 00	BOOL	0	I
...	...			
SUBBO_15	Substitute value 15	BOOL	0	I
SUBBO_16	Substitute value 16	BOOL	0	I
...				
SUBBO127	Substitute value 127	BOOL	0	I
SUBS_ON	1 = substitute values on in the event of error	BOOL	0	I

2.2 REC_R: Receiving 32 BOOLEAN and 32 REAL values with BRCV

2.2.1 Description of REC_R

Object name (type + number)

FB 210

- REC_R block I/Os (Page 26)

Area of application

The REC_R block represents a simple user interface to SFB 13 "BRCV".

It receives 32 BOOL and 32 REAL values via an MPI, PROFIBUS, or Ethernet connection from another S7 CPU. This CPU needs to call the function-block type "SEND_R" (FB 209) from the *PCS 7 Library* in order to send data. In STEP 7, a homogeneous connection must be configured for both communication partners and transferred to the AS.

Data are only available after the job has been completed, and after a 0 -> 1 signal transition at the **NDR** output.

Calling OBs

This is the cyclic-interrupt OB in which you install the block (for example, OB 35).

How it works

The internal SFB 13 "BRCV" allows 32 BOOL and 32 REAL values to be exchanged between communication partners. Data are received by the CPU operating system and entered in the instance DB of the receive FB (REC_R). Before new data can be received, the operating system must transmit an internal acknowledgment of data received.

Data are entered in the instance DB asynchronously to user-program execution. After "REC_R" has been called, instance-DB data may not be computed as long as the job is in progress (**NDR** = 0). If the job completes without errors, the **NDR** output is set to 1 for the duration of one cycle. In the next cycle, the FB automatically outputs the receive enable signal to the CPU operating system (**NDR** is reset to 0 as of this call).

The receive enable signal can be effective prior to the first incoming receive job. In this case, the receive enable signal is stored by the operating system.

The **ID** parameter represents the connection number specified in your connection configuration and is applied only at the first call after a cold restart.

The **R_ID** parameter is a random number (suggestion: message-frame ID); however, it must be identical at the corresponding send and receive blocks and is applied only at the first call after a cold restart.

The "REC_R" block must be called for each **ID/R_ID** pair and in each program cycle (cyclically or also via timeout alarms). Each message frame requires two calls of "REC_R".

The **ERR** (error) and **STAT** (status) outputs indicate specific error information relevant to SFB 13 (see "Error handling").

If an error occurs, substitute values can be output as received data.

Error handling

Error handling of the REC_R block is restricted to the error information of the lower-level SFB 13 "BRCV". You can find additional information in the manual titled *System Software for S7-300/400 - System and Standard Functions*, which describes the **ERR** and **STAT** outputs.

If input SUBS_ON = TRUE, substitute values will be output to REC_MON (number of cycles) if receive errors occur or if new data are not received. While the REC_MON cycles are running, the last valid values are applied to the output.

If the SUBS_ON input is set to FALSE, the last valid values are always applied to the output if an error has occurred.

Startup characteristics

Not available.

Time response

Not available.

Message response

Not available.

Operating and monitoring

Not available.

2.2.2 I/Os of REC_R

The factory setting of the block display in CFC is identified in the "I/O" column:
I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description".

I/O (parameter)	Meaning	Data type	Default	Type
ERR	1 = error (for error type see STAT)	BOOL	0	O
ID	Connection ID	WORD	0	I
NDR	1 = new data received	BOOL	0	O
QNO_REC	1 = no data received	BOOL	0	I
QSUBS_ON	1 = substitute values	BOOL	0	I
R_ID	Message-frame ID	DWORD	0	I
RD_BO_00	Received value BOOL 00	BOOL	0	O
...				
RD_BO_07	Received value BOOL 07	BOOL	0	O
RD_BO_08	Received value BOOL 08	BOOL	0	O
...				
RD_BO_31	Received value BOOL 31	BOOL	0	O
RD_R_00	Received value REAL 00	REAL	0	O
...				
RD_R_07	Received value REAL 07	REAL	0	O
RD_R_08	Received value REAL 08	REAL	0	O
...				
RD_R_31	Received value REAL 31	REAL	0	O
REC_MON	Reception monitoring (value = number of block calls)	INT	3	I
STAT	Error ID	WORD	0	O
SUBBO_00	Substitute value BOOL 00	BOOL	0	I
...				
SUBBO_07	Substitute value BOOL 07	BOOL	0	I
SUBBO_08	Substitute value BOOL 08	BOOL	0	I
...				
SUBBO_31	Substitute value BOOL 31	BOOL	0	I
SUBR_00	Substitute value REAL 00	REAL	0	I
...				
SUBR_07	Substitute value REAL 07	REAL	0	I
SUBR_08	Substitute value REAL 08	REAL	0	I
...				
SUBR_31	Substitute value REAL 31	REAL	0	I
SUBS_ON	1 = substitute values on in the event of error	BOOL	0	I

See also

General Information About Block Description (Page 15)

2.3 SEND_BO: Sendin 128 BOOLEAN values with BSEND

2.3.1 Description of SEND_BO

Object name (type + number)

FB 207

- SEND_BO block I/Os (Page 30)

Area of application

The SEND_BO block represents a simple user interface to SFB 12 "BSEND".

It sends up to 128 BOOL values via an MPI, PROFIBUS, or Industrial Ethernet connection to another S7 CPU. This CPU needs to call the function-block type "REC_BO" (FB 208) from the *PCS 7 Library* in order to receive data.

Data are only consistently available at "REC_BO" once the job has been completed (i.e. after the acknowledgment DONE = TRUE has been received). This acknowledgment is indicated by the signal at output **CIW** changing to 0.

If the **FAST** parameter = 1, the FB enables the transmission of one message frame with each FB call, since it calls SFB 12 "BSEND" twice internally (a positive edge at its **REQ** control input is required in order to enable SFB 12). However, such frequent send requests should only be initiated if sufficient time is available between two FB calls to transfer the message frame.

Calling OBs

This is the cyclic-interrupt OB in which you install the block (for example, OB 35).

How it works

The internal SFB 12 "BSEND" allows 128 BOOL values to be exchanged between communication partners. Data are sent to the communication partner by the operating system of the CPU and entered in the instance DB of the receive FB (REC_BO) automatically. Before any new 128 BOOL values are sent, the system waits for the operating system to transfer an internal acknowledgment for the values that have just been sent.

Data transfer is initiated if control input **COM** = 1 when the block is called. The block must be called at least once (regardless of the cycle) in order to transfer all data. It transfers the job to the CPU operating system for complete processing of the job. Further calls of the block with the same **ID** and **R_ID** during data transfer are permitted; however, they do not have any function, i.e. the block can be called only once in each cycle. Instead, the value 11 is output at **STAT**. Data are read from the user memory asynchronously to user-program execution. **CIW** is set to 0 if the job completes without errors. (If an error has occurred, ERR = 1. A new job will be initiated with the actual data automatically until all data have been transferred.) If the **COM** input = 0, current data transfer will be aborted and not resumed; **CIW** is then 0.

The **ID** parameter represents the connection number specified in your connection configuration and is applied only at the first call after a cold restart.

The **R_ID** parameter is a random number (suggestion: message-frame ID); however, it must be identical at the corresponding send and receive blocks and is applied only at the first call after a cold restart.

If the **FAST** parameter = 1, a new message frame can be transferred with each call of the FB. In this case the FB calls SFB 12 "BSEND" twice internally (a positive edge at its **REQ** control input is required to enable SFB 12). Calling the FB in each cycle enables **one message frame to be transferred per cycle**. However, such frequent send requests are only advisable in the following situations:

- There is sufficient time between two FB calls to transfer the message frame.
- The call of the "REC_BO" in the receiving CPU is faster than that of the "SEND_BO" in the sending CPU (the receiving CPU requires two calls (= 2 cycles) of "REC_BO" for each message frame).

If the **FAST** parameter = 0, a new send job can be initiated only at every second FB call.

Error handling

Error handling of the block is restricted to the error information of the lower-level SFB 12 "BSEND". You can find additional information in the manual titled *System Software for S7-300/400 - System and Standard Functions*, which describes the **ERR** and **STAT** outputs.

If an error has occurred, a new job will be initiated with the actual data automatically until all data have been transferred.

Startup characteristics

Not available.

Time response

Not available.

Message response

Not available.

Operating and monitoring

Not available.

2.3.2 I/Os of SEND_BO

The factory setting of the block display in CFC is identified in the "I/O" column:
I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description".

I/O (parameter)	Meaning	Data type	Default	Type
BO_00	Input 00	BOOL	0	I
...	...			
BO_15	Input 15	BOOL	0	I
BO_16	Input 16	BOOL	0	I
...	...	...	...	
BO_127	Input 127	BOOL	0	I
CIW	1 = job is busy	BOOL	0	O
COM	1 = send continuously, 0 = do not send	BOOL	1	I
DONE	Job is completed	BOOL	0	O
ERR	Error (for error type see STAT)	BOOL	0	O
FAST	Transmission type: 1 = one message frame can be transferred per FB call 0 = one message frame can be transferred with two FB calls	BOOL	0	I
ID	Connection ID	WORD	0	I
R_ID	Message-frame ID	DWORD	0	I
STAT	Error ID	WORD	0	O

See also

General Information About Block Description (Page 15)

2.4 SEND_R: Send 32 BOOL and 32 REAL values, driven by changes, with BSEND

2.4.1 Description of SEND_R

Object name (type + number)

FB 209

- SEND_R block I/Os (Page 34)

Area of application

The SEND_R block represents a simple user interface to SFB 12 "BSEND".

It is driven by changes and sends up to 32 BOOL and 32 REAL values via an MPI, PROFIBUS, or Industrial Ethernet connection to another S7 CPU. This CPU needs to call the function-block type "REC_R" (FB 210) from the *PCS 7 Library* in order to receive data.

Data are only consistently available at "REC_R" once the job has been completed (i.e. after the acknowledgment DONE = TRUE has been received). This acknowledgment is indicated by the signal at output **CIW** changing to 0.

If the **FAST** parameter = 1, the FB enables the transmission of one message frame with each FB call, since it calls SFB 12 "BSEND" twice internally (a positive edge at its **REQ** control input is required in order to enable SFB 12). However, such frequent send requests should only be initiated if sufficient time is available between two FB calls to transfer the message frame.

Calling OBs

This is the cyclic-interrupt OB in which you install the block (for example, OB 35).

How it works

The block monitors the data at 32 BOOL and 32 REAL inputs in order to detect changes to the previous values which were successfully sent. A hysteresis HYS_R_xx (absolute value) is set for REAL values (R_xx) in every change monitoring cycle. The default value for HYS_R_xx is zero (no hysteresis). Data transfer can be locked or forced via the EDC_MIN and EDC_MAX parameters.

At EDC_MIN, you set the number of cycles to wait until the next send request of actual input data, regardless of any changes in data.

At EDC_MAX, you set the number of cycles to wait after the last valid data transfer until the next transfer of actual input data, regardless of any changes in data or whether changes to REAL values are within the set hysteresis.

The theoretical time based on the set number of cycles cannot be guaranteed due to the asynchronous data transfer between "SEND_R" and "REC_R" (see below).

The internal SFB 12 "BSEND" allows 32 BOOL and 32 REAL values to be exchanged between communication partners. Data are sent to the communication partner by the operating system of the CPU and entered in the instance DB of the receive FB (REC_R) automatically. Before the new values are sent, the system waits for the operating system to transfer an internal acknowledgment for the values that have just been sent.

Data transfer is initiated if control input **COM** = 1 when the block is called. The block must be called at least once (regardless of the cycle) in order to transfer all data. It transfers the job to the CPU operating system for complete processing of the job. Further calls of the block with the same **ID** and **R_ID** during data transfer are permitted; however, they do not have any function, i.e. the block can be called once in each cycle. Instead, the value 11 is output at **STAT**. Data are read from the user memory asynchronously to user-program execution. **CIW** is set to 0 if the job completes without errors. (If an error has occurred, ERR = 1. A new job will be initiated with the actual data automatically until all data have been transferred.) If the **COM** input = 0, all incomplete data transfers will be aborted and not resumed; **CIW** is then 0.

The **ID** parameter represents the connection number specified in your connection configuration and is applied only at the first call after a cold restart.

The **R_ID** parameter is a random number (suggestion: message-frame ID); however, it must be identical at the corresponding send and receive blocks and is applied only at the first call after a cold restart.

If the **FAST** parameter = 1, a new message frame can be transferred with each call of the FB. In this case the FB calls SFB 12 "BSEND" twice internally (a positive edge at its **REQ** control input is required to enable SFB 12). Calling the FB in each cycle enables **one message frame to be transferred per cycle**.

However, such frequent send requests are only advisable in the following situations:

- There is sufficient time between two FB calls to transfer the message frame.
- The call of the "REC_R" in the receiving CPU is faster than that of the "SEND_R" in the sending CPU (the receiving CPU requires two calls (= 2 cycles) of "REC_R" for each message frame).

If the **FAST** parameter = 0, a new send job can be initiated only at every second FB call.

Error handling

Error handling of the block is restricted to the error information of the lower-level SFB 12 "BSEND". You can find additional information in the manual titled *System Software for S7-300/400 - System and Standard Functions*, which describes the **ERR** and **STAT** outputs.

If an error has occurred, a new job will be initiated with the actual data automatically until all data have been transferred.

Startup characteristics

Not available.

Time response

Not available.

Message response

Not available.

Operating and monitoring

Not available.

2.4.2 I/Os of SEND_R

The factory setting of the block display in CFC is identified in the "I/O" column:
I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description".

I/O (parameter)	Meaning	Data type	Default	Type
BO_00	BOOL input 00	BOOL	0	I
...	…			
BO_07	BOOL input 07	BOOL	0	I
BO_08	BOOL input 08	BOOL	0	I
...	…			
BO_31	BOOL input 31	BOOL	0	I
CIW	1 = job is busy	BOOL	0	O
COM	1 = send continuously, 0 = do not send	BOOL	1	I
DONE	1 = job is completed	BOOL	0	O
EDC_MAX	Force transfer after n cycles without changes	INT	10	I
EDC_MIN	Earliest transfer after n cycles if changes were made	INT	1	I
ERR	Error (for error type see STAT)	BOOL	0	O
FAST	Transmission type: 1 = one message frame can be transferred per FB call 0 = one message frame can be transferred with two FB calls	BOOL	0	I
HYS_R_00	Hysteresis input 00	REAL	0	I
...	…			
HYS_R_07	Hysteresis input 07	REAL	0	I
HYS_R_08	Hysteresis input 08	REAL	0	I
...	…			
HYS_R_31	Hysteresis input 31	REAL	0	I
ID	Connection ID	WORD	0	I
R_00	REAL input 00	REAL	0	I
...	…			
R_07	REAL input 07	REAL	0	I
R_08	REAL input 08	REAL	0	I
...	…			
R_31	REAL input 31	REAL	0	I
R_ID	Message-frame ID	DWORD	0	I
STAT	Error ID	WORD	0	O

See also

General Information About Block Description (Page 15)

Family: CONTROL

3

3.1 Controller Optimization Using the PID Tuner

Using the PID tuner, you can tune the following control blocks:

- CTRL_PID
- CTRL_S
- FMCS_PID
- FMT_PID

How it works

With the "Controller tuning" function, you determine the optimum controller settings for a particular process. A wizard guides you through the individual steps.

The measured data of the controlled system is recorded. The optimum PID parameters are calculated based on this data and prepared for further use.

You will find additional information on the use of the PID tuner in the corresponding HELP file. You can call up this help by selecting a block on the chart in CFC and activating the menu command **Edit > Controller Optimization**.

3.2 CPM: Monitoring of control-loop performance

3.2.1 Description of CPM

Object name (type + number)

FB 140

- CPM block I/Os (Page 51)

Area of application of CPM

The block is used to continuously monitor the control performance of control loops.

The CPM block calculates:

- Stochastic characteristics of the control performance with the process in a steady state
  - Mean value of the controlled variable, variance and standard deviation of the 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 displayed in the faceplate of the CPM block.

In an overview representation of a plant or unit, you can obtain a clear picture of the state of all control loops based on CPM block icons (indicator light function).

The aim is to detect errors before they occur 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 CPM 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 quality 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 CPM 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. The CPM block is linked to the assigned controller block as a result of the naming convention: The name of the CPM instance is exactly the same as the name of the controller instance with "_cpm" added.

Example

The TIC501_cpm monitoring block belongs to controller TIC501.

After successful commissioning and optimization of the PID controller to be monitored, the CPM block is initialized while the process is in a steady state and then 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 in the CFC trend view in the engineering system or in WinCC online trend control on the OS and wait until the process has settled.
- To specify the length TimeWindow of the sliding time window, monitor the block output PV_Variance of the CPM block 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 quality is detected by the CPM 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.
- The CPM block can be initialized if the controller
  - Has been optimized
  - Has reached a steady state
  - Has specified the time window and
  - Is filled with values from the steady state.

  Initialization is performed via the "Initialization" button in the parameter view of the CPM faceplate or if parameter InitRefVar = true at the block. This saves the PV_Variance parameter in the current time window as a reference value for calculating the control quality in the block, along with reference values for manipulated variable and process value.

The Control Performance Index 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 Functions (Page 40).

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 CPM block after the CPI value has settled to a constant level.
- 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 TimeWindowCPI_FiltFactor.

The CPM faceplate is opened from the faceplate of the assigned PID controller so that the CPM block icons do not need to be installed separately in each OS picture. It is, in fact, advisable to group all CPM 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 quality 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. This can also be viewed in the example projects (Page 632).

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. This means that the block must be called in the startup OB. In CFC engineering, this is handled by the CFC.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SampleTime parameter by CFC during compilation.

Status word allocation

Bit number:	3	2	1	0
Parameter:	-	-	BatchEn	Occupied

Bit number:	7	6	5	4
Parameter:	-	-	CPI_Suppress	PID_AutAct

Bit number:	11	10	9	8
Parameter:	MV_known	-	-	-

Bit number:	15	14	13	12
Parameter:	-	-	-	-

Bit number:	19	18	17	16
Parameter:	CPI_AlmAct	CPI_WrnAct	OvsWrnAct	OvsAlmAct

Bit number:	23	22	21	20
Parameter:	AlmOffDelayAct	AlmOnDelayAct	-	-

Bit number:	27	26	25	24
Parameter:	UserStatus bit 4	UserStatus bit 3	UserStatus bit 2	UserStatus bit 1

Bit number:	31	30	29	28
Parameter:	UserStatus bit 8	UserStatus bit 7	UserStatus bit 6	UserStatus bit 5

See also

CPM reporting (Page 50)
Error handling of CPM (Page 49)
CPM modes (Page 40)

3.2.2 CPM modes

CPM modes

This block does not provide any mode changes of its own. The block adopts the mode from the higher-level PID controller.

The mode "Manual/Automatic" is displayed in the block icon of the CPM block (A/M).

See also

CPM I/Os (Page 51)

CPM reporting (Page 50)

Error handling of CPM (Page 49)

Functions of CPM (Page 40)

Description of CPM (Page 36)

3.2.3 Functions of CPM

Functions of CPM

The block provides the following functions:

- Monitoring of stochastic characteristics of the control quality
- Monitoring of deterministic characteristics of the control quality
- Graphic evaluation and long-term statistics
- Alternatives for determining the reference variance
- Special cases cascade control and multivariable control
- Calculating the quality code
- Alarm delay

Monitoring of stochastic characteristics of the control quality

The mean value of a variable of an ergodic stochastic process can be determined from a sliding time window with the length n= TimeWindow / SampleTime, for example for the controlled variable y= PV:

$$\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y(i)$$

A recursive formulation of this calculation is included in the function block MEANTM_P that is called by the CPM block. 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 other than zero at a constant setpoint is an indication of problems in the control loop, if the controller has I action. You should then check the following potential causes:

- Actuator intervention is limited and inadequate; in other words, the actuator stops at its limit. This can be caused by unsuitably dimensioned actuators or may simply be a consequence of 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 process gain if it is assumed that only disturbances with zero mean have an effect:

$$\mathrm{StatGain} = \frac{\mathrm{PV_{Mean}} - \mathrm{PV_{Ref}}}{\mathrm{MV_{Mean}} - \mathrm{MV_{Ref}}}$$

Normally the reference operating point is obtained during the initialization of the CPM block. Estimation of the process again is then, however, impossible precisely at this operating point. As an alternative, you can also enter 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 = PVamb for MV=0, in other words, ambient temperature

If the process gain gradually deteriorates over time, this is an indication of wear and tear in the process, for example, fouling of heat exchangers, deposits on valves or flaps, reduced effectiveness of units, etc. If, for example, a temperature control loop is connected via a heat exchanger and deposits form on the heat-exchanger surfaces, the heat transfer coefficient is reduced and with it the system gain. Within certain limits, this can be compensated by a closed control loop (so that the controller initially "hushes up" the problem). Although the original control-loop dynamics can be restored (to a certain extent) by a 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 process 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, or second central moment, requires the differences between each current measured value and the (constant) mean value to be calculated:

$$\sigma_y^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(y(i)-\bar{y}\right)^2 = \frac{1}{n}\left(\sum_{i=1}^{n} y^2(i)\right) - \bar{y}^2$$

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 in the unit [%] describes the current variance of the control quality relative to a benchmark. It is defined as

$$\varsigma = \frac{\sigma_{ref}^2}{\sigma_y^2} 100\%$$

The CPI moves in the range $0 < \varsigma \leq 100\%$. If the current variance corresponds to the benchmark value, the index reaches the value 100; if the current variance, on the other hand, increases, the control performance index drops accordingly. Ideally, the reference variance is obtained in a defined good state of the control loop and stored when the CPM block is initialized. It does not matter if the CPI temporarily reaches values 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 reference variance will be explained in a separate section.

The disadvantage of these stochastic characteristics is that they assume an ergodic, in other words, a steady state in the process act 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 CPM block 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 CPM block freezes the CPI value and automatically suppresses all messages relating to this. As the user, you can also force the suppression of the messages manually via the ManSupprCPI = 1 binary input. This is useful to avoid false alarms when known disturbances occur, for example at a load change in a continuous process or a dosing procedure in a batch process. With such events, it is natural that the variance of the controlled variable rises temporarily and this should not be interpreted as a degrading of the control performance.

Monitoring of deterministic characteristics of the control quality

The assessment of the control performance based on the response to a step change in the setpoint is relatively simple even with the naked eye. In the sense of automatic monitoring, the CPM 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,

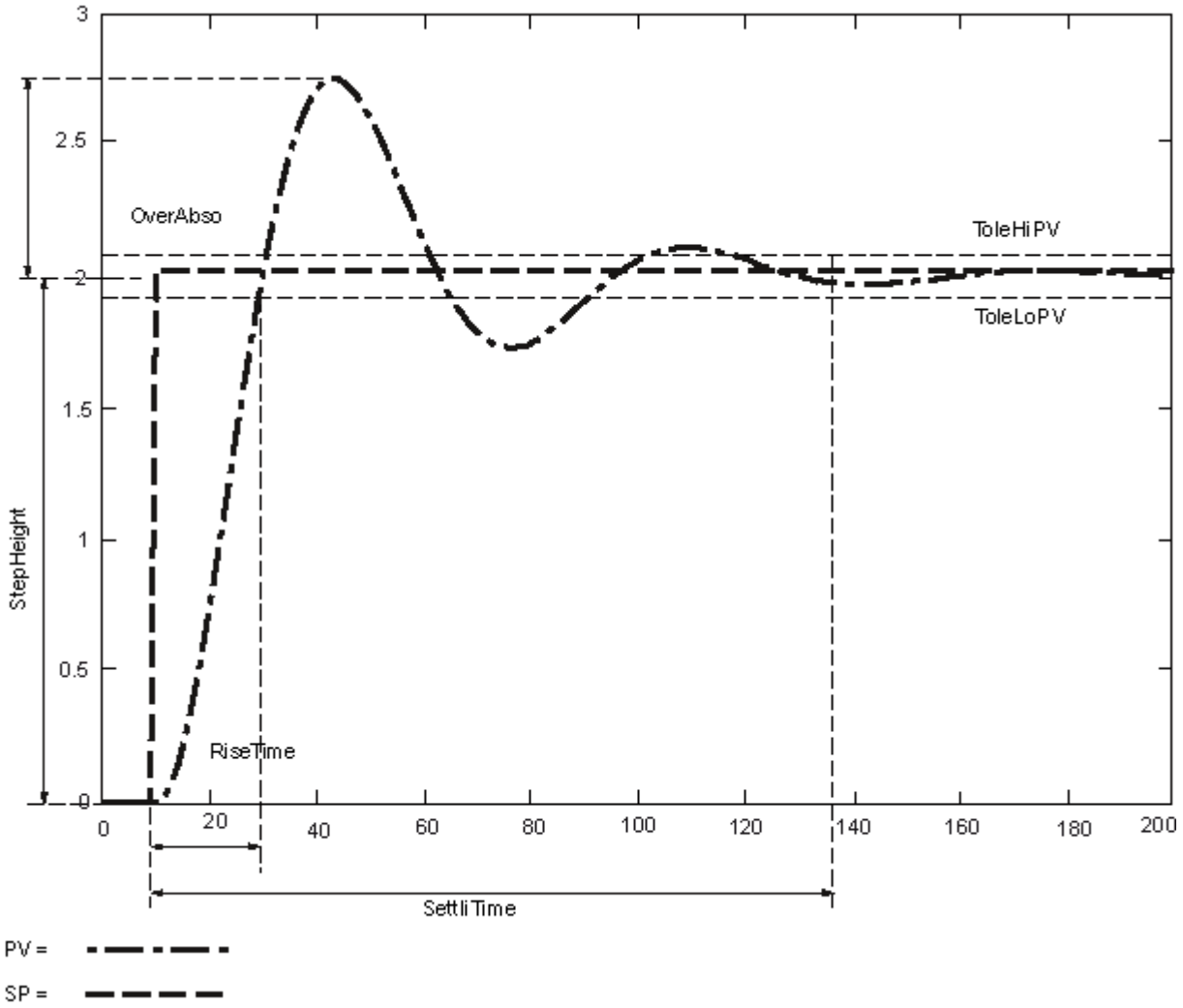
OverAbso = max(PV) - SP > 0

is output where is for a negative step response (step response down), and negative values

OverAbso = min(PV) - SP < 0

are also output. The absolute overshoot is consulted for normalization; related to the height of the step change in the setpoint, it is always positive. The relative 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 value.



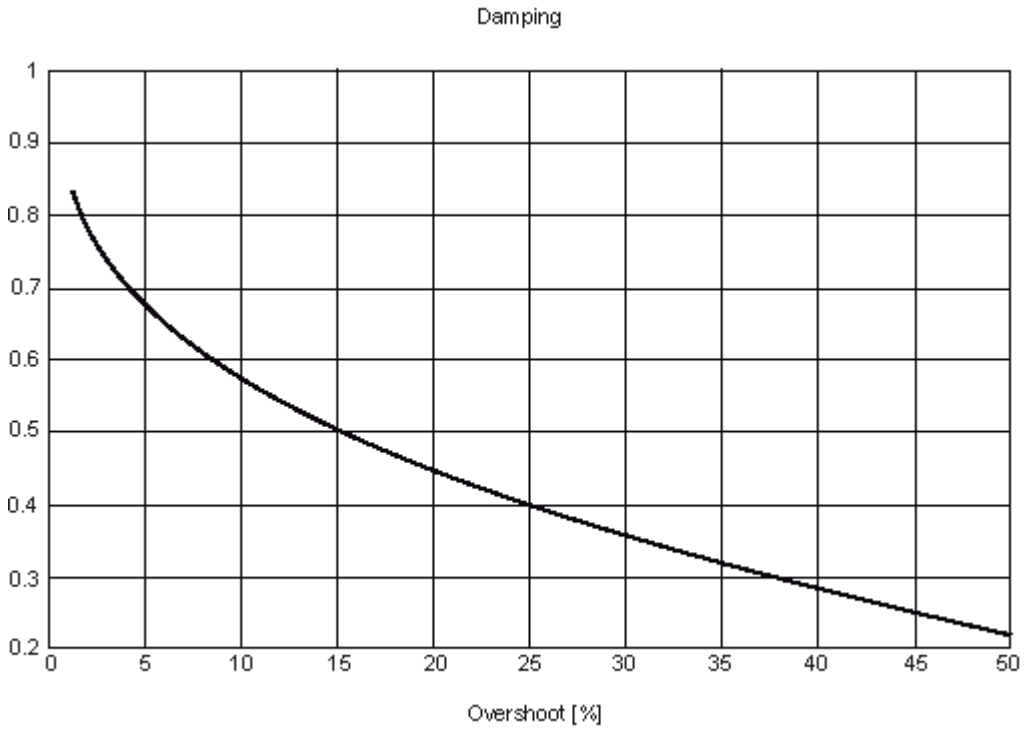
In every control loop, there is a general correlation between overshoot and phase reserve: The higher the overshoot, the lower the phase reserve. If the response of the closed control loop can be described approximately by a 2nd order transfer function

$$g_{cl}(s) = \frac{PV(s)}{SP(s)} = \frac{1}{\frac{1}{\omega_0^2}s^2 + 2\frac{\delta}{\omega_0}s + 1}$$

the following relationships are known:

1. If δ ≥ 1, the overshoot is equal to zero and the settling response is asymptotic.
2. if δ < 1, overshoot and oscillations occur. The damping of the closed loop can be determined approximately from the overshoot:

$$\delta = \frac{-\ln\left(\frac{\text{Overshoot}}{100\%}\right)}{\sqrt{\ln^2\left(\frac{\text{Overshoot}}{100\%}\right) + \pi^2}}$$



A useful controller setting typically aims for overshoot between 5 and 25%, In other words, damping between 0.7 and 0.4.

If overshoot is too high, it is often helpful to reduce the gain of the controller.

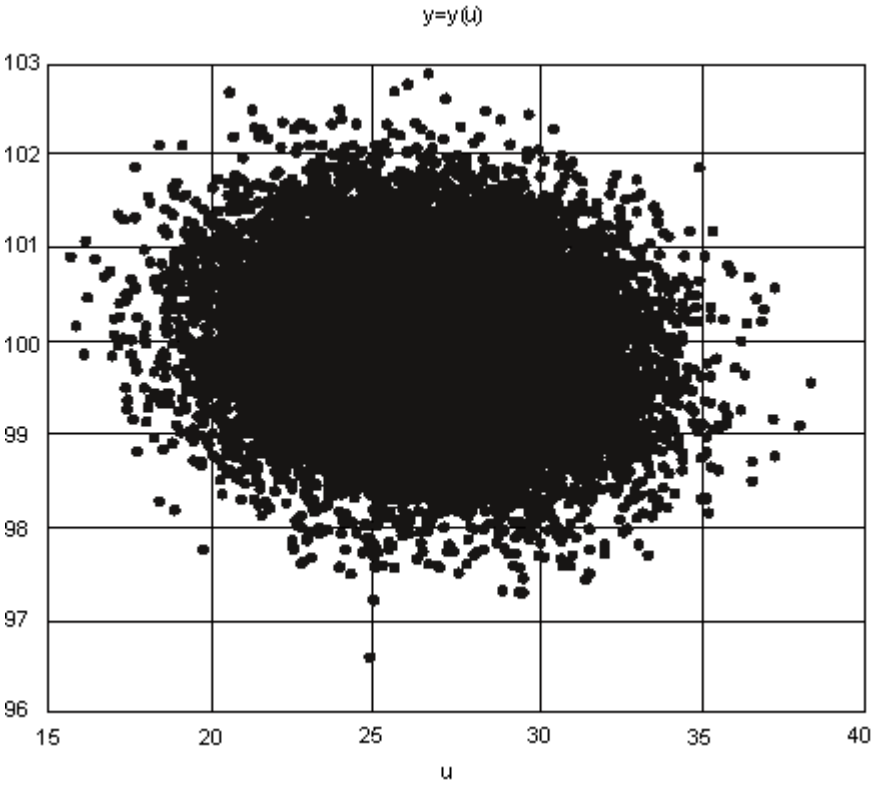
While overshoot primarily serves to check controller gain, there is a further characteristic that provides information on the setting of the I action: If the setting of the reset time is unsuitable, the process value will creep towards the new setpoint following a step change in the setpoint. To allow normalization, the settling time SettliTime is related to the rise time RiseTime of the step response of the control loop. 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 long. 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 CPM 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 output and the stochastic evaluation is activated again.

Graphic evaluation and long-term statistics

The statistics view of the faceplate displays the most important measured values of the monitored control loop and these are stored automatically in the process value archive. Setpoint (SP), process value (PV) with 3σ tolerance band (PV_ToHi and PV_ToLo) manipulated variable (MV) and CPI along with the binary states automatic mode (AutAct), high/low manipulated variable limits (HiLiAct and LoLiAct) active. You can evaluate these curves with the aid of the statistical functions of the WinCC online trend control over freely selectable (including longer) periods:

- Mean value, variance and standard deviation of the process value and manipulated variable that are only calculated by the block itself in a short time window.
- You can interpret the mean value of binary variables directly as time components. If, for example, the mean value of AutAct is 0.8 over a certain period, the controller was in automatic mode for 80% of this time; in other words, the "service factor" of the controller is 80%. If the mean value of MV_HiLim is 0.6, the controller was stuck at the high limit for 60% of the time. If such a problem was not caused by an unsuitably dimensioned actuator, it may indicate wear and tear on the actuator.



A scatter plot is a two-dimensional representation of all measured-value pairs, process value over manipulated variable. In a well set control loop, the data points should form an elliptical bulge. In a scatter plot, non-linearities such as the effects of friction in valves (that cause a contour like a parallelogram) can be recognized as well as quantization effects and other unusual statistical distributions.

Alternatives for determining the reference variance

During planned commissioning of a plant with integrated CPM, following controller optimization, the CPM block is initialized for every control loop and the calculated variance stored as the reference variance for calculating the CPI.

As an alternative, a reference variance can set via the RefVarExt input by setting the RefVarExtOn = 1 input. There are various ways of obtaining numeric values for the reference variance:

- Take the lowest variance that was ever measured in this control loop since the initialization of the CPM block. This is displayed at the PV_VarMin output. 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 CPM block.
- Take the variance of the control loop with a theoretical minimum variance controller as can be obtained based on archived data with an external CPM tool. 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 the value 100% is seldom reached 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, it is not therefore desirable to bring this as close as possible to 100% at any price.

Special cases cascade control and multivariable control

In a cascade control, you should only use the CPM block for the primary controller and not for the secondary controller. The CPM 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 quality should be monitored while the control quality 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.

The mathematical concept of the CPM block is intended for monovariable control loops. If the variance in a control loop is found to be too high, the block cannot recognize whether the actual cause is within the control loop or whether influences are being brought in from the outside due to interactions. If, therefore, you notice strong interactions between various control loops in your plant or even use multivariable controllers, the information provided by the CPM block should be treated with caution.

It can nevertheless make sense to equip a multivariable controller such as the MPC 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 CPM block. Several additional logic functions need to be configured upstream from the ManSuprCPI input as shown in the corresponding simulation template:

- 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, the temporarily increased variance cannot be avoided in this controller channel and should not lead to a CPI message.
- If one or more other channels of the multivariable controller has a higher variance (poor control performance) indicated by the appropriate CPI_WrnAct = 1 output, due to the interaction these variances cause a higher variance in this controller channel as well, which also 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 process gains from the monovariable observation are irrelevant. By setting the input value StatGainValid = 0, this is also displayed in the operator picture as quality "Uncertain, process related".

Calculating the quality code

The block can calculate and display several quality codes. The following inputs are available for parameter assignment:

- MV_Known: Set this input to 0 if you use a step controller without position feedback. This makes the calculated output values MV_Mean and StatGain invalid.
- StatGainValid: Set this input to 0 if you use a multivariable controller or observe strong interactions between neighboring control loops. This makes the calculated output value StatGain invalid. If known disturbances affect your process, for example, dosing procedures in a batch process, you can also set this input temporarily using recipe control.
- The quality code of the QC_CPI output depends primarily on the CPI_Suppress input: If CPI_Suppress=true, the output is invalid. Apart from this, the CPI can also become invalid in occasional situations when there are numeric problems in the calculation of the variance.

The quality code of the QC_OverAbso output 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.

Alarm delay

If the CPI temporarily exceeds 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 function, you can make sure that an alarm is triggered only after the cause exists for longer than a configured period AlmDelayTime. To do this, set the input AlmOnDelayAct = 1. If you set the AlmOffDelayAct = 1 input, the resetting of the message is also delayed, in other words, the cause must be cleared for a period longer than the configured delay before the message is marked as having exited the state.

See also

CPM I/Os (Page 51)
CPM reporting (Page 50)
Error handling of CPM (Page 49)
CPM modes (Page 40)
Description of CPM (Page 36)

3.2.4 Error handling of CPM

CPM troubleshooting

The block does not have any error numbers.

See also

CPM I/Os (Page 51)
CPM reporting (Page 50)
Functions of CPM (Page 40)
CPM modes (Page 40)
Description of CPM (Page 36)

3.2.5 CPM reporting

Message response

If the control quality 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.

Message texts and message classes

Message identifier	Message class		Event	
SIG 1	Alarm - high	yes	$$Block Comment$$ Overshoot - high alarm limit exceeded	no
SIG 2	Warning - high	yes	$$Block Comment$$ Overshoot - high warning limit exceeded	no
SIG 3	Warning - low	yes	$$Block Comment$$ Overshoot - low warning limit exceeded	no
SIG 4	Alarm - low	yes	$$Block Comment$$ Overshoot - low alarm limit exceeded	no
SIG 5	< no message >	no		no
SIG 6	< no message >	no		no
SIG 7	< no message >	no		no
SIG 8	< no message >	no		no

See also

CPM I/Os (Page 51)
Error handling of CPM (Page 49)
Functions of CPM (Page 40)
CPM modes (Page 40)
Description of CPM (Page 36)

3.2.6 CPM I/Os

CPM I/Os

Inputs

I/O (parameter)	Description	Data type	Default Value
AlmDelayTime	Alarm delay time [s]	REAL	0
AlmOffDelayAct	1 = delay for outgoing alarms active	BOOL	1
AlmOnDelayAct	1 = delay for incoming alarms active	BOOL	1
BA_ID	Current batch ID (number)	DWORD	0
BA_NA	Current batch name	STRING(32)	''
BATCH_EN	1 = activate control by means of batch	BOOL	0
BreakSuppress	Break suppression of the control quality index alarm at the end of the step response loop	BOOL	0
CPI_AlmHys	Alarm hysteresis of the control quality index [%]	REAL	5
CPI_LowAlm	Low alarm limit of the control quality index [%]	REAL	30
CPI_LowWrn	Low warning limit of the control quality index [%]	REAL	70
CpiAlm_En	1 = activate alarm for low limit	BOOL	1
CpiAlm_MsgEn	1 = activate alarm message for the low limit of the control-performance index	BOOL	1
CpiWrn_En	1 = activate warning for low limit	BOOL	1
CpiWrn_MsgEn	1 = activate warning message for the low limit of the control-performance index	BOOL	1
InitOpEn	1 = operator may initialize block	BOOL	1
InitRefVar	1 = initialization of the block. The reference variance of the process value and the reference values of process value and manipulated variable are measured in the steady state.	BOOL	0
ManSupprCPI	1 = manual suppression of the CPI calculation and message, e.g. during known disturbances	BOOL	0
MsgEvID	Message number (assigned automatically)	DWORD	16#29
MV_Known	0 = MV unknown (specifically for step controller without position feedback)	BOOL	1
MV_Mon	Manipulated variable of the controller for monitoring	REAL	0
MV_Ref	Reference value of the manipulated variable	REAL	0
OCCUPIED	Occupied by Batch	BOOL	0
OvsAlm_En	1 = activate alarm for overshoot of low limit	BOOL	1
OvsAlm_MsgEn	1 = activate alarm message if the value overshoots the low limit	BOOL	1
OvsHiAlm	Overshoot alarm limit [%]	REAL	25
OvsHiWrn	Overshoot warning limit [%]	REAL	15
OvsWrn_MsgEn	1 = activate warning message if the value undershoots the low limit	BOOL	0
PID_AutAct	1 = closed-loop controller is in auto mode	BOOL	1

I/O (parameter)	Description	Data type	Default Value
PV_Mon	Process value of the controller for monitoring	REAL	0
PV_Ref	Reference value for the process value	REAL	0
RefVarExt	Reference value in good state of the control loop	REAL	0
RefVariance	Reference value for PV_Variance in good state of the control loop	REAL	0
ReVaExOn	1 = use the external reference value of RefVarExt	BOOL	0
RunUpCyc	Number of startup cycles	INT	3
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SP_Mon	Setpoint of the corresponding closed-loop controller for monitoring	REAL	0
STEP_NO	Step number for batch	DWORD	0
StGainValid	0 = StatGain invalid, process failed	BOOL	1
TimeWindow	Width of the time window [s]	REAL	120
UserStatus	Free user area for status word (bit 24 to bit 31)	BYTE	16#0

Outputs

I/O (parameter)	Description	Data type	Default Value
CPI	Control quality index	REAL	100
CPI_AlmAct	1 = alarm of the control quality index is active	BOOL	0
CPI_Suppress	1 = message for the control performance index is suppressed; hold last valid value	BOOL	1
CPI_SuRoot	Suppression of control-loop CPI messages	BOOL	1
CPI_WrnAct	1 = message for the low limit of the control quality index is active	BOOL	0
ER_Mean	Mean value of the control error in the time window [units of PV]	REAL	0
ErrorNum	I/O for error messages	INT	-1
MsgAckn	ALARM_8P: ACK_STATE output	WORD	0
MsgErr	1 = message processing error occurred	BOOL	0
MsgStatus	ALARM_8P: STATUS output Error information of ALARM_8P	WORD	0
MV_Mean	Mean value of the manipulated variable in the time window [units of PV]	REAL	0
OverAbso	Absolute overshoot of the step response [units of PV]	REAL	0
Overshoot	Relative overshoot of the step response [%]	REAL	10
OvsAlmAct	1 = high limit overshoot alarm is active	BOOL	0
OvsWrnAct	1 = high limit overshoot warning is active	BOOL	0
PV_Mean	Mean value of the process value in the time window [units of PV]	REAL	0
PV_StdDev	Standard deviation of the process value	REAL	0
PV_Variance	Variance of the process value	REAL	0
PV_VarMin	Smallest value of the process value variance up to now	REAL	10000.0
QC_CPI	Quality code for CPI	BYTE	16#80
QC_MV_Mean	Quality code for MV_MEAN	BYTE	16#80
QC_OverAbso	Quality code for OverAbso	BYTE	16#80
QC_StatGain	Quality code for StatGain	BYTE	16#80
RefStdDev	Reference value for the standard deviation of the process value for the control loop	REAL	0
RefVariance	Reference value for the PV_Variance parameter for the control loop	REAL	0
RiseTime	Rise time of the step response [s]	REAL	0
SettliTime	Settling time of the step response [s]	REAL	0
SettlRatio	Ratio = RiseTime / SettliTime *100%	REAL	0.3
StatGain	Steady-state process gain [units of PV / MV]	REAL	1
Status	Status word	DWORD	0
StepPhase	Phase of the step response 0 = ready 1 = rising 2 = overshoot 3 = settled	INT	0

I/O (parameter)	Description	Data type	Default Value
ToleHiPV	High limit of the 3σ band around the setpoint	REAL	0
ToleLoPV	Low limit of the 3σ band around the setpoint	REAL	0

See also

CPM reporting (Page 50)
Error handling of CPM (Page 49)
Functions of CPM (Page 40)
CPM modes (Page 40)
Description of CPM (Page 36)

3.3 CTRL_PID: PID controller block

3.3.1 Description of CTRL_PID

Object name (type + number)

FB61

- CTRL_PID block I/Os (Page 68)
- CTRL_PID block icon
- CTRL_PID faceplate (Page 495)

Function

Block CTRL_PID is a continuous PID controller used for setting up the following standard controller circuits:

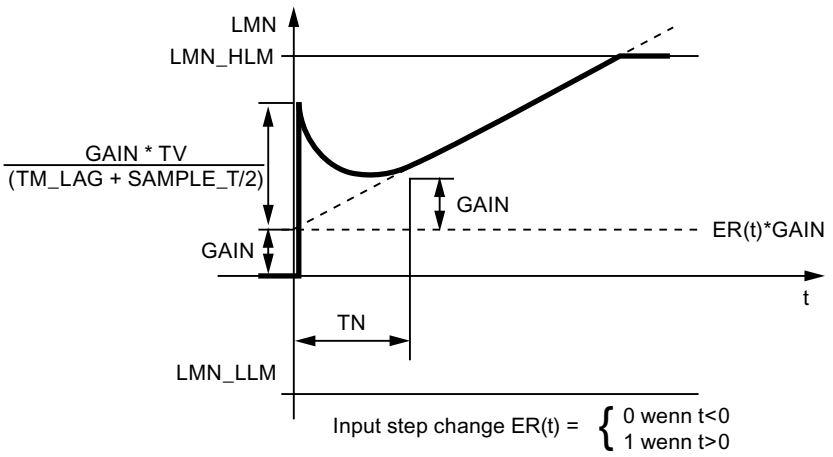
- Fixed setpoint control
- Cascade control (single/multiple cascades)
- Ratio control
- Synchronization control
- Proportional control

In addition to its actual controller functions, the controller block provides the following processing options:

- Modes: Manual mode, automatic or tracking
- Limit monitoring of the controlled variable and error signal as well as message generation via the ALARM_8P block
- Feedforward control
- Setpoint tracking (SP = PV_IN)
- Setpoint value and process value range setting (physical normalization)
- Setting the range of values for the manipulated variable (physical denormalizing)
- Dead band (response threshold) in the error signal branch
- P, I and D actions can be enabled and disabled individually
- P and D action can be placed in the feedback path
- Operating point setting for P or PD controller mode

How it works

The block operates as PID controller (with delayed derivative action) and has the step response shown below. The integrator operates according to the trapezoid rule.



Step response of CTRL_PID

Note

The input parameter LMNR_IN is displayed in the faceplate (loop display) as the manipulated variable. If there is no position feedback available from the process, you can interconnect the manipulated-variable output LMN with LMNR_IN in CFC in order to display the manipulated variable in the loop display.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Additional information

You will find more information in:

Signal processing in the setpoint and process-value branches of CTRL_PID (Page 57)

Generation of manipulated variables for CTRL_PID (Page 59)

Manual, auto, and tracking mode, and cascading CTRL_PID (Page 61)

CTRL_PID mode change (Page 63)

Error handling of CTRL_PID (Page 65)

Startup characteristics, time response, and message response of CTRL_PID (Page 66)

Block diagram of CTRL_PID (Page 67)

I/Os of CTRL_PID (Page 68)

Message texts and associated values of CTRL_PID (Page 72)

VSTATUS for CTRL_PID (Page 73)

Archiving process values (Page 613)

Operating and monitoring CTRL_PID (Page 73)

3.3.2 Signal Processing in the Setpoint and Actual-Value Branches of CTRL_PID

Setpoint generation

The setpoint SP can be obtained from three different sources. The table below defines the source used, depending on the states of the SP_TRK_ON and SPEXTSEL_OP inputs:

SP_TRK_ON	SPEXTSEL_OP	SP=	State
0	0	SP_OP	Internal setpoint
Irrelevant	1	SP_EXT	External setpoint
1	0	PV_IN **	Tracked setpoint
		** in manual mode only and when SPBUMPON = 1	

Internal setpoint

The internal setpoint SP_OP is set and limited using OP_A_LIM (Page 469) (range SP_LLM - SP_HLM).

External setpoint

The external setpoint SP_EXT can be interconnected and is limited to the range (SPEXTLLM, SPEXTHLM).

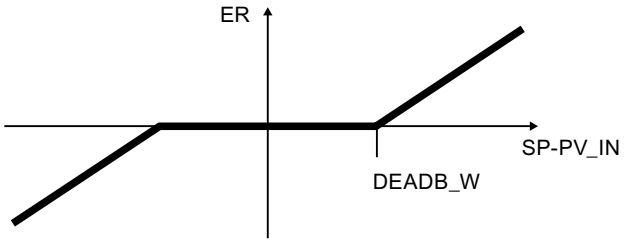
The change in the internal or external setpoint is limited to a maximum gradient (SPDRLM, SPURLM), if the setpoint ramp has been activated (SPRAMPOF = 0).

Tracked setpoint

The control variable PV_IN is used as setpoint if SP_TRK_ON = 1. Tracking the setpoint to the process value is enabled only in manual mode (with an internal setpoint and when SPBUMPON = 1), and is primarily used to provide a suitable setpoint when switching from manual to automatic mode.

Error signal generation

The error signal is generated based on the effective setpoint SP and the process value PV_IN and is available at the output ER after the deadband DEADB_W.



Error signal monitoring

The alarm limits (ERL_ALM, ERH_ALM) of error signal ER are monitored with of a common hysteresis (ER_HYS) and are indicated at the corresponding outputs (QERL_ALM, QERH_ALM).

Process value monitoring

The warning and alarm limits (PVL_ALM, PVL_WRN, PVH_WRN, PVH_ALM) of the process value PV_IN are monitored by means of a common hysteresis (HYS) and are displayed at the corresponding outputs (QPVL_ALM, QPVL_WRN, QPVH_WRN, QPVH_ALM).

Physical normalization

The error signal ER is normalized to a percentage based on the physical measuring range of the process value (NM_PVHR, NM_PVLR).

$$ER_{normalized} = \frac{ER}{NM\_PVHR - NM\_PVLR} \cdot 100$$

After the PID algorithm has been executed, the manipulated variable is converted back from a percentage to the physical measuring range of the manipulated value (NM_LMNHR, NM_LMNLR).

$$LMN = NM\_LMNLR + \frac{LMN_{normalized}}{100} \cdot (NM\_LMNHR - NM\_LMNLR)$$

The internal or external setpoint, the process value and the corresponding parameters are entered in 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 value.

The controller GAIN is specified in normalized (dimensionless) format.

3.3.3 Generation of Manipulated Variables for CTRL_PID

Manipulated variable LMN

The manipulated variable LMN can be obtained from three different sources. The table below specifies which source is used, depending on the statuses of the LMN_SEL, LIOP_MAN_SEL, AUT_L, and AUT_ON_P inputs:

LMN_SEL	LIOP_MAN_SEL	AUT_L	AUT_ON_OP	LMN=	State
0	0	X	0	MAN_OP (is limited)	Manual mode, set via the OS
0	0	X	0	MAN_OP (is limited)	Manual mode, set via the OS
0	0	X	1	Calculated by PID algorithm	Automatic mode, set via the OS
0	1	0	X	MAN_OP (is limited)	Manual mode, set when AUT_L = 0
0	1	0	X	MAN_OP (is limited)	Manual mode, set when AUT_L = 0
0	1	1	X	Calculated by PID algorithm	Automatic mode, set when AUT_L = 1
1	X	X	X	LMN_TRK	Manipulated variable tracked
X = any state					

- If LIOP_MAN_SEL = 0, the AUT_ON_OP parameter is used to change over between manual and automatic mode on the OS.
- If LIOP_MAN_SEL = 1, the AUT_L parameter is used to change over between manual and automatic mode via an interconnection in the CFC.
- Tracking mode can be enabled only over an interconnection via the LMN_SEL parameter. Tracking has priority over manual and automatic mode.

In automatic mode, the normalized manipulated variable is generated according to the following algorithm:

$$LMN_{normalized} = GAIN \bullet \left(1 + \frac{1}{TN \bullet s} + \frac{TV \bullet s}{1 + TM\_LAG \bullet s} \right) \bullet ER_{normalized}$$

s: Complex number

The manipulated variable is then denormalized.

Note

This formula describes a standard case in which the P. I and D actions are enabled, and the P and D actions are not in the feedback path. (P_SEL = TRUE, TN <> 0, PFDB_SEL = FALSE and DFDB_SEL = FALSE).
For example, if TN = 0, an offset calculated from the physical variable LMN_OFF (operating point) is added. You will find more information in "Block diagram of CTRL_PID (Page 67)".

Feedforward control and limitation

In automatic mode the disturbance variable DISV is added at the output of the PID algorithm. The result is limited to the range LMN_LLM to LMN_HLM.

3.3.4 Manual, Auto and Following mode, and Cascading of CTRL_PID

Manual mode

The manipulated variable is set by the operator at OS via the input MAN_OP. The manipulated variable is set and limited via the OP_A_LIM (Page 469) block (range MAN_HLM - MAN_LLM). The QVHL and QVLL output values of the OP_A_LIM block are passed on to the QLMN_HLM and QLMN_LLM outputs.

Automatic mode

The PID algorithm calculates the manipulated variable. The control parameters GAIN, TN, TV and TM_LAG cannot be interconnected by default. If they need to be interconnected for exceptional applications such as gain scheduling, the corresponding system attribute "s7_link" must be modified. Please note that parameter changes during automatic mode may lead to step changes in the manipulated variable.

- The effective direction of control action can be reversed (rising error signal leads to falling manipulated variable) by setting a negative proportional GAIN value. The P action can be disabled by setting P_SEL = 0. The I action can be disabled by setting TN(TI) = 0. If the manipulated variable LMN is limited in automatic mode, the integrator is set to hold (anti-wind-up). The direction of action of the integrator is reversed by inverting the sign at parameter TN(TI).
- Operating point (input LMN_OFF): This value replaces the I action of the PID algorithm in auto mode if the I action is deactivated. The operating point is entered within the measuring range of the manipulated variable.
- The D action is implemented as a delaying differential element. It can be deactivated by setting TV(TD) = 0. The direction of action of the differentiator is reversed by inverting the sign of the value at parameter TV(TD). The time lag constant TM_LAG should have a suitable relationship to the derivative time TV(TD). This ratio is also known as derivative gain (maximum of the unit step response of the D action) and normally lies within the range 5 < TV(TD)/TM_LAG < 10.
- Applying the P action to the feedback path: When PFDB_SEL = TRUE, the P action is enabled in the feedback path. A control value step change does not affect the P action, so that overshoot resulting from setpoint step changes can be reduced or avoided without changing the disturbance characteristics. The changeover of PFDB_SEL in automatic mode leads to extreme changes in the manipulated variable. It is therefore advisable to change PFDB_SEL in manual mode.
- Applying the D action to the feedback path: When DFDB_SEL = TRUE, the D action is enabled in the feedback path. In this case, a control step change does not affect the D action. The changeover of DFDB_SEL is not bumpless.

Tracking mode

In this state (LMN_SEL = 1) the manipulated variable is fetched from the interconnected tracked value LMN_TRK and set at the output. The outputs QLMN_HLM and QLMN_LLM are set to FALSE. The "Track" mode takes priority over all other modes, so that you can use this input to configure a safety shutdown circuit for the plant.

Placing the D and P action in the feedback path

The P and D actions can be located in the feedback path to reduce or avoid overshoot of the process value as a result of a setpoint step change. The setpoint step change no longer has any effect on the P or D action in this mode. The manipulated value no longer responds with a step change after a setpoint step change. The P action is enabled in the feedback path by setting PFDB_SEL = 1. The D action is enabled in the feedback path by setting DFDB_SEL = 1.

The feedback of the P action is ignored if TI = 0.

Cascading PID controllers

The manipulated variable LMN of the primary controller is connected to input SP_EXT of the secondary controller. The primary controller must also change to tracking mode if the cascade is opened. A QCAS_CUT signal is generated in the secondary controller and interconnected with LMN_SEL input of the primary controller. This disconnection can be caused in manual or tracking mode by setpoint input, or by setpoint tracking of the secondary controller:

QCAS_CUT = NOT(QMAN_AUT) OR LMN_SEL OR SP_TRK_ON OR NOT(QSPEXT_ON)

The tracking input LMN_TRK of the primary controller is interconnected with the SP output of the secondary controller to prevent a step change when the cascade is closed again.

It is also advisable to set a directional lock of the integrator in the primary controller which is activated as soon as the secondary controller reaches a manipulated variable limit. With a positive control action, interconnect the INT_HPOS and INT_HNEG inputs of the primary controller with the QLMN_HLM and QLMN_LLM outputs of the secondary controller.

3.3.5 CTRL_PID Mode Change

Mode change

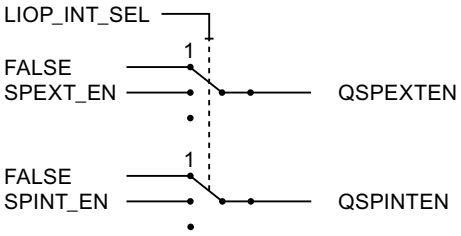
The mode change can be triggered either by the operator or via interconnected inputs.

External/internal setpoint

A changeover between external and internal setpoint is initiated by the OS operator setting the SPEXTSEL_OP input by interconnecting SPEXON_L. You set the corresponding enable inputs SPINT_EN, SPEXT_EN or the selection input LIOP_INT_SEL to enable these changeovers.

If SPBUMPON = 1, the internal setpoint is replaced with the effective setpoint in order to allow a bumpless changeover from external or tracking mode to internal mode.

Enabling changeover between internal and external setpoint



QSPEXTEN = TRUE: SPEXTSEL_OP can be changed from FALSE (internal setpoint) to TRUE (external setpoint).

QSPINTEN = TRUE: SPEXTSEL_OP can be changed from TRUE (external setpoint) to FALSE (internal setpoint).

SPEXTSEL_OP tracks or is reset as required.

Enabling setpoint control via the operator input SP_OP_ON

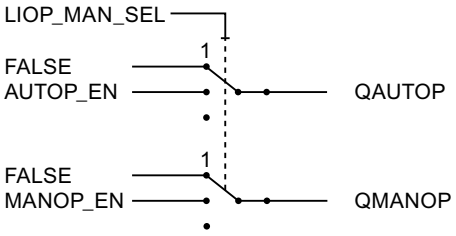
Q_SP_OP = TRUE: The setpoint SP_OP can be adjusted.

The setpoint SP_OP is tracked or reset as required.

Manual/auto

Changeover between manual and auto mode is initiated by the OS operator at input AUT_ON_OP or by interconnecting AUT_L. You set the corresponding enable inputs MANOP_EN, AUTOP_EN or the selection input LIOP_MAN_SEL to enable this changeover.

Enabling the changeover between manual and automatic mode

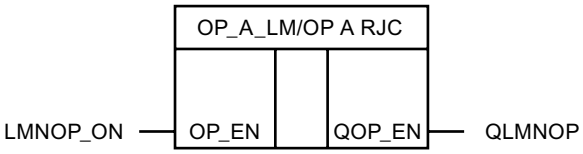


QAUTOP = TRUE: AUT_ON_OP can be changed from FALSE (manual mode) to TRUE (automatic mode).

QMANOP = TRUE: AUT_ON_OP can be changed from TRUE (automatic mode) to FALSE (manual mode).

AUT_ON_OP tracks or is reset as required.

Enabling manual value operator control via the operator control input



QLMNOP = TRUE: MAN_OP can be set.

MAN_OP is tracked or reset as required.

Special measures are applied for the modes listed below in order to ensure a bumpless changeover:

- Setpoint external/setpoint tracking: If SPBUMPON = TRUE, the internal setpoint SP_OP is set to the effective (external or tracked) setpoint.
- Automatic mode: The manual value MAN_OP is tracked to the effective manipulated variable.
- Tracking mode: The manual value MAN_OP is tracked to the effective manipulated variable.
- Manual or tracking mode: The integrator is tracked to allow a bumpless changeover to automatic mode.

  Action = value of manipulated variable (percentage) minus P action minus the disturbance variable (percentage)

  Notice: When this formula is applied, the integrator may be loaded with extremely high numeric values if at the time of changeover there is a maverick measured value, in other words, there is an extremely high P action. Additional measures have been implemented as of V6.0 to allow flexible limiting of the I action.

  The D action component is disabled and compensated.

3.3.6 Error Handling of CTRL_PID

Operator input error

QOP_ERR = 1 is set if at least one operator error occurs when changing one of the parameters SPEXTSEL_OP, AUT_ON_OP, SP_OP, or MAN_OP. Otherwise QOP_ERR = 0 is set. An operator error exists only for one cycle.

The block algorithm handles the following situations:

- Parameter assignment error NM_PVHR <= NM_PVLR:

  The error signal ER is set to zero and ENO = 0 or QERR = 1 is set.

- NM_LMNHR <= NM_LMNLR:

  In automatic mode, the disturbance variable will be output and ENO=0 or QERR=1 is set.

- Absolute value (TN) < SAMPLE_T/2:

  For TN(TI) > 0 is calculated with TN(TI) = SAMPLE_T/2, for TN(TI) < 0 with TN(TI) = -SAMPLE_T/2.
  For TN(TI) = 0 the integrator is shut down and the operating point LMN_OFF is active.

- Absolute value (TV) < SAMPLE_T:

  When TV(TD) > 0, TV(TD) = SAMPLE_T is used for calculation, when TV(TD) < 0, TV(TD) = -SAMPLE_T is used.
  When TV(TD) = 0, the differentiator is shut down.

- TM_LAG < SAMPLE_T/2:

  When TM_LAG < SAMPLE_T/2, TM_LAG = SAMPLE_T/2 is used internally for calculation. In these cases, the D action acts as an ideal differentiator.

3.3.7 Startup Behavior, Dynamic Response and Message response of CTRL_PID

Startup characteristics

During CPU startup, CTRL_PID is set to manual mode with an internal setpoint. This means that the block must be called in the startup OB. In CFC engineering, this is handled by the CFC. When using basic STEP 7 tools, you enter this call in the startup OB.
Following startup, messages will be suppressed during the number of cycles set at RUNUPCYC.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SAMPLE_T parameter.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for CTRL_PID (Page 73)".

Message response

The CTRL_PID block uses the ALARM_8P block for generating messages.

The following message triggers exist:

- The functions for monitoring the process value and system deviation limits
- The CSF signal that is received as a control system error via the interconnection

Limit violation messages can be suppressed individually using the the relevant M_SUP_xx inputs. Process messages (not control system messages!) can be disabled centrally using MSG_LOCK.

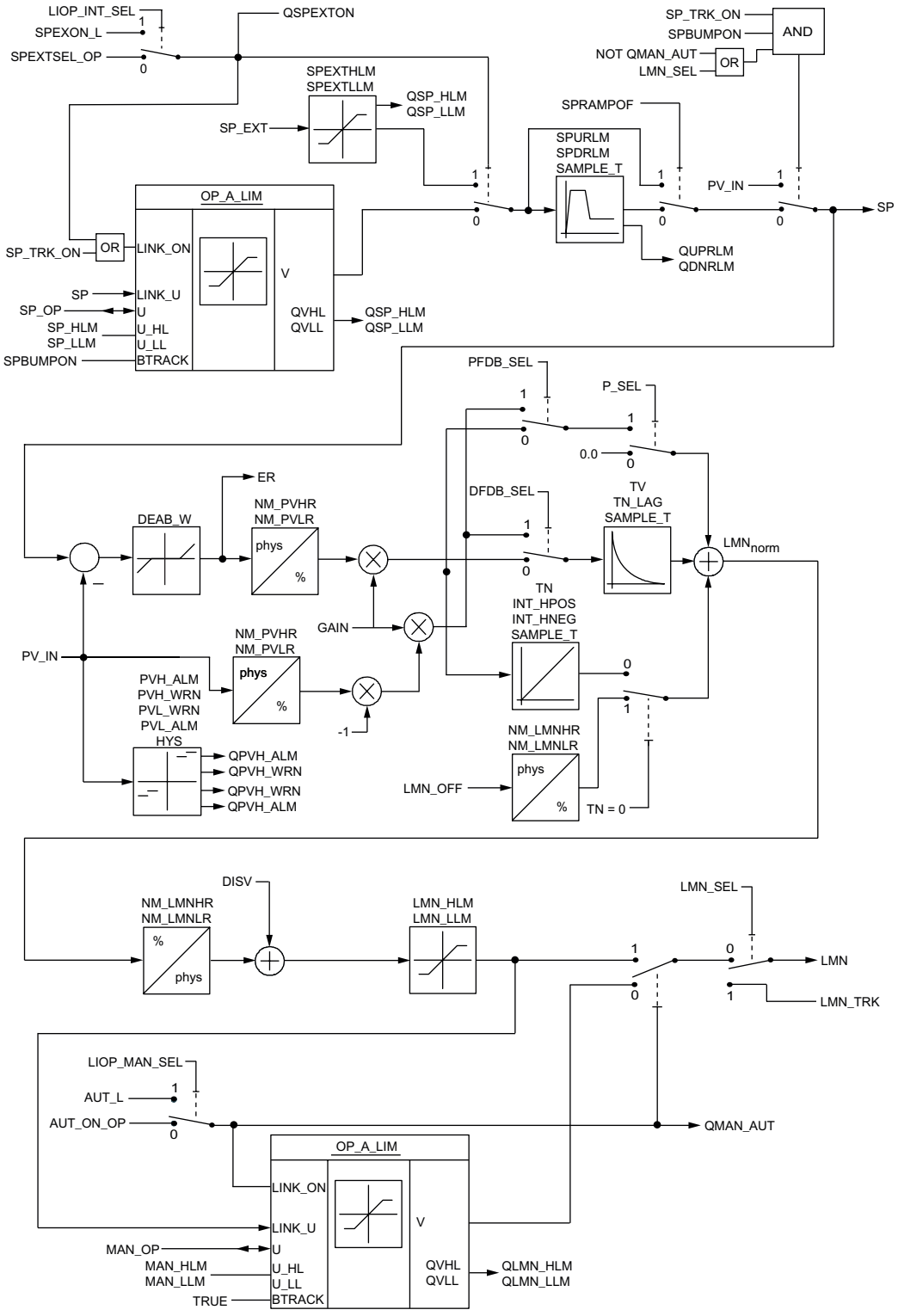
QMSG_SUP is set if the RUNUPCYC cycles have not expired yet since a restart, MSG_LOCK = TRUE or MSG_STAT = 21.

The table lists the message texts (Page 72) of the CTRL_PID block and their assignment to the block parameters.

Monitoring process values

Not available

3.3.8 Block Diagram of CTRL_PID



3.3.9 I/Os of CTRL_PID

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
AUT_L	Interconnectable input for MANUAL/AUTO: (0=Manual/1=Auto)	BOOL	0	I		
AUT_ON_OP	Operator input: 0 = manual; 1 = auto	BOOL	0	IO	+	
AUTOP_EN	1 = operator control enable for Auto	BOOL	1	I		
AUX_PRx	Associated value x	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: Consecutive batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	''	I	+	
CSF	Control system error	BOOL	0	I		
DEADB_W	Dead band	REAL	0	I	+	>= 0
DFDB_SEL	Place D action in feedback (1: active)	BOOL	0	I		
DISV	Disturbance variable	REAL	0	I		
ER	Error signal	REAL	0	O	+	
ER_HYS	Hysteresis for monitoring the error signal	REAL	0.1	I	+	>= 0
ERH_ALM	Error signal: High alarm limit	REAL	100	I	+	> DEADBW
ERL_ALM	Error signal: Low alarm limit	REAL	-100	I	+	< -DEADBW
GAIN	Proportional gain	REAL	1	I	+	
HYS	Hysteresis	REAL	5	I	+	>= 0
INT_HNEG	Freeze I action (neg. direction); 1 = active	BOOL	0	I		
INT_HPOS	Freeze I action (pos. direction); 1 = active	BOOL	0	I		
LIOP_INT_SEL	1 = interconnection active, 0 = operator control active	BOOL	0	I		
LIOP_MAN_SEL	1 = interconnection active, 0 = operator control active	BOOL	0	I		
LMN	Manipulated value output	REAL	0	O		
LMN_HLM	High limit for value of manipulated variable	REAL	100	I	+	LMN_HLM > LMN_LLM
LMN_LLM	Low limit for value of manipulated variable	REAL	0	I	+	LMN_LLM < LMN_HLM
LMN_OFF	Operating point	REAL	0	I	+	
LMN_SEL	External manipulated variable; 1 = active	BOOL	0	I		
LMN_TRK	External manipulated variable	REAL	0	I		
LMNOP_ON	1 = operator control enable for value of manipulated variable MAN_OP	BOOL	1	I		
LMNR_IN	Position feedback for display on OS	REAL	0	I		
M_SUP_AH	1 = message suppression high limit alarm process value	BOOL	0	I	+	

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
M_SUP_AL	1 = message suppression low limit alarm process value	BOOL	0	I	+	
M_SUP_ER	Suppression of messages for error signal alarms	BOOL	1	I	+	
M_SUP_WH	1 = message suppression: high warning process value	BOOL	0	I	+	
M_SUP_WL	1 = message suppression low limit warning process value	BOOL	0	I	+	
MAN_HLM	High limit of manual value manipulated variable	REAL	100	I	+	
MAN_LLM	Low limit of manual value manipulated variable	REAL	0	I	+	
MAN_OP	Operator input for manipulated variable	REAL	0	IO	+	
MANOP_EN	1 = operator control enable for manual mode	BOOL	1	I		
MO_PVHR	High display limit (measuring range)	REAL	110	I	+	
MO_PVLR	Low display limit (measuring range)	REAL	-10	I	+	
MSG_ACK	Acknowledge messages	WORD	0	O		
MSG_EVID	Message number	DWORD	0	I		
MSG_LOCK	1 = message suppression dependent on process state	BOOL	0	I	+	
MSG_STAT	Error message status	WORD	0	O		
NM_LMNHR	High normalization limit of manipulated variable (measuring range)	REAL	100	I		
NM_LMNLR	Low normalization limit of manipulated variable (measuring range)	REAL	0	I		
NM_PVHR	High normalization limit of process value (measuring range)	REAL	100	I		
NM_PVLR	Low normalization limit of process value (measuring range)	REAL	0	I		
OCCUPIED	ID for occupied by BATCH	BOOL	0	I	+	
OOS	Reserve	BOOL	0	I	+	
OPTI_EN	1 = controller tuning on, 0 = off	BOOL	0	I	+	
P_SEL	Activate P action (1: active)	BOOL	1	I		
PFDB_SEL	Place P action in feedback; 1 = active	BOOL	0	I		
PV_IN	Process value	REAL	0	IO	+	
PVH_ALM	Process value: High alarm limit	REAL	100	I	+	PVH_ALM > PVL_ALM
PVH_WRN	Process value: High warning limit	REAL	95	I	+	PVH_WRN > PVL_WRN
PVL_ALM	Process value: Low alarm limit	REAL	0	I	+	PVL_ALM < PVH_ALM
PVL_WRN	Process value: Low warning limit	REAL	5	I	+	PVL_WRN < PVH_WRN
Q_SP_OP	1: Operator control enable for setting setpoint	BOOL	0	O	+	

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
QAUT_OP	1 = operator control enable for automatic mode	BOOL	0	O	+	
QC_LMN	Quality code for LMN	BYTE	16#80	O		
QC_LMN_I	Quality code for output LMN	BYTE	16#80	I		
QC_LMNR_IN	Quality code for LMNR_IN	BYTE	16#80	I		
QC_PV_IN	Quality code for PV_IN	BYTE	16#80	I		
QCAS_CUT	1 = cascade opened	BOOL	1	O		
QDNRLM	1 = negative setpoint ramp limited	BOOL	0	O		
QERH_ALM	Error signal, 1 = high limit alarm	BOOL	0	O	+	
QERL_ALM	Error signal, 1 = low limit alarm	BOOL	0	O	+	
QERR	1 = error output (inverted ENO)	BOOL	1	O	+	
QLMN_HLM	1 = high value of manipulated-variable output limited	BOOL	0	O	+	
QLMN_LLM	1 = low value of manipulated-variable output limited	BOOL	0	O	+	
QLMNOP	1 = operator-control enable for setting manipulated variable	BOOL	0	O	+	
QMAN_AUT	0 = manual, 1 = auto	BOOL	0	O	+	
QMANOP	1 = operator-control enable for manual mode	BOOL	0	O	+	
QMSG_ERR	1 = message error	BOOL	0	O		
QMSG_SUP	1 = message suppression	BOOL	0	O	+	
QOP_ERR	1 = group operator input error	BOOL	0	O		
QPVH_ALM	1 = high alarm	BOOL	0	O		
QPVH_WRN	1 = high warning	BOOL	0	O		
QPVL_ALM	1 = low alarm	BOOL	0	O		
QPVL_WRN	1 = low warning	BOOL	0	O		
QSP_HLM	1 = high setpoint output limited	BOOL	0	O		
QSP_LLM	1 = low setpoint output limited	BOOL	0	O		
QSPEXTEN	1 = operator control enable for external	BOOL	0	O	+	
QSPEXTON	0 = internal, 1 = external	BOOL	0	O	+	
QSPINTEN	1 = operator control enable for internal	BOOL	0	O	+	
QUPRLM	1 = positive setpoint ramp limited	BOOL	0	O		
RUNUPCYC	Number of run-up cycles	INT	3	I		
SAMPLE_T	Sampling time in seconds	REAL	1	I		>= 0.001
SP	Active setpoint	REAL	0	O	+	
SP_EXT	External setpoint	REAL	0	I		
SP_HLM	Setpoint high limit	REAL	100	I	+	SP_HLM > SP_LLM
SP_LLM	Setpoint low limit	REAL	0	I	+	SP_LLM < SP_HLM
SP_OP	Operator input for setpoint	REAL	0	IO	+	
SP_OP_ON	Operator control enable for setpoint SP_OP	BOOL	1	I		

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
SP_TRK_ON	1 = track setpoint SP_OP	BOOL	0	I	+	
SPBUMPON	1 = bumpless setpoint	BOOL	1	I	+	
SPDRLM	Max. negative setpoint ramp rate [1/s]	REAL	100	I	+	
SPEXON_L	Interconnectable input for internal/external: 0 = internal, 1 = external	BOOL	0	I		
SPEXT_EN	1 = operator control enable for external setpoint	BOOL	1	I		
SPEXTHLM	High limit of external setpoint	REAL	100	I		SPEXTHLM > SPEXTLLM
SPEXTLLM	Low limit of external setpoint	REAL	0	I		SPEXTLLM < SPEXTHLM
SPEXTSEL_OP	Operator input: 0 = internal, 1 = external	BOOL	0	IO	+	
SPINT_EN	1 = operator-control enable for internal	BOOL	1	I		
SPRAMPOF	1 = deactivate setpoint ramp limitation	BOOL	1	I	+	
SPURLM	Max. positive setpoint ramp rate [1/s]	REAL	100	I	+	
STEP_NO	BATCH step number	DWORD	0	I	+	
TM_LAG	Time lag of D action in seconds	REAL	1	I	+	≥±SAMPLE_T/2
TN	Reset time in seconds	REAL	10	I	+	TN=0, ≥±SAMPLE_T/2
TV	Derivative time in seconds	REAL	0	I	+	TV=0, ≥±SAMPLE_T
USTATUS	Status word in VSTATUS; can be configured user-specific	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.3.10 Message Texts and Associated Values of CTRL_PID

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QPVH_ALM	PV:$$BlockComment$$ HighHigh alarm	AH	M_SUP_AH, MSG_LOCK
2	QPVH_WRN	PV:$$BlockComment$$ High alarm	WH	M_SUP_WH, MSG_LOCK
3	QPVL_WRN	PV:$$BlockComment$$ Low alarm	WL	M_SUP_WL, MSG_LOCK
4	QPVL_ALM	PV:$$BlockComment$$ LowLow alarm	AL	M_SUP_AL, MSG_LOCK
5	CSF	$$BlockComment$$ External fault	S	-
6	QERH_ALM	ER:$$BlockComment$$ HighHigh alarm	AH	M_SUP_ER, MSG_LOCK
7	QERL_ALM	ER:$$BlockComment$$ LowLow alarm	AL	M_SUP_ER, MSG_LOCK

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data, the fourth is reserved for PV_IN and the remaining ones (AUX_PRx) can be freely assigned by the user.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	PV_IN
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR010

3.3.11 VSTATUS for CTRL_PID

The 32-bit status word extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	-	-	-	QSPEXTON	QMAN_AUT	MSG_LOCK	BA_EN	OCCUPIED

Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	LMN_SEL	-	-	-	-	-

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.3.12 Operating and Monitoring of CTRL_PID

Additional information

You will find more information in the following sections:

- CTRL_PID block icon (Page 582)
- CTRL_PID faceplate (Page 495)

3.4 CTRL_S: PID step controller block

3.4.1 Description of CTRL_S

Object name (type + number)

FB76

- CTRL_S block I/Os (Page 91)
- CTRL_S block icon (Page 580)
- CTRL_S faceplate (Page 613)

Function

The CTRL_S controller block is a step controller for process control systems in which integral action actuators (for example, motor-driven valves) are used. The valves are controlled by two binary control signals.

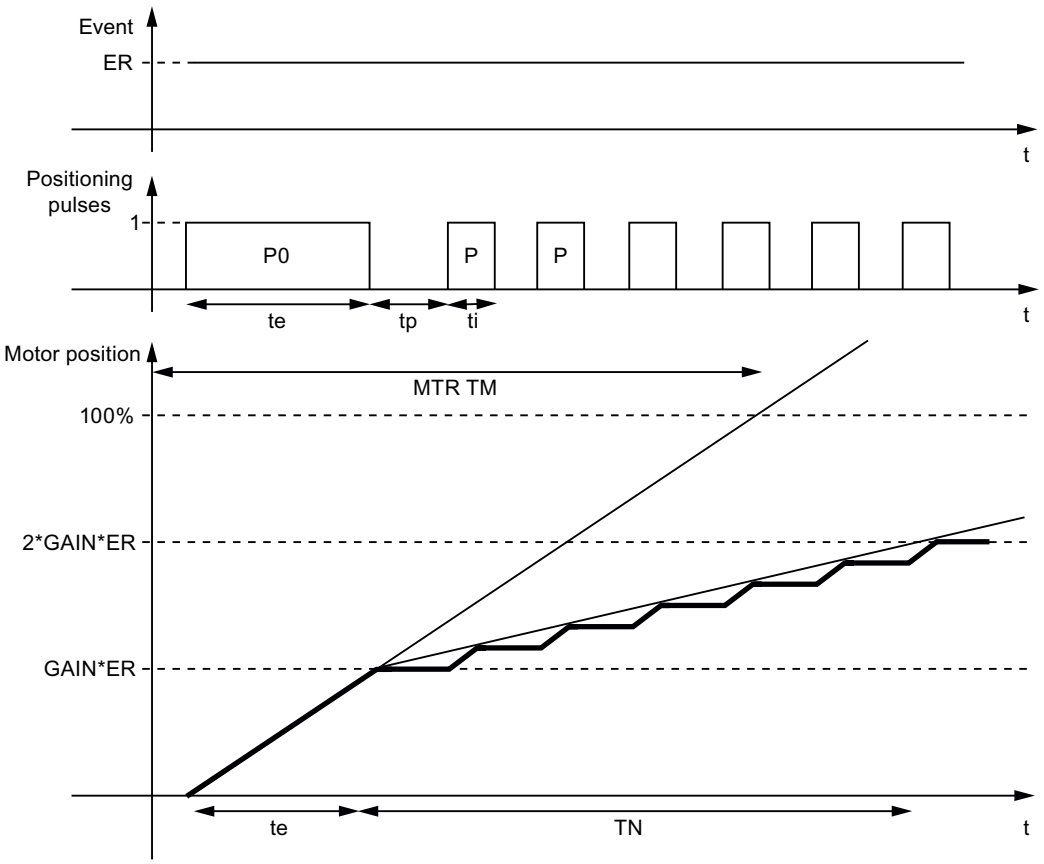
The operating principle of the step controller is based on a combination of the PID algorithm of a sampling controller and a downstream position controller. The continuous control signal is converted into a sequence of control pulses.

With suitable parameter settings, the following sub-functions of the PID algorithm can be enabled or disabled so that they are adapted to the controlled system:

- Modes: Manual mode, automatic or tracking
- Monitoring limits of the controlled variable and error signal as well as message generation via the ALARM_8P block
- Feedforward control
- Setpoint tracking (SP = PV_IN)
- Setpoint value and process value range setting (physical normalization)
- P, I and D actions can be enabled and disabled individually
- P and D action can be applied to the feedback path (P action only in step controllers with position feedback).
- Operating point setting for P or PD controller mode
- The downstream positioning controller allows the following applications:
  - Controlling with position feedback signal
  - Controlling without position feedback signal
  - Direct signal adjustment by operator or interconnected signals
- Suppression of the control signals based on corresponding feedback signals from the motor (motor protection) or valve (limit stop signals)
- Reduction of the number of control pulses by means of adaptive response threshold

How it works

PI step controllers are commonly used applications. In this mode, the step response of the controller is as follows:



Designations:

P0	Starting pulse
P	Sequential pulse
t0	Starting instant
te	Duration of the starting pulse
ti	Pulse duration (= PULSE_TM)
tp	Pause duration (depends on the parameter assignment, does therefore not correspond to BREAK_TM)

Note

The faceplate (loop display) is displayed as the effective manipulated variable of input parameter LMNR_IN. The position feedback is interconnected to this parameter.

Control input LMNR_ON can be used to set this variable in the control algorithm. If LMNR_ON = 0, the controller operates without position feedback.

If oscillation develops in the control valve action due to a lag in position feedback, use the step controller without position feedback and switch the feedback to input PV_IN. You can compensate the dead time response of the control valve with the PI parameters GAIN and TN. Use CTRL_PID as the primary controller (see also cascade control) for controlling the actual process.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Additional information

You will find more information in:

Signal processing in the setpoint and process value branches of CTRL_S (Page 77)

Generation of control signals for CTRL_S (Page 79)

Manual, auto and tracking mode, and cascading CTRL_S (Page 82)

CTRL_S mode change (Page 85)

Error handling of CTRL_S (Page 87)

Startup behavior, dynamic response and message response of CTRL_S (Page 88)

Block diagram of CTRL_S (Page 89)

I/Os of CTRL_S (Page 91)

Message texts and associated values of CTRL_S (Page 96)

VSTATUS for CTRL_S (Page 97)

Archiving process values (Page 613)

Operating and monitoring CTRL_S (Page 503)

See also

Operating and Monitoring of CTRL_S (Page 97)

3.4.2 Signal Processing in the Setpoint and Actual-Value Branches of CTRL_S

Setpoint generation

The setpoint SP can be obtained from three different sources. The table below defines the source used, depending on the states of the SP_TRK_ON and SPEXTSEL_OP inputs:

SP_TRK_ON	SPEXTSEL_OP	SP=	State
0	0	SP_OP	Internal setpoint
Irrelevant	1	SP_EXT	External setpoint
1	0	PV_IN **	Tracked setpoint
		** in manual mode only and when SPBUMPON = 1	

Internal setpoint

The internal setpoint SP_OP is set and limited using OP_A_LIM (Page 469) (range SP_LLM - SP_HLM).

External setpoint

The external setpoint SP_EXT can be interconnected and is limited to the range (SPEXTLLM, SPEXTHLM).

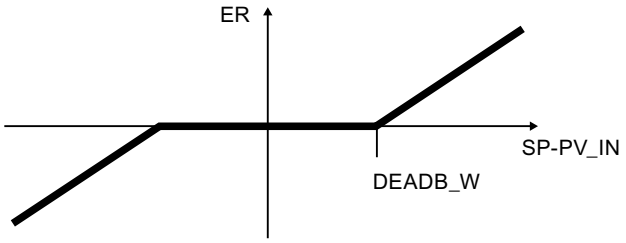
Changes to the internal or external setpoint are limited to a maximum gradient (SPDRLM, SPURLM) if the setpoint ramp is activated (SPRAMPOF = 0).

Tracked setpoint

The control variable PV_IN is used as setpoint if SP_TRK_ON = 1. Setpoint tracking of the process value is only active in manual mode (with internal setpoint and if SPBUMPON = 1) and is primarily used to provide a suitable setpoint when changing from manual to auto mode.

Error signal generation

The error signal is generated based on the effective setpoint SP and the process value PV_IN and is available at the output ER after the deadband DEADB_W.



Error signal monitoring

The alarm limits (ERL_ALM, ERH_ALM) of error signal ER are monitored with of a common hysteresis (ER_HYS) and are indicated at the corresponding outputs (QERL_ALM, QERH_ALM).

Process value monitoring

The warning and alarm limits (PVL_ALM, PVL_WRN, PVH_WRN, PVH_ALM) of the process value PV_IN are monitored by means of a common hysteresis (HYS) and are displayed at the corresponding outputs (QPVL_ALM, QPVL_WRN, QPVH_WRN, QPVH_ALM).

Physical normalization

The error signal ER is normalized to a percentage based on the physical measuring range of the process value (NM_PVHR, NM_PVLR).

$$ER_{normalized} = \frac{ER}{NM\_PVHR - NM\_PVLR} * 100$$

The internal or external setpoint, the process value and the corresponding parameters are entered in the physical measuring range of the process value.

The operating range of the valve is normalized to 0 to 100. Manual values, the tracking value of the manipulated variable, and the feedforward are entered as percentages.

The controller GAIN is specified in normalized (dimensionless) format.

3.4.3 Generation of Control Signals for CTRL_S

Control signals

Control signals can be generated from various sources. The tables below specify which source is used, depending on the statuses of the control inputs:

No.	How it works	Setpoint internal/ external	Position feedback	Effective source	Tracked signals	Remark
				Generally: Depending on the setting, LMN_OFF and DISV can also be effective in automatic mode.	Generally: SP_OP tracks only when SPBUMPON=1.	
2	Startup/restart	-	-	Startup values	Startup values	
4	Manual/tracking	-	Yes	MAN_OP	SP_OP=SP_EXT/ PV_IN	Operator control of manipulated variable
8			Yes	LMN_TRK	SP_OP=SP_EXT/ PV_IN MAN_OP=LMNR_IN	Track external manipulated variable
5			-	LMNUP_OP/LMNDN_OP	SP_OP=SP_EXT/ PV_IN MAN_OP= LMNR_IN	Operator control of the control signal on OS
6			-	LMNUP/LMNDN	SP_OP=SP_EXT/PV_IN MAN_OP = LMNR_IN	Direct signal adjustment via interconnection
9	Automatic	Internal	-	SP_OP, control signal from PID algorithm	MAN_OP = LMNR_IN	Setpoint from OS
12				SP_EXT, control signal from PID algorithm	SP_OP=SP_EXT MAN_OP = LMNR_IN	Interconnected setpoint

How it works	Tracking		Manual				Automatic					
Source	LMN_TRK	LMNUP/ LMNDN	MAN_OP		LMNUP_OP/ LMNDN_OP		SP_EXT		SP_OP		PV_IN	
Internal/external	External	External	Ext.	Int.	Ext.	Int.	Ext.	Int.	Ext.	Int.	Ext.	Int.
Control inputs:												
AUT_L	-	-	0	-	0	-	1	-	1	-	1	-
AUT_ON_OP	-	-	-	0	-	0	-	1	-	1	-	1
AUTOP_EN	-	-	-	-	-	-	-	(1)	-	(1)	-	-
MANOP_EN	-	-	-	(1)	-	(1)	-	-	-	-	-	-
LIOP_MAN_SEL	-	-	1	0	1	0	1	0	1	0	1	0
SPEXON_L	-	-	-	-	-	-	1	-	0	-	-	-
SPEXTSEL_OP	-	-	-	-	-	-	-	1	-	0	-	-
SPINT_EN	-	-	-	-	-	-	-	-	-	(1)	-	-
SPEXT_EN	-	-	-	-	-	-	-	(1)	-	-	-	-
LIOP_INT_SEL	-	-	-	-	-	-	1	0	1	0	-	-

How it works	Tracking		Manual				Automatic					
LMN_SEL	1	-	0	0	0	0	0	0	0	0	0	0
LMNS_ON	0	1	0	0	0	0	0	0	0	0	0	0
SP_OP_ON	-	-	-	-	-	-	-	-	1	1	-	-
LMNOP_ON	-	-	1	1	-	-	-	-	-	-	-	-
LMNSOPON	-	-	(2)	(2)	1	1	-	-	-	-	-	-
LMNR_ON	1	-	1	1	-	-	-	-	-	-	-	-
SP_TRK_ON	-	-	-	-	-	-	0	0	0	0	1	1

Explanations:

External: The setpoint is program-controlled and is set via interconnected inputs or by configuration.

Internal: The setpoint is set via operator control on the OS

"-": Random status

(1): The setting is only checked when there is a changeover at the OS.

(2): Operator control of the control signal must not be active. (= not (LMNSOPON and (LMNUP_OP xor LMNDN_OP))) Operator control of the control signal via LMNUP_OP or LMNDN_OP takes priority over operator control of the manipulated variable via MAN_OP.

The analog manipulated variable of the PID algorithm is generated as follows:

$$LMN_{unlimited} = GAIN \bullet \left(1 + \frac{1}{TN \bullet s} + \frac{TV \bullet s}{1 + TM\_LAG \bullet s} \right) \bullet ER_{normalized}$$

s: Complex number

Feedforward control and limitation

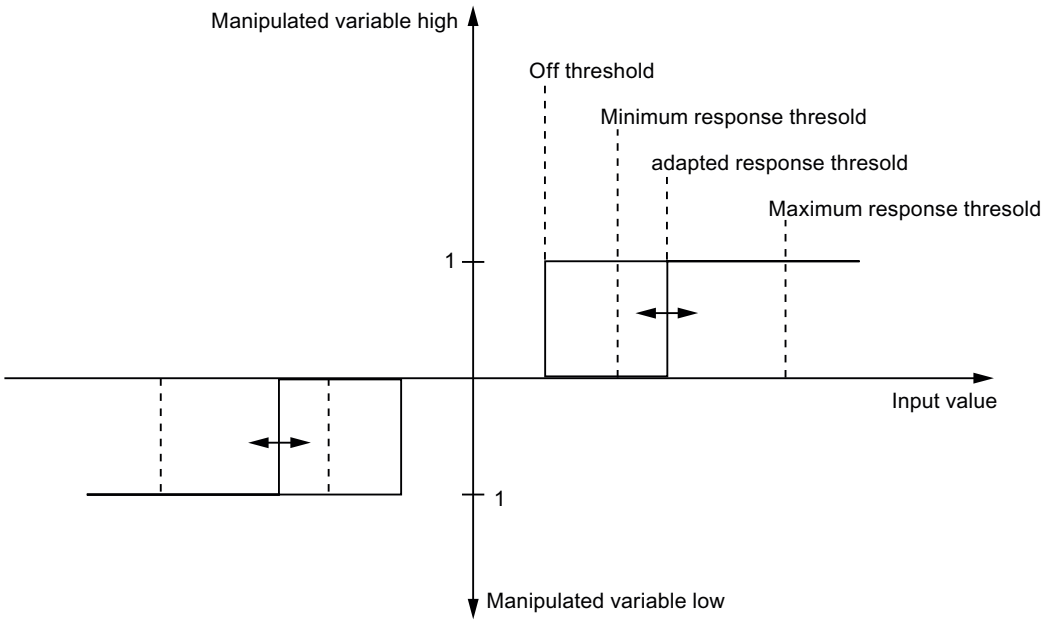
In automatic mode the disturbance variable DISV is added at the output of the PID algorithm. The result is limited to the range LMN_LLM to LMN_HLM.

The three-step element with threshold adaptation

The three-point element has an on and off threshold, which can be adapted by the block to a value between.

Note

This formula describes a standard case in which the P, I and D actions are active, and the P and D actions are not in the feedback path. (P_SEL = TRUE, TN <> 0, PFDB_SEL = FALSE and DFDB_SEL = FALSE).
For example, if TN = 0, an offset calculated from the physical variable LMN_OFF (operating point) is added. You will find more information about this in "Block diagram of CTRL_PID (Page 89)".



Off threshold: 55.0 / MTR_TM * SAMPLE_T

Minimum response threshold: 100.0 / MTR_TM * Maximum(PULSE_TM, SAMPLE_T)

Maximum response threshold: 10.0

To reduce the frequency of switching when compensating larger errors, the response threshold is adapted automatically during operation.

Adaptation of the response threshold can be switched off by setting THRADA_ON = FALSE. This sets the response threshold to the minimum value.

The currently effective response threshold can be monitored at the THRESON output.

The response threshold is set to minimum in the following situations (also if THRADA = TRUE):

- Manual or tracking mode of step controllers without position feedback
- Controllers I action (TN = 0 s)

3.4.4 Manual, Auto and Following Mode, and Cascading of CTRL_S

Manual mode

Manual mode provides you with three ways of influencing the control signals:

- Manipulated variable input with MAN_OP
- Jogging mode of MAN_OP
- Direct activation of control signals using control commands

Operation of MAN_OP by means of manipulated variable input or jogging is only possible in control systems with position feedback (LMNR_ON = 1). It is set and limited via OP_A_LIM (Page 469) (range MAN_HLM - MAN_LLM). The QVHL and QVLL output values of OP_A_LIM are passed on to the QLMN_HLM and QLMN_LLM outputs. The value of MAN_OP is set at output LMN and the motor is driven by means of the control signals until the value of position feedback LMNR_IN has reached the value of MAN_OP.

However, direct operation of the control signals by means of actuating commands is also possible in control systems operating with or without position feedback. Signal manipulation is enabled with LMNSOPON; the control signals are set at LMNUP_OP or LMNDN_OP. The valve is adjusted until operator input is canceled or the end position is reached.

Control signal input takes priority over manipulated variable input. If MAN_OP is not the effective input signal, the value is tracked to LMNR_IN.

Automatic mode

The PID algorithm calculates the manipulated variable LMN. The control parameters GAIN, TN, TV and TM_LAG cannot be interconnected by default. In order to interconnect these parameters in exceptional situations, for example, for gain scheduling, you must change the corresponding system attribute "s7_link". Note that parameter changes may lead to step changes in the manipulated variable if automatic mode is active:

- The effective direction of control action can be reversed (rising error signal leads to falling manipulated variable) by setting a negative proportional GAIN value. The P action can be disabled by setting P_SEL = 0. The I action can be disabled by setting TN(TI) = 0. The I action of control systems with position feedback can be deactivated for a particular direction by interconnecting input INTH_POS or INTH_NEG.
- Within the scope of the anti-windup measures, the integrator is automatically stopped when the limit switch is tripped LMNR_HS or LMNR_LS or if the position feedback LMNR_IN is active and exits the LMN_HLM, LMN_LLM range.
- Operating point (input LMN_OFF): This value replaces the I action of the PID algorithm in auto mode if the I action is deactivated. The operating point is entered within the measuring range of the manipulated variable.
- The D action is implemented as a delaying differential element. It can be deactivated by setting TV(TD) = 0. The time lag constant TM_LAG should have a suitable relationship to the derivative time TV(TD). This ratio is also known as derivative gain (maximum of the unit step response of the D action) and normally lies within the range 5 < TV(TD)/TM_LAG < 10.
- Activating the P action in the feedback path of the closed-loop controller: The P action is enabled in the feedback path of the closed-loop controller by setting PFDB_SEL = TRUE. A control value step change does not affect the P action, so that overshoot resulting from setpoint step changes can be reduced or avoided without changing the disturbance

characteristics. The changeover of PFDB_SEL in automatic mode leads to extreme changes in the manipulated variable. It is therefore advisable to change PFDB_SEL in manual mode.

- Enabling the D action in the feedback path of the closed-loop controller: The D action is enabled in the feedback path of the closed-loop controller by setting DFDB_SEL = TRUE. In this case, a control step change does not affect the D action. The changeover of DFDB_SEL is not bumpless.

The calculated manipulated variable is converted into a sequence of actuating pulses. The algorithm for generating these actuating pulses is influenced by the following parameters:

- MTR_TM: Motor runtime = time required to cover the maximum travel distance of the valve.
- PULSE_TM: Minimum pulse duration; the smallest step distance that the valve can travel is 100%*PULSE_TM/MTR_TM.
- BREAK_TM: Minimum break time; after an actuating pulse has terminated, this interval must expire before a new pulse can be output.
- LMNR_HS, LMNR_LS: Limit switches; if one of the limit switches is set, the corresponding output signal QLMNUP or QLMNDN is disabled.
- A negative edge of the motor protection signal MSS sets the motor protection fault signal retentively and transfers this signal to output QMSS_ST. Parameter MSS_SIG is used to define whether only to display an error message (MSS_SIG = 0), or whether the motor is to be limited irrespective of any other inputs and system states (MSS_SIG = 1). The motor protection fault (QMSS_ST = 1) is reported to the OS. QMSS_ST is either reset by the operator at the RESET input, or automatically by interconnecting L_RESET with "1" when there a positive edge at MSS.
- LMNR_ON: The closed-loop control operates with position feedback if MNR_ON is set. The state at this control input must not be changed while the system is in operation.
- DEADB_W: The deadband in the error signal is necessary to reduce switching frequency of the controller as a result of minor error signal fluctuations around zero. At the operating point of the closed loop, the error signal changes by the value: (100% *PULSE_TM/MTR_TM) * process gain. The deadband should therefore be greater than 50% of this value. The deadband is ignored if DEADB_W assumes negative values.

Tracking mode

There are two ways of influencing actuating signals in tracking mode:

- Tracking via the external manipulated variable LMN_TRK
- Direct control of the actuating signals via interconnected inputs LMNS_ON, LMNUP and LMNDN

Tracking mode with the external manipulated variable LMN_TRK is only possible in control systems with position feedback. If LMN_SEL = 1, the manipulated variable is fetched from the interconnected tracking value LMN_TRK and applied to output LMN. The valve is adjusted by actuating signals until the value of position feedback LMNR_IN reaches the value of LMN_TRK.

Tracking mode using LMNS_ON with direct connection of the control signals via the interconnected LMNUP and LMNDN inputs has highest priority of all modes. If LMNS_ON is set, the actuating signals can only be set via the LMNUP or LMNDN inputs. Any other influence on the control signals is suppressed as long as LMNS_ON is set.

Placing the D and P action in the feedback path

The P and D actions can be located in the feedback path to reduce or avoid overshoot of the process value as a result of a setpoint step change. The setpoint step change no longer has any effect on the P or D action in this mode. The manipulated value no longer responds with a step change after a setpoint step change.

The P action is enabled in the feedback path by setting PFDB_SEL = 1. PFDB_SEL is ignored by step controllers without position feedback. The P action cannot be enabled in the feedback path in this mode.

The D action is enabled in the feedback path by setting DFDB_SEL = 1.

Cascading PID controllers

The manipulated variable LMN of the primary controller is connected to input SP_EXT of the secondary controller (Note: the secondary controller is a step controller whereas the primary controller is not!). The primary controller must also change to tracking mode if the cascade is opened. A QCAS_CUT signal is generated in the secondary controller and interconnected with LMN_SEL input of the primary controller. This disconnection can be caused in manual or tracking mode by setpoint input, or by setpoint tracking of the secondary controller:

QCAS_CUT = LMNS_ON or LMN_SEL or (not QMAN_AUT) or (QMAN_AUT and SP_TRK_ON)

The tracking input LMN_TRK of the primary controller is interconnected with the SP output of the secondary controller to prevent a step change when the cascade is closed again.

It is also advisable to set a directional lock of the integrator in the primary controller which is activated as soon as the secondary controller reaches a manipulated variable limit. With a positive control action, interconnect the INT_HPOS and INT_HNEG inputs of the primary controller with the QLMN_HLM and QLMN_LLM outputs of the secondary controller.

3.4.5 CTRL_S Mode Change

Mode change

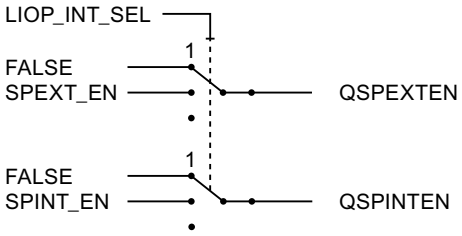
The mode change can be triggered either by the operator or via interconnected inputs.

External/internal setpoint

Depending on the setting of the LIOP_INT_SEL selection input, changeover is carried out between the external and internal setpoint by OS operation of the SPEXTSEL_OP input or interconnection of SPEXON_L. You must set the corresponding enable inputs SPINT_EN, SPEXT_EN to enable this changeover.

If SPBUMPON = 1, the internal setpoint is replaced with the effective setpoint in order to allow a bumpless changeover from external or tracking mode to internal mode.

Enabling changeover between internal and external setpoint



QSPEXTEN = TRUE: SPEXTSEL_OP can be changed from FALSE (internal setpoint) to TRUE (external setpoint).

QSPINTEN = TRUE: SPEXTSEL_OP can be changed from TRUE (external setpoint) to FALSE (internal setpoint).

SPEXTSEL_OP tracks or is reset as required.

Enabling setpoint control via the operator input SP_OP_ON

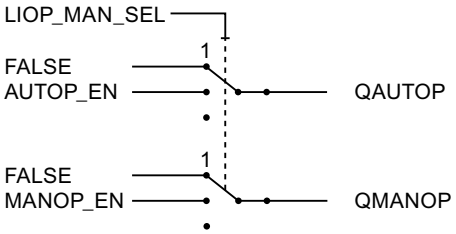
Q_SP_OP = TRUE: SP_OP operation is enabled.

SP_OP is tracked or reset as required.

Manual/auto

Depending on the setting of the LIOP_MAN_SEL selection input, changeover is carried out between manual and automatic mode by OS operation of the AUT_ON_OP input or interconnection of AUT_L. You must set the corresponding enable inputs MANOP_EN, AUTOP_EN to enable this changeover.

Enabling the changeover between manual and automatic mode



QAUTOP = TRUE: AUT_ON_OP can be changed from FALSE (manual mode) to TRUE (automatic mode).

QMANOP = TRUE: AUT_ON_OP can be changed from TRUE (automatic mode) to FALSE (manual mode).

AUT_ON_OP tracks or is reset as required.

Enabling Manual Mode

Manipulated variable operation is enabled at the faceplate via MAN_OP by setting QLMNVOP, the control signal is controlled via QLMNSOP instructions:

	QLMNVOP	QLMNSOP
LMNOP_ON	1	-
LMNSOPON	-	1
LMNR_ON	1	-
LMNSOPON and (LMNUP_OP xor LMNDN_OP)	0	-
LMN_SEL	0	0
LMNS_ON	0	0
QMAN_AUT	0	0

Special measures are applied for the modes listed below in order to ensure a bumpless changeover:

- Setpoint external/setpoint tracking: If SPBUMPON = TRUE, the internal setpoint SP_OP is set to the effective (external or tracked) setpoint.
- Automatic mode: The manual value MAN_OP is tracked to the value of the effective manipulated variable LMNR_IN.
- Tracking mode: When LMN_SEL is set, the manually input value MAN_OP is tracked to the value of LMN_TRK. You can thus see up to which value the valve is traveled. When LMN_SEL is canceled, MAN_OP will be reset to the value of LMNR_IN, in order to ensure a bumpless changeover when MAN_OP is set.
- Manual mode: In control systems with position feedback the integrator is tracked to allow a bumpless changeover to automatic mode.

  Integrated component = Manipulated variable (in %)
  minus the proportional component minus the disturbance variable (in %). The derivative component is disabled and compensated.

3.4.6 Error Handling of CTRL_S

Operator input error

QOP_ERR = 1 is set if at least one operator error occurs when changing one of the parameters SPEXTSEL_OP, AUT_ON_OP, SP_OP, or MAN_OP. Otherwise QOP_ERR = 0 is set. An operator error exists only for one cycle.

The block algorithm handles the following situations:

- Parameter assignment error: NM_PVHR <= NM_PVLR:
  The error signal ER is set to zero. ENO = 0 or QERR = 1 is set.
- Parameter assignment error: SAMPLE_T < 0.001:
  The sampling time SAMPLE_T is set to 0.001. ENO = 0 or QERR = 1 is set.
- Parameter assignment error: GAIN = 0:
  The error signal ER is set to zero. ENO = 0 or QERR = 1 is set.
- TN(TI) < SAMPLE_T/2:
  If TN > 0, TN = SAMPLE_T/2 is used for calculation. If TN(TI) = 0, the integrator is deactivated and the operating point LMN_OFF is active.
- TV(TD) < SAMPLE_T:
  If TV > 0, TV = SAMPLE_T is used for calculation. The D action is deactivated if TV(TD) = 0.
- TM_LAG < SAMPLE_T/2:
  When TM_LAG < SAMPLE_T/2, TM_LAG < SAMPLE_T/2 is used for calculation. In these cases, the D action acts as an ideal differentiator.

  MTR_TM, PULSE_TM and BREAK_TM are restricted downwards to the value of SAMPLE_T.

A reset of the *_EN enable inputs during active operation is indicated at the QMAN_ERR or QAUT_ERR outputs.

QMAN_ERR is also set if the control input for position feedback LMNR_ON is canceled while the controller tracks LMN_TRK. The valve is stopped in this case.

3.4.7 Startup Behavior, Dynamic Response and Message response of CTRL_S

Startup characteristics

During CPU startup, CTRL_S is set to manual mode with an internal setpoint. This means that the block must be called in the startup OB. In CFC engineering, this is handled by the CFC. When using basic STEP 7 tools, you enter this call in the startup OB.

After startup, the messages will be suppressed during the number of cycles set at RUNUPCYC.

MAN_OP and LMN with LMNR_IN are initialized during the startup. The integral action is set to zero.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SAMPLE_T parameter.

Assignment of the 32-bit status word VSTATUS

You can find additional information in "VSTATUS for CTRL_S (Page 97)".

Message response

The CTRL_S block uses the ALARM_8P block for generating messages.

The following message triggers exist:

- The functions for monitoring the limits of process values and of system deviation
- the CSF signal QMSS_ST which is received as a control system fault via the interconnection

Limit violation messages can be suppressed individually using the the relevant M_SUP_xx inputs. Process messages (not control system messages!) can be disabled centrally using MSG_LOCK.

QMSG_SUP is set if the RUNUPCYC cycles have not expired yet since a restart, MSG_LOCK = TRUE or MSG_STAT = 21.

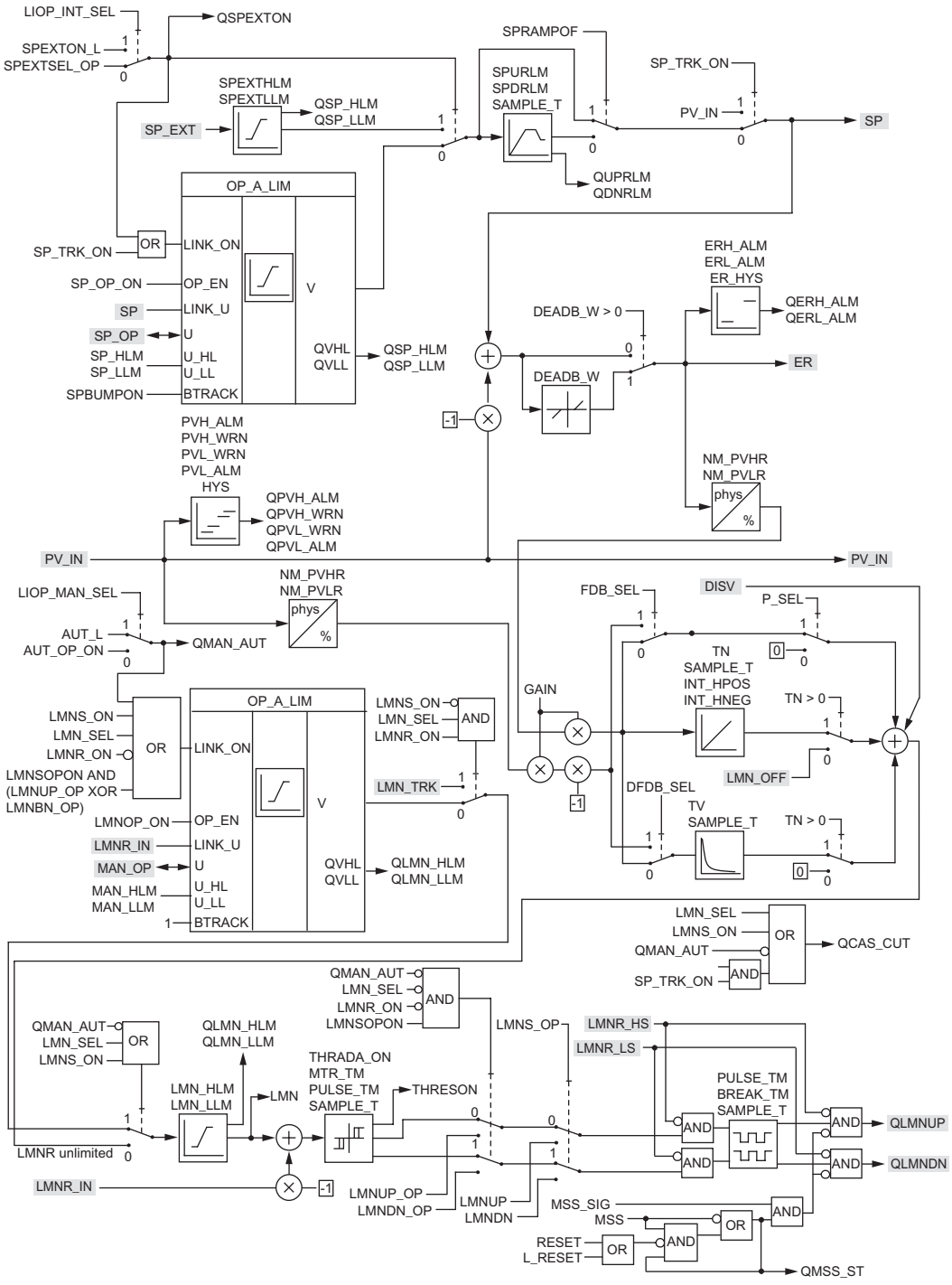
The table lists the message texts (Page 96) of the CTRL_S block and their assignment to the block parameters.

Monitoring process values

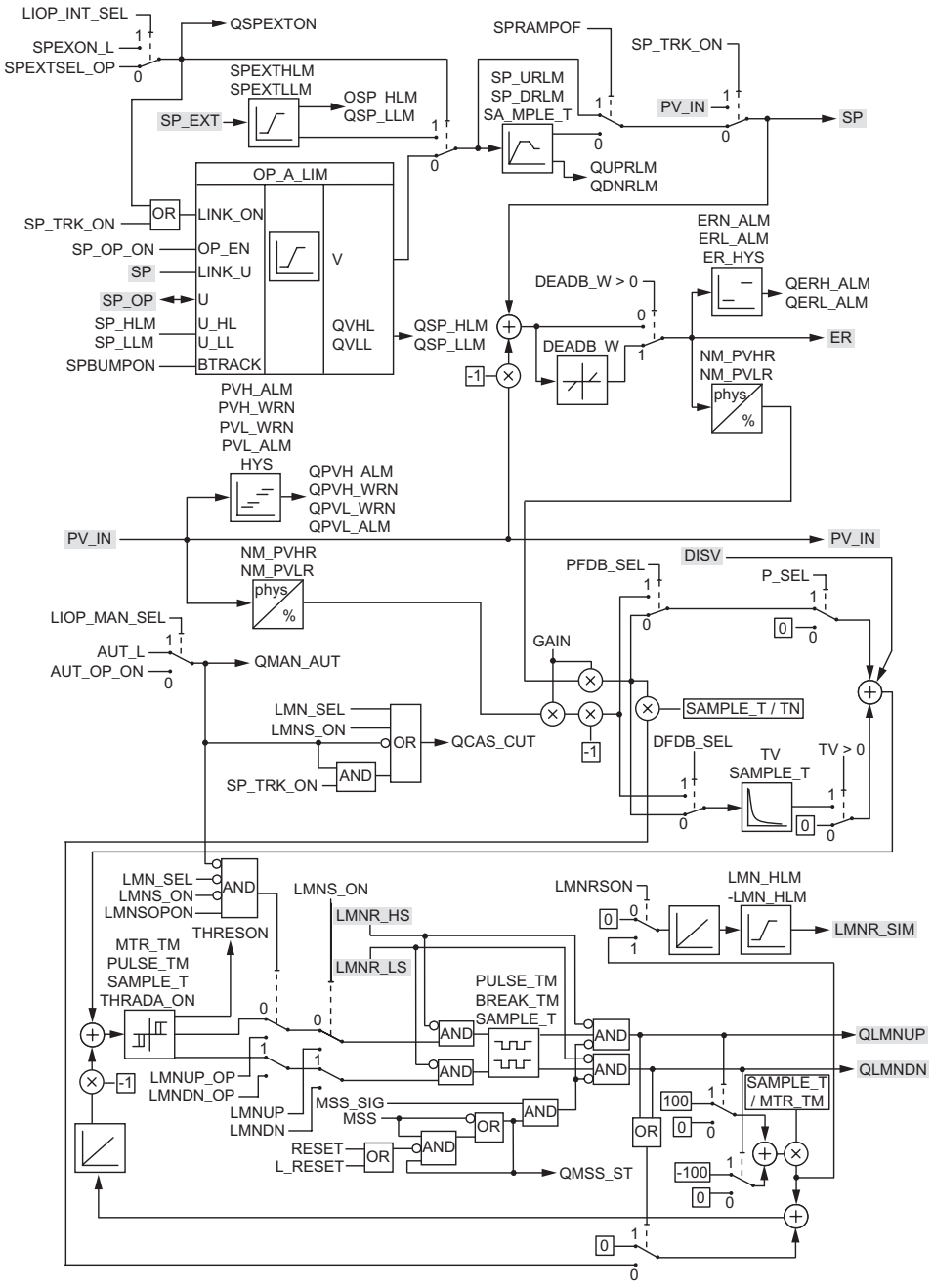
Not available

3.4.8 Block Diagram of CTRL_S

Block Diagram of CTRL_S with Position Feedback



Block Diagram of CTRL_S without Position Feedback



3.4.9 I/Os of CTRL_S

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
AUT_L	Interconnectable input for MANUAL/AUTO: 0 = manual; 1 = auto	BOOL	0	I		
AUT_ON_OP	Operator input: 0 = manual; 1 = auto	BOOL	0	IO	+	
AUTOP_EN	1 = operator-control enable for auto mode	BOOL	1	I		
AUX_PRx	Associated value x	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: Consecutive batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	''	I	+	
BREAK_TM	Minimum break time in seconds	REAL	1	I	+	
CSF	Control-system error	BOOL	0	I		
DEADB_W	Dead band	REAL	0	I	+	≥ 0
DFDB_SEL	Place D action in feedback, 1 = active	BOOL	0	I		
DISV	Disturbance variable	REAL	0	I		
ER	Error signal	REAL	0	O	+	
ER_HYS	Hysteresis for monitoring the error signal	REAL	0.1	I	+	≥ 0
ERH_ALM	Error signal: High alarm limit	REAL	100	I	+	> 0
ERL_ALM	Error signal: Low alarm limit	REAL	-100	I	+	< 0
GAIN	Proportional gain	REAL	1	I	+	
HYS	Hysteresis	REAL	5	I	+	≥ 0
INT_HNEG	Lock dependent on direction, negative	BOOL	0	I		
INT_HPOS	Lock dependent on direction, positive	BOOL	0	I		
L_RESET	Interconnectable input RESET for motor-protection error (QMSS_ST=0)	BOOL	0	I		
LIOP_INT_SEL	1 = interconnection active, 0 = operator control active	BOOL	0	I		
LIOP_MAN_SEL	1 = interconnection active, 0 = operator control active	BOOL	0	I		
LMN	Manipulated-variable output	REAL	0	O		
LMN_HLM	High limit for manipulated variable	REAL	100	I	+	LMN_HLM > LMN_LLM
LMN_LLM	Low limit for manipulated variable	REAL	0	I	+	LMN_LLM < LMN_HLM
LMN_OFF	Operating point	REAL	0	I	+	
LMN_SEL	1 = External manipulated variable active	BOOL	0	I		
LMN_TRK	External manipulated variable	REAL	0	I		
LMNDN	Interconnected control signal low	BOOL	0	I		

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
LMNDN_OP	Adjustable control signal low	BOOL	0	IO	+	
LMNOP_ON	1 = operator-control enable for manipulated variable MAN_OP	BOOL	1	I		
LMNR_HS	High limit stop signal of position feedback	BOOL	0	I		
LMNR_IN	Position feedback for display on OS	REAL	0	I		
LMNR_LS	Low limit stop signal of position feedback	BOOL	0	I		
LMNR_ON	Controlling with position feedback	BOOL	0	I		
LMNR_SIM	Simulated position feedback	REAL	0	O	+	
LMNRSON	Activate simulation of position feedback	BOOL	0	I	+	
LMNS_ON	Interconnected control signals ON (LMNDN, LMNUP)	BOOL	0	I		
LMNSOPON	Activate operator control of the control signal	BOOL	1	I		
LMNUP	Interconnected control signal high	BOOL	0	I		
LMNUP_OP	Adjustable control signal high	BOOL	0	IO	+	
M_SUP_AH	1 = message suppression high limit alarm process value	BOOL	0	I	+	
M_SUP_AL	1 = message suppression low limit alarm process value	BOOL	0	I	+	
M_SUP_ER	Suppression of messages for error-signal alarms	BOOL	1	I	+	
M_SUP_WH	1 = message suppression high limit warning process value	BOOL	0	I	+	
M_SUP_WL	1 = message suppression low limit warning process value	BOOL	0	I	+	
MAN_HLM	High limit of manual value manipulated variable	REAL	100	I	+	
MAN_LLM	Low limit of manual value manipulated variable	REAL	0	I	+	
MAN_OP	Operator input for manipulated variable	REAL	0	IO	+	
MANOP_EN	1 = operator-control enable for manual mode	BOOL	1	I		
MO_PVHR	High display limit (measuring range)	REAL	110	I	+	
MO_PVLR	Low display limit (measuring range)	REAL	-10	I	+	
MSG_ACK	Acknowledge messages	WORD	0	O		
MSG_EVID	Message number	DWORD	0	I		
MSG_LOCK	1 = message suppression dependent on process state	BOOL	0	I	+	
MSG_STAT	Message error information	WORD	0	O		
MSS	Motor protection active low	BOOL	1	I		
MSS_SIG	Reset control signals if motor protection active	BOOL	0	I		
MTR_TM	Motor actuating time in seconds	REAL	60	I	+	
NM_PVHR	High normalization limit of process value (measuring range)	REAL	100	I		
NM_PVLR	Low normalization limit of process value (measuring range)	REAL	0	I		

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
OCCUPIED	ID for occupied by BATCH	BOOL	0	I	+	
OOS	Reserve	BOOL	0	I	+	
OPTI_EN	1 = enable controller tuning	BOOL	0	I	+	
P_SEL	Activate P action, 1 = action	BOOL	1	I		
PFDB_SEL	Place P action in feedback; 1 = active	BOOL	0	I		
PULSE_TM	Minimum pulse duration in seconds	REAL	1	I	+	
PV_IN	Process value	REAL	0	IO	+	
PVH_ALM	Process value: High alarm limit	REAL	100	I	+	PVH_ALM ≥ PVL_ALM
PVH_WRN	Process value: High warning limit	REAL	95	I	+	PVH_WRN ≤ PVL_WRN
PVL_ALM	Process value: Low alarm limit	REAL	0	I	+	PVL_ALM ≤ PVH_ALM
PVL_WRN	Process value: Low warning limit	REAL	5	I	+	PVL_WRN ≤ PVH_WRN
Q_SP_OP	1 = operator-control enable for setting setpoint	BOOL	0	O	+	
QAUT_ERR	Missing enable signals for automatic mode	BOOL	0	O		
QAUTOP	1 = operator control enable for automatic mode	BOOL	0	O	+	
QC_LMN	Quality code for LMN	BYTE	16#80	O		
QC_LMN_I	Quality code for output LMN	BYTE	16#80	I		
QC_LMNR_IN	Quality code for LMNR_IN	BYTE	16#80	I		
QC_PV_IN	Quality code for PV_IN	BYTE	16#80	I		
QCAS_CUT	1 = cascade cut	BOOL	1	O		
QDNRLM	1 = negative setpoint ramp limited	BOOL	0	O		
QERH_ALM	Error signal, 1 = high limit alarm	BOOL	0	O	+	
QERL_ALM	Error signal, 1 = low limit alarm	BOOL	0	O	+	
QERR	1 = error output (inverted ENO)	BOOL	1	O	+	
QLMN_HLM	1 = high value of manipulated-variable output limited	BOOL	0	O		
QLMN_LLM	1 = low value of manipulated-variable output limited	BOOL	0	O		
QLMN_SEL	1 = Tracking LMN_TRK active	BOOL	0	O	+	
QLMNDN	Control signal low	BOOL	0	O	+	
QLMNOP	1 = operator-control enable for manipulated variable or control signal	BOOL	0	O	+	
QLMNR_HS	High limit stop signal of position feedback set	BOOL	0	O	+	
QLMNR_LS	Low limit stop signal of position feedback set	BOOL	0	O	+	
QLMNR_ON	Position feedback is activated	BOOL	0	O	+	
QLMNRSON	Simulation of position feedback activated	BOOL	0	O	+	
QLMNS_ON	1 = signal adjustment active	BOOL	0	O	+	
QLMNSOP	Output: Operator control of the control signal activated	BOOL	1	O	+	

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
QLMNUP	Control signal high	BOOL	0	O	+	
QLMNVOP	Output: Operator control of the manipulated variable activated	BOOL	1	O	+	
QMAN_AUT	0 = manual, 1 = auto	BOOL	0	O	+	
QMAN_ERR	Missing enable signals for manual mode	BOOL	0	O		
QMANOP	1 = operator-control enable for manual mode	BOOL	0	O	+	
QMAN_ERR	Error from Alarm_8P	BOOL	0	O		
QMSG_SUP	1 = message suppression active	BOOL	0	O	+	
QMSS_ST	Motor protection active (0 = reset with RESET)	BOOL	0	O	+	
QOP_ERR	1 = group operator-input error	BOOL	0	O		
QPVH_ALM	1 = high alarm	BOOL	0	O		
QPVH_WRN	1 = high warning	BOOL	0	O		
QPVL_ALM	1 = low alarm	BOOL	0	O		
QPVL_WRN	1 = low warning	BOOL	0	O		
QSP_HLM	1 = high setpoint output limited	BOOL	0	O		
QSP_LLM	1 = low setpoint output limited	BOOL	0	O		
QSPEXTEN	1 = operator control enable for external	BOOL	0	O	+	
QSPEXTON	0 = internal 1 = external	BOOL	0	O	+	
QSPINTEN	1 = operator-control enable for internal	BOOL	0	O	+	
QUPRLM	1 = positive setpoint ramp limited	BOOL	0	O		
RESET	Enabled reset input for motor-protection error (QMSS_ST=0)	BOOL	0	IO	+	
RUNUPCYC	Number of run-up cycles	INT	3	I		
SAMPLE_T	Sampling time in seconds	REAL	1	I		≥ 0.001
SP	Active setpoint	REAL	0	O	+	
SP_EXT	External setpoint	REAL	0	I		
SP_HLM	Setpoint high limit	REAL	100	I	+	SP_HLM > SP_LLM
SP_LLM	Setpoint low limit	REAL	0	I	+	SP_LLM < SP_HLM
SP_OP	Operator input for setpoint	REAL	0	IO	+	
SP_OP_ON	Operator-control enable for setpoint SP_OP	BOOL	1	I		
SP_TRK_ON	1 = track setpoint SP_OP	BOOL	0	I	+	
SPBUMPON	1 = bumpless setpoint	BOOL	1	I	+	
SPDRLM	Max. negative setpoint ramp rate [1/s]	REAL	100	I	+	
SPEXON_L	Interconnectable input for internal/external setpoint (0 = internal/1 = external)	BOOL	0	I		
SPEXT_EN	1 = operator-control enable for external setpoint	BOOL	1	I		
SPEXTHLM	High limit of external setpoint	REAL	100	I		SPEXTHLM > SPEXTLLM

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
SPEXTLLM	Low limit of external setpoint	REAL	0	I		SPEXTLLM < SPEXTHLM
SPEXTSEL_OP	Operator input: 0 = internal, 1 = external	BOOL	0	IO	+	
SPINT_EN	1 = operator-control enable for internal	BOOL	1	I		
SPRAMPOF	1 = deactivate setpoint ramp limitation	BOOL	1	I	+	
SPURLM	Max. positive setpoint ramp rate [1/s]	REAL	100	I	+	
STEP_NO	BATCH step number	DWORD	0	I	+	
THRADA_ON	Adaptation of the response threshold: 0 = keep constant	BOOL	1	I		
THRESON	Adaptive threshold	REAL	0.0	O		
TM_LAG	Time lag of D action in seconds	REAL	1	I	+	≥ SAMPLE_T/2
TN	Reset time in seconds	REAL	10	I	+	TN=0, ≥ SAMPLE_T/2
TV	Derivative time in seconds	REAL	0	I	+	TV=0, ≥ SAMPLE_T
USTATUS	Status word in VSTATUS; can be configured user-specific	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.4.10 Message Texts and Associated Values of CTRL_S

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QPVH_ALM (when PV_IN ≥ PVH_ALM)	PV:$$BlockComment$$ HighHigh alarm	AH	M_SUP_AH, MSG_LOCK
2	QPVH_WRN (when PV_IN ≥ PVH_WRN)	PV:$$BlockComment$$ High alarm	WH	M_SUP_WH, MSG_LOCK
3	QPVL_WRN (when PV_IN ≤ PVL_WRN)	PV:$$BlockComment$$ Low alarm	WL	M_SUP_WL, MSG_LOCK
4	QPVL_ALM (when PV_IN ≤ PVL_ALM)	PV:$$BlockComment$$ LowLow alarm	AL	M_SUP_AL, MSG_LOCK
5	CSF	$$BlockComment$$ External fault	S	-
6	QERH_ALM (when ER ≥ ERH_ALM)	ER:$$BlockComment$$ HighHigh alarm	AH	M_SUP_ER, MSG_LOCK
7	QERL_ALM (when ER ≤ ERL_ALM)	ER:$$BlockComment$$ LowLow alarm	AL	M_SUP_ER, MSG_LOCK
8	QMSS_ST	$$BlockComment$$ Motor protection	S	-

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data, the fourth is reserved for PV_IN and the remaining ones (AUX_PRx) can be freely assigned by the user.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	PV_IN
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.4.11 VSTATUS for CTRL_S

The 32-bit status word extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	QLMNR_ON	-	QMSS_ST	QSPEXTON	QMAN_AUT	MSG_LOCK	BA_EN	OCCUPIED

Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	LMN_SEL	-	QLMNR_LS	QLMNR_HS	QLMNDN	QLMNUP

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.4.12 Operating and Monitoring of CTRL_S

Additional information

You will find more information in the following sections:

- CTRL_S block icon (Page 580)
- CTRL_S faceplate (Page 503)

3.5 DEADT_P: Dead-time element

3.5.1 Description of DEADT_P

Object name (type + number)

FB37

- DEADT_P block I/Os (Page 99)

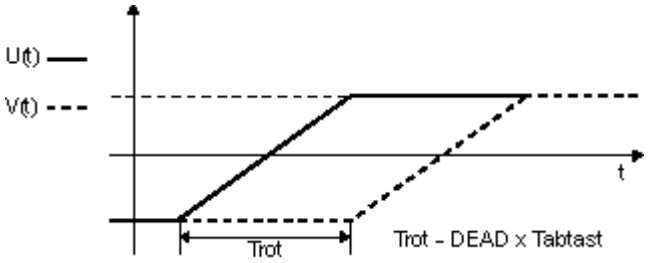
Function

An analog value of the input U is not passed to output V until a definable number of cycles DEADT have expired. The following formula applies:

V(t) = U(t-Tdead), where Tdead = DEADT * Tsampling

You will find more information about time 0 < t < Tdead in "Startup characteristics".

How it works



Operating principle of DEADT_P

- The block fetches the analog input value U during the current cycle, buffers it and outputs it to output V after the set number of DEADT cycles have expired. The maximum number of buffered values is limited to 16. You will find more information in "Error handling".
- If the DEADT parameter is modified while the block is being executed, the block responds in the same way as it would during a CPU startup.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Error handling

If the parameter DEADT is < 0 or if DEADT > 16, the value DEADT = 16 is used internally for calculation, ENO = 0 or QERR = 1 is indicated.

Startup characteristics

During a CPU startup or when the dead time parameter DEADT is modified, the active input value U is written to the internal dead time buffer.

Time response

To allow it to perform its required function, the block is called in a cyclic interrupt OB. You can calculate the dead time Tdead according to the following formula:

- Tdead = DEADT * Tsampling
  Tsampling is the sampling time of the block.
- When configuring with CFC, the higher-level runtime group of the block and its sampling parameter may have to be taken into consideration.

3.5.2 I/Os of DEADT_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameters)	Meaning	Data type	Default	Type	Permissible values
DEADT	Dead time in cycles	INT	0	I	< 16
QERR	1 = error	BOOL	1	O	
U	Input	REAL	0	I	
V	Output	REAL	0	O	

3.6 DIF_P: Differentiation

3.6.1 Description of DIF_P

Object name (type + number)

FB38

- DIF_P block I/Os (Page 102)

Function

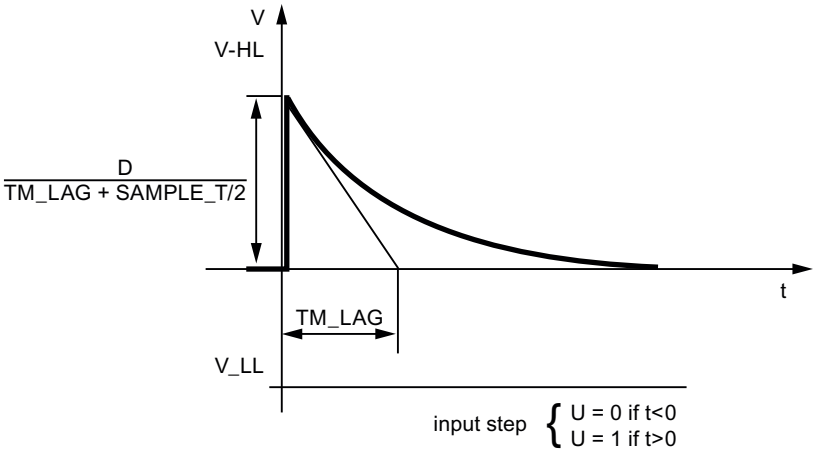
The DIF_P block approximates a DT1 action and operates according to the trapezoid rule:

$v(s) = TD * s / (TM\_LAG * s + 1) * u(s)$

How it works

The algorithm operates according to the trapezoid rule. The following steps are also carried out:

- The high limit of output V is limited to V_HL, the low limit to V_LL. Internal calculation is not affected by the limiting function.
- The assigned output Q_HL or Q_LL is set when limiting is active.
- The parameter setting TD = 0 is allowed and results in V = 0 = const. Negative values for TD are also allowed and have the effect of inverting the sign of the results.



Step response of DIF_P

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Error handling

In case of an overflow/underflow, the violated range limit of the type REAL is used and ENO is set to "0". The following configuration errors also lead to ENO = 0 (QERR = TRUE) and V=0:

- V_LL > 0
- V_HL < 0

Startup characteristics

After a CPU startup, the internal memory bit for the old value of input U is tracked to this. This ensures that the output value V is "0" during the first cyclic operation.

Time response

The block is called from a cyclic interrupt OB.

3.6.2 I/Os of DIF_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
QERR	1 = Error	BOOL	1	O		
QVHL	Initial value high limit	BOOL	0	O		
QVLL	Initial value low limit	BOOL	0	O		
SAMPLE_T	Sampling time in seconds	REAL	1	I		> 0
TD	Differential time in seconds	REAL	1	I		
TM_LAG	Time lag in seconds	REAL	10	I		≥ 0
U	Input	REAL	0	I		
V	Output value	REAL	0	O		
V_HL	High limit of V	REAL	100	I		V_HL ≥ V_LL
V_LL	Low limit of V	REAL	-100	I		V_LL ≥ V_HL

3.7 DIG_MON: Digital value monitoring

3.7.1 Description of DIG_MON

Object name (type + number)

FB62

- DIG_MON block I/Os (Page 105)
- DIG_MON block icon (Page 583)
- DIG_MON faceplate (Page 509)

Function

The DIG_MON block is used to monitor a digital process tag with flutter suppression. Both the signal status and the state of the control system (external control system errors, channel errors) belong to the process tag. The MSG_CLAS parameter can be used to determine the message class of the process tag.

How it works

The block monitors changes of the digital value at input Ix. The timer is retriggered at each edge of the input signal. Once the idle time set under SUPPTIME has expired, the input value I is written to the output Q.

This ensures that only the signals held at least until the period set in SUPPTIME has expired will be passed to the output. Signals changing at a faster rate will not be passed on.

If SUPPTIME < SAMPLE_T, the input signal I will be passed to output Q.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB 32) and also OB100.

Error handling

QERR=1 if the message class configuration is invalid. An error message is not output in this case. You will find more information in "Message response".

Startup characteristics

During CPU startup the old initial value Q is retained. Monitoring of changes begins again after the restart. This means that the block must be called in the startup OB. In CFC engineering, this is handled by the CFC. With simple STEP 7 tools, you will have to enter the call in the startup OB.
Following startup, the messages are suppressed for the number of cycles set in the RUNUPCYC value.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for DIG_MON (Page 108)".

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SAMPLE_T parameter.

Message response

The DIG_MON block uses the ALARM_8P block (MSG_EVID) for generating messages (1 to 8, except 3, 4 and 7). Messages 3, 4 and 7 do not require acknowledgement and are generated via NOTIFY_8P (MSG_EVID1).

The following message triggers exist:

- Change to the output signal Q
- The CSF signal received as a control system error via the interconnection.

Process messages (not control system messages!) can be locked with MSG_LOCK.

QMSG_SUP is set if the RUNUPCYC cycles have not expired yet since a restart, MSG_LOCK = TRUE or MSG_STAT = 21.

Message classes

A change at output Q can be signaled to the OS with a selectable message class by configuring the MSG_CLAS input.

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of DIG_MON (Page 105)

Message texts and associated values of DIG_MON (Page 106)

VSTATUS for DIG_MON (Page 108)

Operating and monitoring DIG_MON (Page 108)

3.7.2 I/Os of DIG_MON

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
AUX_PRx	Associated value x block	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: current batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	0	I	+	
CSF	1 = external control system error	BOOL	0	I		
I	Input signal	BOOL	0	I	+	
MSG_ACK	Acknowledge messages	WORD	0	O		
MSG_CLAS	Message class of the signal	INT	0	I		1 - 7
MSG_EVID	Message number	DWORD	0	I		
MSG_LOCK	1 = message suppression dependent on process state	BOOL	0	I	+	
MSG_STAT	Message error information	WORD	0	O		
OCCUPIED	ID for occupied by BATCH	BOOL	0	I	+	
OOS	Reserve	BOOL	0	I	+	
Q	Output signal	BOOL	0	O	+	
QC_I	Quality code for I	BYTE	16#80	I		
QERR	1 = error output (inverted ENO)	BOOL	1	O	+	
QMSG_ERR	1 = message error	BOOL	0	O	+	
QMSG_SUP	1 = message suppression	BOOL	0	O	+	
RUNUPCYC	Number of run-up cycles	INT	3	I		
SAMPLE_T	Sampling time in seconds	REAL	1.0	I		> 0
STEP_NO	BATCH step number	DWORD	0	I	+	
SUPPTIME	Time in seconds to expire before an edge transition at the input is passed to the output.	REAL	0	I	+	
USTATUS	Status word in VSTATUS; can be configured user-specific	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.7.3 Message Texts and Associated Values of DIG_MON

Assignment of message texts and message classes to the block parameters

Message no. MSG_EVID	Block parameters	Default message text	Message class	Can be suppressed by
1	Q & MSG_CLAS = 1	$$BlockComment$$ HighHigh Alarm	AH	MSG_LOCK
2	Q & MSG_CLAS = 2	$$BlockComment$$ High alarm	WH	MSG_LOCK
3			No message	
4			No message	
5	Q & MSG_CLAS = 5	$$BlockComment$$ Low alarm	WL	MSG_LOCK
6	Q & MSG_CLAS = 6	$$BlockComment$$ LowLow alarm	AL	MSG_LOCK
7			No message	
8	CSF	$$BlockComment$$ External fault	S	-
Message no. MSG_EVID1				
1			No message	
2			No message	
3	Q & MSG_CLAS = 3	$$BlockComment$$ Tolerance high	TH	MSG_LOCK
4	Q & MSG_CLAS = 4	$$BlockComment$$ Tolerance low	TL	MSG_LOCK
5			No message	
6			No message	
7	Q & MSG_CLAS = 7	$$BlockComment$$ Operator request	OR	MSG_LOCK
8			No message	

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data and the remaining ones (AUX_PRx) can be freely assigned by the user.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	AUX_PR04
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.7.4 VSTATUS for DIG_MON

The 32-bit status word extends the status display in the block icons and faceplates. The 16 low bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	I	-	-	-	-	MSG_LOCK	BA_EN	OCCUPIED

Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	-	-	-	-	Q

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.7.5 Operating and Monitoring of DIG_MON

Additional information

You will find more information in the following sections:

- DIG_MON block icon (Page 583)
- DIG_MON faceplate (Page 509)

3.8 DOSE: Dosing process

3.8.1 Description of DOSE

Object name (type + number)

FB63

- DOSE block I/Os (Page 113)
- DOSE block icon (Page 583)
- DOSE faceplate (Page 510)

Function

The DOSE block is used for upsizing or downsizing batches in single-component dosing with weighing devices and also for dosing using volumetric measuring devices. When flow meters are used, the integral flow value should be made available at input PV_IN. As long as dosing is active, output Q is set. This output makes it possible to control the equipment that allows dosing. At the end of dosing an automatic correction for dribbling can be made which will become active at the next dosing. The initial dribble is preset at the DRIBB input.

The dosing value is monitored against the setpoint for exceeding or falling below tolerance and the results supplied to two corresponding outputs at the end of the dosing process.

Operating modes

The internal/external modes can be set either with the operator input SPEXTSEL_OP or the switched input SPEXON_L. The switched input SPEXTON_L is only effective if the input SPEXT_ON = True. The result switches between the "internal setpoint" and "external setpoint":

- Internal. The setpoint (SP) is entered by the operator at SP_OP and limited to (SP_LLM, SP_HLM).
- External. The setpoint (SP) is obtained from SP_EXT and limited as described above.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Start of dosing

Dosing starts as follows:

- Dosing is started by setting the START_OP input or by the rising edge of the interconnected signal L_START in the same block.
- There is a wait time until the scales stand still, i.e. STNDSTLL = 1. If there is no standstill signal available, this input must be set to 1. This is followed by taring, in other words, the current process value PV_IN is entered into the tare memory.
- The actual dosing value PV_OUT is calculated as follows, depending on whether the operation causes rise or fall (selected with REVERSE):
  - REVERSE = 0: PV_OUT = PV_IN - TARA (PV_IN rises)
  - REVERSE = 1: PV_OUT=TARA - PV_IN (PV_IN falls)
- Q, QSTRTDOS is set and QEND_DOS, QTOL_P, and QTOL_N reset. You will find more information in the section "Component change".

End of dosing

The final phase of dosing takes place in the following steps:

- As soon as PV_OUT ≥ SP - DRIBB_F, Q is reset. If STNDSTLL = 1 is set in addition to this, QSTRTDOS is also reset.
- As soon as a standstill is reported (STNDSTLL = 1), a counter is initialized with the time [s] defined at RELAXTME and is decremented cyclically by the sampling time SAMPLE_T. The settling time (QRELXING = 1) runs as long as the counter is > 0.
- On expiration of the settling time any underdosing or overdosing is evaluated according to set tolerance limits TOL_N and TOL_P, and a dribbling correction follows if DRIB_COR = 1.
- If the dosing quantity is within the tolerance band, the end of dosing (QEND_DOS = 1) is set.

Component change

If you change a component, set COMP_CHG = 1 before you start dosing. The dribbling value currently set at DRIBB is transferred to output DRIBB_F at the start of dosing (QSTRTDOS = 1).

Dribbling correction

If a dribbling correction is requested (DRIB_COR = 1), the dribbling value is calculated as follows:

DRIBB_F := DRIBB_F - (SP - PV_OUT) * DCF/100, where the following condition is met:

0 ≤ DRIBB_F ≤ DRIBBMAX

The correction factor DCF is limited internally to 0 - 100.

You will find more information on the dribbling value in the section "Component change".

Overdosing/underdosing

- When overdosing occurs (PV_OUT > SP + TOL_P), QTOL_P and QEND_DOS are set.
- In the event of underdosing (PV_OUT < SP - TOL_N) only QTOL_N is set. Operators can correct the dosing volume manually. The end of the dosing is reported (QEND_DOS = 1) as soon as this correction is completed.. The block outputs are no longer updated if no further dosing operation is started.

Post-dosing

Operators can only correct the dosing volume if underdosing is detected. This operation is carried out by setting POSTDOSE, or via the interconnectable L_PDOSE input.

- Set DRIB_COR = 1
- When the signal edge rises, the QSTRT_DOS signal for the start of dosing is set for the duration of PDOS_TME. This operation can be repeated until either the setpoint has been exceeded, or until the operator has acknowledged the operation by setting the ACK_TOL_OP input or via the interconnectable ACK_TOL input.
- Following the acknowledgment, the end of dosing (QEND_DOS = 1) is indicated and the outputs are no longer updated.

Pause

- When necessary, dosing can be interrupted by the PAUSE_OP command or via the interconnectable PAUSE input. The interconnectable PAUSE input is only effective if input LIOP_SEL = True. When overdosing occurs (PV_OUT > SP + TOL_P), QTOL_P and QEND_DOS are set.
- Dosing is resumed after the pause "End" signal is set. If overdosing has occurred, the pause must be ended before a new dosing operation can be started.

Cancel

Dosing can be cancelled prematurely with of CANCEL_OP command input or via the interconnectable CANCEL input. A new dosing operation can then be started..

Error handling

Operator errors detected while controlling the blocks are logically ORed and applied to the QOP_ERR group output. The outputs ENO = 0 and QERR = 1 are set if mathematical errors are detected.

Startup characteristics

When the CPU starts up, "Abort dosing" will be simulated but without generating a message. This means that the block must be called in the startup OB. In CFC engineering, this is handled by the CFC. When using basic STEP 7 tools, you enter this call in the startup OB. After startup, the messages will be suppressed during the number of cycles set at RUNUPCYC.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SAMPLE_T parameter.

Assignment of the 32-bit status word VSTATUS

You can find additional information in "VSTATUS for DOSE (Page 117)".

Message response

The DOSE block uses the ALARM8_P (MSG_EVID) block to generate messages. The "Acknowledgment request" message does not require acknowledgment and is generated by NOTIFY (MSG_EVID1).

The following message triggers exist:

- Dosing limit monitoring functions
- The end or cancellation of dosing
- The CSF signal, which is referenced as a control-system error by interconnection

The messages reporting limit violations can be suppressed individually via the relevant inputs M_SUP1 to 3. Process messages (not control system messages!) can be disabled centrally using MSG_LOCK.

QMSG_SUP is set if the RUNUPCYC cycles have not expired yet since a restart and MSG_LOCK = TRUE or MSG_STAT = 21.

Monitoring process values

n.a.

Additional information

You will find more information in:

I/Os of DOSE (Page 113)

Message texts and associated values of DOSE (Page 116)

VSTATUS for DOSE (Page 117)

Operator Control and Monitoring of DOSE (Page 117)

3.8.2 I/Os of DOSE

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
ACK_TOL	Interconnectable input for ACK_TOL_OP	BOOL	0	I		
ACK_TOL_OP	Acknowledgement underdosing	BOOL	0	IO	+	
AK_OP_EN	1= operator-control enable for acknowledgement	BOOL	1	I		
AUX_PRx	Associated value x	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: Consecutive batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	''	I	+	
CANCEL	Interconnectable input for CANCEL	BOOL	0	I		
CANCEL_OP	Cancel current dosing operation at positive edge	BOOL	0	IO	+	
CN_OP_EN	1 = operator-control enable for CANCEL	BOOL	1	I		
COMP_CHG	1 = component change at next dosing start	BOOL	0	I	+	
CSF	Control-system error	BOOL	0	I		
DCF	Dribbling correction factor in %	REAL	25	I	+	0...100
DRIB_COR	1 = activate dribbling correction	BOOL	0	I	+	
DRIBB	Dribbling initial value	REAL	0	I	+	
DRIBB_F	Current dribbling value	REAL	0	O	+	
DRIBBMAX	Maximum dribbling value (default is selected at random here, since the dimension is not known until instancing is carried out)	REAL	999	I	+	
ER	Dosing error (ER = SP - PV_OUT)	REAL	0	O	+	
L_PDOSE	Interconnectable input post-dosing	BOOL	0	I		
L_START	Interconnectable input for START	BOOL	0	I		
LIOP_SEL	1 = interconnection (i.e. PAUSE) is active, 0 = operator control is active	BOOL	0	I	+	
M_SUP_1	Message suppression normal dosing	BOOL	0	I	+	
M_SUP_2	Message suppression overdosing	BOOL	0	I	+	
M_SUP_3	Message suppression underdosing	BOOL	0	I	+	
MO_PVHR	High display limit (measuring range)	REAL	110	I	+	
MO_PVLR	Low display limit (measuring range)	REAL	-10	I	+	
MSG_ACK	Acknowledge messages	WORD	0	O		
MSG_EVID	ALARM_8P event ID	DWORD	0	I		
MSG_LOCK	1 = message lock dependent on process state	BOOL	0	I	+	
MSG_STAT	Message error information	WORD	0	O		
OCCUPIED	ID for occupied by BATCH	BOOL	0	I	+	
OOS	Reserve	BOOL	0	I	+	
P_OFF_EN	1 = operator-control enable for "Continue"	BOOL	1	I		

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
P_ON_EN	1 = operator-control enable for "Pause"	BOOL	1	I		
PAUSE	Interconnectable input for PAUSE_OP	BOOL	0	I		
PAUSE_OP	1 = stop current dosing operation (pause), 0 = continue	BOOL	0	IO	+	
PD_OP_EN	1 = operator-control enable for post-dosing	BOOL	1	I		
PDOS_TME	Post-dosing time [s]	REAL	0	I	+	
POSTDOSE	Post-dosing at positive edge	BOOL	0	IO	+	
PV_IN	Process value weight (weighing input)	REAL	0	I	+	
PV_OUT	Dosing process value	REAL	0	O	+	
Q	1 = control dosing device	BOOL	0	O	+	
Q_SP_OP	1 = operator-control enable setting setpoint	BOOL	0	O	+	
QAK_OP	1 = operator-control enable acknowledgment	BOOL	0	O	+	
QC_PV_IN	Quality code for PV_IN	BYTE	16#80	I		
QC_Q	Quality code for Q	BYTE	16#80	O		
QC_Q_I	Quality code for output Q	BYTE	16#80	I		
QCN_OP	1 = operator-control enable CANCEL	BOOL	0	O	+	
QEND_DOS	1 = dosing completed	BOOL	1	O	+	
QERR	1 = error output (inverted ENO)	BOOL	1	O	+	
QMSG_ERR	1 = message error	BOOL	0	O		
QMSG_SUP	1 = message suppression active	BOOL	0	O	+	
QOP_ERR	1 = operator error	BOOL	0	O		
QP_OFF_EN	1 = operator-control enable "Continue"	BOOL	0	O	+	
QP_ON_EN	1= operator-control enable "Pause"	BOOL	0	O	+	
QPD_OP	1 = operator-control enable post-dosing	BOOL	0	O	+	
QRELXING	1 = setting time active	BOOL	0	O	+	
QSP_HLM	1 = high operator setpoint limited	BOOL	0	O		
QSP_LLM	1 = low operator setpoint limited	BOOL	0	O		
QSPEXTEN	1 = operator-control enable external changeover	BOOL	0	O	+	
QSPEXTON	0 = internal, 1 = external	BOOL	0	O	+	
QSPINTEN	1 = operator-control enable internal changeover	BOOL	0	O	+	
QSTRT_OP	1 = operator-control enable dosing start	BOOL	0	O	+	
QSTRTDOS	1 = dosing started	BOOL	0	O	+	
QTOL_N	1 = underdosed after dose end	BOOL	0	O	+	
QTOL_P	1 = overdosed after dose end	BOOL	0	O	+	
RELAXTME	Idle time after dosing stop in seconds	REAL	3	I	+	
REVERSE	0 = gain in weight, 1 = loss in weight	BOOL	0	I	+	
RUNUPCYC	Number of run-up cycles	BYTE	3	I		
SAMPLE_T	Cycle time in seconds	REAL	1	I		> 0
SP_EXT	External setpoint	REAL	0	I	+	
SP_HLM	High limit setpoint	REAL	100	I	+	
SP_LLM	Low limit setpoint	REAL	0	I	+	
SP_OP	Setpoint	REAL	0	IO	+	

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
SP_OP_ON	1 = operator-control enable for internal setpoint	BOOL	1	I		
SPBUMPON	1 = bumpless setpoint ON	BOOL	0	I	+	
SPEXON_L	Interconnectable input for selecting SP_EXT (0 = internal, 1 = external)	BOOL	0	I		
SPEXT_EN	1 = operator-control enable for external setpoint	BOOL	1	I		
SPEXT_ON	1 = interconnection: SP_EXT is active; 0 = operator control is active	BOOL	0	I	+	
SPEXTSEL_OP	Enabled input for selecting SP_EXT (0 = internal, 1 = external)	BOOL	0	IO	+	
SPINT_EN	1 = operator-control enable for internal	BOOL	1	I		
ST_OP_EN	1 = operator-control enable for dosing start	BOOL	0	I		
START_OP	1 = dosing start at positive edge	BOOL	0	IO	+	
STEP_NO	BATCH step number	DWORD	0	I	+	
STNDSTLL	1 = standstill	BOOL	1	I	+	
TOL_N	Lower tolerance band	REAL	0	I	+	
TOL_P	Upper tolerance band	REAL	0	I	+	
USTATUS	Status word in VSTATUS, can be configured by user	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.8.3 Message Texts and Associated Values of DOSE

Assignment of message texts and message classes to the block parameters

Message no. MSG_EVID	Block parameters	Default message text	Message class	Can be suppressed by
1	(SP-TOL_N ≤ PV_OUT ≤ SP+TOL_P)	$$BlockComment$$ Dosing okay	PM	M_SUP_1, MSG_LOCK
2	QTOL_P	$$BlockComment$$ Overdosed	AH	M_SUP_2, MSG_LOCK
3	QTOL_N	$$BlockComment$$ Underdosed	AL	M_SUP_3, MSG_LOCK
4	CSF	$$BlockComment$$ External fault	S	-
5	CANCEL	$$BlockComment$$ Dosing stopped	PM	MSG_LOCK
6,7,8			No message	-
Message no. MSG_EVID1	**Block parameters**	**Default message text**	**Message class**	**Can be suppressed by**
	(PV_OUT < SP-TOL_N)	$$BlockComment$$ Acknowledge Request	OR	-

The first three of the associated values of the message block are assigned SIMATIC BATCH data, the fourth is reserved for PV_OUT, and the remaining ones (AUX_PRx) can be assigned freely.

Assignment of the associated values to block parameters

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	PV_OUT
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.8.4 VSTATUS for DOSE

The 32-bit statusword extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	QSTRTDOS	-		QSPEXTON	-	MSG_LOCK	BA_EN	OCCUPIED

Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	-	QTOL_N	QTOL_P	QEND_DOS	QRELXING

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.8.5 Operator Control and Monitoring of DOSE

Additional information

You can find additional information in the following sections:

- DOSE block icon (Page 583)
- DOSE faceplate (Page 510)

3.9 ELAP_CNT: Operating hours counter

3.9.1 Description of ELAP_CNT

Object name (type + number)

FB64

- ELAP_CNT block I/Os (Page 120)
- ELAP_CNT block icon (Page 584)
- ELAP_CNT faceplate (Page 120)

Function

The ELAP_CNT block detects the operating hours of units.

How it works

Block ELAP_CNT detects the time as long as the ON_OFF input = 1, i.e. as long as the connected unit is operational. The value SAMPLE_T[s]/3600 is added to the value HOURS at every execution. The output HOURS thus specifies the number of operating hours.

Calling OBs

This is the cyclic alarm OB in which you install the block (for example, OB32) and in OB100.

Setting the Counter

Under certain circumstances (for example, after maintenance or replacement of the unit), the initial value of the operating-hours counter has to be specified (as a rule, 0). On the OS, the operator sets the tracking value at the HOURS_OP input. This value is then passed to the HOURS output via the TRACK_OP input or by interconnecting the TRACK input with the HOURS output.

Error handling

Arithmetic errors are indicated by ENO = 0 or QERR = 1.

Startup characteristics

No special measures. After startup, the messages will be suppressed during the number of cycles set at RUNUPCYC.

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.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for ELAP_CNT (Page 122)".

Message response

The ELAP_CNT block uses the ALARM_8P block to generate messages.

Messages are triggered by the functions for monitoring the operating hour limits.

Limit violation messages can be suppressed individually using the the relevant M_SUP_xx inputs. Process messages (not control system messages!) can be disabled centrally using MSG_LOCK.

QMSG_SUP is set if the RUNUPCYC cycles have not expired yet since a restart and MSG_LOCK = TRUE or MSG_STAT = 21.

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of ELAP_CNT (Page 120)

Message texts and associated values of ELAP_CNT (Page 121)

VSTATUS for ELAP_CNT (Page 122)

Operating and monitoring ELAP_CNT (Page 122)

See also

ELAP_CNT (All Views) (Page 516)

3.9.2 I/Os of ELAP_CNT

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
AUX_PRx	Associated value x	ANY	0	IO	
HOURS	Duty period (hours)	REAL	0	O	+
HOURS_AH	Alarm high limit (hours)	REAL	100	I	+
HOURS_OP	Tracking value (hours)	REAL	0	IO	+
HOURS_WH	Warning high limit (hours)	REAL	95	I	+
M_SUP_AH	1 = Suppress HL alarm	BOOL	0	I	+
M_SUP_WH	1 = Suppress HL warning	BOOL	0	I	+
MO_HOUHR	High display limit	REAL	120	I	+
MO_HOULR	Low display limit	REAL	0	I	+
MSG_ACK	Acknowledge messages	WORD	0	O	
MSG_EVID	Message number	DWORD	0	I	
MSG_LOCK	1 = Process messages locked	BOOL	0	I	+
MSG_STAT	Message error information	WORD	0	O	
ON_OFF	Equipment status: 1 = ON, 0 = OFF	BOOL	0	I	+
OOS	Reserve	BOOL	0	I	+
QC_ON_OFF	Quality Code for ON_OFF	BYTE	16#80	I	
QERR	1 = error output (inverted ENO)	BOOL	1	O	+
QH_ALM	1 = HL alarm active	BOOL	0	O	
QH_WRN	1 = HL warning active	BOOL	0	O	
QMSG_ERR	1 = message error	BOOL	0	O	+
QMSG_SUP	1 = message suppression	BOOL	0	O	+
RUNUPCYC	Number of run-up cycles	BYTE	3	I	
SAMPLE_T	Sampling time in seconds	REAL	1	I	
TRACK	Interconnectable input for TRACK	BOOL	0	I	
TRACK_OP	1 = Set HOURS to HOURS_OP	BOOL	0	IO	+
USTATUS	Status word in VSTATUS, can be configured by user	WORD	0	I	
VSTATUS	Extended status display in block icons	DWORD	0	O	+

3.9.3 Message Texts and Associated Values of ELAP_CNT

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QH_ALM	$$BlockComment$$ HighHigh Alarm	M	M_SUP_AH, MSG_LOCK
2	QH_WRN	$$BlockComment$$ High alarm	M	M_SUP_WH, MSG_LOCK

All associated values (AUX_PRx) of the message block can be freely assigned by the user.

Assignment of the associated values to block parameters

Associated value	Block parameters
1	AUX_PR01
2	AUX_PR02
3	AUX_PR03
4	AUX_PR04
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.9.4 VSTATUS for ELAP_CNT

The 32-bit statusword extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	ON_OFF	-	-	-	-	MSG_LOCK	-	-
Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	-	-	-	-	-

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.9.5 Operator Control and Monitoring of ELAP_CNT

Additional information

You can find additional information in the following sections:

- ELAP_CNT block icon (Page 584)
- ELAP_CNT faceplate (Page 516)

3.10 FMCS_PID: Controller block

3.10.1 Description of FMCS_PID

Object name (type + number)

FB114

- FMCS_PID block I/Os (Page 138)
- FMCS_PID block icon (Page 585)
- FMCS_PID faceplate (Page 518)

Area of application

Block FMCS_PID is used to interface the FM 355 controller block.

It can be used for the C (continuous controllers) and S (step and pulse controllers) module types. It does not in itself contain any control algorithms, since the control functions are carried out exclusively on the module. You can use the FMCS_PID 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 input and output parameters can be found in the manual of the controller module FM 355.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The parameter CO_NO is set
- The input EN_CO is interconnected with the output EN_CO_x of the FM_CO block (x = number of the rack)
- The output ENCO is connected to the input ENCOx_yy of the FM_CO block (x = number of the rack, yy = coordination number).

Implementing the FM 355 C Controller Module as Continuous Control

The block provides the following displays and setting options:

- Display of the result of the limit monitoring carried out on the module for two limit pairs for the process value PV or the error signal ER (QH_ALM, QH_WRN, QL_WRN, QL_ALM outputs). MONERSEL is used to specify whether PV or ER is monitored.
- Disabling of the generation of individual messages when limits are exceeded
- Split-range function
- Dead band (DEADB_W, on threshold) in the error-signal branch
- Specification of the control algorithm: PID algorithm (QFUZZY = 0) or FUZZY algorithm (QFUZZY = 1)
- Manipulated variable tracking
- Disabling of the integral action
- Setpoint tracking (SP = PV)

Implementing the FM 355 S Controller Module as Pulse Control

Using the controller module as a pulse controller rather than a continuous controller means that split-range control is not possible.

You can use the pulse controller to generate pulse width modulated control signals. The control signal is converted into a binary output signal so that the ratio between the pulse length and the configured period at the assigned digital output corresponds with the manipulated value LNM.

Implementing the FM 355 S controller module as step controller

Using the controller module as a step controller rather than a continuous controller results in the following differences:

- Output QLMNR_ON indicates whether a feedback signal is available (1: exists, 0: does not exist).
- Split-range operation is not possible.

When used as a step-action controller without a position feedback (QLMNR_ON = 0), manual adjustment of the manipulated variable is only possible at the end positions. In this case the safety position or value of the external manipulated variable value is interpreted by the controller module as follows:

Value < 40 %: Close actuating element completely

Value > 60 %: Open actuating element completely

40% ≤ Value ≤ 60%: Hold current setting

Calling OBs

Cyclic interrupt OB: For example OB 32

The block is be installed with the same instance in the startup block OB100.

Take note of the dependencies on the FM_CO block too.

Additional information

You will find more information in:

Addressing (Page 126)
Function (Page 126)
Acquisition of process values via the process image (Page 128)
Generation of setpoints, limits, error signals, and manipulated variables (Page 129)
Manual, auto and tracking mode (Page 131)
Mode change (Page 133)
Safety mode (Page 134)
Download parameters to the module (Page 134)
Read data from the module (Page 135)
Error handling (Page 135)
Startup behavior, dynamic response and message response (Page 136)
Backup mode of the FM 355 (Page 138)
I/Os of FMCS_PID (Page 138)
Message texts and associated values of FMCS_PID (Page 143)
VSTATUS for FMCS_PID (Page 145)
Archiving process values (Page 613)
Operating and monitoring FMCS_PID (Page 145)

3.10.2 Addressing

The controller channel of an FM 355 belonging to the instance is addressed via its logical base address (set in HW Config) (LADDR input) and the controller channel number (CHANNEL input). 1 to 4 are the values permitted for the CHANNEL input. ACC_MODE must be set to TRUE after a change is made to CHANNEL.

The FM 355 module is monitored with the blocks of the *PCS 7 Library*. The MODE input is interconnected with the OMODE output of the MOD_D1 block. The block communicates only via the FM 355 control channel. As a result, the measurement range coding in the low word of the output OMODE is irrelevant and assigned zero.

3.10.3 Function of FMCS_PID

Function

The FMCS_PID block forms the interface between the control module (FM 355) and the blocks of the SIMATIC PCS 7 Library. It can also be interconnected with other SIMATIC S7 blocks.

The block and the controller module operate asynchronously to each other.

All relevant process and disturbance variables are provided by the module and can only be read by the block. The block can also transfer different operating modes and settings to the controller module.

The process values are read cyclically. (Exceptions: SP (setpoint from the FM), ER (error signal), DISV (disturbance), LMN_A and LMN_B). They are, however, only updated after every 4th cycle. You will find more information about this in Acquisition of process values via the process image (Page 128).

The parameters SP (setpoint from the FM), ER (error signal), DISV (disturbance variable), LMN_A, and LMN_B can only be read from the FM 355 using "Read data record" (SFC59). You will find more information in Read via data record.

A quality code is generated for each process value PV and LMN and this can have the following states:

State	Quality code
Valid value	16#80
Invalid value	16#00

OP_SEL Input

As a rule, the FM 355 obtains its parameters via the block. Each change to a parameter in the block is passed to the module. If, however, you want to change a parameter directly at the FM 355 via the operator panel (OP), you first have to enable this function at OP_SEL (OP_SEL = 1). OP_SEL must be reset if you want to restrict operator control and parameter assignment via the block again after the OP has been used.

When configuration via the OP is enabled, the controller module does not accept any parameters from the block. However, the blocks continues to update the process values SP_OP_ON, LMNOP_ON, SP_OP and LMN, thus allowing a bumpless changeover to the mode in which configuration is carried out by the block. The remaining parameters (*for example* GAIN) are overwritten with the data of the block instance as soon as you set OP_SEL = 0. The entries made with the OP are lost, if you have not entered the data in the block instance before reversing OP_SEL.

SDB_SEL Input

Some of the parameters can be specified via the FMCS_PID block in addition to the configuration tool. These two parameter sets may differ. The SDB_SEL input of the block is available to avoid such a conflict. SDB_SEL = 1 is used to specify that the module only reads these parameters from the block and not from the configuration tool. The setting SDB_SEL = 0 specifies that the module accepts these parameters from the parameter assignment tool and from the block. Please note that the parameters from the configuration tool are sent to the module after every STOP-RUN transition of the CPU. The parameters of the block, on the other hand, are transferred to the module each time there is a change at the block input.

When configuration via the OP is permitted, the operator inputs are disabled with the following exception: The operation OP_SEL = 0 (operation disable via OP) is possible.

3.10.4 Acquisition and Writing of Process Values Via the Process Image

Reading process values

The process values (except SP (setpoint from the FM), ER (error signal), DISV (disturbance), LMN_A and LMN_B) are read cyclically from the process image. They are only updated after every 4th cycle. In a normal situation, this takes place in the cyclic interrupt OB in which the FMCS_PID is installed. To achieve higher accuracy, it is possible to install the FMCS_PID additionally in a second cyclic interrupt OB with a higher clock rate.

After a restart or when ACC_MODE is set to 1, the block determines the cyclic interrupts used (up to 2 are permitted); you will find more information in "Startup characteristics (Page 136)".

The following must be considered when selecting cyclic interrupts:

- The cyclic interrupt OB in which the block is installed must not run slower than 30 s or faster than 25 ms.

  Note: OB1 must not be used!

- The process image partition (TPA) in HW Config must be set for the faster OB.
- Reading via the process image for a block is complete after 4 block cycles (after 4 s in the example with OB32 [1000 ms]) if the block cycle is longer than the cycle time of the module, otherwise after 4 module cycles.
- **Example**: With 4 processed analog inputs on the module, the cycle time of the module is 400 ms (see "Module parameters" command button on the configuration interface of the FM 355). If the block is installed in OB32 (1000 ms), for example, you can accelerate reading of the process image by means of an additional installation in OB33 (500 ms).

Writing Process Values

The following process values are written to the process image in every second cycle:

- Setpoint (automatic mode only)
- Manipulated value (manual mode only)
- SP_OP_ON, SAFE_ON, QMAN_AUT, LMNTRKON, LMN_REON, LMNRHSRE, LMNRLSRE
- LMNSOPON
- LMNUP or LMNUP_OP
- LMNDN or LMNDN_OP
- FUZID_ON
- LMNRS_ON

In the following cases, a restart must be performed or ACC_MODE must be set to 1 following a compile and download:

- if you have moved blocks to a different cycle
- if you have also installed blocks in a fast cycle
- if you have also deleted installed blocks

3.10.5 Generation of Setpoints, Limits, Error Signals, and Manipulated Variables

Setpoint generation by the FMCS_PID block

The setpoint SP can be obtained from four different sources:

- From the controller module (you have configured this on the OP, or the module is in back-up mode). In this case setpoint adjustment is disabled and the applied value is written to the operator input SP_OP of the block.
- The three other sources depend on the inputs SP_TRK_ON, SPEXTSEL_OP and SP_OP_ON.

You will find more information about the **external/internal** setpoint in "Mode change (Page 133)".

SP_TRK_ON	SPEXTSEL_OP	SP_OP_ON	SP=	State
Irrelevant	0	0	SP_INT	Internal setpoint
0	0	1	SP_OP	Internal (operator controlled) setpoint
Irrelevant	1	Irrelevant	SP_EXT	External setpoint
1	0	1	PV **	Tracked setpoint
			** in manual mode only and when SPBUMPON = 1	

The effective setpoint is limited to the range (SP_LLM, SP_HLM).

If SP_TRK_ON is set, in the manual mode (when SP_OP_ON = 1, SPEXTSEL_OP = 0 and SPBUMPON = 1) the setpoint SP_OP is tracked. This enables bumpless changeover from manual to automatic mode.

Bumpless changeover to manual operation is ensured by writing back the active setpoint and manipulated variables.

Limit generation

Depending on the input MONERSEL, the controller module monitors either the process value PV (MONERSEL = 0) or the error signal ER (MONERSEL = 1) for warning and alarm limits (L_WRN, H_WRN, L_ALM, H_ALM). Monitoring is carried out with the common hysteresis HYS.

The block makes the monitoring result available at the QL_WRN, QH_WRN, QL_ALM and QH_ALM outputs. While monitoring the process value PV, the block signals any violation of the high and low limits, unless message suppression has been enabled.

Error signal generation

The error signal is generated by the controller module, based on the active setpoint SP and the process value PV and is made available at output ER of the block.

After the dead band DEADB_W has expired, the error signal is processed further in the PID algorithm. A disturbance variable is not added.

Manipulated-Variable Generation by the FMCS_PID Block

The manipulated value LMN is derived from various sources. If several control inputs are TRUE simultaneously, the priority is as follows:

Priority	Control input	LMN	State
1	LMNS_ON = 1	LMNUP or LMNDN	Interconnected control signals with step controller
2	SAFE_ON = 1	= LMN_SAFE	Safety mode
3	Manual	= LMN_OP	Manual mode (QMAN_AUT = 0)
4	LMNTRKON = 1	= analog input on module or 0.0	Tracking mode
5	LMN_REON = 1	= LMN_RE	External manipulated variable
6	Automatic	= PID algorithm	Automatic mode (QMAN_AUT = 1)

- If LIOP_MAN_SEL = 0, the AUT_ON_OP parameter is used to change over between manual and automatic mode on the OS.
- If LIOP_MAN_SEL = 1, the AUT_L parameter is used to change over between manual and automatic mode via an interconnection in the CFC.

In step controllers the manipulated variable is converted into control pulses (QLMNUP; QLMNDN) under consideration of the motor-specific parameters "Motor actuating time" (MOTOR_TM), Minimum pulse duration (PULSE_TM), Minimum break duration (BREAK_TM).

3.10.6 Manual, auto and tracking mode of FMCS_PID

Manual mode

The manipulated variable is set by the operator at input LMN_OP (jogging mode is also possible here). When changing over to automatic mode, the module takes over the manipulated variable set "manually" as the operating point.

"Manual mode" takes priority over "Tracking mode".

Manual mode for step controller:

Actuating commands can be used for direct control of the control signal for step controllers. Signal manipulation is enabled with LMNSOPON; the control signals are set at LMNUP_OP or LMNDN_OP. The valve is adjusted until operator input is canceled or the end position is reached.

The operator adjustment of the control signal has priority over adjustment of the manipulated value via LMN_OP.

The LMN_OP input cannot be controlled when step controllers without position feedback are used.

Automatic mode

The manipulated variable is calculated by the PID or fuzzy algorithm of the module. The control parameters GAIN, TI, TD and TM_LAG can be interconnected.

The controller's direction of control can be reversed (rising error signal causes a falling manipulated variable) by setting a negative proportional GAIN.

The I action can be deactivated by setting TI = 0.

The operator-controllable manipulated variable input LMN_OP tracks the LMN output so that bumpless changeover from automatic to manual mode is ensured.

External setpoint (LMN_RE)

The block transfers the value LMN_RE to the FM 355. The FM 355 accepts the external manipulated variable LMN_RE manipulated variable as LMN, if LNM_REON = 1.

"External manipulated variable" mode takes priority over "Automatic mode".

Manipulated variable tracking

In manipulated variable tracking mode (LMNTRKON = 1), the manipulated variable tracks an analog input of the module or the value 0.0.

"Tracking" mode takes priority over the "external manipulated variable" mode.

Safety mode (LMN_SAFE)

The block transfers the value LMN_SAFE to the FM 355. The FM 355 accepts the safety manipulated variable LMN_SAFE as manipulated variable LMN, if SAFE_ON = 1.

"Safety mode" takes priority over "tracking" mode.

Deactivating operator control enable for control signals and manual value

You must set LMNOP_ON and LMNSOPON to 0.

Interconnectable control signal setting for the step controller

Tracking mode using LMNS_ON with direct connection of the control signals via the interconnected LMNUP and LMNDN inputs has highest priority of all modes. If LMNS_ON is set, the control signals can only be set via the inputs LMNUP or LMNDN. Any other influence on the control signals is suppressed as long as LMNS_ON is set.

3.10.7 Mode change of FMCS_PID

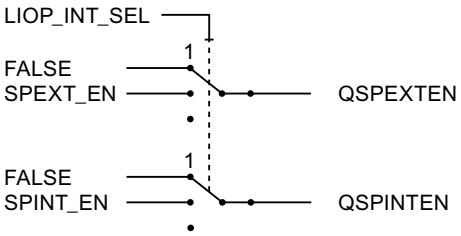
The mode change can be triggered either by the operator or via interconnected inputs. The mode is changed by means of the operator control blocks assigned to the modes.

External/internal setpoint

A changeover between external and internal setpoint is initiated by the OS operator setting the SPEXTSEL_OP input by interconnecting SPEXON_L. You must set the corresponding enable inputs SPINT_EN, SPEXT_EN or the selection input LIOP_INT_SEL to enable these changeovers.

SP_OP_ON must be set TRUE to enable operator control of the setpoint.

Enabling changeover between internal and external setpoint



QSPEXTEN = TRUE: SPEXTSEL_OP can be changed from FALSE (internal setpoint) to TRUE (external setpoint).

QSPINTEN = TRUE: SPEXTSEL_OP can be changed from TRUE (external setpoint) to FALSE (internal setpoint).

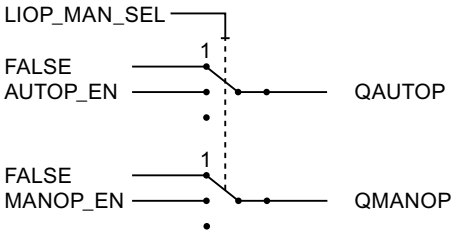
SPEXTSEL_OP tracks or is reset as required.

Manual/auto

The operating mode is changed over by the OS operator setting the AUT_ON_OP input or by interconnection of AUT_L. You must set the corresponding enable inputs MANOP_EN, AUTOP_EN or the selection input LIOP_MAN_SEL to enable this changeover.

LMNOP_ON must be set TRUE to enable manual control of the variable.

Enabling the changeover between manual and automatic mode



QAUTOP = TRUE: AUT_ON_OP can be changed from FALSE (manual mode) to TRUE (automatic mode).

QMANOP = TRUE: AUT_ON_OP can be changed from TRUE (automatic mode) to FALSE (manual mode).

AUT_ON_OP tracks or is reset as required.

Measures for bumpless changeover

Special measures are applied for the modes listed below in order to ensure a bumpless changeover:

- Setpoint external/setpoint tracking:
  If SPBUMPON = TRUE, the internal setpoint SP_OP is set to the effective (external or tracking) setpoint.
- Automatic mode, safety mode, tracking mode or external manipulated variable LMN_RE:
  The manipulated value LMN_OP is tracked to the effective manipulated variable.
- Manual mode, safety mode, tracking mode or external manipulated variable LMN_RE:
  The integrator tracks so that a bumpless changeover to automatic mode is possible (not with a step controller). Using the configuration tool, you can disable the default bumpless changeover function. The error signal can then be corrected more quickly.

3.10.8 Safety mode

The interconnectable input SAFE_ON is used to activate the safety mode. This is executed by the controller module with highest priority. In safety mode, the value present at the LMN_SAFE input of the block is output at the control output.

3.10.9 Download Parameters to the Module

The channel-specific controller and operating parameters are transferred to the controller module whenever a corresponding block parameter changes. As long as operator control via OP is disabled, the module rejects the parameters written by the block.

The download of controller and operating parameters to the controller module can require several block calls.

3.10.10 Read Data From the Module

The channel-specific process values are read by the controller module whenever a block is called. Particularly in distributed operation reading may require several block calls.

If channel-specific controller and operating parameters on the module have been changed at an OP, the block also reads the current parameters from the controller module. It then updates the SP_OP_ON (setpoint-value operation on), LMNOP_ON (manipulated-variable operation on), SP_OP (operating setpoint) and LMN_OP (operating manipulated variable) inputs.

3.10.11 Error Handling of FMCS_PID

Error displays

The FMCS_PID block provides the following error displays:

Error display	Meaning
QOP_ERR = 1	Operator input error. If there is no new operator input error, QOP_ERR is reset in the next block cycle.
QPARF_FM = 1	• Error when assigning parameters directly to the controller module using the parameter assignment tool. • Invalid control channel number (CHANNEL) was set at the block. • Error when installing in OBs; you can find additional information in "Startup Characteristics".
QCH_F = 1	Channel error. Due to a hardware fault the controller channel belonging to the instance cannot return valid results.
QMODF = 1	Controller module has been removed or is faulty.
QPERAF = 1	I/O access error. The block could not access the controller module.

3.10.12 Startup Behavior, Dynamic Response and Message response of FMCS_PID

Startup characteristics

During CPU startup when the FMCS_PID block runs the first time, the operating modes MANUAL and INTERNAL are set.

In OB100, QDONE is set to 0.

After a CPU restart or when ACC_MODE is set to 1, the block is not operable for up to approximately 30 seconds.

The block uses the first 30 seconds after a restart (OB100) or after ACC_MODE is set to 1 to detect whether the block instance is installed in more than one OB. If you then move the blocks to another OB, a restart must be performed or ACC_MODE must be set to 1 after compiling and downloading (otherwise you will receive the error message QPARF_FM = 1, and the block will not provide any more data).

Note

The parameter settings of the configuration tool can be written to the SDB (system data) by selecting the menu command **Save/Compile and Download** in HW Config. These SDB parameters can deviate from the block parameters. When SDB_SEL = 0, the SDB parameters are sent to the module at every STOP-RUN transition of the CPU. The block parameters are written to the module, however, a few cycles later. With SDB_SEL = 1, the module does not load the SDB parameters at a STOP-RUN transition of the CPU. This avoids a jump in the manipulated variable when the SDB parameters deviate from the block parameters.

The following cases are differentiated:

- The FM 355 has failed and had not had settings changed via the OP before it failed.
  The block transfers the current controller and operating parameters to the FM 355.
- The FM 355 has failed and settings had been changed via the OP before it failed.
  The block reads the current values from the FM 355 and updates its SP, LMN, Q_SP_OP and QLMNOP outputs.
- The FM 355 has not failed and settings had been changed via the OP.
  The block reads the current values from the FM 355 and updates its SP, LMN, Q_SP_OP and QLMNOP outputs.
- The FM 355 has not failed and settings had not been changed via the OP.
  Controller and operating parameters of the FM 355 and block are identical. The block does nothing.

The block transfers the controller and operating parameters to the FM 355 during the startup (however not during the initial run).

Time response

Not available

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for FMCS_PID (Page 145)".

Message response

The FMCS_PID block uses the ALARM_8P block for generating messages.

The following message triggers exist:

- Process-value or system-deviation limit monitoring functions
- The function for monitoring the module hardware (handled mainly by the MOD_D1 block).

Limit violation messages can be suppressed individually using the the relevant M_SUP_xx inputs. Process messages (not control system messages!) can be disabled centrally using MSG_LOCK.

QMSG_SUP is set if the RUNUPCYC cycles have not been completed since the last restart and if MSG_LOCK = TRUE or MSG_STAT = 21.

Monitoring process values

Not available

3.10.13 Backup Mode of the FM 355

If the CPU changes over to STOP or fails, the FM 355 changes over to backup mode. In this case, FM 355 automatically enables operator control via the OP (acts as if OP_SEL = 1).

3.10.14 I/Os of FMCS_PID

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
ACC_MODE	SUBN1_ID, SUBN2_ID, RACK_NO, SLOT_NO and CHANNEL included in internal processing	BOOL	0	IO		
AUT_L	Interconnectable input for MANUAL/AUTO: 0 = Manual, 1 = Auto	BOOL	0	I		
AUT_ON_OP	Operator input for MANUAL/AUTO: 0 = Manual, 1 = Auto	BOOL	0	IO	+	
AUTOP_EN	1 = operator control enable for AUTO	BOOL	0	I		
AUX_PRxx	User-specific associated process value 6 … 10	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: current batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	0	I	+	
BREAK_TM	Minimum break time in seconds	REAL	2	I		
CHANNEL	Controller channel number	INT	1	I		
CO_NO	Read coordination number for data record	INT	0	I		
D_EL_SEL	D-action input	INT	0	I		
DEADB_W	Dead band width	REAL	0	I	+	
DISV	Disturbance variable	REAL	0	O		
EN_CO	Current coordination number	STRUCT		IO		
ENCO	Coordination number "Enable Coordination" in CFC	BYTE	16#00	O		
ER	Error signal	REAL	0	O		
FUZID_ON	Fuzzy identification on	BOOL	0	I		
GAIN	Proportional gain	REAL	1	I	+	
H_ALM	High limit alarm	REAL	100	I	+	H_ALM > H_WRN > L_WRN > L_ALM
H_WRN	Low limit warning	REAL	90	I	+	H_ALM > H_WRN > L_WRN > L_ALM
HYS	Hysteresis	REAL	1	I	+	>= 0

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
L_ALM	Low limit alarm	REAL	0	I	+	H_ALM > H_WRN > L_WRN > L_ALM
L_WRN	Low limit warning	REAL	10	I	+	H_ALM > H_WRN > L_WRN > L_ALM
LADDR	Logical address FM 355	INT	0	I		
LIOP_INT_SEL	1 = interconnection active, 0 = operator control active	BOOL	0	I		
LIOP_MAN_SEL	1 = interconnection active, 0 = operator control active	BOOL	0	I		
LMN	Manipulated value	REAL	0	O	+	
LMN_A	Manipulated value A: Split-range function/position feedback	REAL	0	O		
LMN_B	Manipulated value B: Split-range function	REAL	0	O		
LMN_HLM	Manipulated value high limit	REAL	100	I	+	
LMN_LLM	Manipulated value low limit	REAL	0	I	+	
LMN_OP	Operator input for manipulated variable	REAL	0	IO	+	
LMN_RE	External manipulated variable	REAL	0	I		
LMN_REON	Enable external setpoint	BOOL	0	I		
LMN_SAFE	Safety manipulated value	REAL	0	I	+	
LMNDN	Interconnected control signal low	BOOL	0			
LMNDN_OP	Adjustable control signal low	BOOL	0	IO		
LMNOP_ON	1 = Operator may input LMN_OP	BOOL	0	IO		
LMNRHSRE	High limit stop signal of position feedback	BOOL	0	I		
LMNRLSRE	Low limit stop signal of position feedback	BOOL	0	I		
LMNRS_ON	Turn on simulation of position feedback	BOOL	0	I		
LMNRSVAL	Start value of simulated position feedback	REAL	0	I		
LMNS_ON	Interconnected control signals ON (LMNDN, LMNUP)	BOOL	0	I		
LMNSOPON	Activate operator control enable for control signals	BOOL	0	I		
LMNTRKON	Tracking (manipulated variable via analog input)	BOOL	0	I		
LMNUP	Interconnected control signal high	BOOL	0	I		
LMNUP_OP	Adjustable control signal high	BOOL	0	IO		
M_SUP_AH	1 = Message suppression: High limit alarm	BOOL	0	I	+	
M_SUP_AL	1 = Message suppression: Low limit alarm	BOOL	0	I	+	
M_SUP_WH	1 = message suppression high warning	BOOL	0	I	+	
M_SUP_WL	1 = message suppression low warning	BOOL	0	I	+	
MANOP_EN	1 = operator control enable for MANUAL	BOOL	0	I		
MO_PVHR	High display limit (measuring range)	REAL	110	I	+	

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
MO_PVLR	Low display limit (measuring range)	REAL	-10	I	+	
MODE	Mode	DWORD	0			
MODE_CS	Mode: 0 = continuous controller, 1 = step controller	BOOL	0	I	+	
MONERSEL	Monitoring: 0 = process value, 1 = error signal	BOOL	0	I		
MSG_ACK	Acknowledge messages	WORD	0	O		
MSG_EVID	Message number	DWORD	0	I		
MSG_LOCK	1 = message suppression dependent on process state	BOOL	0	I	+	
MSG_STAT	Message error information	WORD	0	O		
MTR_TM	Motor actuating time in seconds	REAL	60	I		
OCCUPIED	ID for occupied by BATCH	BOOL	0	I	+	
OOS	Reserve	BOOL	0	I	+	
OP_SEL	1 = operator control via OP ON, 0 = OFF	BOOL	0	I	+	
OPTI_EN	1 = controller tuning on, 0 = off	BOOL	0	I	+	
P_SEL	Activate P action	BOOL	1	I		
PFDB_SEL	P action in feedback path	BOOL	0	I		
PULSE_TM	Minimum pulse duration in seconds	REAL	2	I		
PV	Process value	REAL	0	O	+	
Q_SP_OP	1 = operator control enable for setpoint activated	BOOL	0	O	+	
QAUTOP	1 = operator-control enable for AUTO mode activated	BOOL	0	O	+	
QC_LMN	Quality code for LMN	BYTE	16#80	O		
QC_PV	Quality code for PV	BYTE	16#80	I		
QCH_F	Channel error	BOOL	0	O		
QDNRLM	Negative setpoint ramp limit triggered	BOOL	0	O		
QDONE	1 = parameter read	BOOL	0	O		
QERR	Inverted ENO	BOOL	1	O		
QFUZZY	0 = PID algorithm 1 = Fuzzy	BOOL	0	O		
QID	1 = Fuzzy identification in process	BOOL	0	O		
QH_ALM	High limit: Alarm triggered	BOOL	0	O		
QH_WRN	High limit: Warning triggered	BOOL	0	O		
QL_ALM	Low limit: Alarm triggered	BOOL	0	O		
QL_WRN	Low limit: Warning triggered	BOOL	0	O		
QLMN_HLM	High limit of manipulated value triggered	BOOL	0	O	+	
QLMN_LLM	Low limit of manipulated value triggered	BOOL	0	O	+	
QLMN_RE	0 = manual, 1 = auto	BOOL	0	O		
QLMNDN	Manipulated value low	BOOL	0	O		
QLMNOP	1 = operator control enable for adjusting value of manipulated variable activated	BOOL	0	O	+	
QLMNOPON	Manipulated variable operation turned on	BOOL	0	O		
QLMNR_HS	High limit stop signal of position feedback	BOOL	0	O		

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
QLMNR_LS	Low limit stop signal of position feedback	BOOL	0	O		
QLMNR_ON	Position feedback is activated	BOOL	0	O		
QLMNS_ON	1 = LMNS_ON active	BOOL	0	O		
QLMNSAFE	Safety mode	BOOL	0	O		
QLMNSOP	Operator control enable for control signal activated	BOOL	1	O	+	
QLMNTRK	Tracking mode	BOOL	0	O		
QLMNUP	Control signal high	BOOL	0	O		
QLMNVOP	Operator control enable for control signal LMN_OP activated	BOOL	1	O	+	
QMAN_AUT	0 = manual, 1 = auto	BOOL	0	O	+	
QMAN_FC	1 = tracking mode or anti-reset windup by secondary controller	BOOL	0	O		
QMANOP	1 = operator control enable for manual mode activated	BOOL	0	O	+	
QMODF	1 = module fault	BOOL	0	O		
QMSG_ERR	1 = message error	BOOL	0	O	+	
QMSG_SUP	1 = message suppression active	BOOL	0	O	+	
QOP_ERR	1 = group operator input error	BOOL	0	O		
QOP_SEL	Operator control via OP: 0 = Off, 1 = On	BOOL	0	O		
QPARF_FM	1 = direct module parameter assignment error or CHANNEL false	BOOL	0	O		
QPERAF	1 = I/O access error	BOOL	0	O		
QSP_HLM	1 = high limit setpoint triggered	BOOL	0	O		
QSP_LLM	1 = low limit setpoint triggered	BOOL	0	O		
QSPEXTEN	1 = operator control enable for external activated	BOOL	0	O	+	
QSPINTEN	1 = operator control enable for internal activated	BOOL	0	O	+	
QSPINTON	Internal setpoint ON	BOOL	0	O	+	
QSPLEPV	Fuzzy controller display: Setpoint < process value	BOOL	0	O		
QSPOPON	Setpoint adjustment activated	BOOL	0	O		
QSPR	Split-range mode	BOOL	0	O		
QUPRLM	Limit of positive setpoint ramp	BOOL	0	O		
RACK_NO	Rack number	BYTE	255	I		
RET_VALU	Return value of RD_REC	WORD	0	O		
RUNUPCYC	Number of run-up cycles	INT	3	I		
SAFE_ON	Adopt safety position	BOOL	0	I		
SDB_SEL	1 = the SDB parameters are not adopted by the module at STOP-RUN transition of the CPU	BOOL	1	I	+	
SLOT_NO	Slot number	BYTE	255	I		
SP	Setpoint	REAL	0	O	+	
SP_EXT	External setpoint	REAL	0	I		

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
SP_HLM	Setpoint high limit	REAL	100	I	+	
SP_INT	Internal setpoint	REAL	0	I		
SP_LLM	Setpoint low limit	REAL	0	I	+	
SP_OP	Manual setpoint	REAL	0	IO	+	
SP_OP_ON	Operator control enable for manual setpoint SP_ON	BOOL	1	IO		
SP_TRK_ON	1 = SP_OP tracks PV	BOOL	0	I	+	
SPBUMPON	1 = bumpless setpoint	BOOL	1	I	+	
SPEXON_L	Interconnectable input for SP_EXT, 1 = SP_EXT is active	BOOL	0	I		
SPEXT_EN	1 = operator control enable for external setpoint	BOOL	0	I		
SPEXTSEL_OP	Operator input: 0 = internal setpoint, 1 = external setpoint	BOOL	0	IO	+	
SPINT_EN	1 = operator control enable for internal setpoint	BOOL	0	I		
STEP_NO	BATCH step number	DWORD	0	I	+	
SUBN1_ID	ID of the primary DP master system	BYTE	255	I		
SUBN2_ID	ID of the redundant DP master system	BYTE	255	I		
TD	Differential time in seconds	REAL	0	I	+	0 or >= 1.0
TI	Integration time in seconds	REAL	3000	I	+	0 or >= 0.5
TM_LAG	Time lag of the derivative action (s)	REAL	5	I	+	0 or >= 0.5
USTATUS	Status word in VSTATUS; can be configured user-specific	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

Parameters with the same names as those related to FB "PID_CS" also have the same meanings. You will find more information in the manual *Controller Module FM 355, Structuring and Configuring*.

3.10.15 Message Texts and Associated Values of FMCS_PID

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QPERAF/QMODF	@6%d@/@7%d@/@8%d@ error controller channel @5%d@	S	-
2	QPARF_FM	@6%d@/@7%d@/@8%d@ Configuration error controller channel @5%d@	S	-
3			No message	-
4			No message	-
5	QH_ALM	$$BlockComment$$ HighHigh Alarm	AH	M_SUP_AL, MSG_LOCK
6	QH_WRN	$$BlockComment$$ High alarm	WH	M_SUP_ER, MSG_LOCK
7	QL_WRN	$$BlockComment$$ Low alarm	WL	M_SUP_ER, MSG_LOCK
8	QL_ALM	$$BlockComment$$ LowLow alarm	AL	M_SUP_AL, MSG_LOCK

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data, the fourth is reserved for the process value and the fifth for the controller channel number. The remaining associated values can be assigned freely.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	PV
5	CHANNEL
6	SUBNET_ID
7	RACK_NO
8	SLOT_NO
9	AUX_PR09
10	AUX_PR10

Note

The FM 355 can be configured in HW Config by means of a configuration tool. Other than the standard counting method 0 to n in PCS 7, this configuration tool counts the controller and signal channels on a module from 1 to n. If there is an FM 355 hardware error, the displayed channel number in the message text is therefore one number lower than in the configuration tool.

Example:

12.10.2002 10.20 Origin FM 355 Wire Break AE Channel 02 Entering State.

However, this corresponds to wire break of the analog input channel 3 in the configuration tool.

3.10.16 VSTATUS for FMCS_PID

The 32-bit status word extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	-	-	-	QSPINTON	QMAN_AUT	MSG_LOCK	BA_EN	OCCUPIED
Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	-	-	-	-	-

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.10.17 Operating and Monitoring of FMCS_PID

Additional information

You will find more information in the following sections:

- FMCS_PID block icon (Page 585)
- FMCS_PID faceplate (Page 518)

3.11 FMT_PID: Temperature controller block

3.11.1 Description of FMT_PID

Object name (type + number)

FB77

- FMT_PID block I/Os (Page 162)
- FMT_PID block icon (Page 586)
- FMT_PID faceplate (Page 527)

Area of application

The FMT_PID block is used to integrate the FM 355-2 temperature control modules.

It can be used for the module types FM 355-2 C (C controller) and FM 355-2 S (S and P controller). It does not itself contain a control algorithm, since the PID control function is performed only on the module. You can use it to monitor all relevant process values and to change all relevant controller parameters. You can find sample applications for FM 355-2 and detailed descriptions of I/O parameters in the *Temperature-Controller Module FM 355-2* manual.

The controller parameters can be set via the "Auto-tuning" function on the module (TUN_ON = TRUE).

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The parameter CO_NO is set
- The input EN_CO is interconnected with the output EN_CO_x of the FM_CO block (x = number of the rack)

The output ENCO is connected to the input ENCOx_yy of the FM_CO block (x = number of the rack, yy = coordination number).

Using the FM 355-2 C Controller Module as a Continuous Controller (C controller)

The block provides the following displays and setting options:

- Display of the result of the limit monitoring carried out on the module for two limit pairs for the process value PV or the error signal ER (QH_ALM, QH_WRN, QL_WRN, QL_ALM outputs). MONERSEL is used to specify whether PV or ER is monitored.
- Disabling of the generation of individual messages when limits are exceeded
- Split-range function
- Dead band (DEADB_W, on threshold) in the error-signal branch
- PID algorithm
- Attenuation of the P-action to setpoint changes via PFAC_SP < 1.0 (avoids overshoot; optimized control and error response)
- Manipulated variable tracking
- Disabling of the integral action
- Setpoint tracking in manual mode (SP = PV)

Using the FM 355-2 S Controller Module as a Pulse Controller

Using the controller module as a pulse controller rather than a continuous controller means that split-range control is not possible.

You can use the pulse controller to generate pulse width modulated control signals. This variable is converted into a binary output signal, so that the ratio between the pulse width and configured period corresponds to the value of the manipulated variable LNM.

Using the FM 355-2 S Controller Module as a Step Controller

Using the controller module as a step controller rather than a continuous controller results in the following differences:

- Output QLMNR_ON indicates whether a feedback signal is available (1 = exists, 0 = does not exist).
- Split-range operation is not possible.

When used as a step-action controller without a position feedback (QLMNR_ON = 0), manual adjustment of the manipulated variable is only possible at the end positions. In this case the safety position LMN_SAFE or the external control value LMN_RE is interpreted by the controller module as follows:

Value < 40 %: Close actuating element completely

Value > 60 %: Open actuating element completely

40% ≤ Value ≤ 60%: Hold current setting

Calling OBs

Cyclic interrupt OB: For example OB 32

For startup detection, the block is also installed in OB100 by the driver generator.

Take note of the dependencies on the FM_CO block too.

Additional information

You will find more information in:

Addressing (Page 148)
Function (Page 149)
Acquisition of process values via the process image (Page 150)
Generation of setpoints, limits, error signals, and manipulated variables (Page 151)
Manual, auto, and tracking mode (Page 153)
Mode change (Page 155)
Safety mode (Page 157)
Download parameters to the module (Page 157)
Read data from the module/Working with the configuration tool (Page 157)
Optimization (overview) (Page 158)
Switching between different PID parameter sets (Page 159)
Error handling (Page 159)
Startup characteristics, time response, and message response (Page 160)
Backup mode of the FM 355-2 (Page 162)
I/Os of FMT_PID (Page 162)
Message texts and associated values of FMT_PID (Page 167)
VSTATUS for FMT_PID (Page 168)
Archiving process values (Page 613)
Operating and monitoring FMT_PID (Page 168)

3.11.2 Addressing

The controller channel of an FM 355 belonging to the instance is addressed via its logical base address (set in HW Config) (LADDR input) and the controller channel number (CHANNEL input, valid values = 1 to 4). 0 to 3 are the allowed values for the CHANNEL input. ACC_MODE must be set to TRUE after a change is made to CHANNEL.

The FM 355-2 module is monitored with the PCS7 blocks of the *PCS 7 Library*. The MODE input is interconnected with the OMODE output of the MOD_D1 block. The block communicates only via the FM 355-2 control channel. Hence, the measuring-range coding in the low word of the OMODE output is irrelevant and assigned zero.

3.11.3 Function of FMT_PID

The FMT_PID block forms the interface between the temperature controller module (FM 355-2) and the blocks of the SIMATIC PCS 7 Library. It can also be interconnected with other SIMATIC S7 blocks.

The block and the FM 355-2 operate asynchronously to each other.

All relevant process and disturbance variables are provided by the module and can only be read by the block. The block can also transfer different operating modes and settings to the controller module.

The block can be used to read and write the FM 355-2 operator control and control parameters. Each change to a parameter in the block is passed to the module.

The process values (except SP (setpoint from the FM), ER (error signal), DISV (disturbance), LMN_A and LMN_B, PHASE, STATUS_H, STATUS_C, STATUS_D and ZONE_TUN) are read cyclically. They are, however, only updated after every 4th cycle. You will find more information in "Acquisition and writing of process values via the process image (Page 150)":

The parameters SP (setpoint from the FM), ER (error signal), DISV (disturbance variable), LMN_A, LMN_B, PHASE, STATUS_H, STATUS_C, STATUS_D, and ZONE_TUN can only be read from the FM 355-2 using "Read data record" (SFC 59).

A quality code is generated for each process value PV and LMN and this can have the following states:

State	Quality code
Valid value	16#80
Invalid value	16#00

Parameter assignment

The FM 355-2 usually receives its parameters from the block. However, you can also bypass the block (using the configuration tool, *for example*). The parameters of FMT_PID are then updated automatically. This ensures that the parameters on the FM 355-2 and in the block are always synchronized.

Some of the parameters can be specified not only with the configuration tool but can be set in the block. These two parameter sets may differ. The SDB_SEL input of the block is available to avoid such a conflict. The setting SDB_SEL = 0 specifies that the module accepts these parameters from the parameter assignment tool and from the block. Note that the parameters are transferred from the parameter assignment tool to the module at each STOP to RUN transition of the CPU. The parameters of the block, on the other hand, are transferred to the module each time there is a change at the block input.

SDB_SEL = 1 (not activated internally because the module firmware does not support this function) defines that the module accepts these parameters only from the function block and not from the parameter assignment tool.

Note

After HW Config in Run (CiR) is downloaded, the block parameters are not adjusted to the active parameters on the module.

3.11.4 Acquisition and Writing of Process Values Via the Process Image

Reading process values

The process values (except SP (setpoint from the FM), ER (error), DISV (disturbance), LMN_A and LMN_B, PHASE, STATUS_H, STATUS_C, STATUS_D and ZONE_TUN) are read cyclically from the process image. They are, however, only updated after every 4th cycle. In a normal situation, this takes place in the cyclic interrupt OB in which the FMT_PID is installed. To achieve higher accuracy, it is possible to install the FMT_PID in a second cyclic interrupt OB with a higher clock rate.

After a restart or when ACC_MODE is set to 1, the block determines the cyclic interrupts used (up to 2 are permitted; you can find additional information in "Startup Characteristics".

The following must be considered when selecting cyclic interrupts:

- The cyclic-interrupt OB in which the block is installed must not run slower than 30 s or faster than 25 ms.

  Note: OB 1 must not be used!

- The process-image partition (TPA) in HW Config must be set for the faster OB.
- Reading via the process image for a block is complete after 4 block cycles (after 4 s in the example with OB 32 [1,000 ms]) if the block cycle is longer than the cycle time of the module, otherwise after 4 module cycles.
- **Example:** With 4 processed analog inputs on the module, the cycle time of the module is typically 400 ms (see "Module parameters" button on the parameter-assignment interface of the FM 355). If the block is installed in OB 32 (1,000 ms), for example, you can accelerate reading via the process image by means of an additional installation in OB 33 (500 ms).

Writing process values

The following process values are written to the process image in every second cycle:

- Setpoint (automatic mode only)
- Manipulated variable (manual mode only)
- SAFE_ON, LMNTRKON, LMN_REON, LMNRHSRE, LMNRLSRE
- LMNS_ON or TRUE if set by control signals
- LMNUP or LMNUP_OP
- LMNDN or LMNDN_OP

In the following cases, a restart must be performed or ACC_MODE must be set to 1 following a compile and download:

- If you have moved blocks to a different cycle
- If you have also installed blocks in a fast cycle
- If you have also deleted installed blocks

3.11.5 Generation of Setpoints, Limits, Error Signals, and Manipulated Variables

Setpoint Generation by the FMT_PID Block

Setpoint SP can be fetched from three different sources, depending on the SP_TRK_ON and SPEXTSEL_OP inputs. You can find additional information about the external/internal setpoint in "Mode Change (Page 155)":

SP_TRK_ON	SPEXTSEL_OP	SP=	State
0	0	SP_OP	Internal (operator-controlled) setpoint
Irrelevant	1	SP_EXT	External setpoint
1	0	PV **	Tracked setpoint
		** in manual mode only and when SPBUMPON = 1	

The effective setpoint is limited to the range (SP_LLM, SP_HLM).

When SP_TRK_ON is set the SP_OP setpoint will be tracked in manual mode (for internal setpoint and when SPBUMPON = 1). This enables bumpless changeover from manual to automatic mode.

Generation of limits

Depending on the input MONERSEL, the controller module monitors either the process value PV (MONERSEL = 0) or the error signal ER (MONERSEL = 1) for warning and alarm limits (L_WRN, H_WRN, L_ALM, H_ALM). Monitoring is carried out with the common hysteresis HYS.

The block makes the monitoring result available at the QL_WRN, QH_WRN, QL_ALM, and QH_ALM outputs. While monitoring the process value PV, the block signals any violation of the high and low limits, unless message suppression has been enabled.

Generation of error signals

The error signal is generated by the controller module, based on the active setpoint SP and the process value PV, and is made available at output ER of the block.

After the dead band DEADB_W has expired, the error signal is processed further in the PID algorithm. A disturbance variable is not added.

Generation of manipulated variables by the FMT_PID block

The manipulated variable LMN is derived from various sources. If several control inputs are set to TRUE simultaneously, the priority is as follows:

Priority	Control input	LMN	State
1	LMNS_ON = 1	LMNUP or LMNDN	Interconnected control signals for step controller
2	SAFE_ON = 1	= LMN_SAFE	Safety mode
3	TUN_ON = 1 **	= LMN_{old} + TUN_DLMN	Optimization mode (PHASE = 2)
4	LMNTRKON = 1	= analog input on module or 0.0	Tracking mode
5	Manual	= LMN_OP	Manual mode (QMAN_AUT=0)
6	LMN_REON = 1	= LMN_RE	External manipulated variable
7	Automatic	= PID algorithm	Automatic mode (QMAN_AUT=1)

** The optimization also has to be set by a step change in the setpoint or TUN_ST/TUN_CST = 1 in phase 2.

- If LIOP_MAN_SEL = 0, the AUT_ON_OP parameter is used to change over between manual and automatic mode on the OS.
- If LIOP_MAN_SEL = 1, the AUT_L parameter is used to change over between manual and automatic mode via an interconnection in the CFC.

In step controllers the manipulated variable is converted into control pulses (QLMNUP; QLMNDN) under consideration of the motor-specific parameters motor actuating time (MOTOR_TM), minimum pulse duration (PULSE_TM), and minimum break duration (BREAK_TM).

3.11.6 Manual, auto and tracking mode of FMT_PID

Manual mode

The manipulated variable is set by the operator at input LMN_OP (jogging mode is also possible here). When changing over to automatic mode, the module takes over the manipulated variable set "manually" as the operating point.

"Manual mode" takes priority over "External setpoint".

Manual mode for step controller:

Actuating commands can be used for direct control of the control signals for step controllers. Signal manipulation is enabled with LMNSOPON; the control signals are set at LMNUP_OP or LMNDN_OP. The valve is driven until the command is disabled or the end position is reached.

The operator adjustment of the control signal has priority over adjustment of the manipulated value via LMN_OP.

The LMN_OP input cannot be controlled when step controllers without position feedback are used.

Automatic mode

The manipulated variable is calculated by the PID algorithm of the module. The control parameters GAIN, TI, TD and TM_LAG can be interconnected.

The controller's direction of control can be reversed (rising error signal causes a falling manipulated variable) by setting a negative proportional GAIN.

The I action can be deactivated by setting TI = 0.

The operator-controllable manipulated variable input LMN_OP tracks the LMN output so that bumpless changeover from automatic to manual mode is ensured.

External setpoint (LMN_RE)

The block transfers the value LMN_RE to the FM 355-2. The FM 355-2 uses the external manipulated variable LMN_RE as the manipulated variable LMN, if LNM_REON = 1 is set.

"External manipulated variable" mode takes priority over "Automatic mode".

Manipulated variable tracking

In manipulated variable tracking mode (LMNTRKON = 1), the manipulated variable tracks an analog input of the module or the value 0.0

"Tracking" mode takes priority over "Manual mode".

Safety mode (LMN_SAFE)

The block transfers the value LMN_SAFE to the FM 355-2. The FM 355-2 accepts the safety manipulated variable LMN_SAFE as manipulated variable LMN, if SAFE_ON = 1.

"Safety mode" takes priority over "tracking" mode.

Interconnectable control signal setting for the step controller

The tracking mode using LMNS_ON with direct connection of the control signals over interconnected inputs LMNUP and LMNDN has the highest priority of all modes. If LMNS_ON is set, the control signals can only be set via the inputs LMNUP or LMNDN. Any other influence on the control signals is suppressed as long as LMNS_ON is set.

3.11.7 Mode change of FMT_PID

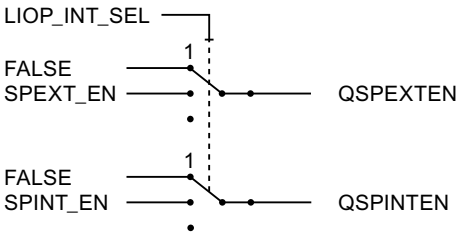
The mode change can be triggered either by the operator or via interconnected inputs. The mode is changed by means of the operator control blocks assigned to the modes.

External/internal setpoint

A changeover between external and internal setpoint is initiated by the OS operator setting the SPEXTSEL_OP input by interconnecting SPEXON_L. You must set the corresponding enable inputs SPINT_EN, SPEXT_EN or the selection input LIOP_INT_SEL to enable these changeovers.

SP_OP_ON must be set TRUE to enable operator control of the setpoint.

Enabling changeover between internal and external setpoint



QSPEXTEN = TRUE: SPEXTSEL_OP can be changed from FALSE (internal setpoint) to TRUE (external setpoint).

QSPINTEN = TRUE: SPEXTSEL_OP can be changed from TRUE (external setpoint) to FALSE (internal setpoint).

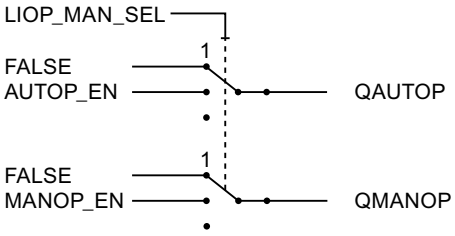
SPEXTSEL_OP tracks or is reset as required.

Manual/auto

The operating mode is changed over by the OS operator setting the AUT_ON_OP input or by interconnection of AUT_L. You must set the corresponding enable inputs MANOP_EN, AUTOP_EN or the selection input LIOP_MAN_SEL to enable this changeover.

LMNOP_ON must be set TRUE to enable manual control of the variable.

Enabling the changeover between manual and automatic mode



QAUTOP = TRUE: AUT_ON_OP can be changed from FALSE (manual mode) to TRUE (automatic mode).

QMANOP = TRUE: AUT_ON_OP can be changed from TRUE (automatic mode) to FALSE (manual mode).

AUT_ON_OP tracks or is reset as required.

Measures for bumpless changeover

Special measures are applied for the modes listed below in order to ensure a bumpless changeover:

- Setpoint external/setpoint tracking:
  If SPBUMPON = TRUE, the internal setpoint SP_OP is set to the effective (external or tracking) setpoint.
- Automatic mode, safety mode,
  tracking mode or external manipulated variable LMN_RE: The manual value LMN_OP tracks the effective manipulated variable.
- Manual mode, safety mode, tracking mode or external manipulated variable LMN_RE:
  The integrator tracks so that a bumpless changeover to automatic mode is possible (not with a step controller). Using the configuration tool, you can disable the default bumpless changeover function. The error signal can then be corrected more quickly.

3.11.8 Safety mode

The interconnectable input SAFE_ON is used to set safety mode. This is executed by the controller module with highest priority. In safety mode, the value present at the LMN_SAFE input of the block is output at the control output.

3.11.9 Download Parameters to the Module

The channel-specific controller and operating parameters are transferred to the controller module whenever a corresponding block parameter changes.

The download of controller and operating parameters to the controller module may require several block calls.

3.11.10 Read Data From the Module/Working with the Configuration Tool

The channel-specific process values are read by the controller module. You can find additional information in "Function (Page 149)". Particularly in distributed operation, reading may require several block calls.

You can also bypass the block using the configuration tool, for example. The parameters of the FMT_PID are then read and updated automatically by the module. This ensures that the parameters on the FM 355-2 and in the block are always synchronized.

Note

Proceed according to the following sequence:

- Use the menu command **Upload to PG** to align the data management of the configuration tool with the parameters that are effective in the module.
- In the configuration tool, click the "Download to module" button. The parameters are written.
- Transfer the modified parameters to the offline data management of the CFC by selecting the menu command **Chart > Readback...**.

3.11.11 Optimization

Optimization sequence

The optimization sequence is as follows:

1. Create a stationary state
2. Set PID_ON = TRUE (if PID parameters are required)
3. Configure TUN_DLMN/TUN_CLMN
4. Set TUN_ON = TRUE (phase 1, ready for optimization)
5. Start the optimization using a step change in the setpoint or by setting TUN_ST

   If you have not made any configuration errors, the controller optimization is now in phase 2 and STATUS_H is 0.
6. When the point of inflection has been reached (PHASE >= 3), evaluate the diagnostics display at the STATUS_H 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 STATUS_H > 20000, a valuation error has occurred or the point of inflection has not been reached. In this case, repeat the procedure.

Result

Once the optimization is completed, the parameters PFAC_SP, GAIN, TI, TD, D_F, CON_ZONE, and CONZ_ON are updated (for both the module and at FMT_PID). Furthermore, the PI or PID parameter sets are saved on the FM 355-2.

More detailed information about the optimization procedure is available in the manual for the FM 355-2 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 by selecting the menu command **Chart > Readback...**.

3.11.12 Switching Between Different PID Parameter Sets

Another parameter set is stored on the FM 355-2 in addition to the effective PID parameter set. The parameter set is saved using SAVE_PAR and reset using UNDO_PAR. This affects the following parameters:

PFAC_SP, GAIN, TI, TD, D_F, CON_ZONE, RATIOFAC, CONZ_ON, P_SEL

Following optimization both the PI and the PID parameters sets are saved on the FM 355-2. These parameter sets are loaded by setting LOAD_PID. If PID_ON = TRUE, the PID parameter set is copied to the effective controller parameters, otherwise the PI parameter set is loaded. This affects the following parameters:

GAIN, TI, TD, CON_ZONE

3.11.13 Error handling of FMT_PID

The block supplies the following error displays:

Error display	Meaning
QOP_ERR = 1	Operator input error. If there is no new operator input error, QOP_ERR is reset in the next block cycle.
QPARF_FM = 1	• Error when assigning parameters directly to the controller module using the parameter assignment tool. • Invalid control channel number (CHANNEL) was set at the block. • Error when installing in OBs; you can find additional information in "Startup Behavior".
QCH_F = 1	Channel error. Due to a hardware fault the controller channel belonging to the instance cannot return valid results.
QMODF = 1	Controller module has been removed or is faulty.
QPERAF = 1	I/O access error. The block could not access the controller module.

3.11.14 Startup Behavior, Dynamic Response and Message response of FMT_PID

Startup characteristics

During CPU startup, startup of the FM or the first time the block is run, the operating modes MANUAL and INTERNAL are set and the controller parameters are written from the block to the module.

In OB100, QDONE is set to 0.

After a CPU restart or when ACC_MODE is set to 1, the block is not operable for up to approximately 30 seconds.

The block uses the first 30 seconds after a restart (OB100) or after ACC_MODE is set to 1 to detect whether the block instance is installed in more than one OB. If you then move the blocks to another OB, a restart must be performed or ACC_MODE must be set to 1 after compiling and downloading (otherwise you will receive the error message QPARF_FM = 1, and the block will not provide any more data).

Note

The configuration tool can be configured in HW Config with Save/Compile and Download to the SDB (system data). These SDB parameters can deviate from the block parameters. When SDB_SEL = 0, the SDB parameters are sent to the module at every STOP-RUN transition of the CPU. The block parameters are written to the module, however, a few cycles later. With SDB_SEL = 1, the module does not load the SDB parameters at a STOP-RUN transition of the CPU. This avoids a jump in the manipulated variable when the SDB parameters deviate from the block parameters.

Time response

Not available

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for FMT_PID (Page 168)".

Message response

The FMT_PID block uses the ALARM_8P block for generating messages.

The following message triggers exist:

- Monitoring of the process value or system deviation
- Module access error if no higher-level error is active

Limit violation messages can be suppressed individually using the the relevant M_SUP_xx inputs. Process messages (not control system messages!) can be disabled centrally using MSG_LOCK.

QMSG_SUP is set if the RUNUPCYC cycles were not completed since the last restart and if MSG_LOCK = TRUE or MSG_STAT = 21.

Monitoring process values

Not available

3.11.15 Backup Mode of the FM 355-2

After a CPU failure or transition to STOP, the FM 355-2 switches to backup mode.

3.11.16 I/Os of FMT_PID

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Valid values
ACC_MODE	SUBN1_ID, SUBN2_ID, RACK_NO, SLOT_NO and CHANNEL included in internal processing	BOOL	0	IO		
AUT_L	Interconnectable input for MANUAL/AUTO: 0 = manual; 1 = auto	BOOL	0	I		
AUT_ON_OP	Operator input: MANUAL/AUTO: 0 = manual; 1 = auto	BOOL	0	IO	+	
AUTOP_EN	1 = operator control enable for AUTO	BOOL	1	I		
AUX_PRxx	User-specific associated value 6 to 10	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: Consecutive batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	''	I	+	
BREAK_TM	Minimum break time in seconds	REAL	0	IO		
CHANNEL	Controller channel number	INT	0	I		
CO_NO	Coordination number for data record reading	INT	0	O		
CON_ZONE	Control zone	REAL	100	IO		
CONZ_ON	Control zone ON	BOOL	0	IO		
D_EL_SEL	D-action input	INT	0	IO		
D_F	Derivative factor	REAL	5	IO		
DEADB_W	Dead band width	REAL	0	IO	+	
DISV	Disturbance variable	REAL	0	O		
EN_CO	Current coordination number	STRUCT				
ENCO	Coordination number "Enable Coordination" in CFC	BOOL	0	IO		
ER	Error signal	REAL	0	O		
GAIN	Proportional gain	REAL	1	IO	+	
H_ALM	High limit alarm	REAL	100	IO	+	H_ALM > H_WRN > L_WRN > L_ALM
H_WRN	Low limit warning	REAL	95	IO	+	H_ALM > H_WRN > L_WRN > L_ALM

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Valid values
HYS	Hysteresis	REAL	1	IO	+	>= 0
L_ALM	Low limit alarm	REAL	-5	IO	+	H_ALM > H_WRN > L_WRN > L_ALM
L_WRN	Low limit warning	REAL	-3	IO	+	H_ALM > H_WRN > L_WRN > L_ALM
LADDR	Logical address of FM 355-2	INT	0	I		
LIOP_INT_SEL	1 = interconnection active, 0 = operator control active	BOOL	0	I		
LIOP_MAN_SEL	1 = interconnection active, 0 = operator control active	BOOL	0	I		
LMN	Manipulated variable	REAL	0	O	+	
LMN_A	Manipulated variable A of split-range function/position feedback	REAL	0	O		
LMN_B	Manipulated variable B of split-range function	REAL	0	O		
LMN_HLM	Manipulated value high limit	REAL	100	IO	+	
LMN_LLM	Manipulated value low limit	REAL	0	IO	+	
LMN_OP	Operator input for manipulated variable	REAL	0	IO	+	
LMN_RE	External manipulated variable	REAL	0	I		
LMN_REON	Activate external manipulated variable	BOOL	0	I		
LMN_SAFE	Safety manipulated value	REAL	0	IO	+	
LMNDN	Interconnected control signal low	BOOL	0	I		
LMNDN_OP	Control signal low operation	BOOL	0	IO		
LMNOP_ON	1 = Operator may input LMN_OP	BOOL	1	IO		
LMNRHSRE	High limit stop signal of position feedback	BOOL	0	I		
LMNRLSRE	Low limit stop signal of position feedback	BOOL	0	I		
LMNS_ON	Interconnected control signals on (LMNDN, LMNUP)	BOOL	0	I		
LMNSOPON	Activate operator control enable for control signals	BOOL	0	IO		
LMNTRKON	Tracking (manipulated variable via analog input)	BOOL	0	I		
LMNUP	Interconnected control signal high	BOOL	0	I		
LMNUP_OP	Adjustable control signal high	BOOL	0	IO		
LOAD_PID	Load optimized PI/PID parameters	BOOL	0	IO		
M_SUP_AH	1 = message suppression high limit alarm	BOOL	0	I	+	
M_SUP_AL	1 = message suppression low limit alarm	BOOL	0	I	+	
M_SUP_WH	1 = message suppression high limit warning	BOOL	0	I	+	
M_SUP_WL	1 = message suppression low limit warning	BOOL	0	I	+	
MANOP_EN	1 = operator control enable for MANUAL	BOOL	1	I		
MO_PVHR	High display limit (measuring range)	REAL	110	I	+	

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Valid values
MO_PVLR	Low display limit (measuring range)	REAL	-10	I	+	
MODE	Mode	DWORD	0	I		
MONERSEL	Monitoring: 0 = process value, 1 = error signal	BOOL	0	IO		
MSG_ACK	Message acknowledged	WORD	0	O		
MSG_EVID	Message number	DWORD	0	I		
MSG_LOCK	1 = Message suppression dependent on process state	BOOL	0	I	+	
MSG_STAT	Message error information	WORD	0	O		
MTR_TM	Motor actuating time in seconds	REAL	60	IO		
OCCUPIED	ID for occupied by BATCH	BOOL	0	I	+	
OOS	Reserve	BOOL	0	I	+	
P_SEL	Activate P action	BOOL	1	IO		
PFAC_SP	Proportional gain	REAL	1	IO		
PHASE	Phase of auto-tuning (0..7)	INT	0	O		
PID_ON	1 = activate PID mode	BOOL	0	IO		
PULSE_TM	Minimum pulse duration in seconds	REAL	0	IO		
PV	Process value	REAL	0	O	+	
Q_SP_OP	1 = operator control enable for setpoint is activated	BOOL	0	O	+	
QAUTOP	1 = operator control enable for AUTO is activated	BOOL	0	O	+	
QC_LMN	Quality code for LMN	BYTE	16#80	O		
QC_PV	Quality code for PV	BYTE	16#80	O		
QCH_F	Channel error	BOOL	0	O		
QDNRLM	Negative setpoint ramp limit triggered	BOOL	0	O		
QDONE	1 = parameter read	BOOL	0	O		
QERR	Negated value of ENO	BOOL	1	O		
QH_ALM	High limit alarm triggered	BOOL	0	O		
QH_WRN	High limit warning triggered	BOOL	0	O		
QL_ALM	Low limit alarm triggered	BOOL	0	O		
QL_WRN	Low limit warning triggered	BOOL	0	O		
QLMN_HLM	Manipulated variable high limit triggered	BOOL	0	O	+	
QLMN_LLM	Manipulated variable low limit triggered	BOOL	0	O	+	
QLMN_RE	1 = external manipulated value activated	BOOL	0	O		
QLMNDN	Manipulated-variable signal low	BOOL	0	O		
QLMNOP	1 = operator-control enable for setting manipulated variable activated	BOOL	0	O	+	
QLMNR_HS	High limit stop signal of position feedback	BOOL	0	O		
QLMNR_LS	Low limit stop signal of position feedback	BOOL	0	O		
QLMNR_ON	Position feedback is activated	BOOL	0	O		
QLMNS_ON	1 = LMNS_ON activated	BOOL	0	O		
QLMNSAFE	Safety mode	BOOL	0	O		

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Valid values
QLMNSOP	Operator control enable for control signals is activated	BOOL	1	O	+	
QLMNTRK	Tracking mode	BOOL	0	O		
QLMNUP	Control signal high	BOOL	0	O		
QLMNVOP	Operator-control enable for manipulated variable LMN_OP is activated	BOOL	1	O	+	
QMAN_AUT	0 = manual, 1 = auto	BOOL	0	O	+	
QMAN_FC	1 = tracking mode or anti-reset windup by secondary controller	BOOL	0	O		
QMANOP	1 = operator-control enable for manual mode is activated	BOOL	0	O	+	
QMODF	1 = module error	BOOL	0	O		
QMSG_ERR	1 = message error	BOOL	0	O	+	
QMSG_SUP	1 = message suppression active	BOOL	0	O	+	
QOP_ERR	1 = group operator-input error	BOOL	0	O		
QPAR_ACT	1 = update controller parameters	BOOL	0	O		
QPARF_FM	1 = direct module-parameter-assignment error or CHANNEL false	BOOL	0	O		
QPERAF	1 = I/O access error	BOOL	0	O		
QSP_HLM	1 = high limit setpoint triggered	BOOL	0	O		
QSP_LLM	1 = low limit setpoint triggered	BOOL	0	O		
QSPEXTEN	1 = operator-control enable for external activated	BOOL	0	O	+	
QSPEXTON	External setpoint activated	BOOL	0	O		
QSPINTEN	1 = operator-control enable for internal activated	BOOL	0	O	+	
QSPR	Split-range mode	BOOL	0	O		
QSTEPCON	1 = step controller	BOOL	0	O		
QTUN_ON	1 = Tuning running	BOOL	0	O		
QUPRLM	1 = setpoint ramp rate limit triggered	BOOL	0	O		
RACK_NO	Rack number	BYTE	255	I		
RATIOFAC	Ratio factor	REAL	0	IO		
RET_VALU	Return value of SFC 58/59 (WR_REC/RD_REC)	WORD	0	O		
RUNUPCYC	Number of run-up cycles	INT	3	I		
SAFE_ON	Adopt safety position	BOOL	0	I		
SAVE_PAR	Save control parameters	BOOL	0	IO		
SDB_SEL	1 = the SDB parameters are not adopted by the module at STOP-RUN transition of the CPU	BOOL	1	I	+	
SLOT_NO	Slot number	BYTE	255	I		
SP	Setpoint	REAL	0	O	+	
SP_EXT	External setpoint	REAL	0	I		
SP_HLM	Setpoint high limit	REAL	100	IO	+	

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Valid values
SP_LLM	Setpoint low limit	REAL	0	IO	+	
SP_OP	Manual setpoint	REAL	0	IO	+	
SP_OP_ON	Operator control enable for manual setpoint SP_ON	BOOL	1	I		
SP_TRK_ON	1 = SP_OP tracks PV	BOOL	0	I	+	
SPBUMPON	1 = bumpless setpoint	BOOL	1	I	+	
SPEXON_L	Interconnectable input for SP_EXT, 1 = SP_EXT is active	BOOL	0	I		
SPEXT_EN	1 = operator control enable for external setpoint	BOOL	1	I		
SPEXTSEL_OP	Operator input:0: internal setpoint, 1 = external setpoint	BOOL	0	IO	+	
SPINT_EN	1 = operator control enable for internal setpoint	BOOL	1	I		
STATUS_C	Status of cooling tuning	INT	0	O		
STATUS_D	Status of controller design	INT	0	O		
STATUS_H	Status of heating tuning	INT	0	O		
STEP_NO	BATCH step number	DWORD	0	I	+	0 or >= 1.0
SUBN1_ID	ID of the primary DP master system	BYTE	255	I		
SUBN2_ID	ID of the redundant DP master system	BYTE	255	I		
TD	Differential time in seconds	REAL	0	IO		0 or >= 0.5
TI	Integration time in seconds	REAL	3,000	IO		
TUN_CLMN	Delta manipulated value for cooling tuning	REAL	-20	IO		
TUN_CST	Start cooling tuning	BOOL	0	IO		
TUN_DLMN	Delta manipulated value for process excitation	REAL	20	IO		
TUN_ON	Enable controller tuning	BOOL	0	IO		
TUN_ST	Start controller tuning	BOOL	0	IO		
UNDO_PAR	Undo controller parameter changes	BOOL	0	IO		
USTATUS	Status word in VSTATUS; can be configured user-specific	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	
ZONE_TUN	Controller channels grouped in one zone for parallel tuning	WORD	0	O		

You can find additional information about the FM 355-2 parameters in the *Temperature Controller FM 355-2* manual.

3.11.17 Message Texts and Associated Values of FMT_PID

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QPERAF/QMODF	@6%d@/@7%d@/@8%d@ error controller channel @5%d@	S	-
2	QPARF_FM	@6%d@/@7%d@/@8%d@ Configuration error controller channel @5%d@	S	-
3			No message	-
4			No message	-
5	QH_ALM	$$BlockComment$$ HighHigh Alarm	AH	M_SUP_AL, MSG_LOCK,
6	QH_WRN	$$BlockComment$$ High alarm	WH	M_SUP_ER, MSG_LOCK,
7	QL_WRN	$$BlockComment$$ Low alarm	WL	M_SUP_ER, MSG_LOCK,
8	QL_ALM	$$BlockComment$$ LowLow alarm	AL	M_SUP_AL, MSG_LOCK

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data, the fourth is reserved for the process value, and the fifth for the controller-channel ID. The remaining associated values can be assigned freely.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	PV
5	CHANNEL
6	SUBNET_ID
7	RACK_NO
8	SLOT_NO
9	AUX_PR09
10	AUX_PR10

3.11.18 VSTATUS for FMT_PID

The 32-bit statusword extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	-	-	-	QSPEXTON	QMAN_AUT	MSG_LOCK	BA_EN	OCCUPIED
Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	-	-	-	-	-

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.11.19 Operating and Monitoring of FMT_PID

Additional information

You can find additional information in the following sections:

- FMT_PID block icon (Page 586)
- FMT_PID faceplate (Page 527)

3.12 GAIN_SHD - gain scheduling

3.12.1 Description of GAIN_SHD

Object name (type + number)

FB 141

- GAIN_SHD block I/Os (Page 173)

Area of application of GAIN_SHD

The block is used for the following applications:

- Adaptation of parameter values for the
  - Controller gain
  - Integral action time
  - Derivative action time
- Continuous adaptation to the current operating point of a non-linear process

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 the GAIN_SHD block in the form of a table ("timetable"). The current operating point is represented by a continuously measurable variable X, typically by the process 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 GAIN_SHD faceplate is called from the parameter view of the corresponding PID controller using the "GAIN_SHD" button.

Internally, the GAIN_SHD block consists essentially of three instances of the POLYG_P block. In contrast to all other function blocks, it is implemented as a CFC chart and is generated with the "Compile chart as block type" function. The source chart "fbGAIN_SHD" is supplied with the library so that you have several options open to you:

- You can use the precompiled function block GAIN_SHD 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.

Configuration

The GAIN_SHD block is placed in the same CFC chart as the assigned controller and interconnected with it as shown in the corresponding template: The outputs Link2Gain, Link2TI, Link2TD and Link2TM_LAG are connected to the inputs GAIN, TN, TV and TM_LAG of the PID controller. The X input of GAIN_SHD is supplied with the measured value for the operating point, typically with the same value as PV_IN of the controller.

The GAIN_SHD instance in the CFC is connected to the instance of the assigned controller by a naming convention: The name of the GAIN_SHD instance in the CFC chart is exactly the same as the name of the PID instance with "_gsc" added.

Example:

This means that monitoring block TIC501_gsc would belong to the controller TIC501. The "GAIN_SHD" button in the controller faceplate is visible only for controllers for which there is a gain scheduling block that keeps to this naming convention.

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 GAIN_SHD block. The table is clearly displayed in the standard view of the GAIN_SHD 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 enter them manually at the inputs of the CFC block.

Note: 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 GAIN_SHD 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 GAIN_SHD 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 to 3 is currently required from the GAIN_SHD block. However, the numeric values of the parameters are not anchored in the recipe. In this case, input parameter X of the GAIN_SHD block can be used as the number of the required data record and assigned by the recipe instead of being linked with a measurable process value. In this case, there are only three fixed 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 GAIN_SHD block are not used.

In general, it is not necessary to manage the batch parameters (batch number, batch name, etc.) in the GAIN_SHD block because the block does not generate any separate messages and there is always a 1:1 relationship with a controller block that knows the batch parameters.

Startup characteristics

The block does not have any startup characteristics.

Time response

The block does not have any time response.

Message response

The block does not have any message response.

See also

Functions of GAIN_SHD (Page 172)

3.12.2 Functions of GAIN_SHD

Functions of GAIN_SHD

The block provides the following functions.

- Manual setting of the controller parameters

Manual setting of the controller parameters

For controllers without gain scheduling, the numeric values of the controller parameters can be changed in the parameter view of the controller faceplate. When a GAIN_SHD block is used, the parameters are, however, interconnected and therefore no longer accessible in the faceplate. If you want to enter controller parameter manually, independent of the values specified in the gain scheduling table, put a check mark in "Manual parameter assignment" in the standard view of the GAIN_SHD faceplate. The parameter values in the input fields, gain factor, integration time, and derivative time are the passed on to the controller.

See also

GAIN_SHD I/Os (Page 173)

Description of GAIN_SHD (Page 169)

3.12.3 GAIN_SHD I/Os

GAIN_SHD I/Os

Inputs

I/O (parameters)	Description	Data type	Default Value	Attributes
Gain1	PID gain for operating point 1	REAL	1	S7_m_c := 'true' S7_link := 'true'
Gain2	PID gain for operating point 2	REAL	1	S7_m_c := 'true' S7_link := 'true'
Gain3	PID gain for operating point 3	REAL	1	S7_m_c := 'true' S7_link := 'true'
GainOp	PID gain: Input for manual mode	REAL	1	S7_m_c := 'true' S7_link := 'true'
ManParOn	1 = input of PID parameters in manual mode 0 = planning gain in auto mode	BOOL	0	S7_m_c := 'true'
TD_Op	PID derivative time is constant [s]: Manual input for the operator	REAL	0	S7_m_c := 'true'
TD1	PID derivative time is constant [s] for operating point 1	REAL	0	S7_m_c := 'true' S7_link := 'true'
TD2	PID derivative time is constant [s] for operating point 2	REAL	0	S7_m_c := 'true' S7_link := 'true'
TD3	PID derivative time is constant [s] for operating point 3	REAL	0	S7_m_c := 'true' S7_link := 'true'
TI_Op	Time for which PID will be integrated [s]: Manual input for the operator	REAL	10	S7_m_c := 'true'
TI1	Time for which PID will be integrated [s]: For operating point 1	REAL	10	S7_m_c := 'true' S7_link := 'true'
TI2	Time for which PID will be integrated [s]: For operating point 2	REAL	10	S7_m_c := 'true' S7_link := 'true'
TI3	Time for which PID will be integrated [s]: For operating point 3	REAL	10	S7_m_c := 'true' S7_link := 'true'
X	Process value defines the operating point	REAL	0	S7_m_c := 'false' S7_link := 'true'
X1	Operating point 1 (interpolation point) for X	REAL	0	S7_m_c := 'true'
X2	Operating point 2 (interpolation point) for X	REAL	50	S7_m_c := 'true'
X3	Operating point 3 (interpolation point) for X	REAL	100	S7_m_c := 'true'

Outputs

I/O (parameters)	Description	Data type	Default Value	Attributes
Link2Gain	Calculated controller gain	REAL	1	S7_m_c := 'false' S7_link := 'true'
Link2TD	Calculated derivative action time	REAL	0	S7_m_c := 'false' S7_link := 'true'
Link2TI	Calculated integral action time	REAL	10	S7_m_c := 'false' S7_link := 'true'

See also

Functions of GAIN_SHD (Page 172)
Description of GAIN_SHD (Page 169)

3.13 INT_P: Integration

3.13.1 Description of INT_P

Object name (type + number)

FB40

- INT_P block I/Os (Page 178)

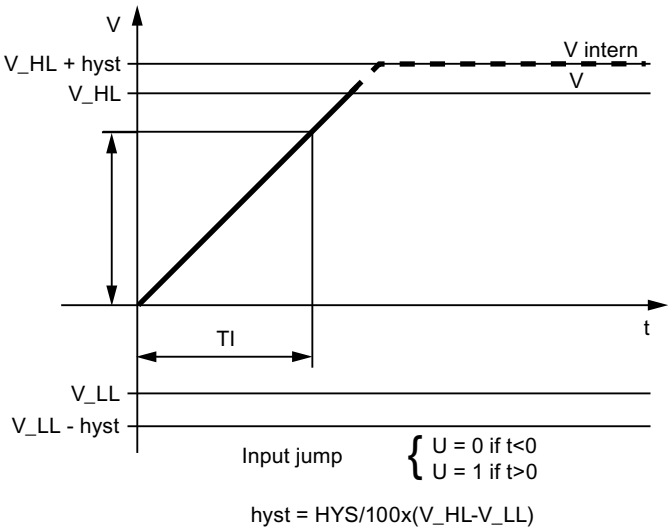
Function

Forms the time integral of the connected input value:

v(s) = 1 / (TI * s) * u(s)

How it works

Block INT_P functions by means of sum generation in accordance with the trapezoid rule per sampling interval (SAMPLE_T). The result, V internal, lies within the range V_HL - hyst to V_LL + hyst, as you can see in the figure. Subsequently the value is written to the output V after an additional limitation to between V_LL and V_HL.



Jump response of the INT_P

In addition, the internal result V internal is monitored for violation of the limits V_LL and V_HL and displayed via the Boolean outputs QVLL and QVHL, as you can see in the figure.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Error handling

Apart from the errors recognized by the operating system, the following configuration errors are also indicated by the block algorithm via ENO = 0 and QERR = 1:

- V_LL ≥ V_HL (V = 0)
- SAMPLE_T ≤ 0 (calculation continues internally with the substitute value = 1)
- Hysteresis HYS < 0 (calculation continues internally with the substitute value = 1)

If TI = 0 then V = 0, if the value is within the limits of V_LL and V_HL, otherwise V = limit value.

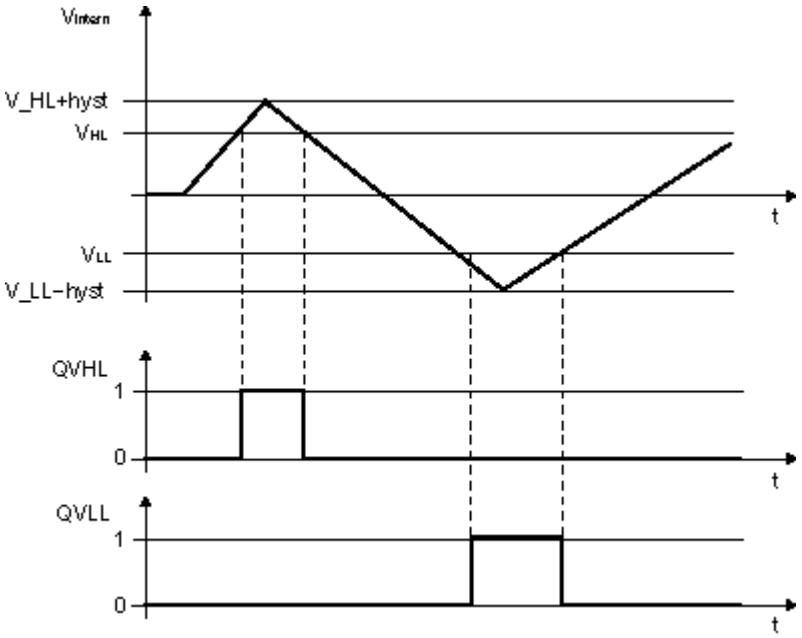


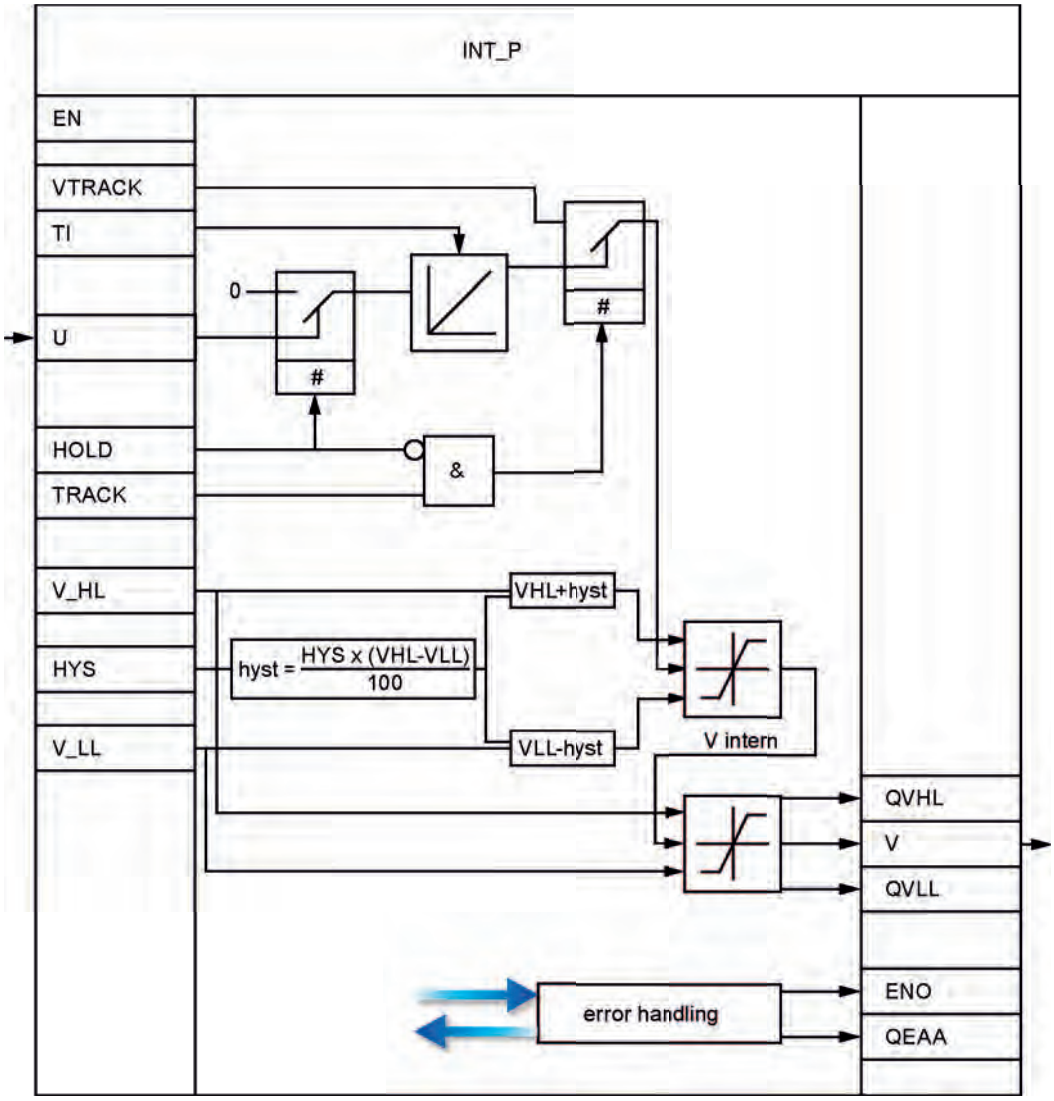
Figure 3-1 Limit monitoring of INT_P

Startup characteristics

If the input parameter RESET = TRUE, the internal historical process data as well as the output V are reset during startup. The last value is retained if RESET = FALSE. This means that the block must also be called in the startup OB (OB100).

Time response

The block must be installed in a cyclic interrupt OB.



INT_P structure

3.13.2 I/Os of INT_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameters)	Meaning	Data type	Default	Type	Permissible values
HOLD	1 = Hold integration (takes priority over TRACK)	BOOL	0	I	
HYS	Hysteresis of V_INTERN in [%]	REAL	1	I	≥ 0
QERR	1 = Error	BOOL	1	O	
QVHL	1 = High limit output value	BOOL	0	O	
QVLL	1 = Low limit output value	BOOL	0	O	
RESET	1 = RESET (restart)	BOOL	1	I	
SAMPLE_T	Sampling time in seconds	REAL	1	I	> 0
TI	Integration time in seconds	REAL	1	I	≥ 0
TRACK	1 = tracking	BOOL	0	I	
U	Input value	REAL	0	I	
V	Output value	REAL	0	O	
V_HL	High limit for V	REAL	100	I	V_HL > V_LL
V_LL	Low limit for V	REAL	0	I	V_LL < V_HL
VTRACK	Tracking value	REAL	0	I	

3.14 INTERLOK: Status Display Lock

3.14.1 Description of INTERLOK

Object name (type + number)

FB 75

- INTERLOK block I/Os (Page 181)
- INTERLOK block icon (Page 586)
- INTERLOK_Standard_View faceplate (Page 532)

Function

The INTERLOK block is used to implement a standardized interlock display, which can be called on the OS. The block can be assigned a maximum of 10 input signals, which can each be inverted as required.

How it works

The first five inputs I1_1 to I1_5 form a group. Each signal can be linked logically either directly or inverted by setting the corresponding inputs NEG1_1 to NEG1_5.
The type of logic operation of the first group is set at the AND_OR1 parameter. NEGRES_1 = 1 inverts the result of Q1 used to form Q via AND_OR3. Output Q1, however, is not inverted.

The same applies to the second group of five inputs as to the first group.
The results of both groups can be linked logically via an and/or operation.

With input OVERWRITE = 1, the output Q can be set to 0 when an interlock is active (Q = 1). This is only possible if OVERW_EN = 1. Input OVERWRITE = 0 if OVERW_EN = 0 or if the interlock condition is not satisfied. Q_OVERWR = 1 at the output indicates that output Q has been overwritten.

The following applies only if input CHECK_EN = TRUE:
The output parameter FIRST_I contains the number (1 to 10) of the input Ix, which was first TRUE or inverted FALSE. If several conditions are set simultaneously, the lowest number is entered in FIRST_I. A positive edge at input RESET sets FIRST_I to zero, if none of the above conditions are satisfied. Output Q is usually interconnected with RESET.

Calling OBs

The calling OB is in the same OB and after the last block whose signals are to be displayed on the INTERLOK.

Error handling

Only by means of the operating system

Startup characteristics

No special measures are required.

Time response

The block does not have a time response.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for INTERLOK (Page 182)".

Message response

Not available

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of INTERLOK (Page 181)

VSTATUS for INTERLOK (Page 182)

Operating and monitoring INTERLOK (Page 182)

3.14.2 I/Os of INTERLOK

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

Detailed information about the abbreviations used is available in the section "General information pertaining to the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
AND_OR1	1 = AND, 0 = OR first group	BOOL	0	I	+
AND_OR2	1 = AND, 0 = OR second group	BOOL	0	I	+
AND_OR3	1 = AND, 0 = OR of both groups	BOOL	0	I	+
CHECK_EN	1 = FIRST_I check enable	BOOL	0	I	+
FIRST_I	First input signal that was TRUE (inverted FALSE)	BYTE	16#00	O	+
I1_1	Input signal 1, first group	BOOL	0	I	+
I1_2	Input signal 2, first group	BOOL	0	I	+
I1_3	Input signal 3, first group	BOOL	0	I	+
I1_4	Input signal 4, first group	BOOL	0	I	+
I1_5	Input signal 5, first group	BOOL	0	I	+
I2_1	Input signal 1, second group	BOOL	0	I	+
I2_2	Input signal 2, second group	BOOL	0	I	+
I2_3	Input signal 3, second group	BOOL	0	I	+
I2_4	Input signal 4, second group	BOOL	0	I	+
I2_5	Input signal 5, second group	BOOL	0	I	+
NEG1_1	1 = I1_1 is inverted	BOOL	0	I	+
NEG1_2	1 = I1_2 is inverted	BOOL	0	I	+
NEG1_3	1 = I1_3 is inverted	BOOL	0	I	+
NEG1_4	1 = I1_4 is inverted	BOOL	0	I	+
NEG1_5	1 = I1_5 is inverted	BOOL	0	I	+
NEG2_1	1 = I2_1 is inverted	BOOL	0	I	+
NEG2_2	1 = I2_2 is inverted	BOOL	0	I	+
NEG2_3	1 = I2_3 is inverted	BOOL	0	I	+
NEG2_4	1 = I2_4 is inverted	BOOL	0	I	+
NEG2_5	1 = I2_5 is inverted	BOOL	0	I	+
NEGRES_1	1 = result of first group is inverted	BOOL	0	I	+
NEGRES_2	1 = result of second group is inverted	BOOL	0	I	+
OVERW_EN	1 = OVERWRITE enabled	BOOL	0	I	+
OVERWRITE	1 = OVERWRITE	BOOL	0	IO	+
Q	Output signal	BOOL	0	O	+
Q_OVERWR	1 = Q is overwritten	BOOL	0	O	+
Q1	Interim result, first group	BOOL	0	O	+
Q2	Interim result, second group	BOOL	0	O	+
QC_I1_1	Quality Code of I1_1	BYTE	16#80	I	
QC_I1_2	Quality Code of I1_2	BYTE	16#80	I	

I/O (parameter)	Meaning	Data type	Default	Type	OCM
QC_I1_3	Quality Code of I1_3	BYTE	16#80	I	
QC_I1_4	Quality Code of I1_4	BYTE	16#80	I	
QC_I1_5	Quality Code of I1_5	BYTE	16#80	I	
QC_I2_1	Quality Code of I2_1	BYTE	16#80	I	
QC_I2_2	Quality Code of I2_2	BYTE	16#80	I	
QC_I2_3	Quality Code of I2_3	BYTE	16#80	I	
QC_I2_4	Quality Code of I2_4	BYTE	16#80	I	
QC_I2_5	Quality Code of I2_5	BYTE	16#80	I	
QC_Q	Quality Code of Q	BYTE	16#80	O	
QC_Q_I	Quality Code of output Q	BYTE	16#80	I	
RESET	Positive edge = reset FIRST_I	BOOL	0	I	+
USTATUS	Status word in VSTATUS; can be configured user-specific	WORD	0	I	
VSTATUS	Extended status display in block icons	DWORD	0	O	+

3.14.3 VSTATUS for INTERLOK

The 32-bit statusword extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	-	-	-	I1_5	I1_4	I1_3	I1_2	I1_1
Bit no.:	15	14	13	12	11	10	9	8
Parameters	-	-	Q	I2_5	I2_4	I2_3	I2_2	I2_1

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.14.4 Operator Control and Monitoring of INTERLOK

Additional information

You can find additional information in the following sections:

- INTERLOK block icon (Page 586)
- INTERLOK faceplate (Page 532)

3.15 LIMITS_P: Limiting

3.15.1 Description of LIMITS_P

Object name (type + number)

FB41

- LIMITS_P block I/Os (Page 184)

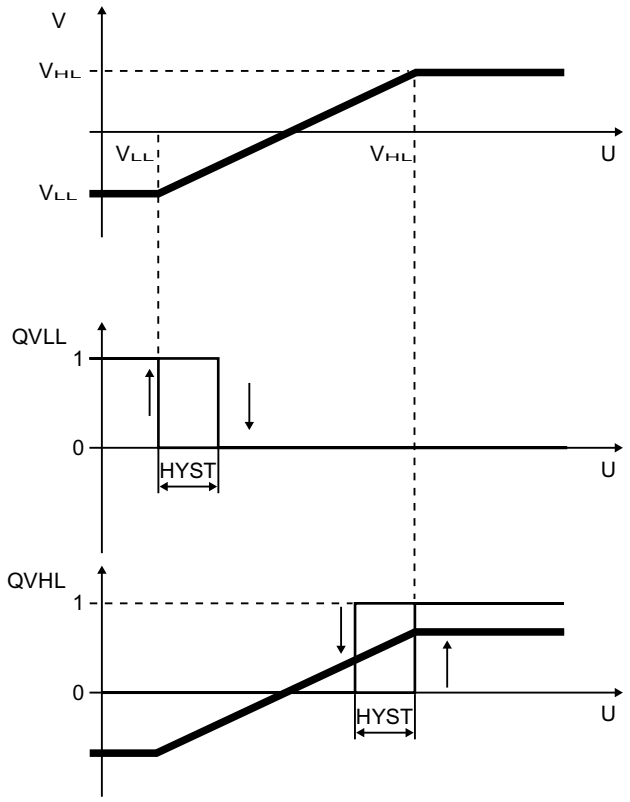
Function

The LIMITS_P block is used to limit an analog variable to an adjustable range.

How it works

Block LIMITS_P passes the analog input value U to the output V as long as it lies within the set limits.

- If the value is outside the low limit, the low limit value is output. If the value is outside the high limit, the high limit value is output.
- The active limitation is indicated at the set binary outputs. A hysteresis can be set in order to avoid dithering of the display when the input value fluctuates around the limit.



Operating principle of the LIMITS_P

Calling OBs

The OB in which you install the block.

Error handling

If V_HL - V_LL ≤ HYS, QVHL and QVLL can be 1 simultaneously. A plausibility test of V_LL and V_HL is not implemented.

Arithmetic errors are indicated by ENO = 0 or QERR = 1. Arithmetic errors occur when the range limits of the REAL data type exceed the results of the formula described above. The value of V from the previous cycle is retained in this case. If V takes the value = #+INF or #-INF because of the corresponding value of U, there is also an arithmetic error in the next cycle.

3.15.2 I/Os of LIMITS_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
HYS	Hysteresis	REAL	0.0	I
QERR	1 = error	BOOL	1	O
QVHL	High limit triggered	BOOL	0	O
QVLL	Low limit triggered	BOOL	0	O
U	Input value	REAL	0.0	I
V	Output value	REAL	0.0	O
V_HL	High limit	REAL	100.0	I
V_LL	Low limit	REAL	0.0	I

3.16 MEAS_MON: Measured value monitoring

3.16.1 Description of MEAS_MON

Object name (type + number)

FB 65

- MEAS_MON block I/Os (Page 187)
- MEAS_MON block icon (Page 587)
- MEAS_MON faceplate (Page 533)

Function

The MEAS_MON block is used to monitor a measured value (analog signal) with regard to the limit pairs:

- Warning limit (high/low)
- Alarm limit (high/low)

How it works

The block monitors the measured value at input U. The high or low transgression of a limit is indicated at a corresponding output and signaled if applicable. Further information is available in the "Message response" section.

Calling OBs

The calling OB is in the same OB and after the block whose measured value is to be monitored, and also in OB 100.

Error handling

In the event of arithmetical errors the outputs ENO = 0 and QERR = 1 will be set.

Startup characteristics

After startup, messages will be suppressed for the number of cycles set at RUNUPCYC.

Time response

No time response. The block is to run in the same runtime group (see CFC) with the measured-value producer.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for MEAS_MON (Page 189)".

Message response

The MEAS_MON block uses the ALARM_8P block to generate messages.

The following message triggers exist:

- Functions for monitoring the measured-value limits
- The CSF signal that is received as a control-system error via the interconnection

Messages regarding limit infringements can be suppressed individually via the corresponding M_SUP_xx inputs. Process messages (not control-system messages) can be disabled centrally using MSG_LOCK.

QMSG_SUP is set if the RUNUPCYC cycles since restart have not yet elapsed or MSG_LOCK = TRUE or MSG_STAT = 21.

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of MEAS_MON (Page 187)

Message texts and associated values of MEAS_MON (Page 188)

VSTATUS for MEAS_MON (Page 189)

Operating and monitoring MEAS_MON (Page 189)

3.16.2 I/Os of MEAS_MON

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
AUX_PRx	Associated value x	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: current batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	0	I	+	
CSF	Control system error	BOOL	0	I		
HYS	Hysteresis	REAL	5	I	+	≥ 0
M_SUP_AH	1 = message suppression high alarm	BOOL	0	I	+	
M_SUP_AL	1 = message suppression low alarm	BOOL	0	I	+	
M_SUP_WH	1 = message suppression high warning	BOOL	0	I	+	
M_SUP_WL	1 = message suppression low warning	BOOL	0	I	+	
MO_PVHR	High display limit (measuring range) - only for OS	REAL	110	I	+	
MO_PVLR	Low display limit (measuring range) - only for OS	REAL	-10	I	+	
MSG_ACK	Messages acknowledged	WORD	0	O		
MSG_EVID	Message number	DWORD	0	I		
MSG_LOCK	1 = process messages locked	BOOL	0	I	+	
MSG_STAT	Message error information	WORD	0	O		
OCCUPIED	ID for occupied by BATCH	BOOL	0	I	+	
OOS	Reserve	BOOL	0	I	+	
QC_U	Quality code for U	BYTE	16#80	I		
QERR	1 = error output (inverted ENO)	BOOL	1	O	+	
QH_ALM	1 = high alarm	BOOL	0	O		
QH_WRN	1 = high warning	BOOL	0	O		
QL_ALM	1 = low alarm	BOOL	0	O		
QL_WRN	1 = low warning	BOOL	0	O		
QMSG_ERR	1 = message error	BOOL	0	O	+	
QMSG_SUP	1 = message suppression active	BOOL	0	O	+	
RUNUPCYC	Number of run-up cycles	INT	3	I		
STEP_NO	BATCH step number	DWORD	0	I	+	
U	Analog input (measured value)	REAL	0	I	+	
U_AH	High alarm limit	REAL	100	I	+	U_AH>U_WH
U_AL	Low alarm limit	REAL	0	I	+	U_AL<U_WL
U_WH	High warning limit	REAL	95	I	+	U_AH>U_WH>U_WL
U_WL	Low warning limit	REAL	5	I	+	U_WH>U_WL>U_AL

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
USTATUS	Status word in VSTATUS, can be configured by user	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.16.3 Message Texts and Associated Values of MEAS_MON

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QH_ALM	$$BlockComment$$ HighHigh Alarm	AH	M_SUP_AH, MSG_LOCK
2	QH_WRN	$$BlockComment$$ High alarm	WH	M_SUP_WH, MSG_LOCK
3	QL_WRN	$$BlockComment$$ Low alarm	WL	M_SUP_WL, MSG_LOCK
4	QL_ALM	$$BlockComment$$ LowLow alarm	AL	M_SUP_AL, MSG_LOCK
5	CSF	$$BlockComment$$ External fault	S	-

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data, the fourth is reserved for U, and the remaining ones (AUX_PRx) can be freely assigned by the user.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	U
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.16.4 VSTATUS for MEAS_MON

The 32-bit status word extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	-	-	-			MSG_LOCK	BA_EN	OCCUPIED

Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	-	-	-	-	-

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.16.5 Operating and Monitoring of MEAS_MON

Additional information

You can find additional information in the following sections:

- MEAS_MON block icon (Page 587)
- MEAS_MON faceplate (Page 533)

3.17 MOT_REV: Reversing motor

3.17.1 Description of MOT_REV

Object name (type + number)

FB67

- MOT_REV block I/Os (Page 195)
- MOT_REV block icon (Page 588)
- MOT_REV faceplate (Page 536)

Function

The block is used to control reversible motors (clockwise/counterclockwise direction). You can optionally monitor up to two feedback signals generated by auxiliary contactors.

NOTICE
With the PCS 7 it is not intended that other blocks will be inserted between the plant block and the output driver. If you deviate from this principle, ensure that when interconnecting the block, that from the outputs of the plant block until the output driver, that all blocks that form the output signal are installed in the same OBs.

How it works

Various inputs are available for controlling the motor. They are implemented in a concrete hierarchical relationship to each other and to the motor states. In particular the interlock, the feedback or rotary direction monitoring and the motor protective circuit-breaker influence the control signals QSTART (1: on, 0: off) and QDIR (1: counterclockwise, 0: clockwise).

The allocation of priorities to individual input variables and events with regard to their influence on the control signals is shown in the following table. The sections following provide further details.

Priority:	Event:
High	Motor protection fault, if MSS_OFF = 1
⇑	Wait time at change of rotary direction
	LOCK = 1
	LOCK_ON = 1 (with LOCK_DIR)
⇓	Monitoring error, if FAULT_OFF = 1
Low	Automatic/Manual mode
No effect	Motor protection fault, if MSS_OFF = 0
	Monitoring error, if FAULT_OFF = 0
	Control system error, operator error

Manual/auto

The operating mode is changed either by the operator setting AUT_ON_OP on the OS, or via the interconnection at input AUT_L, provided the functions required are enabled. The set mode is indicated at output QMAN_AUT (1 = Auto, 0 = Manual).

- **Manual mode:** This mode allows control by the operator on the OS or via linkable inputs.
  - **OS operator control** (LINK_MAN = 0): Set the FORW_ON (clockwise) and REV_ON (anticlockwise) inputs on the OS to control the rotary direction of the motor, or set MOT_OFF to shut it down. The corresponding enables FW_OP_EN, RV_OP_EN or OFFOP_EN must be be available.
  - **Operation via interconnectable inputs** (LINK_MAN = 1): The commands are received via the L_FORW, L_REV and L_OFF inputs. You can interconnect these inputs to allow tracking or local control, for example. Note that you must set the switches LINK_MAN, LIOP_SEL and AUT_L by means of a suitable logic.
- **Automatic mode:** An automatic control system provides the automatic commands via the interconnection to the AUTO_ON (1 = on, 0 = off) or AUTO_DIR (1 = anticlockwise, 0 = clockwise) inputs.

Interlock

The interlock function takes priority over all other control signals and errors, with the exception of the motor protection switch with corresponding enable signal (MSS_OFF = 1) and time monitoring during reversal of the rotary direction. When LOCK is set, the motor is switched off directly. The motor is switched on directly when LOCK_ON is set, provided that LOCK is not also set. LOCK_DIR is used to set the desired rotary direction at LOCK_ON = 1.

Monitoring

The monitoring logic monitors consistency between the control commands QSTART or QDIR and the process value feedback FB_ON or FB_DIR, and outputs the actual status at QRUN and QSTOP. It sets a monitoring error (QMON_ERR = 1) if a feedback signal is not returned for QSTART or QDIR within the time set at TIME_ON, or if the feedback signal changes unexpectedly without being requested by QSTART or QDIR.

If there is no feedback, you can interconnect either QSTART with FB_ON and QDIR with FB_DIR, or disable monitoring with MONITOR = 0.

The FAULT_OFF parameter defines the relevance of the monitoring error. The motor is shut down if FAULT_OFF = 1 and a fault is detected. This fault state does not affect the control outputs if FAULT_OFF = 0.

Motor protection

A negative edge of the motor protection signal MSS sets the motor protection fault signal retentively and transfers this signal to output QMSS_ST. Parameter MSS_OFF defines whether only to indicate the fault state (MSS_OFF = 0), or whether to shut down the motor regardless of any other inputs and system states (MSS_OFF = 1).

Bumpless changeover

To ensure bumpless changeover to manual mode in all operating phases, the manual values FORW_ON, REV_ON and MOT_OFF always track the current values of QSTART and QDIR (exception: reversal of the rotary direction).

Reversal of the rotary direction

If reversal of the rotary direction is selected, the following occurs:

- The motor is stopped (QSTART = 0).
- The internal OFF monitor waits for the time period TIME_OFF to expire and then starts the motor in the reverse direction, provided the OFF monitor does not report an error. Note that the TIME_OFF parameter defines the time the motor is expected to take before it actually comes to a standstill, so that the rotary direction can be reversed without causing any damage. The motor standstill feedback does not reduce this time, as this message is generated by means of an auxiliary contactor and does not provide any information about the actual state of the motor.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Error handling

The motor protection error (QMSS_ST = 1) and the monitoring error (QMON_ERR = 1) are reported to the OS and influence how the block works as described above. The operator can either reset these states with RESET, or automatically at the next positive edge at MSS by interconnecting L_RESET with a "1" signal. The control system fault CSF is merely reported to the OS and applied to the group error QGR_ERR along with the motor protection and monitoring errors. It does not have any further influence on the block algorithm.

Operator errors are indicated at output QOP_ERR without a message.

Startup after error state

Startup after error state depends on the mode set at the time of the reset:

- The motor can restart after a reset in auto mode if a corresponding start signal is provided by the automatic function.
- In manual mode, the motor must be explicitly turned on, because manual operation was tracked to "OFF".

Startup characteristics

When the CPU starts, the MOT_REV block is switched to manual mode and the OFF command is output. A prerequisite is that the block must be called in the startup OB. In CFC engineering, this is handled by the CFC. When using basic STEP 7 tools, you enter this call in the startup OB. After startup, the messages will be suppressed during the number of cycles set at RUNUPCYC.

The START_OFF input is used to define whether to shut down the motor during CPU startup (START_OFF = 1) or whether to retain the last operating state.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SAMPLE_T parameter.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for MOT_REV (Page 199)".

Message response

The MOT_REV block uses the ALARM_8P block to generate messages.

The following message triggers exist:

- Control system error
- Motor circuit-breaker fault and monitoring error (runtime error)
- the CSF signal received via the interconnection

QMSG_SUP is set if the RUNUPCYC cycles have not expired since the restart or if MSG_STAT = 21.

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of MOT_REV (Page 195)
Message texts and associated values of MOT_REV (Page 198)
VSTATUS for MOT_REV (Page 199)
Operating and monitoring MOT_REV (Page 199)

3.17.2 I/Os of MOT_REV

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
AUT_L	Interconnectable input for MANUAL/AUTO: 0 = manual; 1 = auto	BOOL	0	I		
AUT_ON_OP	Operator input: 0 = manual; 1 = auto	BOOL	0	IO	+	
AUTO_DIR	AUTOMATIC mode rotary direction: 1 = anticlockwise, 0 = clockwise	BOOL	0	I		
AUTO_ON	AUTOMATIC value: 1 = ON, 0 = OFF	BOOL	0	I		
AUTOP_EN	1 = operator control enable for Auto	BOOL	1	I		
AUX_PRx	Associated value x	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: Consecutive batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	0	I	+	
CSF	Control-system error	BOOL	0	I		
FAULT_OFF	1 = Motor OFF in case of fault	BOOL	1	I		
FB_DIR	Feedback of rotary direction: 1 = anticlockwise, 0 = clockwise	BOOL	0	I		
FB_ON	Feedback: 1 = ON	BOOL	0	I		
FORW_ON	1 = Set clockwise direction	BOOL	0	IO	+	
FW_OP_EN	1 = Operator-control enable for clockwise direction	BOOL	1	I		
L_FORW	AUTOMATIC mode: 1 = set clockwise direction	BOOL	0	I		
L_OFF	AUTOMATIC mode: 1 = Motor off	BOOL	0	I		
L_RESET	Interconnectable RESET input	BOOL	0	I		
L_REV	AUTOMATIC value 1 = Anticlockwise rotation ON	BOOL	0	I		
LINK_MAN	0 = operator input active, 1 = manual control via L_FORW, L_REV, L_OFF	BOOL	0	I		
LIOP_SEL	Interconnectable input for manual/auto changeover (AUT_L): 1 = interconnection active, 0 = operator control active	BOOL	0	I		
LOCK	1 = Lock (OFF)	BOOL	0	I	+	
LOCK_DIR	Direction of rotation when LOCK_ON = 1: 1 = forward, 0 = reverse	BOOL	0	I	+	
LOCK_ON	1 = Lock (ON)	BOOL	0	I	+	
MANOP_EN	1 = operator control enable for manual mode	BOOL	1	I		
MONITOR	Monitoring: 1 = ON	BOOL	1	I	+	
MOT_OFF	Operator input: 1 = motor off	BOOL	0	IO	+	
MSG_ACK	Messages acknowledged	WORD	0	O		

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
MSG_EVID	Message number	DWORD	0	I		
MSG_STAT	Message error information	WORD	0	O		
MSS	Motor protective circuit breaker (active low, i.e. 0 = error)	BOOL	1	I		
MSS_OFF	1 = In case of MSS fault: Motor stop	BOOL	1	I		
OCCUPIED	ID for occupied by BATCH	BOOL	0	I	+	
OFFOP_EN	1 = Operator-control enable for motor OFF	BOOL	1	I		
OOS	Reserve	BOOL	0	I	+	
QAUTOP	1 = operator control enable for automatic mode	BOOL	0	O	+	
QC_FB_DIR	Quality code for FB_DIR	BYTE	16#80	I		
QC_FB_ON	Quality code for FB_ON	BYTE	16#80	I		
QC_QDIR	Quality code for QDIR	BYTE	16#80	O		
QC_QDIR_I	Quality code for output QDIR	BYTE	16#80	I		
QC_QSTART	Quality code for QSTART	BYTE	16#80	O		
QC_QSTART_I	Quality code for output QSTART	BYTE	16#80	I		
QDIR	Direction control output: 1 = anticlockwise	BOOL	0	O	+	
QERR	1 = Error output (Inverted ENO)	BOOL	1	O	+	
QFORW_OP	1 = operator-control enable for activating forward	BOOL	0	O	+	
QGR_ERR	1 = Group error	BOOL	0	O		
QMAN_AUT	0 = manual, 1 = auto	BOOL	0	O	+	
QMANOP	1 = operator control enable for manual mode	BOOL	0	O	+	
QMON_ERR	1 = Monitoring error	BOOL	0	O	+	
QMSG_ERR	1 = ALARM_8P error	BOOL	0	O	+	
QMSG_SUP	1 = message suppression	BOOL	0	O	+	
QMSS_ST	Stored motor protective circuit breaker, 1 = error	BOOL	0	O	+	
QOFF_OP	1 = Operator-control enable for motor OFF	BOOL	0	O	+	
QOP_ERR	1 = group operator input error	BOOL	0	O		
QREV_OP	1 = operator-control enable for activating reverse	BOOL	0	O	+	
QRUN	1 = Motor is running	BOOL	0	O	+	
QSTART	Control Output: 1 = ON	BOOL	0	O	+	
QSTOP	1 = Motor STOP	BOOL	0	O	+	
RESET	Enabled input of error reset	BOOL	0	IO	+	
REV_ON	Operator input: 1 = activate reverse	BOOL	0	IO	+	
RUNUPCYC	Number of run-up cycles	INT	3	I		
RV_OP_EN	1 = Operator-control enable for anticlockwise direction	BOOL	1	I		
SAMPLE_T	Sampling time in seconds	REAL	1.0	I		> 0
START_OFF	1 = during startup: Motor OFF	BOOL	1	I		
STEP_NO	BATCH step number	DWORD	0	I	+	
TIME_OFF	Monitoring time for OFF in seconds	REAL	3.0	I	+	≥ 0

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
TIME_ON	Monitoring time for ON in seconds	REAL	3.0	I	+	≥ 0
USTATUS	Status word in VSTATUS, can be configured by user	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.17.3 Message Texts and Associated Values of MOT_REV

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QMSS_ST	$$BlockComment$$ Motor protection	S	-
2	QMON_ERR	$$BlockComment$$ Monitoring Fault	S	-
3	CSF	$$BlockComment$$ External fault	S	-

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data and the remaining ones (AUX_PRx) can be freely assigned by the user.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	AUX_PR04
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.17.4 VSTATUS for MOT_REV

The 32-bit statusword extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	QMON_ERR	-	QMSS_ST		QMAN_AUT	-	BA_EN	OCCUPIED
Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	-	QDIR	QSTOP	QRUN	LOCK

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.17.5 Operating and Monitoring of MOT_REV

Additional information

You can find additional information in the following sections:

- MOT_REV block icon (Page 588)
- MOT_REV faceplate (Page 536)

3.18 MOT_SPED: Two-speed motor

3.18.1 Description of MOT_SPED

Object name (type + number)

FB68

- MOT_SPED block I/Os (Page 204)
- MOT_SPED block icon (Page 589)
- MOT_SPED faceplate (Page 538)

Function

Block MOT_SPED is used to control two-speed motors (slow/rapid). The two feedback signals output by the contactor relays can be monitored.

NOTICE
With the PCS 7 it is not intended that other blocks will be inserted between the plant block and the output driver. If you deviate from this principle, ensure when interconnecting the block that from the outputs of the plant block until the output driver all blocks that form the output signal are installed in the same OB.

How it works

Various inputs are available for controlling the motor. They are implemented in a concrete hierarchical relationship to each other and to the motor states. In particular the interlock, the feedback monitoring and the motor protective circuit-breaker functions influence the control signals QSTART (1: on, 0: off) and QSPEED (1: fast, 0: slow).

The allocation of priorities to individual input variables and events with regard to their influence on the control signals is shown in the following table. The sections following provide further details.

Priority:	Event:
High	Motor protection fault, if MSS_OFF = 1
⇑	LOCK = 1
	LOCK_ON = 1 (with LOCK_SPD)
⇓	Monitoring error, if FAULT_OFF = 1
Low	Automatic/Manual mode
No effect	Motor protection fault, if MSS_OFF = 0
	Monitoring error, if FAULT_OFF = 0
	Control system error, operator error

Note

Deceleration times during changeover from rapid speed to slow speed must be set by means of an external timer (on delay).

Manual/auto

The mode is changed over either by the operator setting AUT_ON_OP on the OS , or by way of the interconnection at input AUT_L, provided the necessary enables are set. The set operating mode is indicated at the output QMAN_AUT (1: Auto, 0: Manual).

- **Manual mode**: This operating mode permits control either at the OS or via interconnectable inputs.
  - **OS operator control** (LINK_MAN = 0): The following inputs can be operated at the OS: SP1_ON for slow speed, SP2_ON for fast speed or MOT_OFF for switching off the motor. The appropriate parameters (S1_OP_EN, S2_OP_EN or OFFOP_EN) must be set accordingly.
  - **Operation via interconnectable inputs** (LINK_MAN = 1): In this case the commands are input at L_SP1, L_SP2 and L_OFF. You can interconnect these inputs to allow tracking or local control, for example. Note that you must set the switches LINK_MAN, LIOP_SEL and AUT_L by means of a suitable logic.
- **Automatic mode**: An automatic function block interconnection outputs the automatic mode instructions to the inputs AUTO_ON (1: ON, 0: OFF) or AUTO_SPD (1: fast, 0: slow) by interconnecting an automatic system.

Interlock

The interlock function takes priority over all other control signals and errors - with the exception of the motor protection when the corresponding enable signal is set (MSS_OFF = 1). When LOCK is set, the motor is switched off directly. The motor is switched on directly when LOCK_ON is set, provided that LOCK is not also set. LOCK_SPD is used to specify the desired speed for LOCK_ON = 1 (1: fast, 0: slow).

Monitoring

The monitoring logic monitors consistency between the control commands QSTART or QSPEED and the actual-value feedback signals FB_ON or FB_SPEED and outputs the actual state via QRUN and QSTOP. It sets a monitoring error (QMON_ERR = 1) if no feedback is set for QSTART or QSPEED within the time set at TIME_MON, or if it changes unexpectedly without being requested by QSTART or QSPEED.

If there is no feedback, either QSTART can be interconnected with FB_ON and QSPEED with FB_SPEED or monitoring can be disabled by setting MONITOR = 0.

The FAULT_OFF parameter determines the significance of a monitoring error. If FAULT_OFF = 1, the motor is switched off when a fault occurs. This error status does not affect the control outputs when FAULT_OFF = 0.

Motor protection

A negative edge of the motor protection signal MSS sets the motor protection fault signal retentively and transfers this signal to output QMSS_ST. Parameter MSS_OFF defines whether the fault state should only be indicated (MSS_OFF = 0), or whether the motor should be shut down regardless of any other inputs or system states (MSS_OFF = 1).

Bumpless changeover

To ensure bumpless changeover to manual mode in all operating phases, the manual values SP1_ON, SP2_ON and MOT_OFF always track the current values of QSTART and QSPEED.

Error handling

The motor protection error (QMSS_ST = 1) and the monitoring error (QMON_ERR = 1) are reported to the OS and influence how the block works as described above. The operator can either reset these states with RESET, or automatically at the next positive edge at MSS by interconnecting L_RESET with a "1" signal. The control system fault CSF is merely reported to the OS and applied to the group error QGR_ERR along with the motor protection and monitoring errors. It does not have any further influence on the block algorithm.

Operator errors are indicated at output QOP_ERR without a message.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Startup after error state

Startup after error state depends on the mode set at the time of the reset:

- In automatic mode, the motor can restart after the reset, provided a corresponding start signal is supplied by the automatic mode.
- In manual mode, the motor must be explicitly turned on, because manual operation was tracked to "OFF".

Startup characteristics

When the CPU starts, the MOT_SPED block is switched to manual mode and the OFF command is output. This means that the block must be called in the startup OB. In CFC engineering, this is handled by the CFC. When using basic STEP 7 tools, you enter this call in the startup OB. After startup, the messages will be suppressed during the number of cycles set at RUNUPCYC.

The START_OFF input is used to define whether to shut down the motor during CPU startup (START_OFF = 1) or whether to retain last operating state.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SAMPLE_T parameter.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for MOT_SPED (Page 208)".

Message response

The MOT_SPED block uses the ALARM_8P block to generate messages.

The following message triggers exist:

- Control system error
- Motor circuit-breaker fault and monitoring error (runtime error)
- The CSF signal that is received via the interconnection

QMSG_SUP is set if the RUNUPCYC cycles have not expired since the restart or if MSG_STAT = 21.

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of MOT_SPED (Page 204)

Message texts and associated values of MOT_SPED (Page 207)

VSTATUS for MOT_SPED (Page 208)

Operating and monitoring MOT_SPED (Page 208)

3.18.2 I/Os of MOT_SPED

The factory setting of the block display in CFC is identified in the "I/O" column:
I/O name**bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameters)	Meaning	Data type	Default	Type	Attr.	OCM	Permissible values
AUT_L	Interconnectable input for MAN/AUTO 0 = manual, 1 = auto	BOOL	0	I	Q		
AUT_ON_OP	Operator input: 0 = manual, 1 = auto	BOOL	0	IO	F	+	
AUTO_ON	AUTOMATIC value: 1 = ON, 0 = OFF	BOOL	0	I	Q		
AUTO_SPD	Automatic speed value: 1 = rapid	BOOL	0	I	Q		
AUTOP_EN	1 = operator control enable for Auto	BOOL	1	I	Q		
AUX_PRx	Associated value x	ANY	0	IO	Q		
BA_EN	Release by BATCH	BOOL	0	I	Q	+	
BA_ID	BATCH: current batch number	DWORD	0	I	Q	+	
BA_NA	BATCH name	STRING[32]	0	I	Q	+	
CSF	Control system error	BOOL	0	I	Q		
FAULT_OFF	1 = Motor OFF in case of fault	BOOL	1	I	Q		
FB_ON	Feedback: 1 = ON	BOOL	0	I	Q		
FB_SPEED	Speed feedback: 1 = rapid	BOOL	0	I	Q		
L_OFF	AUTOMATIC mode: 1 = Motor off	BOOL	0	I	Q		
L_RESET	Interconnectable RESET input	BOOL	0	I	Q		
L_SP1	AUTOMATIC mode: 1 = set clockwise direction	BOOL	0	I	Q		
L_SP2	AUTOMATIC value 1 = Anticlockwise rotation ON	BOOL	0	I	Q		
LINK_MAN	0 = operator input enabled, 1 = manual control via L_SP1, L_SP2, L_MOTOFF	BOOL	0	I	Q		
LIOP_SEL	Interconnectable input for manual/auto changeover (AUT_L) 1 = interconnection is active 0 = operator control is active	BOOL	0	I	Q		
LOCK	1 = Lock (OFF)	BOOL	0	I	Q	+	
LOCK_ON	1 = Lock (ON)	BOOL	0	I	Q	+	
LOCK_SPD	Speed at LOCK_ON = 1 1 = fast, 0 = slow	BOOL	0	I	Q	+	
MANOP_EN	1 = operator control enable for manual mode	BOOL	1	I	Q		
MONITOR	Monitoring: 1 = ON	BOOL	1	I		+	
MOT_OFF	Operator input: 1 = motor off	BOOL	0	IO	F	+	
MSG_ACK	Messages acknowledged	WORD	0	O			
MSG_EVID	Message number	DWORD	0	I	M		
MSG_STAT	Message error information	WORD	0	O			

I/O (parameters)	Meaning	Data type	Default	Type	Attr.	OCM	Permissible values
MSS	Motor protective circuit-breaker (active low, i.e., 0 = error)	BOOL	1	I	Q		
MSS_OFF	1 = In case of motor protection fault, stop motor	BOOL	1	I	Q		
OCCUPIED	BATCH occupied ID	BOOL	0	I	Q	+	
OFFOP_EN	1 = Operator-control enable for motor OFF	BOOL	1	I	Q		
OOS	Reserve	BOOL	0	I		+	
QAUTOP	1 = operator control enable for automatic mode	BOOL	0	O		+	
QC_FB_ON	Quality code for FB_ON	BYTE	16#80	I			
QC_FB_SPEED	Quality code for FB_SPEED	BYTE	16#80	I			
QC_QSPEED	Quality code for QSPEED	BYTE	16#80	O			
QC_QSPEED_I	Quality code for output QSPEED	BYTE	16#80	I			
QC_QSTART	Quality code for QSTART	BYTE	16#80	O			
QC_QSTART_I	Quality code for output QSTART	BYTE	16#80	I			
QERR	1 = Error output (Inverted ENO)	BOOL	1	O		+	
QGR_ERR	1 = Group error	BOOL	0	O			
QMAN_AUT	0 = Manual, 1 = Auto	BOOL	0	O		+	
QMANOP	1 = operator control enable for manual mode	BOOL	0	O		+	
QMON_ERR	1 = Monitoring error	BOOL	0	O		+	
QMSG_ERR	1 = message error	BOOL	0	O		+	
QMSG_SUP	1 = message suppression active	BOOL	0	O		+	
QMSS_ST	Stored motor protective circuit-breaker (1 = error)	BOOL	0	O		+	
QOFF_OP	1 = operator control enable for motor OFF	BOOL	0	O		+	
QOP_ERR	1 = group operator input error	BOOL	0	O			
QRUN	1 = Motor is running	BOOL	0	O		+	
QS1_OP	1 = Operator-control enable for start speed 1	BOOL	0	O		+	
QS2_OP	1 = Operator-control enable for start speed 2	BOOL	0	O		+	
QSPEED	Speed control output: 1 = rapid	BOOL	0	O		+	
QSTART	Control Output: 1 = ON	BOOL	0	O			
QSTOP	1 = Motor STOP	BOOL	0	O		+	
QSTOPING	Reserve message	BOOL	0	O		+	
QSTRTING	Reserve message	BOOL	0	O		+	
RESET	Enabled input of error reset	BOOL	0	IO	F	+	
RUNUPCYC	Number of run-up cycles	INT	3	I			
S1_OP_EN	1 = Operator-control enable for start speed 1	BOOL	1	I	Q		

I/O (parameters)	Meaning	Data type	Default	Type	Attr.	OCM	Permissible values
S2_OP_EN	1 = Operator-control enable for start speed 2	BOOL	1	I	Q		
SAMPLE_T	Sampling time in seconds	REAL	1.0	I			> 0
SP1_ON	Operator input: 1 = Start speed 1 on	BOOL	0	IO	F	+	
SP2_ON	Operator input: 1 = Start speed 2 on	BOOL	0	IO	F	+	
START_OFF	1 = during startup: Motor OFF	BOOL	1	I	Q		
STEP_NO	BATCH step number	DWORD	0	I	Q	+	
TIME_MON	Monitoring time in seconds	REAL	3.0	I		+	≥ 0
USTATUS	Status word in VSTATUS; can be configured user-specific	WORD	0	I			
VSTATUS	Extended status display in block icons	DWORD	0	O		+	

3.18.3 Message Texts and Associated Values of MOT_SPED

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QMSS_ST	$$BlockComment$$ Motor protection	S	-
2	QMON_ERR	$$BlockComment$$ Monitoring Fault	S	-
3	CSF	$$BlockComment$$ External fault	S	-

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data and the remaining ones (AUX_PRx) can be freely assigned by the user.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	AUX_PR04
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.18.4 VSTATUS for MOT_SPED

The 32-bit statusword extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	QMON_ERR	-	QMSS_ST	-	QMAN_AUT	-	BA_EN	OCCUPIED
Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	QSTOPING	QSTRTING	QSPEED	QSTOP	QRUN	LOCK

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.18.5 Operating and Monitoring of MOT_SPED

Additional information

You can find additional information in the following sections:

- MOT_SPED block icon (Page 589)
- MOT_SPED faceplate (Page 538)

3.19 MOTOR: Motor with one control signal

3.19.1 Description of MOTOR

Object name (type + number)

FB66

- MOTOR block I/Os (Page 213)
- MOTOR block icon (Page 590)
- MOTOR faceplate (Page 540)

Function

Block MOTOR is used to control motors by means of a control signal (on/off). The motor speed feedback (on/off) can be monitored optionally. This motor speed feedback signal is provided by a contactor relay.

NOTICE
With the PCS 7 it is not intended that other blocks will be inserted between the plant block and the output driver. If you deviate from this principle, ensure that when interconnecting the block, that from the outputs of the plant block until the output driver, that all blocks that form the output signal are installed in the same OBs.

How it works

Various inputs are available for controlling the motor. They are implemented in a concrete hierarchical relationship to each other and to the motor states. In particular the locking, the feedback monitoring and the motor circuit breaker influence the control signals QSTART.

The allocation of priorities to the individual input variables and events with regard to their influence on the control signal is shown in the following table. The sections following provide further details.

Priority:	Event:
High	Motor protection fault, if MSS_OFF = 1
⇑	LOCK = 1
	LOCK_ON = 1
⇓	Monitoring error, if FAULT_OFF = 1
Low	Automatic/Manual mode
No effect	Motor protection fault, if MSS_OFF = 0
	Monitoring error, if FAULT_OFF = 0
	Control system error, operator error

Manual/auto

The operator modes are changed over either by means of OS operation with AUT_ON_OP (LIOP_SEL = 0) or the interconnection of the AUT_L input (LIOP_SEL = 1). If the OS system is used, the corresponding enable signals AUTOP_EN and MANOP_EN must be set. The set operating mode is indicated at the output QMAN_AUT (1: Auto, 0: Manual).

- Manual mode: Operations are performed by the OS system via the input MAN_ON, if the corresponding enable signals ON_OP_EN and OFFOP_EN are set.
- Automatic mode: An automatic unit outputs the control commands via the interconnected input AUTO_ON.

Interlock

The interlock function takes priority over all other control signals and errors - with the exception of the motor protection when the corresponding enable signal is set (MSS_OFF = 1). When LOCK is set, the motor is switched off directly. The motor is switched on directly when LOCK_ON is set, provided that LOCK is not also set.

Monitoring

The monitoring logic monitors consistency between the control command QSTART and the process value feedback FB_ON and outputs the actual state via QRUN and QSTOP. It sets a monitoring error (QMON_ERR = 1) if after the period TIME_MON no feedback corresponding to QSTART has been set or if it changes unexpectedly without a request by QSTART.

If there is no feedback, either QSTART can be interconnected with FB_ON or monitoring can be disabled by setting MONITOR = 0.

The FAULT_OFF parameter determines the significance of a monitoring error. If FAULT_OFF = 1, the motor is switched off when a fault occurs. This error status does not affect the control outputs when FAULT_OFF = 0.

Motor protection

A negative edge of the motor protection signal MSS sets the motor protection fault signal retentively and transfers this signal to output QMSS_ST. Parameter MSS_OFF defines whether only to indicate the fault state (MSS_OFF = 0), or whether to shut down the motor regardless of any other inputs and system states (MSS_OFF = 1).

Bumpless changeover

In order to ensure a bumpless changeover to manual mode, the manual value MAN_ON is always corrected to the current value of QSTART.

Error handling

The motor protection error (QMSS_ST = 1) and the monitoring error (QMON_ERR = 1) are reported to the OS and influence how the block works as described above. The operator can either reset these states with RESET, or automatically at the next positive edge at MSS by interconnecting L_RESET with a "1" signal. The control system fault CSF is merely reported to the OS and applied to the group error QGR_ERR along with the motor protection and monitoring errors. It does not have any further influence on the block algorithm.

Operator errors are indicated at output QOP_ERR without a message.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Startup after error state

Startup after error state depends on the mode set at the time of the reset:

- The motor can restart after a reset in auto mode if a corresponding start signal is provided by the automatic function.
- In manual mode, the motor must be switched on explicitly since manual operation had been tracked to "OFF".

Startup characteristics

When the CPU starts, the MOTOR block is switched to manual mode and the OFF command is output. A prerequisite is that the block must be called in the startup OB. In CFC engineering, this is handled by the CFC. When using basic STEP 7 tools, you enter this call in the startup OB. After startup, the messages will be suppressed during the number of cycles set at RUNUPCYC.

The START_OFF input is used to define whether to shut down the motor during CPU startup (START_OFF = 1) or whether to retain last operating state.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SAMPLE_T parameter.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for MOTOR (Page 216)".

Message response

The MOTOR block uses the ALARM_8P block for generating messages.

The following message triggers exist:

- Control system error
- Motor protective circuit-breaker error and monitoring error (runtime error)
- The CSF signal which is received as a control system error via the interconnection.

QMSG_SUP is set if the RUNUPCYC cycles have not expired since the restart or if MSG_STAT = 21.

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of MOTOR (Page 213)

Message texts and associated values of MOTOR (Page 215)

VSTATUS for MOTOR (Page 216)

Operating and monitoring MOTOR (Page 216)

3.19.2 I/Os of MOTOR

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
AUT_L	Interconnectable input for MANUAL/AUTO: (0=Manual/1=Auto)	BOOL	0	I		
AUT_ON_OP	Operator input: 0 = manual; 1 = auto	BOOL	0	IO	+	
AUTO_ON	AUTOMATIC value: 1 = ON, 0 = OFF	BOOL	0	I		
AUTOP_EN	1 = operator-control enable for auto mode	BOOL	1	I		
AUX_PRx	Associated value x	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: Consecutive batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	0	I	+	
CSF	Control-system error	BOOL	0	I		
FAULT_OFF	1 = Motor OFF in case of fault	BOOL	1	I		
FB_ON	Feedback: 1 = ON	BOOL	0	I		
L_RESET	Interconnectable RESET input	BOOL	0	I		
LIOP_SEL	Interconnectable input for manual/auto changeover (AUT_L): 1 = interconnection active, 0 = operator control active	BOOL	0	I		
LOCK	1 = lock (OFF)	BOOL	0	I	+	
LOCK_ON	1 = lock (ON)	BOOL	0	I	+	
MAN_ON	Operator input: 1 = ON, 0 = OFF	BOOL	0	IO	+	
MANOP_EN	1 = operator-control enable for manual mode	BOOL	1	I		
MONITOR	Monitoring: 1 = ON	BOOL	1	I	+	
MSG_ACK	Messages acknowledged	WORD	0	O		
MSG_EVID	Message number	DWORD	0	I		
MSG_STAT	Message error information	WORD	0	O		
MSS	Motor protective circuit breaker (active low, i.e. 0 = error)	BOOL	1	I		
MSS_OFF	1 = In case of motor-protection error, stop motor	BOOL	1	I		
OCCUPIED	ID for occupied by BATCH	BOOL	0	I	+	
OFFOP_EN	1 = operator-control enable for motor OFF	BOOL	1	I		
ON_OP_EN	1 = operator control enable for on	BOOL	1	I		
OOS	Reserve	BOOL	0	I	+	
QAUTOP	1 = operator-control enable for automatic mode	BOOL	0	O	+	
QC_FB_ON	Quality code for FB_ON	BYTE	16#80	I		
QC_QSTART	Quality code for QSTART	BYTE	16#80	O		
QC_QSTART_I	Quality code for output QSTART	BYTE	16#80	I		

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
QERR	1 = error output (inverted ENO)	BOOL	1	O	+	
QGR_ERR	1 = group error	BOOL	0	O		
QMAN_AUT	0 = manual, 1 = auto	BOOL	0	O	+	
QMANOP	1 = operator control enable for manual mode	BOOL	0	O	+	
QMON_ERR	1 = monitoring error	BOOL	0	O	+	
QMSG_ERR	1 = message error	BOOL	0	O	+	
QMSG_SUP	1 = message suppression active	BOOL	0	O	+	
QMSS_ST	Stored motor protective circuit-breaker (1 = error)	BOOL	0	O	+	
QOFF_OP	1 = operator-control enable for motor OFF	BOOL	0	O	+	
QON_OP	1 = operator control enable for on	BOOL	0	O	+	
QOP_ERR	1 = group operator-input error	BOOL	0	O		
QRUN	1 = motor is running	BOOL	0	O	+	
QSTART	Control output: 1 = ON	BOOL	0	O	+	
QSTOP	1 = motor is stopping	BOOL	0	O	+	
RESET	Enabled input of error reset	BOOL	0	IO	+	
RUNUPCYC	Number of run-up cycles	INT	3	I		
SAMPLE_T	Sampling time in seconds	REAL	1.0	I		> 0
START_OFF	1 = when starting up: Motor OFF	BOOL	1	I		
STEP_NO	BATCH step number	DWORD	0	I	+	
TIME_MON	Monitoring time in seconds	REAL	3.0	I	+	> 0
USTATUS	Status word in VSTATUS, can be configured by user	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.19.3 Message Texts and Associated Values of MOTOR

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QMSS_ST	$$BlockComment$$ Motor protection	S	-
2	QMON_ERR	$$BlockComment$$ Monitoring Fault	S	-
3	CSF	$$BlockComment$$ External fault	S	-

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data and the remaining ones (AUX_PRx) can be freely assigned by the user.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	AUX_PR04
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.19.4 VSTATUS for MOTOR

The 32-bit statusword extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	QMON_ERR	-	QMSS_ST	-	QMAN_AUT	-	BA_EN	OCCUPIED
Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	-	-	QSTOP	QRUN	LOCK

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.19.5 Operating and Monitoring of MOTOR

Additional information

You will find additional information in the following sections:

- MOTOR block icon (Page 590)
- MOTOR faceplate (Page 540)

3.20 MPC: Model-predictive controller

3.20.1 Description of MPC

Object name (type + number)

FB 142

MPC block I/Os (Page 231)

Area of application of MPC

The block is used in much the same way as a PID controller 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.

Area of application

The block is used for multivariable control of dynamic processes. It can handle up to four dependent manipulated and controlled variables as well as a measurable disturbance variable.

In special situations, the MPC 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 or a strongly oscillating response.

The MPC 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 or includes an integrator (for example, tank level control), the corresponding partial transfer function must be stabilized with a secondary controller.

For an integrating system, a simple P controller is adequate.

Note on the area of application of the controller: Longer execution times

Due to the principle on which the multivariable controller works, its execution time is significantly longer than that of PID controllers. This makes the multivariable controller unsuitable for fast controls.

The computation time required on the CPU is compensated for by the fact that very slow sample times of > 20 s are needed for typical MPC applications (see Advanced process control templates (Page 646)). The MPC is then typically in OB 30 and can be interrupted by faster OBs.

Operating principle

The MPC block is a model-based predictive multivariable controller. It uses a mathematical model of the process including the 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 criterion for a fit (quality)

$$J = (\vec{w} - \vec{y})^T \cdot R \cdot (\vec{w}-\vec{y}) + \Delta\vec{u}^T \cdot Q \, \Delta\vec{u}$$

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 matrix, the controller moves its manipulated variables more cautiously resulting in a slower but more robust control action. Using the weighting factors in the R matrix, you specify the relative significance of the individual controlled variables. A higher weighting (priority) for a controlled variable means that this 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-scheme (Dynamic Matrix Control), 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 limits that must not be violated. This means that precise setpoints or time zones for the controlled variables are taken into account as well as possible in the optimization. Using a reference variable filter for future setpoint settings, the control action of the function block can be finely adjusted during operation.

You can achieve significant improvements in control quality when individual disturbance variables 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 variable on the controlled variables when predicting the controlled variables so that the controller can react preemptively to such disturbances.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB 3x). The block is also installed automatically in the startup OB (OB100) and in OB 1. 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. Start the Engineering Tool ModPreCon for the MPC block in Windows with Start > SIMATIC > STEP 7 > Engineering Tool ModPreCon.
4. Using the tool, create an SCL source code for the user data block (DB). It contains the models and matrices required for an MPC 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 input DB_No of MPC block. The values are adopted by the controller by setting "Reset" at the Restart input.

Note

You will find a detailed description of this procedure in the help on the ModPreCon engineering tool.

The templates in the library include an example of how to use the MPC block.

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 MPC is filled with data.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SampleTime parameter by CFC during compilation.

Note

During controller design in the ModPreCon tool, a controller cycle time and an OB sampling time are calculated and displayed. You yourself are responsible for the MPC block being called in the cyclic interrupt level suitable for the OB sampling time. When necessary, slower controller cycle times will be implemented automatically in the block by an internal downsampling function.

Message response

The MPC block does not have a separate message response. If necessary, you can expand this with downstream MEAS_MON blocks for each controlled variable to be monitored.

Status word allocation

Bit number:	3	2	1	0
Parameter:	-	-	BatchEn	Occupied

Bit number:	7	6	5	4
Parameter:	SP_TrkCV	DV_ModelAvailable	DB_Loaded	No automatic mode

Bit number:	11	10	9	8
Parameter:	MV2TrkOn	MV1TrkOn	-	-

Bit number:	15	14	13	12
Parameter:	-	-	MV4TrkOn	MV3TrkOn

Bit number:	19	18	17	16
Parameter:	-	-	-	-

Bit number:	23	22	21	20
Parameter:	-	-	-	-

Bit number:	27	26	25	24
Parameter:	UserStatus bit 4	UserStatus bit 3	UserStatus bit 2	UserStatus bit 1

Bit number:	31	30	29	28
Parameter:	UserStatus bit 8	UserStatus bit 7	UserStatus bit 6	UserStatus bit 5

See also

MPC error handling (Page 230)
Functions of MPC (Page 223)
MPC modes (Page 221)

3.20.2 MPC modes

MPC modes

The block can be operated in all standard operating modes available for the controller blocks:

- Automatic mode
- Manual mode

In manual mode, the device is controlled by the operator, i.e. manually. The operator decides how to change the block's manipulated variable.

In automatic mode, the device is controlled automatically by the block algorithm; in other words, the process is controlled automatically. When there is a transfer from automatic to manual mode, the last valid control of the device in automatic mode initially remains valid.

The manual and automatic modes apply to the entire block with all control channels. In contrast to PID controllers, it is permitted to run the MPC block in automatic mode without its control signals affecting the process because there is no risk of integrator windup.

The next section provides additional block-specific information relating to the general descriptions. This includes, for example, the parameters for mode changes.

Automatic mode

The controller is operated in automatic mode if

- The AutOnOp = 1 I/O is set or
- Automatic mode is activated in the block's standard view.

Manual mode

The controller is operated in manual mode if

- The AutOnOp = 0 I/O is set or
- Manual mode is activated in the block's standard view.

Changing between modes

The transfer from manual to automatic mode can be bumpless. You set this response at the SP_TrkCV I/O.

- **Bumpless transfer** means that the automatic manipulated variable tracks (SP_TrkCV = 1) the manual manipulated variable. This achieves a gentle transfer from manual to automatic mode without a step change in the value of the manipulated variable in continuous controllers. A bumpless transfer can also be achieved using the faceplate.
- **Practically bumpless transfer** means that the block immediately recalculates the value of the manipulated variable based on the setpoint and process value when the mode is changed, in other words, there can be a step change in the setpoint.

The modes can be changed either by the operator (input AutOnOp) or by the higher-level control system (input AutOnLi). You decide whether to enable operator input or the interconnection by setting input LiOpMod accordingly.

See also

MPC I/Os (Page 231)

MPC error handling (Page 230)

Functions of MPC (Page 223)

Description of MPC (Page 217)

3.20.3 Functions of MPC

Functions of MPC

The block provides the following functions:

- Generating and limiting the manipulated variable
- Manipulated value tracking
- Setting the setpoint internally
- Setpoint tracking in manual mode
- Setpoint filters
- Control error generation and deadband
- Predictive controller algorithm
- Anti-windup
- Model-based disturbance variable compensation
- Control of square and non-square systems
- Control of linear and non-linear systems
- Display and output of the quality code
- BATCH functionality

Generating and limiting the manipulated variable

The manipulated variables MV1 to MV4 can be obtained from different sources:

- From the predictive-controller algorithm in automatic mode,
- From the manual inputs MV1Man to MV4Man in manual mode,
- From the interconnectable tracking inputs MV1Trk to MV4Trk in manipulated variable tracking mode (can be activated individually for each channel).

In automatic mode, the manipulated variables are limited to:

- High limits: MV1HiLim to MV4HiLim
- Low limits: MV1LoLim to MV4LoLim.

In manual mode, the manipulated variables are limited to:

- High limits: MV1ManHiLim to MV4ManHiLim
- Low limits: MV1ManLoLim to MV4ManLoLim.

The limits for automatic mode are normally more restrictive 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 setting range in automatic mode.

The gradients of the manipulated variables (changes per second) are limited to MV1RaLim to MV4RaLim in automatic mode. Gradient limitation applies both to the positive and negative directions.

Manipulated value tracking

Tracking can be activated separately per channel via one of the inputs MV1TrkOn to MV4TrkOn. The corresponding manipulated variable is then set by the interconnected tracking input MV1Trk to MV4Trk. Manual mode has priority over tracking so that the user can still intervene when the value of the manipulated variable is tracking.

A typical use case is cascade control: If the secondary controller assigned to a control channel is no longer in automatic with external setpoint mode, the corresponding control channel of the MPC primary controller must be changed to tracking mode.

Setting the setpoint internally

With this block, the setpoint must always be set internally at the SP1 to SP4 I/Os. These are normally set 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 to SP4 track the assigned controlled variables CV1 to 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 creating a new user data block with the MPC tool and reinitializing the controller. The specified time constant PreFilt1 to PreFilt4 of the setpoint filter can be interpreted as the required settling time of the 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 MPC 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 change if the process value deviates significantly from the setpoint due to disturbances. This means that the filter not only slows down the control action but indirectly also the response to disturbances.

The control action 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 engineering tool with the "Manipulated variable change penalty" parameter and then to optimize this in the software using the function for simulation the closed control loop. The software filter should then only be used for fine modification of the action in the operational system.

Control error generation and deadband

In the predictive controller, the error signal as the deviation between the predicted movement of the process (starting at the current value of the controlled variable CV1 to CV4) and future setpoint settings (ending at SP1 to SP4) is generated over the entire prediction horizon for each control channel and used to calculate the manipulated variable.

In principle, the effect of the deadbands 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 deadband 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 deadband

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 fit criterion (see MPC engineering software), the user can specify that threatened violations of the pressure limits are given a particularly high weighting.

Predictive controller algorithm

The MPC block was derived from the known DMC algorithm (Dynamic Matrix Control). Future changes to the manipulated variables 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 (at constant manipulated variables) in the future
- C is the constant controller matrix calculated by the MPC Engineering Tool. C includes both the process model and the weighting of the manipulated variable changes and the controlled variables from the fit criterion 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.

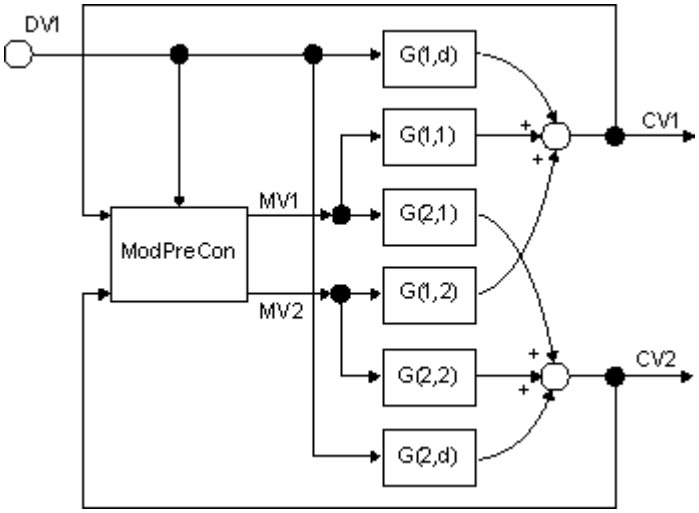
With predictive controllers, the manipulated variable changes are based on the control deviations predicted in the future, while with a PID controller, they are based on control errors of the past (possibly also integrated). This can be interpreted as a "looking ahead" strategy.

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 variable compensation

Model-based disturbance variable 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 to CV4 can be estimated when the controller is put into manual mode. This means that no movements of the controlled variables whatsoever result from changes to the manipulated variables and all movements result from the disturbance variable. If the disturbance variable 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 variable changed.

The identification of the transfer functions from the disturbance variable DV1 to all controlled variables CV1 to CV4 (disturbance model, in the graphic above G(1,d) and G(2,d)) is performed with the MPC Engineering Tool 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 MPC block and disturbance variable 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 countermeasures in advance before the disturbance can have a massive influence on the controlled variables. If there is no disturbance model in the user data block, the DV1 input is ignored.

Typical examples of measurable disturbance variables are inlet volumes in distillation columns or throughput of continuous reactors.

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 to not influence operation, the controller can, in principle, control to the selected setpoints.

If there are less manipulated variables than controlled variables, or 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 MPC algorithm then finds a compromise that can be influenced by the selection of controlled value weights (priorities) in the MPC Engineering Tool: Controlled variables with higher priority will have lower control deviations.

Note

Since MPC 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.

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. A lean predictive controller algorithm, however, cannot recognize this situation explicitly and use the free manipulated variables for optimization. The MPC 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 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 MPC 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.

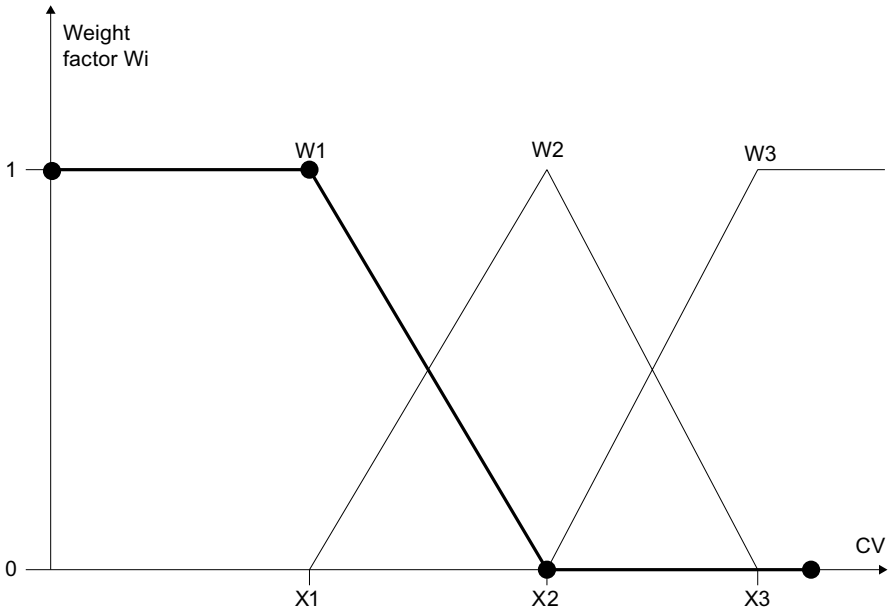
Again analogous to the PID controller, there are, however, several possibilities with which the area of application can be extended with non-linear systems:

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 polyline 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 polyline 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:

Several MPC instances with different models for different operating points run at the same time. The final value of the manipulated variable is formed as a weighted mean value of the manipulated variables proposed by the individual controllers. 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. 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, 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).



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 values; 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 into the MPC block via the inputs MV1Traj to MV4Traj 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 inputs SP1 to 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.

Display and output of the quality code

The block can process and output several quality codes.

The following inputs are available:

- Quality of controlled variable: QC_CV1 to QC_CV4
- Quality of the measurable disturbance variable: QC_DV

The following additional output is also available for further processing:

- Worst quality code: QC_Worst

You will find general information on quality codes in the section Functions of the blocks > Quality codes

BATCH functionality

This block has a Batch interface. You can use SIMATIC BATCH by interconnecting the BatchEn, BatchID, BatchName, StepNo and Occupied I/Os with the corresponding BATCH blocks. Refer to the SIMATIC BATCH documentation.

See also

MPC I/Os (Page 231)

MPC error handling (Page 230)

MPC modes (Page 221)

Description of MPC (Page 217)

3.20.4 MPC error handling

MPC error handling

The ErrorNum I/O can be used to output error numbers.

Overview of error numbers

Error numbers output by this block:

Error number	Meaning of the error number
-1	Default value when the block is installed; this message is irrelevant
0	There is no error.
2	SampleTime < 0,001 [s]
21	The value of MV_Trk1 can no longer be displayed in the REAL number field.
22	The value of MV_Trk2 can no longer be displayed in the REAL number field.
23	The value of MV_Trk2 can no longer be displayed in the REAL number field.
24	The value of MV_Trk3 can no longer be displayed in the REAL number field.
31	The value of CV1 can no longer be displayed in the REAL number field.
32	The value of CV2 can no longer be displayed in the REAL number field.
33	The value of CV3 can no longer be displayed in the REAL number field.
34	The value of CV4 can no longer be displayed in the REAL number field.
90	The controller matrix could not be loaded from the user data block.

See also

MPC I/Os (Page 231)
Functions of MPC (Page 223)
MPC modes (Page 221)
Description of MPC (Page 217)

3.20.5 MPC I/Os

ModPreCon I/Os

Inputs

I/O (parameters)	Description	Data type	Default Value
AutOnLi	1 = automatic mode via interconnection	BOOL	0
AutOpEn	1 = operator may activate automatic mode	BOOL	1
AutOnOp	1 = automatic mode by operator	BOOL	0
BatchEn	1 = activate control by means of batch	BOOL	0
BatchID	Current batch ID (number)	DWORD	16#FF
BatchName	Current batch name	String [32]	''
CVx	(x = 1 - 4) Control variable 1 - 4 (process value)	REAL	0
DB_No	Number of databases in which control data is stored	INT	
DV_On	Select disturbance feedforward	BOOL	
DV1	Disturbance variable	REAL	0
LiOpMod	Mode transfer between: 0 = Operator 1 = interconnection	BOOL	0
ManOpEn	1 = operator may activate manual mode	BOOL	1
MV_ManOpEn	1 = operator may change manipulated variables	BOOL	1
MV1HiLim	High limit of manipulated variable 1	REAL	1
MV1LoLim	Low limit of manipulated variable 1	REAL	0
MVxMan	(x = 1 - 4) 1 = set manipulated variable 1 - 4 in manual mode	REAL	0
MVxManHiLim	(x = 1 - 4) High limit of manipulated variable 1 - 4 in manual mode	REAL	1
MVxManLoLim	(x = 1 - 4) Low limit of manipulated variable 1 - 4 in manual mode	REAL	1
MVxRaLim	(x = 1 - 4) Manipulated variable 1 - 4 rate of change (per s)	REAL	1
MVxTrk	(x = 1 - 4) Manipulated variable 1 - 4 in tracking mode	REAL	0
MVxTrkOn	(x = 1 - 4) 1 = activate tracking mode for manipulated variable 1 - 4	BOOL	0
Occupied	Occupied by Batch	BOOL	0
PreFiltx	(x = 1 - 4) Setpoint 1 - 4 of the settling time of the prefilter [s]	REAL	0
QC_CVx	(x = 1 - 4) Quality code for the signal CV1 - CV4	BYTE	16#80
QC_DV1	Quality code for signal DV1	BYTE	16#80
Restart	Restart with transfer of data from user data block	BOOL	1
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1
SP_OpEn	1 = operator may change setpoints	BOOL	1
SP_TrkCV	Setpoint tracks CV in manual and tracking mode	BOOL	
SPx	(x = 1 - 4) Setpoint 1 - 4	REAL	0
SPxDeadBand	(x = 1 - 4) Radius for range control of CV1 - CV4	REAL	0

I/O (parameters)	Description	Data type	Default Value
SPxHiLim	(x = 1 - 4) High limit for setpoint 1 - 4	REAL	1
SPxLoLim	(x = 1 - 4) Low limit for setpoint 1 - 4	REAL	0
StepNo	Step number for batch	DWORD	16#00
UserStatus	Free user area for status word (bit 24 to bit 31)	BYTE	16#00

Outputs

I/O (parameter)	Description	Data type	Default value
ActInOuts	Status word displays active inputs and outputs on the screen	WORD	
AutAct	1 = automatic mode is active 0 = manual mode is active	BOOL	0
CVxOut	(x = 1 - 4) Copy of control variables 1 - 4 as output parameter	REAL	0
DV_Model_Available	1 = disturbance feedforward can be activated since disturbance model exists	BOOL	1
ErrorNum	Configuration error For error numbers, see Error handling (Page 230)	INT	0
MVx	(x = 1 - 4) Manipulated variable 1 - 4 (controller output)	REAL	0
MVxAutAct	(x = 1 - 4) Manipulated variable 1 - 4 set automatically	BOOL	0
NumberCVs	Number of CVs	INT	0
NumberDVs	Number of DVs	INT	0
NumberMVs	Number of manipulated variables	INT	0
Q_AutOpEn	1 = operator may activate automatic mode	BOOL	1
Q_ManOpEn	1 = operator may activate manual mode	BOOL	1
Q_SP_OpEn	1 = operator may change setpoints	BOOL	1
Q_MV_ManOpEn	1 = operator may change manipulated variables	BOOL	1
SPxOut	(x = 1 - 4) Active setpoint 1 - 4 used by operator	REAL	0
Status	Status word	DWORD	0

See also

Functions of MPC (Page 223)
MPC modes (Page 221)
Description of MPC (Page 217)

3.21 NOISE_GN: Noise-signal generator

3.21.1 Description of NOISE_GN

Object name (type + number)

FB 143

- NOISE_GN block I/Os

How it works

The block serves as a noise generator. It is used to generate simulated signals affected by noise.

These simulated signals are used the control examples.

Startup characteristics

The block does not have any startup characteristics.

Time response

The block does not have any time response.

Status word allocation

The block does not have a status word allocation.

3.22 POLYG_P: Polygon with a maximum of 8 points

3.22.1 Description of POLYG_P

Object name (type + number)

FC271

- POLYG_P block I/Os (Page 235)

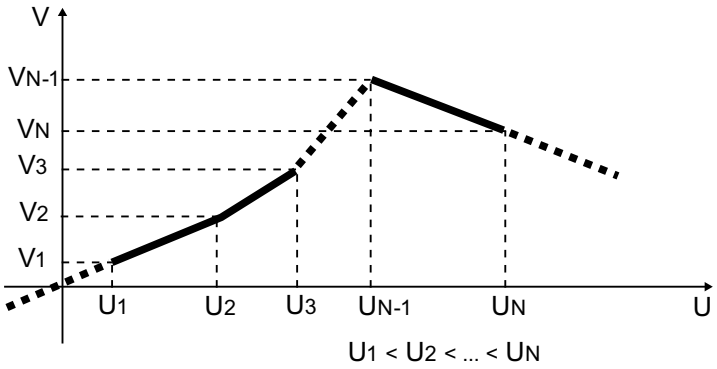
Function

An input U is converted to output V according to a non-linear characteristic curve and with a maximum of 8 vertices.

How it works

After the N vertices have been specified (pairs of coordinates Ui,Vi where i = 1...N in a continuous sequence) and the number N has been set, the block operates as follows:

- Interpolation between the vertices is linear
- Extrapolation outside the end vertices is based on the first two and last two vertices



Characteristic curve representation

Calling OBs

The OB in which you install the block.

Error handling

ENO = 0 and V = U are output if the following conditions are met:

- The number of vertices N < 2 or N > 8
- Ui >= Ui+1 for i = 1, 2...N-1

3.22.2 I/Os of POLYG_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
N	Number of vertices	INT	0	I		2 ≤ N ≤ 8
U	Input	REAL	0.0	I		
U1	U value of vertex 1	REAL	0.0	I		
U2	U value of vertex 2	REAL	0.0	I		
...	...	...	...	...		
U8	U value of vertex 8	REAL	0.0	I		
V1	V value of vertex 1	REAL	0.0	I		
V2	V value of vertex 2	REAL	0.0	I		
...	...	...	...	...		
V8	V value of vertex 8	REAL	0.0	I		
V	Output value	REAL	0.0	O		

3.23 PT1_P: First-order lag element

3.23.1 Description of PT1_P

Object name (type + number)

FB51

- PT1_P block I/Os (Page 237)

Function

The PT1_P block operates according to the formula:

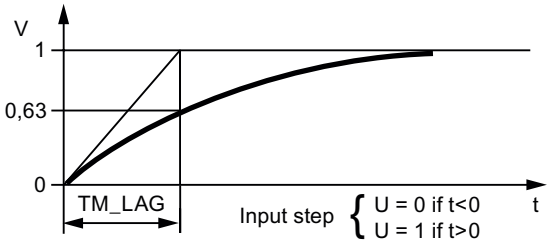
v(s) = 1 / (TM_LAG * s +1) * u(s)

How it works

The input signal U is passed to the output V in accordance with the time constant TM_LAG.

The STOP_RES input has the following effect:

- If STOP_RES = "1", the calculation process is stopped. The output value is retained for the duration of this period.
- The output is reset (V = U) by a falling edge 1 → 0.



Step response of PTI_P

Calling OBs

This is the OB in which you install the block (for example, OB32).

Error handling

The configuration TM_LAG = 0 is permitted and means there is no lag, i.e., V = U. In contrast, negative values for TM_LAG are seen as parameter assignment errors and are acknowledged with QERR = TRUE.

3.23.2 I/Os of PT1_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	Permissible values
QERR	1 = Error	BOOL	1	O	
SAMPLE_T	Sampling time in seconds	REAL	1.0	I	> 0
STOP_RES	Stop / reset PT1 function	BOOL	0	I	
TM_LAG	Length of time constant	REAL	0.0	I	> 0
U	Input value	REAL	0.0	I	
V	Output value	REAL	0.0	O	

3.24 RAMP_P: Ramp generation

3.24.1 Description of RAMP_P

Object name (type + number)

FB52

- RAMP_P block I/Os (Page 240)

Function

Limitation of the ramp of an analog signal

How it works

The block calculates the ramp of the input signal dU/dt and compares it with the two limits URLM for positive changes or DRLM for negative changes (refer to the table below).

- If the ramp (as a quantity) exceeds the relevant maximum ramp (URLM or DRLM), the output V is only changed by the permitted value and the corresponding limitation display QLIM_U or QLIM_D is set.
- If the rate of change is within the valid range, the input value is passed through (U = V) and the values QLIM_U and QLIM_D are reset.
- If the input RATE_OFF=1, the ramp generation is disabled, so that V = U and QLIM_U = QLIM_D = 0.

RATE_OFF	dU/dt	Meaning	Output V	QLIM_D	QLIM_U
0	< - DRLM	Input value dropping too fast	V-(DRLM * SAMPLE_T)	1	0
0	- DRLM to URLM	Rate of change is permitted	U	0	0
0	> URLM	Input value U rises too rapidly	V+(URLM * SAMPLE_T)	0	1
1	No meaning	Ramp disabled	U	0	0

Calling OBs

This is the cyclic alarm OB in which you install the block (for example, OB32) and OB100

Error handling

If SAMPLE_T < 0, ENO = 0 or QERR = 1 is output.

Startup characteristics

Output V is reset during startup. This means that the block must also be called in the startup OB (OB100).

Time response

The block must be called from a cyclic interrupt OB.

3.24.2 I/Os of RAMP_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	Permissible values
DRLM	Maximum negative change of the output value in [units/s]	REAL	3.0	I	DRLM < URLM
QERR	1 = Error	BOOL	1	O	
QLIM_D	Negative gradient too steep	BOOL	0	O	
QLIM_U	Positive gradient too steep	BOOL	0	O	
RATE_OFF	Rate of change monitoring off	BOOL	0	I	
SAMPLE_T	Sampling time in seconds	REAL	1.0	I	> 0
U	Analog input (measured value)	REAL	0.0	I	
URLM	Maximum positive change of the output value in units/seconds	REAL	3.0	I	URLM > DRLM
V	Ramp output	REAL	0.0	O	

3.25 RATIO_P: Ratio control

3.25.1 Description of RATIO_P

Object name (type + number)

FB70

- RATIO_P block I/Os (Page 243)
- RATIO_P block icon (Page 591)
- RATIO_P faceplate (Page 543)

Function

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.

How it works

The RATIO_P block works according to the formula: V = U1 * U2 + BIAS
U1 is obtained over the interconnection while U2 is selected dependent on the internal/external mode.

Internal/wxternal changeover

The **mode** is selected by the following measures and indicated at the output QIN_EX:

- Setting of input IN_EX, when L_IE_ON = 0 and the enable signals IN_OP_EN and EX_OP_EN are valid.
- Interconnection of L_IN_EX, if L_IE_ON = 1.

Internal: The parameter U2 in the formula is specified by the operator and, after limiting to (U2_LL, U2_HL), included in the formula. Operator control enable U2_OP_EN is required.

External: The parameter U2 is specified by interconnecting the input U2_EXT and, after limiting to (U2_LL, U2_HL), included in the formula. The controllable input U2 tracks U2_EXT to allow a smooth change to "internal".

Calling OBs

The RATIO_P block is installed in the OB that contains the block that utilizes the result. The RATIO_P block must be installed before this block (first calculate, then use).

Error handling

Arithmetic errors are indicated by ENO = 0 or QERR = 1.
The operator input errors are indicated collectively at output QOP_ERR.

Startup characteristics

No special measures.

Time response

If the result for the blocks with dynamic response is relevant (for example, ratio control, synchronization control), the block should be installed in the same OB, before these blocks.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for RATIO_P (Page 244)".

Message response

Not available

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of RATIO_P (Page 243)

VSTATUS for RATIO_P (Page 244)

Operating and monitoring RATIO_P (Page 244)

3.25.2 I/Os of RATIO_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
BIAS	Share by which V is moved	REAL	0	I	+	
EX_OP_EN	1 = operator-control enable for external	BOOL	0	I	+	
IN_EX	Operator input: 0 = internal, 1 = external	BOOL	0	IO	+	
IN_OP_EN	1 = operator-control enable for internal	BOOL	0	I	+	
L_IE_ON	1 = interconnection active, 0 = operator control active	BOOL	0	I	+	
L_IN_EX	Interconnectable input for IN_EX	BOOL	0	I		
MO_U1HR	High display limit	REAL	110	I	+	
MO_U1LR	Low display limit	REAL	-10	I	+	
QC_U1	Quality code for U1	BYTE	16#80	I		
QERR	1 = error output (inverted ENO)	BOOL	1	O	+	
QIE_OP	1 = operator-control enable internal/external changeover	BOOL	0	O	+	
QIN_EX	0 = internal, 1 = external	BOOL	0	O	+	
QOP_ERR	1 = operator-error output	BOOL	0	O	+	
QU2_OP	1 = operator-control enable U2 operator control	BOOL	0	O	+	
QVHL	1 = high limit of output value V triggered	BOOL	0	O	+	
QVLL	1 = low limit of output value V triggered	BOOL	0	O	+	
U1	Input value	REAL	0	I	+	
U2	Internal factor	REAL	1	IO	+	
U2_EXT	External factor	REAL	1	I	+	
U2_HL	High limit of U2	REAL	1	I	+	
U2_LL	Low limit of U2	REAL	0	I	+	
U2_OP_EN	1 = operator-control enable U2 operator control	BOOL	1	I		
V	Output value	REAL	0	O	+	
V_HL	High limit of V	REAL	100	I	+	V_HL>V_LL
V_LL	Low limit of V	REAL	0	I	+	V_LL<V_HL
USTATUS	Status word in VSTATUS, can be configured by user	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.25.3 VSTATUS for RATIO_P

The 32-bit status word extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	QIN_EX	-	-			-	-	-
Bit no.:	15	14	13	12	11	10	9	8
Parameters	-	-	-	-	-	-	QVLL	QVHL

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.25.4 Operating and Monitoring of RATIO_P

Additional information

You can find additional information in the following sections:

- RATIO_P block icon (Page 591)
- RATIO_P faceplate (Page 543)

3.26 READ355P: Read Digital and Analog Inputs from FM 355

3.26.1 Description of READ355P

Object name (type + number)

FB 72

- READ355P block I/Os (Page 247)

Function

Block READ355P is used to read the digital and analog inputs of an FM 355 or FM355-2 module.

Dependencies

Once the driver generator is complete, this block - along with blocks FMCS_PID and FMT_PID - is in a block chain controlled by FM_CO.

Take note of the dependencies on the FM_CO block too.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The parameter CO_NO is set
- The input EN_CO is interconnected with the output EN_CO_x of the FM_CO block (x = number of the rack)
- The output ENCO is interconnected with the input ENCOx_yy of the FM_CO block (x = number of the rack, yy = coordination number)

How it works

If the in/out parameter EN_CO.CO_ACT = CO_NO is set, the block begins to determine the following values by reading data records (SFC RED_REC):

- The CJ_TEMP parameter contains the cold-junction temperature measured at the cold junction in degrees C or in degrees F (depending on the selected temperature unit). If no "thermocouple" sensor type was selected or if the set cold-junction temperature was selected for all analog inputs with “thermocouple” sensor type, the CJ_TEMP parameter has the value "0.0".
- The parameters DI_0 through DI_7 indicate the actual state of the digital inputs 0 through 7.
- The parameters PV_PER_0 through PV_PER_3 indicate the value of the analog inputs 0 through 3 in the unit mA or mV.
- The parameters PV_PHY_0 through PV_PHY_3 indicate the preprocessed analog input values 0 through 3 in the physical unit.

Startup characteristics

Not available

Time response

Not available

Message response

Not available

Additional information

You will find more information in:

Addressing (Page 247)

I/Os of READ355P (Page 247)

3.26.2 Addressing

You address the controller channel of an FM 355 belonging to the instance using the logical base address (set with HW Config, LADDR input).

The FM 355 module is monitored with the blocks of the *PCS 7 Library*. The MODE input is connected to the OMODE output of the MOD_D1 block. Since the block communicates with the FM 355 only over read data record, the part of the measurement range coding in the low word of the OMODE output is irrelevant and is set to zero here.

3.26.3 I/Os of READ355P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
CJ_TEMP	Cold-junction temperature measured at the cold junction	REAL	0	O		
CO_NO	Cold-junction temperature measured at the cold junction	INT	0	O		
DI_x	Digital input (x = 0 through 7)	BOOL	0	O		
EN_CO	Current coordination number	STRUCT				
ENCO	Coordination number	BOOL	0	IO		
LADDR	Logical address FM 355	INT	0	I		
MODE	Mode	DWORD	0	I		
PV_PER_x	Analog input (x = 0 through 3)	REAL	0	O		
PV_PHY_x	Physical analog input (x = 0 through 3)	REAL	0	O		
QDONE	1 = parameter read	BOOL	0	O		
QMODF	1 = module error	BOOL	0	O		
RET_VALU	Return value SFC 59	INT	0	O		

3.27 SIG_SMTH: Low pass filter

3.27.1 Description of SIG_SMTH

Object name (type + number)

FB 144

- SIG_SMTH block I/Os (Page 252)

Area of application of SIG_SMTH

The block is used for the following applications:

- Butterworth low pass filter, 2nd order
- Detection and removal of mavericks

How it works

The block is used as a low pass filter. This filter allows signal components with frequencies below its cutoff frequency to pass practically unattenuated, components with high frequencies, on the other hand, are attenuated.

The maverick-detection function monitors adjacent signals. If signal mavericks are detected, they are not processed any further. The block outputs the last valid signal.

Configuration

Use the CFC editor to install the block in a cyclic interrupt OB (OB 3x).

Further addressing is not required.

The templates of the advanced process controls contain an example of how SIG_SMTH is used.

Startup characteristics

When OB 100 is called, the state variables of the Butterworth filter are initialized based on the current process values.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SampleTime parameter by CFC during compilation.

Message response

The block does not have any message response.

See also

Error handling of SIG_SMTH (Page 251)
Functions of SIG_SMTH (Page 250)
Modes of SIG_SMTH (Page 249)

3.27.2 Modes of SIG_SMTH

Modes of SIG_SMTH

This block does not provide any modes.

See also

SIG_SMTH I/Os (Page 252)
Error handling of SIG_SMTH (Page 251)
Functions of SIG_SMTH (Page 250)
Description of SIG_SMTH (Page 248)

3.27.3 Functions of SIG_SMTH

Functions of SIG_SMTH

The block provides the following functions:

- Restart low pass filter
- Activate and deactivate maverick detection

Restart low pass filter

You can recalculate 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 preassigned so that the `CleanPV` output parameter is equal to the PV input parameter.

Activate and deactivate maverick detection

The maverick-detection function monitors the PV process value for consistency. You activate maverick detection by setting the `OutlDetOn = 1` input parameter.

If there are maverick signals in the PV process value, they are selected by the block and and the last valid value is output.

You set the number of maverick signals after which the last valid value should be output via the `OutlCycles` input parameter.

See also

SIG_SMTH I/Os (Page 252)

Error handling of SIG_SMTH (Page 251)

Modes of SIG_SMTH (Page 249)

Description of SIG_SMTH (Page 248)

3.27.4 Error handling of SIG_SMTH

Error handling of SIG_SMTH

The ErrorNum I/O can be used to output error numbers.

Overview of error numbers

Error numbers output by this block:

Error number	Meaning of the error number
0	There is no error.
2	SampleTime < 0,001 [s]
32	The value of PV can no longer be displayed in the real number field or is not a number
72	TimeConstant < 2 · SampleTime

See also

SIG_SMTH I/Os (Page 252)
Functions of SIG_SMTH (Page 250)
Modes of SIG_SMTH (Page 249)
Description of SIG_SMTH (Page 248)

3.27.5 SIG_SMTH I/Os

SIG_SMTH I/Os

Inputs

I/O (parameters)	Description	Data type	Default Value	Attributes
FilterOn	1 = filter activated	BOOL	0	S7_m_c := 'false' S7_link := 'true'
OutlCycles	Number of cycles to avoid special situations	REAL	3	S7_m_c := 'false' S7_link := 'true'
OutlDetOn	1 = special-situation detection activated	BOOL	0	S7_m_c := 'false' S7_link := 'true'
OutlThreshold	Threshold for special-situation detection	REAL	10	S7_m_c := 'false' S7_link := 'true'
PV	Process value	REAL	0	S7_m_c := 'false' S7_link := 'true'
Restart	Reset the filter algorithm	BOOL	0	S7_m_c := 'false' S7_link := 'true'
SampleTime	Sampling time [s] (assigned automatically)	REAL	0.1	S7_m_c := 'false' S7_link := 'true'
TimeConstant	Butterworth: Filter time is constant [s]	REAL	10	S7_m_c := 'false' S7_link := 'true'

Outputs

I/O (parameters)	Description	Data type	Default Value	Attributes
CleanPV	Output of the process value	REAL	0	S7_m_c := 'false' S7_link := 'true'
ErrorNum	Output of the current error number. For the error numbers this block is capable of outputting, see "Error handling".	INT	0	S7_m_c := 'false' S7_link := 'true'
OutlDetected	1 = special situation detected	BOOL	0	S7_m_c := 'false' S7_link := 'true'

See also

Error handling of SIG_SMTH (Page 251)
Functions of SIG_SMTH (Page 250)
Modes of SIG_SMTH (Page 249)
Description of SIG_SMTH (Page 248)

3.28 SPLITR_P: Split range

3.28.1 Description of SPLITR_P

Object name (type + number)

FC272

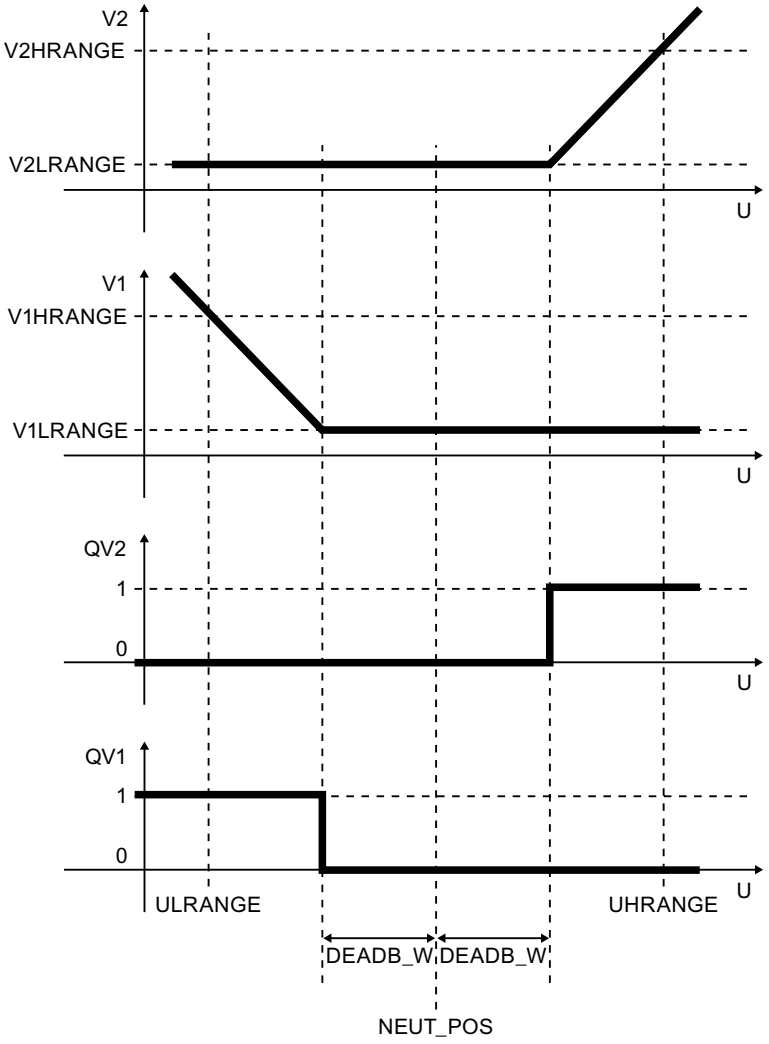
- SPLITR_P block I/Os (Page 257)

Function

Together with a controller block, the block is used to implement a split-range control.

How it works

The block is installed in the run sequence downstream of the controller block. The controller output of the controller block is interconnected with the input U of the SPLITR_P block. The neutral position and the dead band zone are set by means of the corresponding parameters. V1 and V2 are adapted to the physical variable by configuring the high/low limits of V1 and V2. The transfer characteristics have the following appearance:



Transfer characteristics of the SPLITR_P

Calling OBs

The OB in which the controller block runs whose manipulated variable is processed.

Error handling

ENO = 0 is output for the following errors:

- Incorrect calculation of V1 (whereby V1 = V1LRANGE also applies)
- Incorrect calculation of V2 (whereby V2 = V2LRANGE also applies)

3.28.2 I/Os of SPLITR_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

Detailed information about the abbreviations used is available in the section "General information pertaining to the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	Valid values
DEADB_W	Dead band width	REAL	10.0	I	
NEUT_POS	Neutral position	REAL	50.0	I	> ULRANGE < UHRANGE
QV1	1 = output 1 is active	BOOL	0	O	
QV2	1 = output 2 is active	BOOL	0	O	
U	Input value	REAL	0.0	I	
UHRANGE	Measurement range high limit of U	REAL	100.0	I	> ULRANGE
ULRANGE	Measurement range low limit of U	REAL	0.0	I	< UHRANGE
V1	Output 1	REAL	0.0	O	
V1HRANGE	Final value of V1 measurement range	REAL	100.0	I	> V1LRANGE
V1LRANGE	Initial value of V1 measurement range	REAL	0.0	I	< V1HRANGE
V2	Output 2	REAL	0.0	O	
V2HRANGE	Final value of V2 measurement range	REAL	100.0	I	> V2LRANGE
V2LRANGE	Initial value of V2 measurement range	REAL	0.0	I	< V2HRANGE

3.29 SWIT_CNT: Switching cycles counter

3.29.1 Description of SWIT_CNT

Object name (type + number)

FB71

- SWIT_CNT block I/Os (Page 260)
- SWIT_CNT block icon (Page 592)
- SWIT_CNT faceplate (Page 546)

Function

The SWIT_CNT block counts the switching operations of a unit.

How it works

Block SWIT_CNT counts the switching operations at a positive edge (0 → 1) at input ON_OFF and outputs the result at output V. The maximum count value is limited to 2 ^ 31 switching operations, since it is of the data type DINT.

Calling OBs

The calling OB is in the same OB and comes after the block that supplies switching signals it is also in OB100.

Tracking

The counter output V can track the value VTRACK_OP by setting TRACK_OP on the OS or by interconnecting TRACK. VTRACK_OP can, in turn be controlled by the operator.

Monitoring limits

A warning or alarm limit can be set at each of the inputs VWH and VAH. When these are exceeded, the count value V generates a display (at QH_WRN/QH_ALM) and, if necessary, a message. You will find more information in "Message response".

Error handling

If there is a counter overflow at V, the error output QERR is set for one cycle and a control system message is sent. Counting then continues at 0.
Operator input errors are displayed at the QOP_ERR output.

Startup characteristics

During startup (OB100), output V = 0 is set. After startup, the messages will be suppressed during the number of cycles set at RUNUPCYC.

Time response

The block must be executed in the same runtime group (if configured in CFC) as the control block of the unit to be monitored. It must detect the edges reliably.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for SWIT_CNT (Page 262)".

Message response

The SWIT_CNT block uses the ALARM_8P block to generate messages.

Messages are triggered by the functions for monitoring the switching operation limits.

Limit violation messages can be suppressed individually using the the relevant M_SUP_xx inputs. Process messages (not control system messages!) can be disabled centrally using MSG_LOCK.

QMSG_SUP is set if the RUNUPCYC cycles were not completed since the last restart and if MSG_LOCK = TRUE or MSG_STAT = 21.

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of SWIT_CNT (Page 260)

Message texts and associated values of SWIT_CNT (Page 261)

VSTATUS for SWIT_CNT (Page 262)

Operating and monitoring SWIT_CNT (Page 262)

3.29.2 I/Os of SWIT_CNT

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
AUX_PRx	Associated value x	ANY	0	IO		
M_SUP_AH	1 = suppress HL alarms	BOOL	0	I	+	
M_SUP_WH	1 = suppress HL warnings	BOOL	0	I	+	
MO_VHR	High display limit	REAL	120	I	+	
MO_VLR	Low display limit	REAL	0	I	+	
MSG_ACK	Acknowledge messages	WORD	0	O		
MSG_EVID	Message number	DWORD	0	I		
MSG_LOCK	1 = lock process messages	BOOL	0	I	+	
MSG_STAT	Message error information	WORD	0	O		
ON_OFF	Unit state to be counted: 1 = ON, 0 = OFF	BOOL	0	I	+	
OOS	Reserve	BOOL	0	I	+	
QC_ON_OFF	Quality code for ON_OFF	BYTE	16#80	I		
QERR	1 = error output (inverted ENO)	BOOL	1	O	+	
QH_ALM	1 = high alarm limit exceeded	BOOL	0	O		
QH_WRN	1 = high warning limit exceeded	BOOL	0	O		
QMSG_ERR	1 = ALARM_8P error	BOOL	0	O	+	
QMSG_SUP	1 = message suppression	BOOL	0	O	+	
RUNUPCYC	Number of run-up cycles	INT	3	I		
TRACK	Interconnectable input for TRACK	BOOL	0	I		
TRACK_OP	1 = follow V to VTRACK_OP	BOOL	0	IO	+	
V	Number of switching cycles	DINT	0	O	+	
VAH	High limit alarm, count value	DINT	100	I	+	
VTRACK_OP	Tracking value	DINT	0	IO	+	> 0
VWH	High limit warning, count value	DINT	95	I	+	
USTATUS	Status word in VSTATUS, can be configured by user	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.29.3 Message Texts and Associated Values of SWIT_CNT

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QAH	$$BlockComment$$ HighHigh Alarm	M	M_SUP_AH, MSG_LOCK
2	QWH	$$BlockComment$$ High alarm	M	M_SUP_WH, MSG_LOCK

All the associated values (AUX_PRx) of the message block can be freely assigned by the user.

Assignment of the associated values to block parameters

Associated value	Block parameters
1	AUX_PR01
2	AUX_PR02
3	AUX_PR03
4	AUX_PR04
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.29.4 VSTATUS for SWIT_CNT

The 32-bit statusword extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	ON_OFF	-	-	-	-	MSG_LOCK	-	-
Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	-	-	-	-	-

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.29.5 Operator Control and Monitoring of SWIT_CNT

Additional information

Additional information is available in the following sections:

- SWIT_CNT block icon (Page 592)
- SWIT_CNT faceplate (Page 546)

3.30 VAL_MOT: Motor valve control

3.30.1 Description of VAL_MOT

Object name (type + number)

FB 74

- VAL_MOT block I/Os (Page 268)
- VAL_MOT block icon (Page 592)
- VAL_MOT faceplate (Page 548)

Function

Block VAL_MOT is used to control motor-driven valves by means of two control signals. The valve can be stopped at any position. The two position feedback signals (open/closed) can be optionally monitored. The position feedback signals are generated by limit switches.

NOTICE
With the PCS 7 it is not intended that other blocks will be inserted between the plant block and the output driver. If you deviate from this principle, ensure that when interconnecting the block, that from the outputs of the plant block until the output driver, that all blocks that form the output signal are installed in the same OBs.

How it works

Various inputs are available for controlling the motor-driven valve. These are implemented in a defined hierarchical dependency to each other and to system states. In particular the interlock, the feedback or rotary direction monitoring and the motor protective circuit-breaker influence the control signals QSTART (1: motor on, 0: motor off) and QOC (1: open, 0: close).

The allocation of priorities to the individual input variables and events with regard to their influence on the control signal is shown in the following table. The sections following provide further details.

Priority:	Event:
High	Motor protection fault, if MSS_OFF = 1
⇑	Monitoring error, if FAULT_OFF = 1
	Wait time at change of rotary direction
	V_LOCK = 1
	VL_CLOSE = 1
	VL_OPEN = 1
⇓	VL_HOLD = 1
Low	Automatic/Manual mode
No effect	Motor protection fault, if MSS_OFF = 0
	Monitoring error, if FAULT_OFF = 0
	Control system error, operator error

Manual/auto

If the necessary enables are set, the mode is changed over either by the operator setting AUT_ON_OP on the OS, or by way of the interconnection at input AUT_L. The set operating mode is indicated at the output QMAN_AUT (1: Auto, 0: Manual).

- Manual mode: This operating mode permits control either at the OS or via interconnectable inputs.
  - OS operator control (LINK_MAN = 0): The input OPEN_VAL is used for opening, CLOS_VAL for closing or STOP_VAL for stopping. The functions required (OP_OP_EN, CL_OP_EN or ST_OP_EN) must be enabled.
  - Operation via interconnectable inputs (LINK_MAN = 1): In this case the commands are set at the inputs L_OPEN, L_CLOSE and L_STOP, You can interconnect these inputs to allow tracking or local control, for example. Note that you must set the switches LINK_MAN, LIOP_SEL and AUT_L by means of a suitable logic.
- Automatic mode: An interconnected automatic function block outputs its instructions to the inputs AUTO_ON (1: on, 0: off) or AUTO_OC (1: open, 0: close).

Interlock

The interlock function takes priority over all other control inputs and is only overridden by a motor protection fault or monitoring error if the relevant enable signals have been set (MSS_OFF = 1, FAULT_OFF = 1). If V_LOCK is set, the motor-driven valve is brought to its idle position, which is defined by SS_POS. It is opened or closed respectively by VL_OPEN or VL_CLOSE. VL_HOLD blocks the automatic and manual inputs and holds the last status request. The priorities of the individual interlock inputs are described in the “Operating Principle” section.

Monitoring

The monitoring logic (enabled with MONITOR = 1) verifies consistency between the setpoint status (determined in QSTART and QOC) and the process value feedback of the valve (provided by FB_OPEN and FB_CLOSE). If the setpoint status has not been reached after the monitoring time TIME_ON has expired, output QMON_ERR will be set. QMON_ERR is set immediately if the feedback changes without a reason (command).

If no feedback signals are connected, a MONITOR = 0 signal must be output to the monitoring function. The monitor assumes in this case that the setpoint state of the valve has been reached within the time TIME_ON.

If the monitoring function is working correctly, the QOPENING and QCLOSING outputs indicate whether the valve is opening or closing. QOPENED and QCLOSED indicate whether the valve has reached its final position.

If the valve is stopped at an intermediate position, the direction of movement is indicated with QOPENING = 0 or QCLOSING = 0, while 0 will indicate the final positions.

The FAULT_OFF parameter determines the significance of a monitoring error. If FAULT_OFF = 1, the motor is switched off when a fault has been detected and the valve holds its current position. This fault status has no effect on the control outputs when FAULT_OFF = 0, and the block thus behaves as if the monitoring function were switched off and indicates the monitoring error only at the QMON_ERR output.

The TIME_OFF parameter determines the wait time until the motor can be switched on again. QSTART = FALSE if the valve has reached the end position. A motor restart in reverse direction is not performed if QSTART = TRUE unless the set period TIME_OFF has expired. You will find more information in “Reversing the direction of travel”.

Motor protection

A negative edge of the motor protection signal MSS sets the motor protection fault signal retentively and transfers this signal to output QMSS_ST. The MSS_OFF parameter determines only to indicate the error status (MSS_OFF = 0), or that the motor is switched off irrespective of all other inputs and system states (MSS_OFF = 1) and that the valve idles in its current position.

Bumpless changeover

In order to ensure bumpless changeover to manual mode in all operating modes, the manual values OPEN_VAL, CLOS_VAL and STOP_VAL are always tracked to the current values of QSTART and QDIR (exception: reversal of the rotary direction).

Reversing the Direction of Travel

If the direction of travel of the valve is reversed before it reaches its end position, the following steps are taken:

- The motor is stopped (QSTART = 0).
- The internal OFF monitor waits for the time period TIME_OFF to expire and then starts the motor in the reverse direction, provided the OFF monitor does not report an error.

Error handling

The motor protection error (QMSS_ST = 1) and the monitoring error (QMON_ERR = 1) are reported to the OS and influence how the block works as described above. The operator can either reset these states with RESET, or automatically at the next positive edge at MSS by interconnecting L_RESET with a "1" signal. The control system fault CSF is merely reported to the OS and applied to the group error QGR_ERR along with the motor protection and monitoring errors. It does not have any further influence on the block algorithm.

Operator errors are indicated at output QOP_ERR without a message.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Startup after error state

Startup after error state depends on the mode set at the time of the reset:

- In automatic mode, the motor valve cannot start up again unless the monitoring or motor protection error is reset and a corresponding start signal is output by the automation system.
- In manual mode the motor must be switched on explicitly, since manual operation had been tracked to "HOLD".

Startup characteristics

When the CPU starts, the VAL_MOT block is switched to manual mode and the HOLD command is output. This means that the block must be called in the startup OB. In CFC engineering, this is handled by the CFC. When using basic STEP 7 tools, you enter this call in the startup OB. After startup, the messages will be suppressed during the number of cycles set at RUNUPCYC.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SAMPLE_T parameter.

Assignment of the 32-bit status word VSTATUS

You will find more information in "VSTATUS for VAL_MOT (Page 271)".

Message response

The VAL_MOT block uses the ALARM_8P block to generate messages.

The following message triggers exist:

- Control system error
- Motor circuit-breaker fault and monitoring error (runtime error)
- the CSF signal that is received as a control system error via the interconnection

QMSG_SUP is set if the RUNUPCYC cycles have not expired since the restart or if MSG_STAT = 21.

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of VAL_MOT (Page 268)

Message texts and associated values of VAL_MOT (Page 270)

VSTATUS for VAL_MOT (Page 271)

Operating and monitoring VAL_MOT (Page 271)

3.30.2 I/Os of VAL_MOT

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
AUT_L	Interconnectable input for MANUAL/AUTO: 0:Manual/1:Auto)	BOOL	0	I		
AUT_ON_OP	Operator input: (0=Manual/1=Auto)	BOOL	0	IO	+	
AUTO_OC	AUTOMATIC mode rotary direction: 1 = open, 0 = close	BOOL	0	I		
AUTO_ON	AUTOMATIC value: 1: On	BOOL	0	I		
AUTOP_EN	1 = operator control enable for Auto	BOOL	1	I		
AUX_PRx	Associated value x	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: current batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	0	I	+	
CL_OP_EN	1 = operator-control enable for close valve	BOOL	1	I		
CLOS_VAL	Operator input: 1 = close valve	BOOL	0	IO	+	
CSF	1 = external error	BOOL	0	I		
FAULT_OFF	1 = Motor OFF in case of fault	BOOL	1	I		
FB_CLOSE	Feedback: 1 = closed	BOOL	0	I		
FB_OPEN	Feedback: 1 = open	BOOL	0	I		
L_CLOSE	AUTOMATIC value 1 = CLOSE valve	BOOL	0	I		
L_OPEN	AUTOMATIC value 1 = OPEN valve	BOOL	0	I		
L_RESET	Interconnectable RESET input	BOOL	0	I		
L_STOP	AUTOMATIC value 1 = STOP valve	BOOL	0	I		
LINK_MAN	0 = operator input active, 1 = manual control via L_OPEN, L_CLOSE, L_STOP	BOOL	0	I		
LIOP_SEL	Interconnectable input for manual/auto changeover (AUT_L): 1 = interconnection active, 0 = operator control active	BOOL	0	I		
MANOP_EN	1 = operator control enable for manual mode	BOOL	1	I		
MONITOR	1 = Monitoring ON	BOOL	1	I	+	
MSG_ACK	Messages acknowledged	WORD	0	O		
MSG_EVID	Message number	DWORD	0	I		
MSG_STAT	Message error information	WORD	0	O		
MSS	Motor protective circuit breaker (active low, i.e. 0 = error)	BOOL	0	I		
MSS_OFF	1 = In case of MSS fault: Motor stop	BOOL	1	I		
OCCUPIED	BATCH occupied ID	BOOL	0	I	+	
OOS	Reserve	BOOL	0	I	+	

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
OP_OP_EN	1 = Operator-control enable for OPEN valve	BOOL	1	I		
OPEN_VAL	Operator input: 1 = OPEN valve	BOOL	0	IO	+	
QAUTOP	1 = operator control enable for automatic mode	BOOL	0	O	+	
QC_FB_CLOSE	Quality code for FB_CLOSE	BYTE	16#80	I		
QC_FB_OPEN	Quality code for FB_OPEN	BYTE	16#80	I		
QC_QSTART	Quality code for QSTART	BYTE	16#80	O		
QC_QSTART_I	Quality code for output QSTART	BYTE	16#80	I		
QC_QOC	Quality Code for QOC	BYTE	16#80	O		
QC_QOC_I	Quality Code for Output QOC	BYTE	16#80	I		
QCL_OP	1 = operator-control enable for close valve	BOOL	0	0	+	
QCLOSED	1 = valve is closed	BOOL	0	O	+	
QCLOSING	1 = valve is closing	BOOL	0	O	+	
QERR	1 = error output (inverted ENO)	BOOL	1	O	+	
QGR_ERR	1 = group error	BOOL	0	O		
QMAN_AUT	0 = manual, 1 = auto	BOOL	0	O	+	
QMANOP	1 = operator-control enable for manual mode	BOOL	0	O	+	
QMON_ERR	1 = monitoring error	BOOL	0	O	+	
QMSG_ERR	1 = message error	BOOL	0	O	+	
QMSG_SUP	1 = message suppression active	BOOL	0	O	+	
QMSS_ST	Stored motor protective circuit-breaker (1 = error)	BOOL	0	O	+	
QOC	Direction control output: 1 = OPEN	BOOL	0	O		
QOP_ERR	1 = group operator input error	BOOL	0	O		
QOP_OP	1 = Operator-control enable for OPEN valve	BOOL	0	O	+	
QOPENED	1 = Valve is OPENED	BOOL	0	O	+	
QOPENING	1 = Valve is OPENING	BOOL	0	O	+	
QST_OP	1 = Operator-control enable for STOP valve	BOOL	0	O	+	
QSTART	Control Output: 1 = motor ON	BOOL	0	O	+	
RESET	Enabled input of error reset	BOOL	0	IO	+	
RUNUPCYC	Number of run-up cycles	INT	3	I		
SAMPLE_T	Sampling time in seconds	REAL	1.0	I		> 0
SS_POS	0 = idle position closed	BOOL	0	I		
ST_OP_EN	1 = Operator-control enable for STOP valve	BOOL	1	I		
STEP_NO	BATCH step number	DWORD	0	I	+	
STOP_VAL	Operator input: 1 = STOP valve	BOOL	0	IO	+	
TIME_OFF	Motor OFF monitoring time (in seconds)	REAL	3.0	I	+	≥ 0
TIME_ON	Valve-runtime monitoring time in seconds	REAL	3.0	I	+	≥ 0
V_LOCK	1 = Lock (SS_POS)	BOOL	0	I	+	
VL_CLOSE	1 = Lock (closed)	BOOL	0	I	+	
VL_HOLD	1 = Lock (hold/disabled)	BOOL	0	I	+	
VL_OPEN	1 = Lock (open)	BOOL	0	I	+	

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
USTATUS	Status word in VSTATUS, can be configured by user	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.30.3 Message Texts and Associated Values of VAL_MOT

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QMSS_ST	$$BlockComment$$ Motor protection	S	-
2	QMON_ERR	$$BlockComment$$ Monitoring Fault	S	-
3	CSF	$$BlockComment$$ External fault	S	-

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data and the remaining ones (AUX_PRx) can be freely assigned by the user.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	AUX_PR04
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.30.4 VSTATUS for VAL_MOT

The 32-bit statusword extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	QMON_ERR	-	QMSS_ST	-	QMAN_AUT	-	BA_EN	OCCUPIED
Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	QCLOSING	QOPENING	QCLOSED	QOPENED	V_LOCK

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.30.5 Operating and Monitoring of VAL_MOT

Additional information

You can find additional information in the following sections:

- VAL_MOT block icon (Page 592)
- VAL_MOT faceplate (Page 548)

3.31 VALVE: Valve control

3.31.1 Description of VALVE

Object name (type + number)

FB73

- VALVE block I/Os (Page 276)
- VALVE block icon (Page 593)
- VALVE faceplate (Page 550)

Function

Block VALVE is used to operate control valves (open/close fittings) by means of one control signal (open/close). The position of rest of the valve can be the closed or opened state. The two position feedback signals (open/closed) can be optionally monitored. The position feedback signals are generated by limit switches.

NOTICE
With the PCS 7 it is not intended that other blocks will be inserted between the plant block and the output driver. If you deviate from this principle, ensure that when interconnecting the block, that from the outputs of the plant block until the output driver, that all blocks that form the output signal are installed in the same OBs.

How it works

Various inputs are available for controlling the valve. They are implemented in a defined hierarchical relationship to each other and to the valve states. In particular the locking and feedback monitoring influence the control signal QCONTROL.

The allocation of priorities to the individual input variables and events with regard to their influence on the control signal is shown in the following table. The sections following provide further details.

Priority:	Event:
High	V_LOCK = 1
⇑	VL_CLOSE = 1
	VL_OPEN = 1
⇓	Monitoring error, if FAULT_OFF = 1
Low	Automatic/Manual mode
No effect	Monitoring error, if FAULT_OFF = 0
	Control system error, operator error

Position of Rest

The position of rest of the controlled valve is signaled at input SS_POS (1: open, 0: closed). This only affects the definition of the control output QCONTROL (0: rest position, valve terminated). The commands on input side remain unaffected (a "1" signal at the input always means "open").

Example: When SS_POS = 1 (valve in rest position "open"), control output QCONTROL = 1 signal actuates the "Close valve" command.

Manual/auto

If the necessary enables are set, the mode is changed over either by the operator setting AUT_ON_OP on the OS, or by way of the interconnection at input AUT_L. The set operating mode is indicated at the output QMAN_AUT (1: Auto, 0: Manual).

- **Manual mode**: Input MAN_OC is operated via the OS. The corresponding enable parameters (OP_OP_EN or CL_OP_EN) must be set.
- **Automatic mode**: The interconnection of the automation system outputs the control commands to input AUTO_OC (1: open, 0: close).

Interlock

The interlock function takes priority over all other control signals and errors. If V_LOCK is set, the valve is set to its rest position (QCONTROL = 0). If V_LOCK is not set, a locking state (open/closed) can also be selected directly via the inputs VL_OPEN and VL_CLOSE. The signal VL_CLOSE locks VL_OPEN.

Monitoring

The monitoring logic verifies consistency between the output control command QCONTROL and the process value feedback of the valve (FB_OPEN, FB_CLOSE). If the end position has not been reached after the monitoring time TIME_MON has expired, output QMON_ERR will be set. QMON_ERR is set immediately if the feedback changes without a reason (command). The valve is set to its rest position (de-energized).

If no limit feedback signal is connected, a MONITOR = 0 signal must be output to the monitoring function, which then assumes that the limit of the valve has been reached within the time TIME_MON. Until then, QOPENING or QCLOSING is displayed.

If the monitoring function is working correctly, the QOPENING and QCLOSING outputs indicate whether the valve is opening or closing. QOPENED and QCLOSED indicate whether the valve has reached its final position.

The inputs NO_FB_CL and NO_FB_OP are used to set that there will be no feedback of the "open" and "closed" states (NO_FB_xx = 1). Inputs NOMON_CL and NOMON_OP are used to deactivate monitoring of existing feedback (NOMON_xx = 1), due to failure of the limit switch, for example.

The parameter FAULT_SS defines the significance of the monitoring error. If FAULT_SS = 1, the motor is brought to the rest position defined in SS_POS in case of an error. The error has no effect on the control outputs if FAULT_SS = 0.

Bumpless changeover

In order to ensure a bumpless changeover to manual mode, the manual value MAN_OC is always tracked to the current value of QCONTROL.

Error handling

The monitoring error (QMON_ERR = 1) is reported to the OS and influences the block operating principle as described above. It can be reset by operating RESET, or automatically by means of an interconnection with the positive edge of L_RESET. The control system fault CSF is merely reported to the OS and applied to the group error QGR_ERR along with monitoring. It does not have any further influence on the block algorithm.

Operator errors are indicated at output QOP_ERR without a message.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Startup after error state

Startup after error state depends on the mode set at the time of the reset:

- In automatic mode, the motor valve cannot start up again unless the monitoring or motor protection error is reset and a corresponding start signal is output by the automation system.
- In manual mode the motor must be switched on explicitly, since manual operation had been tracked to "HOLD".

Startup characteristics

During a CPU startup, the VALVE block is switched to manual mode and the QCONTROL= 0 (rest position) signal is output. This means that the block must be called in the startup OB. In CFC engineering, this is handled by the CFC. When using basic STEP 7 tools, you enter this call in the startup OB. After startup, the messages will be suppressed during the number of cycles set at RUNUPCYC.

At the START_SS input parameter you can decide either to set the valve to safety state when the CPU is started (START_SS = 1) or to retain its last operating state.

Time response

The block must be called in a cyclic interrupt OB. The sampling time of the block is set in the SAMPLE_T parameter.

Assignment of the 32-bit status word VSTATUS

You can find additional information in "VSTATUS for VALVE (Page 279)".

Message response

The VALVE block uses the ALARM_8P block for generating messages.

Messages are triggered by the following control system errors:

- Monitoring error (runtime error)
- The CSF signal that is received as a control system error via the interconnection

QMSG_SUP is set if the RUNUPCYC cycles have not expired since the restart or if MSG_STAT = 21.

Monitoring process values

Not available

Additional information

You will find more information in:

I/Os of VALVE (Page 276)

Message texts and associated values of VALVE (Page 278)

VSTATUS for VALVE (Page 279)

Operating and monitoring VALVE (Page 279)

3.31.2 I/Os of VALVE

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
AUT_L	Interconnectable input for MANUAL/AUTO: 0 = manual; 1 = auto	BOOL	0	I		
AUT_ON_OP	Operator input: 0 = manual; 1 = auto	BOOL	0	IO	+	
AUTO_OC	Automatic mode direction of rotation: 1 = open, 0 = close	BOOL	0	I		
AUTOP_EN	1 = operator-control enable for auto mode	BOOL	1	I		
AUX_PRx	Associated value x	ANY	0	IO		
BA_EN	Release by BATCH	BOOL	0	I	+	
BA_ID	BATCH: Consecutive batch number	DWORD	0	I	+	
BA_NA	BATCH name	STRING[32]	0	I	+	
CL_OP_EN	1 = operator control enable CLOSE	BOOL	1	I		
CSF	1 = external error	BOOL	0	I		
FAULT_SS	1 = move to safe position in case of error	BOOL	1	I		
FB_CLOSE	Feedback: 1 = closed	BOOL	0	I		
FB_OPEN	Feedback: 1 = open	BOOL	0	I		
L_RESET	Interconnectable RESET input	BOOL	0	I		
LIOP_SEL	Interconnectable input for manual/auto changeover (AUT_L): 1 = interconnection active, 0 = operator control active	BOOL	0	I		
MAN_OC	Operator input: 0 = close, 1 = open	BOOL	0	IO	+	
MANOP_EN	1 = operator-control enable for manual mode	BOOL	1	I		
MONITOR	1 = monitoring ON, 0 = monitoring OFF	BOOL	1	I	+	
MSG_ACK	Messages acknowledged	WORD	0	O		
MSG_EVID	Message number	DWORD	0	I		
MSG_STAT	Message error information	WORD	0	O		
NO_FB_CL	1 = No feedback "closed" present	BOOL	0	I		
NO_FB_OP	1 = no "open" feedback present	BOOL	0	I		
NOMON_CL	1 = no monitoring for "closed" feedback	BOOL	0	I		
NOMON_OP	1 = no monitoring for "open" feedback	BOOL	0	I		
OCCUPIED	ID for occupied by BATCH	BOOL	0	I	+	
OOS	Reserve	BOOL	0	I	+	
OP_OP_EN	1 = operator control enable OPEN	BOOL	1	I		
QAUTOP	1 = operator control enable for AUTO	BOOL	0	O	+	
QC_FB_CLOSE	Quality code for FB_CLOSE	BYTE	16#80	I		
QC_FB_OPEN	Quality code for FB_OPEN	BYTE	16#80	I		

I/O (parameter)	Meaning	Data type	Default	Type	OCM	Permissible values
QC_QCONTROL	Quality code for QCONTROL	BYTE	16#80	O		
QC_QCONTROL_I	Quality code for output QCONTROL	BYTE	16#80	I		
QCL_OP	1 = operator control enable for close	BOOL	0	O	+	
QCLOSED	1 = valve is closed	BOOL	0	O	+	
QCLOSING	1 = valve is closing	BOOL	0	O	+	
QCONTROL	Control output: 0 = rest position	BOOL	0	O		
QERR	1 = error output (inverted ENO)	BOOL	1	O	+	
QGR_ERR	1 = group error	BOOL	0	O		
QMAN_AUT	0 = manual, 1 = auto	BOOL	0	O	+	
QMANOP	1 = operator-control enable for manual mode	BOOL	0	O	+	
QMON_ERR	1 = monitoring error	BOOL	0	O	+	
QMSG_ERR	1 = message error	BOOL	0	O	+	
QMSG_SUP	1 = message suppression active	BOOL	0	O	+	
QOP_ERR	1 = group operator-input error	BOOL	0	O		
QOP_OP	1 = Operator-control enable for OPEN	BOOL	0	O	+	
QOPENED	1 = valve is opened	BOOL	0	O	+	
QOPENING	1 = valve is opening	BOOL	0	O	+	
RESET	Enabled input of error reset	BOOL	0	IO	+	
RUNUPCYC	Number of run-up cycles	INT	3	I		
SAMPLE_T	Sampling time in seconds	REAL	1.0	I		> 0
SS_POS	Rest position: 0 = closed (type C): 1 = open (type O)	BOOL	0	I		
START_SS	1 = startup in safe state and manual mode	BOOL	1	I		
STEP_NO	BATCH step number	DWORD	0	I	+	
TIME_MON	Monitoring time in seconds	REAL	3.0	I	+	≥ 0
V_LOCK	1 = lock (SS_POS)	BOOL	0	I	+	
VL_CLOSE	1 = lock (closed)	BOOL	0	I	+	
VL_OPEN	1 = lock (open)	BOOL	0	I	+	
USTATUS	Status word in VSTATUS, can be configured by user	WORD	0	I		
VSTATUS	Extended status display in block icons	DWORD	0	O	+	

3.31.3 Message Texts and Associated Values of VALVE

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class	Can be suppressed by
1	QMON_ERR	$$BlockComment$$ Monitoring Fault	S	-
2	CSF	$$BlockComment$$ External fault	S	-

Assignment of the associated values to block parameters

The first three of the associated values of the message block are assigned SIMATIC BATCH data and the remaining ones (AUX_PRx) can be freely assigned by the user.

Associated value	Block parameters
1	BA_NA
2	STEP_NO
3	BA_ID
4	AUX_PR04
5	AUX_PR05
6	AUX_PR06
7	AUX_PR07
8	AUX_PR08
9	AUX_PR09
10	AUX_PR10

3.31.4 VSTATUS for VALVE

The 32-bit statusword extends the status display in the block icons and faceplates. The 16 least significant bits (bits 0 - 15) are used by the block as follows:

Bit no.:	7	6	5	4	3	2	1	0
Parameters	QMON_ERR	-			QMAN_AUT	-	BA_EN	OCCUPIED

Bit no.:	15	14	13	12	11	10	9	8
Parameters	OOS	QMSG_SUP	-	QCLOSING	QOPENING	QCLOSED	QOPENED	V_LOCK

The 16-bit input USTATUS (data type WORD) uses the most significant bits 16 to 31. You can use these bits as you wish.

3.31.5 Operating and Monitoring of VALVE

Additional information

You can find additional information in the following sections:

- VALVE block icon (Page 593)
- VALVE faceplate (Page 550)

Family: DRIVER

4

4.1 Notes on Using Driver Blocks

General

- The descriptions of the driver 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 driver blocks of the PCS 7 libraries, you require a 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.

Note

Note that some analog output modules do not support "Response to CPU stop". In this case, when the module driver is used, input START_ON is set to "0" and interconnections to this parameter are deleted.

Signal-processing blocks

The driver blocks of the available PCS 7 library offer three types of channel blocks for signal processing:

1. **Standard channel blocks:**

   CH_AI, CH_AO, CH_DI, CH_DO

   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. **Universal channel blocks:**

   CH_U_AI, CH_U_AO, CH_U_DI, CH_U_DO

   These blocks are used for processing the signals of S7-300/400 SM modules or PA field devices. The advantage of these blocks is that you can create CFC charts irrespective of the hardware I/O to be used later. A disadvantage of universal blocks is that they make increased demands on memory and runtime. The blocks do not have message response.

3. **PA channel blocks:**

   PA_AI, PA_AO, PA_DI, PA_DO, PA_TOT

   Designed especially for use with PA 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 CH blocks, the PA channel blocks do not only process the actual signal but also all variables, according to the desired device configuration selected in the hardware configuration.

4. **FF channel blocks:**

   FF_A_AI, FF_A_AO, FF_A_DI, FF_A_DO

   Designed especially for use with FF field devices and the AB7000 PROFIBUS slave. The blocks are used similar to the corresponding PA channel blocks and are similar in terms of their functionality.

5. **Special channel blocks:**

   CH_CNT, CH_CNT1, CH_MS

   These blocks are required for special applications, such as controlling and reading the count or frequency values of FM 350-1/-2 modules and 8-DI NAMUR modules of the ET 200iSP, as well as for processing the signals of ET 200S motor-starter modules.

4.2 CH_AI: Analog value input

4.2.1 Description of CH_AI

Object name (type + number)

FC275

- CH_AI block I/Os (Page 288)

Area of application

Block CH_AI is used for processing analog input value signals from S7-300/400 SM analog input modules.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32) and the restart OB100.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The MODE input is interconnected with the corresponding OMODE_xx output of the MOD block.

Function and operating principle

The cyclic block CH_AI processes all channel-specific signal functions of an analog input module.

The block 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. The MODE (Page 599) input is used to define the format of the raw value for processing. If the high byte at MODE input parameter = 16#40 (value status = higher-level error, QMOD_ERR = TRUE) the raw value is considered invalid.

The program generates a quality code of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Last valid value	16#44
Substitute value	16#48
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

Addressing

Connect the symbol generated in HW Config (symbol table) for the analog input channel with the VALUE input parameter.

Raw value check

Depending on the measurement type and range of the analog input module, the nominal range sets the range for converting analog signals into digital values (raw values). This includes an overshoot/undershoot range within which an analog signal can still be converted. Values outside this range constitute an overflow or underflow. The block indicates whether the raw value lies inside the nominal range of the module.

Output parameter QCHF_LL = TRUE if the value is outside the nominal low range. Output parameter QCHF_HL = TRUE if the value is outside the nominal high range. QCHF_LL/QCHF_HL remain set to TRUE if a channel error occurs due to the module diagnosis "undershoot/overshoot of the measuring range".

QBAD = TRUE (channel error) is also set when a signal overflow or underflow error occurs.

Note

The reaction of the modules to a wire break in the 4 mA to 20 mA signal line is not uniform. Depending on the module either 16#7FFF (overflow) or 16#8000 (underflow) is written to the process image as a raw value. The CH_AI channel block outputs an overflow (QCHF_HL = TRUE) or an underflow (QCHF_LL = TRUE) as appropriate along with QBAD = TRUE.

Exception: If you enabled "Diagnostic interrupt" at the analog input module in HW Config, only QBAD = TRUE will be set if a diagnostic interrupt is triggered when a “Channel error” is detected (for example, a wire break).

NAMUR limit check

The NAMUR guidelines for analog signal processing define the following limits for life zero (4 to 20 mA) analog signals that have a channel error:

3.6 mA ≤ analog signal ≤ 21 mA

The above NAMUR limits are set as fixed defaults for limit value monitoring. You can define other limits by setting input parameter CH_F_ON = TRUE, and by setting corresponding new limits in [mA] at the CH_F_HL and CH_F_LL input parameters. QBAD = TRUE if a life zero analog signal is outside the current high or low limit range.

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.

Normal value

The raw value is converted to its physical value based on the settings at input parameters VLRANGE, VHRANGE, and MODE (you will find more information in "OMODE_xx (Page 606)"). These values will be written to the outputs OVLRANGE and OVHRANGE to allow the interconnection of the settings of VLRANGE and VHRANGE to other block I/Os. The conversion algorithm depends on a linearized input signal. If VLRANGE = 0 and VHRANGE = 100, you obtain a percentage value. If VHRANGE = VLRANGE is set, you obtain the analog input module's input signal (e.g. mA) according to the MODE (Page 599) setting. If the raw value is already a physical value, set VLRANGE = 0 and VHRANGE = 1. The Quality Code is set to QUALITY = 16#80.

When operating in PTC measurement mode, the analog value contains an encoded binary signal. The output provides the following information:

- If the measured resistance is within the normal range, `PV_Out = 0.0`.
- If the measured resistance is in the prewarning range, `PV_Out = 4.0`.
- If the measured resistance is in the operating range, `PV_Out = 1.0`.

This only applies when you set the input parameters VLRANGE = 0 and VHRANGE = 1. You should only set 0 or 1 for the simulation and substitute values SIM_V and SUBS_V.

Note

In the measuring mode "External or internal comparison of thermocouple values", the physical unit is adapted to the +/- 80 mV range in S7 300 modules. You have to determine the temperature by means of the corresponding conversion tables.

The physical equivalent in [mV] is returned by the module as raw value. Set VHRANGE and VLRANGE to +/- 80 mV.

Simulation

If the input parameter SIM_ON = TRUE, the value of the SIM_V input parameter is output with quality code QUALITY = 16#60. QBAD = TRUE: reset due to a higher priority error. A valid operating mode also has to be set in the low word of the MODE (Page 599) input in simulation mode. Otherwise, QBAD = 1 is output. Simulation takes highest priority. The simulation value is converted into a raw value, based on the operating mode and the input parameters VHRANGE and VLRANGE. This value is verified in the same way as a raw value from the process image. This allows simulation of the QBAD, QCHF_LL and QCHF_HL states.

If a QBAD is to be formed in the negative range with a unipolar measuring range, the value must be set to -119%.

If VLRANGE > VHRANGE, the status QBAD = TRUE can not be simulated. The QCHF_LL and QCHF_HL outputs are set according to the value of SIM_V. If the block is in the simulation state, QSIM = TRUE.

Note

Remember that the simulation value is always output in simulation mode regardless of whether one of the parameters LAST_ON (substitute value) or SUBS_ON (last valid value).

Substitute value

The program outputs the value of input parameter SUBS_V if input parameter SUBS_ON = TRUE, LAST_ON = FALSE and the raw value is invalid. The quality code will be set to QUALITY = 16#48 and QBAD = 1.

Hold last value, V_DELTA parameter

The program outputs the last or next to last valid output value (V_LAST, VLAST1) when input parameter LAST_ON = TRUE and SUBS_ON = FALSE, depending on the setting at the V_DELTA parameter. You can define a valid process value change at the V_DELTA parameter. The function is disabled by setting V_DELTA ≤ 0. The following conditions apply:

Rules for invalid raw values and V_DELTA > 0:

- ABS (V - V_LAST) > V_DELTA: V = V_LAST1 (next to last valid output value), DELTA_ON = 1
- ABS (V - V_LAST) ≤ V_DELTA: V = V_LAST (last valid output value), DELTA_ON = 0
- The quality code is set to QUALITY = 16#44, and QBAD = 1

Rule for valid raw values and V_DELTA > 0:

V_DELTA is also used to limit changes to the valid raw value. If the change to the value between two calls is greater than V_DELTA the last value (V_LAST) will be retained at output V for the duration of one cycle. The V_DELTA value should be selected with due care. If the value is too low, the quality code may flutter between16#80 and 16#44, regardless whether or not the raw value is OK.

- ABS (V - V_LAST) > V_DELTA: For the duration of one cycle V = V_LAST
- The quality code is set to QUALITY = 16#44, DELTA_ON=1 and QBAD = 0.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Delayed Value Acceptance

After a restart or if the quality code is changed from "BAD" to "GOOD", the quality code and value are not updated unless the number of cycles specified in CNT_LIM has expired. When CNT_LIM = 0 (default setting), this function is disabled. During this delay time, the Quality Code = 16#00 and QBAD = 1 and the last value will be retained.

Error handling

The plausibility of input parameters is not checked. If an invalid mode is set in the low word of the MODE (Page 599) input, it is assumed that the raw value is invalid.

Startup characteristics

The accept value delay is started when CNT_LIM is # 0.

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

See also

MODE settings for SM modules

4.2.2 I/Os of CH_AI

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
CH_F_HL	Overshoot high limit of the input value (mA)	REAL	0	IO
CH_F_LL	Undershoot low limit of the input value (mA)	REAL	0	IO
CH_F_ON	1 = activate limit monitoring	BOOL	0	IO
CNT_LIM	Limits of the startup counter	INT	0	IO
CNT_RES	Startup counter	INT	0	IO
DELTA_ON	Last delta process value exceeded	BOOL	0	IO
LAST_BAD	Last QBAD	BOOL	0	IO
LAST_ON	1 = last valid value: Enable injection	BOOL	0	IO
MODE	Value status and mode	DWORD	0	IO
OVHRANGE	High limit of the process value (copy)	REAL	0	O
OVLRANGE	Low limit of the process value (copy)	REAL	0	O
QBAD	1 = invalid process value	BOOL	0	O
QCHF_HL	1 = overshoot of process value	BOOL	0	O
QCHF_LL	1 = undershoot of process value	BOOL	0	O
QLAST	1 = last valid value: Injection active	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QSUBS	1 = substitution active	BOOL	0	O
QUALITY	Process-value status	BYTE	0	O
SIM_ON	1 = activate simulation	BOOL	0	IO
SIM_V	Simulation value	REAL	0	IO
SUBS_ON	1 = enable substitution	BOOL	0	IO
SUBS_V	Substitute value	REAL	0	IO
V	Process value	REAL	0	O
VALUE	Input value	WORD	0	IO
VHRANGE	High limit of the process value	REAL	100	IO
VLRANGE	Low limit of the process value	REAL	0	IO
V_DELTA	Delta (V - V_LAST) of the process value	REAL	0	IO
V_LAST	Last valid process value	REAL	0	IO
V_LAST1	Second to last valid process value	REAL	0	IO

4.3 CH_AO: Analog-value output

4.3.1 Description of CH_AO

Object name (type + number)

FC 276

- CH_AO block I/Os (Page 292)

Area of application

Block CH_AO is used for processing analog-output-value signals from S7-300/400 SM analog output modules.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32) and the restart OB 100.

NOTICE
With PCS 7 it is not intended that other blocks will be inserted between the plant block and the output driver. If you deviate from this principle, ensure when interconnecting the block that from the outputs of the plant block until the output driver all blocks that form the output signal are installed in the same OBs.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The MODE input is interconnected with the corresponding OMODE_xx output of the MOD block.
- The CH_AO block is installed downstream of the MOD block assigned to it in OB 100.

Note

If you do not use the CFC function "Generate module drivers", you have to ensure that the CH_AO block is installed downstream of the MOD block assigned to it in OB 100.

Function and operating principle

Block CH_AO processes all channel-specific signal functions cyclically.

The block writes the process value as an analog raw value to a process image (partition). The MODE (Page 599) input parameter determines the form in which the raw value is to be generated.

If the high byte of input parameter MODE = 0 (value status), the raw value is still written to the process image (partition), but with the quality code "invalid value".

The quality code may assume the following states:

State	Quality code
Valid value	16#80
High value limited	16#56
Low value limited	16#55
Simulation	16#60
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error, or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

Addressing

The symbol (symbol table) generated in HW Config for the analog output channel must be interconnected with the VALUE output parameter.

Normal value

- Parameters ULRANGE and UHRANGE map the process value U to the raw VALUE (quantization steps) of the analog output module according to the MODE (Page 599). Example: In mode 4 mA to 20 mA (16#0203), the raw value for 4 mA is output if U = ULRANGE and the raw value for 20 mA is output if U = UHRANGE.
- The block switches the parameters UHRANGE and ULRANGE through to the outputs OVHRANGE and OVLRANGE. You can interconnect the outputs with the manipulated-variable limits NM_LMNHR and NM_LMNLR of the controller CTRL_PID, for example.
- PHYS_LIM can be used to set the limits for the raw VALUE. The default setting of PHYS_LIM = 0 limits the value at output VALUE to the default limits of the module. According to the example above, the block calculates the raw value for 20 mA if U > UHRANGE and the raw value for 4 mA if U < ULRANGE. As a result, the quality codes 16#56 (high value limited) and 16#55 (low value limited) are applied at the QUALITY output instead of 16#80 (valid value).

- If you want to output analog values outside the default limits up to the physical limits of the module, set PHYS_LIM = 1. The output values are limited only if, taking the example above, you exceed the module limit values by specifying U = 200 (36 mA) or U = -50 (-4 mA) when ULRANGE = 0 and UHRANGE = 100. The output values are then limited to the physical limits specified in the data sheets of the modules and the corresponding quality codes are output.
- The outputs QCHF_HL and QCHF_LL also provide information on whether output value limits have been set.

Simulation

If the input parameter SIM_ON = TRUE is set, the value of SIM_U is output with quality code (QUALITY) = 16#60. QBAD = TRUE is reset. Simulation takes highest priority. If the block is in the simulation state, QSIM = TRUE.

I/O fault

If the high byte of the MODE (Page 599) input parameter is set to 0 (value status), the quality code QUALITY = 16#00 is set. The actual raw value is always written to the process image (partition).

Value limiting

You can limit very low or very high process values that would lead to an error (QBAD = TRUE) before they were entered in the process image (partition).

If the LIMIT_ON switch = TRUE, the process values (U) are limited as follows:

- To V_HL, if U > V_HL and
- To LL_V, if U < V_LL.

Error handling

The plausibility of input parameters is not checked. If an invalid mode has been set in the low word of the MODE (Page 599) input, the digitized output value will be set to 0 and QUALITY = 16#00 is output.

Startup characteristics

The MOD blocks set the LSB in byte 2 of their OMODE_xx (Page 606) output parameters in OB 100. If the block detects this code, it responds with an acknowledgement and:

If START_ON is not set, computes the process value U and writes the result to the process image. If START_ON is set, the raw value corresponding to the START_U process value is written to the process image.

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.3.2 I/Os of CH_AO

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
LIMIT_ON	1 = enable limiting of the process value	BOOL	0	IO
LL_V	Process value, if U < V_LL	REAL	0	IO
MODE	Value status and mode	DWORD	0	IO
OVHRANGE	Output high limit of the process value	REAL	100	O
OVLRANGE	Output low limit of the process value	REAL	0	O
PHYS_LIM	1 = enable physical limits of the module	BOOL	0	IO
QBAD	1 = invalid output value	BOOL	0	O
QCHF_HL	1 = overshoot of process value	BOOL	0	O
QCHF_LL	1 = undershoot of process value	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QUALITY	Value status of the output value	BYTE	0	O
SIM_ON	1 = activate simulation	BOOL	0	IO
SIM_U	Simulation value	REAL	0	IO
START_ON	1 = substitution at startup	BOOL	0	IO
START_U	Substitute value at startup	REAL	0	IO
U	Process value	REAL	0	IO
UHRANGE	High limit of the process value	REAL	0	IO
ULRANGE	Low limit of the process value	REAL	0	IO
V_HL	High limit	REAL	0	IO
V_LL	Low limit	REAL	0	IO
VALUE	PI output value	WORD	0	O

4.4 CH_CNT: Controlling and Reading FM 350 Modules

4.4.1 Description of CH_CNT

Object name (type + number)

FB 127

- CH_CNT block I/Os (Page 297)

Area of application

The block CH_CNT is used for controlling and reading count or measured values of an FM 350-1 or FM 350-2 module.

Calling OBs

Cyclic OB (recommendation 100 ms) in which the data will be received and sent.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed 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.

Function and operating principle

The communication interfaces of the FM 350-1 and FM 350-2 differ as follows:

- 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 parameters LOAD_VAL and CMP_VALx 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.

"FM 350" refers to the FM 350-1 and FM 350-2 modules in the following.

If an FM 350-2 module is being used, the block writes the LOAD_VALx (load count value immediately), PREP_VALx (load count value in preparation), or CMP_VALx (comparison value) parameters to the module (x = channel number) via data records. If the parameter LOAD_DIR = TRUE is set in the block then it writes LOAD_VALx. If LOAD_PRE = TRUE is set, it writes PREP_VALx. The CMP_VALx 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, QMOD_ERR = TRUE), the count or measured value is treated as invalid.

Note

Statuses QCOMP1 (comparator 1), QCOMP2 (comparator 2), QZERO (zero crossing), QOFLW (overflow), and QUFLW (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.

Quality code

Quality code describes the signal states of the MODE input of the CH_CNT block.

The program generates a quality code of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Last valid value	16#44
Substitute value	16#48
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error, or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

The quality code is saved in bytes 2 and 3 of the MODE parameter.

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. 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).
  - Connect the input LATCH in the CFC chart with the previously created icon via "Interconnect to operand...".

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 USE_CNT or USE_MSRV are set to TRUE. Both inputs should be set to FALSE 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 USE_CNT or USE_MSRV inputs are not set, a read-out can be executed via data records if the USE_CNT or USE_MSRV inputs are set at a different instance of CH_CNT (other channel) of the associated FM 350-2.

Simulation

If the input parameter SIM_ON = TRUE is set, the value of the input parameter SIM_V is output with quality code QUALITY = 16#60 and QBAD = FALSE is set. Simulation takes highest priority. If the block is in the simulation state, QSIM = TRUE.

Substitute value

If input parameter SUBS_ON = TRUE, then the value of input parameter SUBS_V is output as a substitute value if the count or measured value is invalid. The quality code will be set to QUALITY = 16#48 and QBAD = 1.

Hold last value

If input parameter LAST_ON = TRUE and the count or measured value is invalid, the last valid output value is output. The quality code will be set to QUALITY = 16#44 and QBAD = 1.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Redundancy

In H systems, the higher-level MOD_D1 block evaluates redundancy of the DP master systems.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

During startup and restart, the parameters CMP_VAL0 and CMP_VAL1 (FM 350-1 only) (comparison value; CMP_VAL1, 2, 3 too for dosing (FM 350-2)) are sent to the FM 350 by the FM_CNT block.

Overload behavior

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.4.2 I/Os of CH_CNT

The factory setting of the block display in CFC is identified in the "I/O" column:
I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
ACT_CNTV	Current load or LATCH value/current measured value	DINT	0	O
ACT_MSRV	Current measured value	DINT	0	O
CHANNEL	Channel FM 350	INT	0	I
CMP_V0	New comparison value 1/high limit	DINT	0	I
CMP_V1	New comparison value 2/updating time	DINT	0	I
CMP_V2	New comparison value 3 (dosing mode)	DINT	0	I
CMP_V3	New comparison value 4 (dosing mode)	DINT	0	I
CTRL_DO0	1 = enable digital output DO	BOOL	0	I
CTRL_DO1	1 = enable digital output DO1 (FM 350-1 or FM 350-2 only, dosing mode)	BOOL	0	I
CTRL_DO2	1 = enable digital output DO2 (FM 350-2 only, dosing mode)	BOOL	0	I
CTRL_DO3	1 = enable digital output DO3 (FM 350-2 only, dosing mode)	BOOL	0	I
ENSET_DN	1 = enable for setting in backward direction	BOOL	1	I
ENSET_UP	1 = enable for setting in forward direction	BOOL	1	I
FM_DATA	Structure FM 350 data	STRUCT		IO
GATE_STP	1 = general GATE stop	BOOL	0	I
LADDR	Logical address FM 350	INT	0	I
LAST_ON	1 = last valid value: Enable injection	BOOL	0	IO
LATCH	Current count value	ANY	0	I
LOAD_DIR	1 = load immediately	BOOL	0	IO
LOAD_PRE	1 = load in preparation	BOOL	0	IO
LOAD_VAL	New load value/low limit	DINT	0	I
MODE	Channel mode	DWORD	0	I
QBAD	1 = invalid values	BOOL	0	O
QCMP1	1 = comparison value 1	BOOL	0	O
QCMP2	1 = comparison value 2	BOOL	0	O
QCMP3	1 = comparison value 3	BOOL	0	O
QCMP4	1 = comparison value 4	BOOL	0	O
QCOMP1	1 = saved status of comparator 1 (corresponds to STS_CMP of FM 350-2)	BOOL	0	O
QCOMP2	1 = saved status of comparator 2 (FM 350-1 or FM 350-2 only), dosing mode	BOOL	0	O
QCOMP3	1 = saved status of comparator 2 (FM 350-2 only), dosing mode	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QCOMP4	1 = saved status of comparator 2 (FM 350-2 only), dosing mode	BOOL	0	O
QDIR	1 = status counter count direction	BOOL	0	O
QGATE	1 = status internal GATE	BOOL	0	O
QLAST	1 = last valid value: Injection active	BOOL	0	O
QLATCH	1 = new LATCH value (in clock-synchronous mode only)	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QOFLW	1 = status overflow	BOOL	0	O
QOP_ERR	1 = operator error	BOOL	0	O
QRUN	1 = status counter working	BOOL	0	O
QSET	1 = status digital input DI set	BOOL	0	O
QSIM	1 = simulation values	BOOL	0	O
QSTA	1 = digital input DI start	BOOL	0	O
QSTP	1 = status digital input DI stop	BOOL	0	O
QSUBS	1 = error substitute values active	BOOL	0	O
QSW_G	1 = status SW GATE	BOOL	0	O
QSYNC	1 = status counter synchronized	BOOL	0	O
QUALITY	Process-value status	BYTE	0	O
QUFLW	1 = status underflow	BOOL	0	O
QZERO	1 = status zero crossing	BOOL	0	O
R_OP_ERR	1 = reset operator error	BOOL	0	IO
RES_SYNC	1 = reset synchronization	BOOL	0	IO
SET_DO0	1 = open DO0	BOOL	0	I
SET_DO1	1 = open DO1 (FM 350-1 or FM 350-2 only, dosing mode)	BOOL	0	I
SET_DO2	1 = open DO2 (FM 350-2 only, dosing mode)	BOOL	0	I
SET_DO3	1 = open DO3 (FM 350-2 only, dosing mode)	BOOL	0	I
SIM_CNT	Simulation count value	DINT	0	I
SIM_MSRV	Simulation measured value	DINT	0	I
SIM_ON	1 = simulation active	BOOL	0	I
SUBS_CNT	Count substitute value	DINT	0	I
SUBS_MSRV	Measured substitute value	DINT	0	I
SUBS_ON	1 = substitute value active	BOOL	0	I
SW_GATE	1 = enable SW GATE	BOOL	0	I
USE_CNT	1 = count value used	BOOL	1	I
USE_MSRV	1 = measured value used	BOOL	1	I

4.5 CH_CNT1: Controlling and Reading an 8-DI-NAMUR Module of the ET 200iSP

4.5.1 Description of CH_CNT1

Object name (type + number)

FB 59

- CH_CNT1 block I/Os (Page 305)

Area of application

Block CH_CNT1 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

Calling OBs

Cyclic OB (recommendation 100 ms) in which the data will be received and sent.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- Inputs LADDR, LADDR1, and CHANNEL are configured.
- The MODE input is interconnected with the OMODEx output of the MOD_D1 block.

Function and operating principle

Depending on the mode setting of the module in HW Config, the user data of the module are stored in the process image. Block CH_CNT1 differentiates between the following modes:

MODE (low word)	Mode	HW Config setting: "Configuration"	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, QMOD_ERR = TRUE), 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
-	C1	Counter pulse counter 1
-	C2	Counter pulse counter 2
GATE_STP (CHANNEL = 0)	GATE1	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.
GATE_STP (CHANNEL = 1)	GATE2	See description of "GATE 1".
RES_CNT (CHANNEL = 0)	RSC1	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.
RES_CNT (CHANNEL = 1)	RSC2	See description of "RSC1".
RES_DO (CHANNEL = 0)	RSO1	On the rising edge of the RSO signal, the assigned output can be reset. The count is not influenced by setting RSO.
RES_DO (CHANNEL = 1)	RSO2	See description of "RSO2".

The in/out parameters RES_CNT and RES_DO 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:

Byte	Bit	Input signal	Block output	Meaning
0, 1	0-15	Proc. value counter 1	ACT_CNTV	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	A1	QZERO	Zero crossing counter 1
	1	A2		Zero crossing counter 2
	2	GATE 1	QGATE	Status gate 1
	3	GATE 2		Status gate 2
	4	RSC1	QRES_CNT	Status reset counter 1
	5	RSC2		Status reset counter 2
	6	RSO1	QRES_DO	Status reset outputs counter 1
	7	RSO2		Status reset outputs counter 2

The LOAD_VAL 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 LOAD_VAL are transferred to the module:

- 16-bit counter: 0 to 65,535
- 32-bit counter: 0 to 2,147,483,647

If the value of LOAD_VAL lies outside of these limits, the last valid value of LOAD_VAL is retained in the module and QOP_ERR = TRUE is set.

Quality code

The program generates a quality code of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Last valid value	16#44
Substitute value	16#48
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error, or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

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 GATE 1 to RSO2. In this configuration of the DI NAMUR module, the states of outputs QGATE, QZERO, QRES_CNT, and QRES_DO are based on the inputs of the block.

When using the digital control signals GATE 1 to RSO2 of the module (HW Config channel 2 to 7 = CONTROL), 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	
GATE 1 = 1	GATE_STP = 0	→	GATE on
GATE 1 = 0	GATE_STP = 1	→	GATE on
GATE 1 = 0	GATE_STP = 0	→	GATE off

Addressing

You must connect the icon (from the icon table) for the count or frequency value with the VALUE input parameter.

Simulation

If the input parameter SIM_ON = TRUE is set, the value of the input parameter SIM_V is output with quality code QUALITY = 16#60 and QBAD = FALSE is set. Simulation takes highest priority. If the block is in the simulation state, QSIM = TRUE is set.

Substitute value

If input parameter SUBS_ON = TRUE, then the value of input parameter SUBS_V is output as a substitute value if the count or measured value is invalid. The quality code will be set to QUALITY = 16#48 and QBAD = 1.

Hold last value

If input parameter LAST_ON = TRUE and the count or measured value is invalid, the last valid output value is output. The quality code will be set to QUALITY = 16#44 and QBAD = 1.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Redundancy

In H systems, the higher-level MOD_D1 block evaluates redundancy of the DP master systems.

Error handling

The plausibility of input parameters is not checked. Exception: LOAD_VAL is checked for valid input values. You will find more information in "Function and operating principle".

Startup characteristics

Not available

Overload behavior

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.5.2 I/Os of CH_CNT1

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
ACT_CNTV	Current count value/frequency value	DINT	0	O
CHANNEL	Channel 8 DI NAMUR	INT	0	I
GATE_STP	1 = GATE on (stop counting)	BOOL	0	I
LADDR	Logical address (inputs)	INT	0	I
LADDR1	Logical address (outputs)	INT	0	I
LAST_ON	1 = last valid value: Enable injection	BOOL	0	IO
LOAD_VAL	Counter load value	DINT	0	I
MODE	Channel mode	DWORD	0	I
QBAD	1 = invalid values	BOOL	0	O
QGATE	1 = GATE on	BOOL	0	O
QLAST	1 = last valid value: Injection active	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QOP_ERR	1 = operator error	BOOL	0	O
QSIM	1 = simulation values	BOOL	0	O
QRES_CNT	1 = count value reset	BOOL	0	O
QRES_DO	1 = digital outputs reset	BOOL	0	O
QSUBS	1 = error substitute values active	BOOL	0	O
QUALITY	Process-value status	BYTE	0	O
QZERO	1 = status zero crossing	BOOL	0	O
RES_CNT	1 = reset count value	BOOL	0	IO
RES_DO	1 = reset digital output	BOOL	0	IO
SIM_CNT	Simulation value	DINT	0	I
SIM_ON	1 = simulation active	BOOL	0	I
SUBS_CNT	Substitute value	DINT	0	I
SUBS_ON	1 = substitute value active	BOOL	0	I
VALUE	Icon count value	ANY	0	I

4.6 CH_CNT2C: Control and read the 1 COUNT 24V/100kHz module for count mode

4.6.1 Description of CH_CNT2C

Object name (type + number)

FB 242

- I/Os of CH_CNT2C (Page 310)

Area of application

The block is used to control and read count and latch values of the "1 COUNT 24V/100kHz" module (as of 6ES7 138-4DA04-0AB0) for the count mode.

Calling OBs

OB 100 and cyclic OB (recommendation 100 ms) in which the data will be received and sent.

Use in CFC

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The LADDR input is assigned a parameter value and
- The MODE input is interconnected with the OMODE_00 output of the MOD_D1 block.

Function and operating principle

The block communicates via the process image. The data are written and read continuously.

The LOAD_VAL parameter is transferred to the module when a positive edge is detected either at the in/out parameter L_PREPAR (= load counter in preparation) or L_DIRECT (= load counter directly). The CMP_VAL1 (comparison value 1) and CMP_VAL2 (comparison value 2) parameters 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)	Mode	Description
1	Continuous counting	In this mode, 1Count24V/100kHz counts endlessly starting at the loaded value
2	One-time counting	In this mode, 1Count24V/100kHz counts once; depending on the set main counting direction
3	Periodic counting	In this mode, 1Count24V/100kHz counts periodically; depending on the set main counting direction

The MODE input indicates how the count or latch value is available in the process image. If the high byte of the MODE input parameter = 16#40 (value status = higher-level error, QMOD_ERR = TRUE), the count and latch values are treated as invalid.

A quality code is generated for each ACT_LATCH and ACT_CNTV result value and this can have the following states:

Quality code

State	Quality code
Valid value	16#80
Simulation	16#60
Last valid value	16#44
Substitute value	16#48
Invalid value	16#00

Note:

The states QSYNC (synchronization), QCMP1 (comparator 1), QCMP2 (comparator 2), QOFLW (overflow), QUFLW (underflow) and QZERO (zero crossing) are acknowledged automatically by the block. They are active for at least one cycle.

Addressing

Connect the symbol (from the symbol table) for the count value with the LATCH input parameter.

Enter the symbol (symbol column) in the symbol table and add the ID base address of the module (for example ID512) in the the row in the Address column.

Simulation

If the input parameter SIM_ON = TRUE is set, the value of the input parameters SIM_CNTV and SIM_LATCH is output with quality code QUALITY = 16#60 and QBAD = FALSE is set. Simulation takes highest priority. If the block is in the simulation state, QSIM = TRUE is set.

Substitute value

When input parameter SUBS_ON = TRUE and the count value is invalid, the value of input parameters SUBS_CNTV and SUBS_LATCH will be output as a value. The quality code will be set to QUALITY = 16#48 and QBAD = 1.

Hold last value

If input parameter LAST_ON = TRUE and SUBS_ON = FALSE and the count value or measured value are invalid, the last valid output value is output. The quality code will be set to QUALITY = 16#44 and QBAD = 1.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Error handling

No plausibility checks are implemented as regards the input parameter. Module error (QERR_24V = TRUE, QERR_DO1 = TRUE), you will need to acknowledge via the EXTF_ACK parameter.

While QERR_24V = 1, the count and latch values are treated as invalid and QBAD = 1 is set, the quality code remains QUALITY valid and QBAD = 0 for QERR_DO1.

The parameter assignment error QERR_PARA is acknowledged by correct parameter assignment and the error in the QERR_LOAD load function is deleted by the subsequent correct operator input.

Startup characteristics

During startup, the CMP_VAL1 parameter (comparison value 1) and CMP_VAL2 (comparison value 2) are transferred to the module.

Overload behavior

Not available.

Time response

Not available.

Message response

Not available.

Operating and monitoring

The block does not have a faceplate.

4.6.2 I/Os of CH_CNT2C

The factory setting of the block display in CFC is identified in the I/O column:

I/O name **bold** = I/O visible, I/O name normal = I/O invisible.

For explanations and the meaning of the abbreviations, refer to "General Information About Block Description (Page 15)"

I/O (parameter)	Meaning	Data type	Default	Type
ACT_CNTV	Current count value	DINT	0	O
ACT_LATCH	Current counted value or stored counted value if the latch function is used at the digital input	DINT	0	O
CMP_VAL1	Comparison value 1	DINT	0	I
CMP_VAL2	Comparison value 2	DINT	0	I
CTRL_DO1	1=enable DO1	BOOL	0	I
CTRL_DO2	1=enable DO2	BOOL	0	I
CTRL_SYN	1=enable synchronization	BOOL	0	I
EXTF_ACK	1=error acknowledgment	BOOL	0	I
L_DIRECT	1=load counter directly	BOOL	0	I
L_PREPAR	1=load counter in preparation	BOOL	0	I
LADDR	Logic address of the module	INT	0	I
LAST_ON	1 = last valid value: Enable injection	BOOL	0	I
LATCH	Interconnected latch value	DWORD	0	I
LOAD_VAL	Load value direct, prepared	DINT	0	I
MODE	Mode	DWORD	0	I
QBAD	1 = invalid process value	BOOL	0	O
QCMP1	1 = status of comparator 1	BOOL	0	O
QCMP2	1 = status of comparator 2	BOOL	0	O
QCNT_DN	1 = status direction down	BOOL	0	O
QCNT_UP	1 = status direction up	BOOL	0	O
QDI	1 = status DI	BOOL	0	O
QDO1	1 = status DO1	BOOL	0	O
QDO2	1 = status DO2	BOOL	0	O
QERR_24V	1= short circuit sensor power supply	BOOL	0	O
QERR_DO1	1= short circuit / wire break / overtemperature	BOOL	0	O
QERR_LOAD	1= error in load function	BOOL	0	O
QERR_PARA	1= parameter assignment error	BOOL	0	O
QGATE	1 = status of internal gate	BOOL	0	O
QLAST	1 = last valid value: Injection active	BOOL	0	O
QMOD_ERR	1= higher-level error	BOOL	0	O
QOFLW	1 = high count value	BOOL	0	O
QSIM	1= simulation active	BOOL	0	O
QSUBS	1= substitute values active	BOOL	0	O
QSYNC	1= status counter synchronized	BOOL	0	O
QUALITY	Value status of the process value	BYTE	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QUFLW	1 = low count value	BOOL	0	O
QZERO	1= status zero crossing	BOOL	0	O
SET_DO1	1= open DO1	BOOL	0	I
SET_DO2	1= open DO2	BOOL	0	I
SIM_CNTV	Simulation count value	DINT	0	I
SIM_LATCH	Simulation latch value	DINT	0	I
SIM_ON	1= simulation on	BOOL	0	I
SUBS_CNTV	Count substitute value	DINT	0	I
SUBS_LATCH	Latch substitute value	DINT	0	I
SUBS_ON	1= substitute value on	BOOL	0	I
SW_GATE	1= enable software gate	BOOL	0	I

4.7 CH_CNT2M: Control and read the 1 COUNT 24V/100kHz module for measurement mode

4.7.1 Description of CH_CNT2M

Object name (type + number)

FB 243

- I/Os of CH_CNT2M (Page 316)

Area of application

The block is used to control and read count and measured values of the "1 COUNT 24V/100kHz" module (as of 6ES7 138-4DA04-0AB0) for the measurement mode.

Calling OBs

OB 100 and cyclic OB (recommendation 100 ms) in which the data will be received and sent.

Use in CFC

When the CFC function "Generate module drivers" is used, the following occurs automatically:

- The LADDR input is assigned a parameter value and
- The MODE input is interconnected with the OMODE_00 output of the MOD_D1 block.

Function and operating principle

The block communicates via the process image. The data are written and read continuously.

The UFLW (low limit) and OFLW (high limit) parameters 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)	Mode	Description
4	Frequency measurement	1Count24V/100kHz determines the frequency of pulse sequence at the input.
5	Speed measurement	1Count24V/100kHz determines the rotational 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 how the count or measured value is available in the process image. If the high byte of the MODE input parameter = 16#40 (value status = higher-level error, QMOD_ERR = TRUE), the count and measured values are treated as invalid.

Quality code

A quality code is generated for each ACT_MSRV and ACT_CNTV result value and this can have the following states:

State	Quality code
Valid value	16#80
Simulation	16#60
Last valid value	16#44
Substitute value	16#48
Invalid value	16#00

Note

The states QCMP1 (measurement completed), QOFLW (overflow) and QUFLW (underflow) are acknowledged automatically by the block. They are active for at least one cycle.

Addressing

Connect the symbol (from the symbol table) for the count value with the MSRV input parameter.

Enter the symbol (symbol column) in the symbol table and add the ID base address of the module (for example ID512) in the the row in the Address column.

Simulation

If the input parameter SIM_ON = TRUE is set, the value of the input parameters SIM_CNTV and SIM_MSRV is output with quality code QUALITY = 16#60 and QBAD = FALSE is set. Simulation takes highest priority. If the block is in the simulation state, QSIM = TRUE is set.

Substitute value

When input parameter SUBS_ON = TRUE and the count value is invalid, the value of input parameters SUBS_CNTV and SUBS_MSRV will be output as a value. The quality code will be set to QUALITY = 16#48 and QBAD = 1.

Hold last value

If input parameter LAST_ON = TRUE and SUBS_ON = FALSE and the count value or measured value are invalid, the last valid output value is output. The quality code will be set to QUALITY = 16#44 and QBAD = 1.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Error handling

No plausibility checks are implemented as regards the input parameter. Module error (QERR_24V = TRUE, QERR_DO1 = TRUE), you will need to acknowledge via the EXTF_ACK parameter.

While QERR_24V = 1, the count and measured values are treated as invalid and QBAD = 1 is set, the quality code remains QUALITY valid and QBAD = 0 for QERR_DO1.

The parameter assignment error QERR_PARA is acknowledged by correct parameter assignment and the error in the QERR_LOAD load function is deleted by the subsequent correct operator input.

Startup characteristics

During startup, the UFLW (low limit) and OFLW (high limit) parameters are transferred to the module.

Overload behavior

Not available.

Time response

Not available.

Message response

Not available.

Operating and monitoring

The block does not have a faceplate.

4.7.2 I/Os of CH_CNT2M

The factory setting of the block display in CFC is identified in the I/O column:

I/O name **bold** = I/O visible, I/O name normal = I/O invisible.

For explanations and the meaning of the abbreviations, refer to "General Information About Block Description (Page 15)"

I/O (parameter)	Meaning	Data type	Default	Type
ACT_CNTV	Current count value	DINT	0	O
ACT_MSRV	Current measured value	DINT	0	O
CTRL_DO1	1=enable DO1	BOOL	0	I
EXTF_ACK	1=error acknowledgment	BOOL	0	I
LADDR	Logic address of the module	INT	0	I
LAST_ON	1 = last valid value: Enable injection	BOOL	0	I
MODE	Mode	DWORD	0	I
MSRV	Interconnected measured value	DWORD	0	I
OFLW	High limit	DINT	0	I
QBAD	1 = invalid process value	BOOL	0	O
QCMP1	1 = measurement completed	BOOL	0	O
QCNT_DN	1 = status direction down	BOOL	0	O
QCNT_UP	1 = status direction up	BOOL	0	O
QDI	1 = status DI	BOOL	0	O
QDO1	1 = status DO1	BOOL	0	O
QERR_24V	1= short circuit sensor power supply	BOOL	0	O
QERR_DO1	1= short circuit / wire break / overtemperature	BOOL	0	O
QERR_LOAD	1= error in load function	BOOL	0	O
QERR_PARA	1= parameter assignment error	BOOL	0	O
QGATE	1 = status of internal gate	BOOL	0	O
QLAST	1 = last valid value: Injection active	BOOL	0	O
QMOD_ERR	1= higher-level error	BOOL	0	O
QOFLW	1 = High limit of measuring range	BOOL	0	O
QSIM	1= simulation active	BOOL	0	O
QSUBS	1= substitute values active	BOOL	0	O
QUALITY	Value status of the process value	BYTE	0	O
QUFLW	Low count limit	BOOL	0	O
SET_DO1	1= open DO1	BOOL	0	I
SIM_CNTV	Simulation count value	DINT	0	I
SIM_MSRV	Simulation measured value	DINT	0	I
SIM_ON	1= simulation on	BOOL	0	I
SUBS_CNTV	Count substitute value	DINT	0	I
SUBS_MSRV	Measured substitute value	DINT	0	I
SUBS_ON	1= substitute value on	BOOL	0	I
SW_GATE	1= enable software gate	BOOL	0	I
UFLW	Low limit	DINT	0	I

4.8 CH_DI: Digital value input

4.8.1 Description of CH_DI

Object name (type + number)

FC277

- CH_DI block I/Os (Page 320)

Area of application

Block CH_DI is used for signal processing of a digital input value of S7-300/400 SM digital input modules.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32).

Use in CFC

If the CFC function "**Generate module drivers**" is used, the MODE input is automatically interconnected with the corresponding OMODE_xx output of the MOD block.

Function and operating principle

Block CH_DI processes all channel-specific signal functions cyclically.

The block reads a digital value of the data type BOOL from the process image (partition). If the high byte of the MODE (Page 599) input parameter = 16#40 (value status = higher-level error, QMOD_ERR = TRUE), the digital value is treated as invalid. If input parameter PQC = TRUE, it reads the value status of the digital value from the process image (partition).

Quality code

The program generates a quality code of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Last valid value	16#44
Substitute value	16#48
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

Addressing

The symbol generated in HW Config (symbol table) for the digital input channel must be interconnected to the VALUE input. If the process image (partition) also contains the value status of the digital input channel, interconnect the corresponding symbol with input VALUE_QC and set input PQC = TRUE.

Normal value

The digital value of the process image (partition) and the quality code QUALITY = 16#80 are applied to output Q.

Simulation

If input parameter SIM_ON = TRUE, the value of input parameter SIM_I is output to the output parameter Q with quality code QUALITY = 16#60. QBAD = TRUE is reset. Simulation takes highest priority. If the block is in the simulation state, QSIM = TRUE is set.

Note

Remember that the simulation value is always output in simulation mode regardless of whether one of the parameters LAST_ON (substitute value) or SUBS_ON (last valid value).

Substitute value

When input parameter SUBS_ON = TRUE and the digital value of the process image (partition) is invalid, the function outputs the signal QBAD = 1 and the value at input parameter SUBS_I with quality code QUALITY = 16#48 to output parameter Q.

Hold last value

If input parameter LAST_ON = TRUE and the count value or measured value are invalid, the last valid output value is output. The quality code will be set to QUALITY = 16#44 and QBAD = 1.

Last valid output value = Q_LAST.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.8.2 I/Os of CH_DI

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
LAST_ON	1 = last valid value: Enable injection	BOOL	0	IO
MODE	Value status and mode	DWORD	0	IO
PQC	1 = use value status in the process image	BOOL	0	IO
Q	Process value	BOOL	0	O
QBAD	1 = invalid process value	BOOL	0	O
QLAST	1 = last valid value injection active	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QSUBS	1 = substitution active	BOOL	0	O
QUALITY	Process-value status	BYTE	0	O
SIM_I	Simulation value	BOOL	0	IO
SIM_ON	1 = activate simulation	BOOL	0	IO
SUBS_I	Substitute value	BOOL	0	IO
SUBS_ON	1 = enable substitution	BOOL	0	IO
VALUE	Input value	BOOL	0	IO
VALUE_QC	Value status in the process image	BOOL	0	IO

4.9 CH_DO: Digital value output

4.9.1 Description of CH_DO

Object name (type + number)

FC 278

- CH_DO block I/Os (Page 323)

Area of application

Block CH_DO processes the digital output signals of S7-300/400 SM digital output modules.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32) and the restart OB 100.

NOTICE
With PCS 7 it is not intended that other blocks will be inserted between the plant block and the output driver. If you deviate from this principle, ensure when interconnecting the block that from the outputs of the plant block until the output driver all blocks that form the output signal are installed in the same OBs.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The MODE input is interconnected with the corresponding OMODE_xx output of the MOD block.
- The CH_DO block is installed downstream of the MOD block assigned to it in OB 100.
- The START_ON input is configured with the corresponding value. The START_I input is only configured if START_ON = 1.

Function and operating principle

Block CH_DO processes all channel-specific signal functions cyclically.

The block writes a digital value to a process image (partition). If the high byte at the MODE settings for SM modules input parameter = 0 (value status), the digital value will still be written to the process image (partition), but an "invalid value" quality code will be set.

Quality code

The quality code may assume the following states:

State	Quality code
Valid value	16#80
Simulation	16#60
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

Addressing

The symbol generated with HW Config in the symbol table for the digital output channel must be interconnected with the VALUE output parameter.

Normal value

The digital value is written to the process image (partition) and quality code (QUALITY) = 16#80.

Simulation

When input parameter SIM_ON = TRUE, the value of input parameter SIM_I will be written to the process image (partition) and quality code QUALITY = 16#60 is set. QBAD = TRUE is reset. Simulation takes highest priority. If the block is in the simulation state, QSIM = TRUE.

I/O fault

If the high byte of the MODE (Page 599) input parameter = 0 (value status), the quality code QUALITY = 16#00 is set. The function always writes the current digital value to the process image (partition).

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

The MOD blocks set the LSB in byte 2 of their OMODE (Page 606) _xx output parameters in OB 100. If the block detects this code, it responds with an acknowledgement and:

If START_ON is not set, it writes the process value I to the process image; otherwise it substitutes this process value with START_I.

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.9.2 I/Os of CH_DO

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
I	Process value	BOOL	0	IO
MODE	Value status and mode	DWORD	0	IO
QBAD	1 = invalid output value	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QUALITY	Value status of the output value	BYTE	0	O
SIM_I	Simulation value	BOOL	0	IO
SIM_ON	1 = activate simulation	BOOL	0	IO
START_I	Substitute value at startup	BOOL	0	IO
START_ON	1 = substitution at startup	BOOL	0	IO
VALUE	PI output value	BOOL	0	O

4.10 CH_MS: Signal processing of the ET 200S motor starter module

4.10.1 Description of CH_MS

Object name (type + number)

FB 60

- CH_MS block I/Os (Page 328)

Area of application

Block CH_MS is used for processing the signals of an ET 200S motor-starter module.

Calling OBs

Cyclic OB (recommendation 100 ms) in which the data will be received and sent.

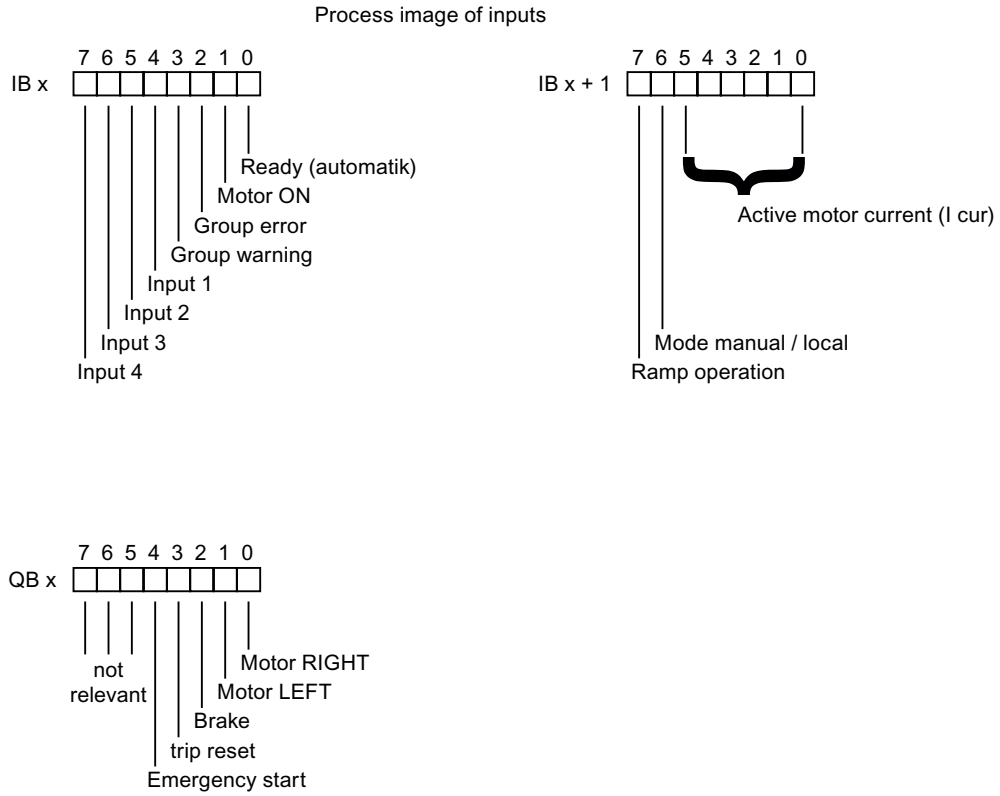
Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The CH_MS block is installed downstream of the MOD block assigned to it in OB 100.
- The MODE input is interconnected with the corresponding OMODE_xx output of the MOD block.
- Inputs IN_x and outputs OUT_x are interconnected with the motor-starter-module icons.

Function and operating principle

A motor-starter module occupies the process image as follows:



The inputs shown in the schematic are acquired from the process image and applied individually to the output.

The 6-bit value supplied by the motor-starter module specifies the motor-current ratio I cur/I rated (I rated = rated operating current set in HW Config). The value is specified with one place before the decimal point (bit 5) and five places after the decimal point (bit 4 to bit 0). The maximum ratio for I cur/I rated is, therefore, 1.96875 (approx. 197%).

I ratio = I rated x value (bit 5 to bit 0)

Bit 5	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	
1	**0,5**	**0,25**	**0,125**	**0,0625**	**0,03125**	**Total = 1.96875**
0	0	0	0	0	0	I ratio = 0
1	0	0	0	0	0	I ratio = 1
1	0	1	1	0	0	I ratio = 1.375
1	1	1	1	1	1	I ratio = 1.96875

The bits for the motor-current ratio are grouped and output as a REAL value.

If the high byte of the OMODE (Page 606) input parameter = 16#40xxxxxx (value status = higher-level error, QMOD_ERR = TRUE), the digital values are treated as invalid.

Quality code

The program generates a quality code of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Last valid value	16#44
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error, or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

The quality code is saved in bytes 2 and 3 of the MODE parameter.

Addressing

You interconnect the first symbol of those generated by HW Config (symbol table) for the inputs of the motor-starter module with the VALUE input.

Simulation

If the input parameter SIM_ON = TRUE, the value of input parameter SIM_U (encoded like the structure of the two bytes of the process input image) is output with quality code QUALITY = 16#60. QBAD = TRUE is reset. Simulation takes highest priority. If the block is in the simulation state, QSIM = TRUE is set.

Substitute value

No substitute value can be set.

Hold last value

If input parameter LAST_ON = TRUE, the last valid output value is output if the digital signals are invalid. The quality code will be set to QUALITY = 16#44 and QBAD = 1.

Output invalid value

If the input parameter LAST_ON = FALSE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Redundancy

In H systems, the higher-level block evaluates redundancy of the DP master systems.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

During startup and the initial run, the current process values of the inputs are written to the process image.

Overload behavior

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

You will find more information in:

Notes on using driver blocks (Page 281)

See also

MODE settings for SM modules (Page 599)

4.10.2 I/Os of CH_MS

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
BRAKE	Brake	BOOL	0	I
EMCY_ST	Emergency start	BOOL	0	I
FORWARD	Motor forward	BOOL	0	I
IN_x	Input value (x = 0 to 15)	BOOL	0	O
IN_NO	Number of bits of the inputs of the motor-starter module	BYTE	0	I
LAST_ON	1 = last valid value: Enable injection	BOOL	0	IO
MODE	Value status and mode	DWORD	0	I
OUT_x	Output value (x = 0 to 15)	BOOL	0	O
OUT_NO	Number of bits of the outputs of the motor-starter module	BYTE	0	I
QAUTO	Ready (automatic)	BOOL	0	O
QBAD	1 = invalid process value	BOOL	0	O
QBRAKE	Brake	BOOL	0	O
QEMCY_ST	Emergency start	BOOL	0	O
QERROR	Group error	BOOL	0	O
QFORWARD	Motor forward	BOOL	0	O
QINPUT_1	Input 1	BOOL	0	O
QINPUT_2	Input 2	BOOL	0	O
QINPUT_3	Input 3	BOOL	0	O
QINPUT_4	Input 4	BOOL	0	O
QLAST	1 = last valid value: Injection active	BOOL	0	O
QMANUAL	Manual/local	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QRAMP	Ramp operation	BOOL	0	O
QRESET	Trip reset	BOOL	0	O
QRES_x	Reserve (x = 1 to 11)	BOOL	0	O
QREVERS	Motor reverse	BOOL	0	O
QRUN	Motor on	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QUALITY	Process-value status	BYTE	0	O
QWARN	Group warning	BOOL	0	O
RESET	Trip reset	BOOL	0	I
REVERS	Motor reverse	BOOL	0	I
RAT_CURR	Motor-current ratio	REAL	0	O
RES_x	Reserve (x = 1 to 11)	BOOL	0	I
SIM_ON	1 = activate simulation	BOOL	0	I

I/O (parameter)	Meaning	Data type	Default	Type
SIM_U	Simulation value	WORD	0	I
VALUE	Input value	BOOL	0	I

4.11 CH_U_AI: Analog value input (universal)

4.11.1 Description of CH_U_AI (Universal)

Object name (type + number)

FC283

- CH_U_AI block I/Os (Page 336)

Area of application

Block CH_U_AI processes the analog input signals of S7-300/400 SM analog input modules of a PA field device (PA profile 3.0 Analog Input) or a HART field device (primary or secondary variables).

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32) and the restart OB 100.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The parameters for input PA_ON are set depending on the I/O devices used - S7 signal modules (= 0) or PA field devices or primary/secondary variables of a HART field device (= 1).
- The icon for the quality code of the analog input channel is interconnected with input QC (for PA devices).
- Input MODE is interconnected with the corresponding output OMODE_xx of the MOD block (or of the PADP block).

Function and operating principle

Block CH_U_AI cyclically processes all channel-specific signal functions or REAL data-type signals of a field device, with or without a quality code.

The block uses a tag (input parameter PA_ON) to distinguish between an analog raw value and a REAL value of a PA field device with quality code. Further information is available in the "Addressing" section.

- PA_ON = TRUE

  The condition PQC = TRUE must be satisfied, since the REAL value of a PA field device or the primary/secondary variable of a HART field device is always defined with a quality code.

  If the high byte of the MODE (Page 599) input parameter = 16#40 (value status = higher-level error, QMOD_ERR = TRUE), the quality code is determined according to PA_ON = FALSE.

- PA_ON = FALSE

  The block reads an analog raw value from the process image (partition) and converts its physical value accordingly or calculates a percentage value. The status at input MODE determines the format of the raw value and how it is processed. If the high byte of the MODE input parameter is 16#40 (value status = higher-level error), the raw value is treated as invalid.

Quality code

The program generates a quality code of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, process related	16#28
Uncertain, device related	16#68
Uncertain, process related	16#78
Last valid value	16#44
Substitute value	16#48
Range overshoot	16#54
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

Addressing

The symbol generated in HW Config for the symbol table of the analog input channel has to be interconnected to the input parameter VALUE or (with PA_ON = TRUE) OUT (process value) and QC (Quality Code).

Raw value check

Only if PA_ON = FALSE: Depending on the measurement type and range of the analog input module, the nominal range sets the range for converting analog signals into digital values (raw values). There is also an overshoot/undershoot range within which an analog signal can still be converted. Values outside this range constitute an overflow or underflow. The block indicates whether the raw value lies inside the nominal range of the module.

Output parameter QCHF_LL = TRUE if the value is outside the nominal low range. Output parameter QCHF_HL = TRUE if the value is outside the nominal high range. QCHF_LL/QCHF_HL remain set to TRUE if a channel error occurs due to the module diagnosis "undershoot/overshoot of the measuring range".

QBAD = TRUE (channel error) is also set when a signal overflow or underflow error occurs.

Note

The reaction of the modules to a wire break in the 4 mA to 20 mA signal line is not uniform. Depending on the module either 16#7FFF (overflow) or 16#8000 (underflow) is written to the process image as a raw value. Channel block CH_U_AI then correspondingly outputs either an overflow (QCHF_HL = TRUE) or an underflow (QCHF_LL = TRUE) signal, each together with QBAD = TRUE.Exception: If you have set the "Diagnostic interrupt" of the analog input module in HW Config, only QBAD = TRUE will be set if a diagnostic interrupt is triggered after a "Channel error" has occurred (e.g., due to a wire break).

NAMUR limit check

Only if PA_ON = FALSE: The NAMUR guidelines for analog signal processing define limits for life zero (4 to 20 mA) analog signals that have a channel error:

3.6 mA ≤ analog signal ≤ 21 mA

The above NAMUR limits are set as fixed defaults for limit value monitoring. If you want to set other limits, you have to set the CH_F_ON input parameter to TRUE and set corresponding new limits in mA at the CH_F_HL and CH_F_LL input parameters. QBAD = TRUE if a life zero analog signal is outside the current high or low limit range.

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.

Normal value

Only if PA_ON = FALSE: The raw value is converted to its physical value based on the settings at input parameters VLRANGE, VHRANGE, and MODE. These values will be written to the outputs OVLRANGE and OVHRANGE to allow the interconnection of the settings of VLRANGE and VHRANGE to other block I/Os. The conversion algorithm depends on a linearized input signal. If VLRANGE = 0 and VHRANGE = 100, you obtain a percentage value. If VHRANGE = VLRANGE is set, you obtain the analog input module's input signal (e.g. mA) according to the MODE (Page 599) setting. If the raw value is already a physical value, set VLRANGE = 0 and VHRANGE = 1. The Quality Code is set to QUALITY = 16#80.

When operating in PTC measurement mode, the analog value contains an encoded binary signal. The output provides the following information:

- If the measured resistance is within the normal range, `PV_Out = 0.0`.
- If the measured resistance is in the prewarning range, `PV_Out = 4.0`.
- If the measured resistance is in the operating range, `PV_Out = 1.0`.

This only applies when you set the input parameters VLRANGE = 0 and VHRANGE = 1. You should only set 0 or 1 for the simulation and substitute values SIM_V and SUBS_V.

Note

In the measuring mode "External or internal comparison of thermocouple values", the physical unit is adapted to the +/- 80 mV range in S7 300 modules. You have to determine the temperature by means of the corresponding conversion tables.

If the physical equivalent in mV is delivered by the module as a raw value, set VHRANGE and VLRANGE +/- 80 mV.

Simulation

If the input parameter SIM_ON = TRUE, the value of the SIM_V input parameter is output with quality code QUALITY = 16#60. QBAD = TRUE: reset due to a higher-level error (QMOD_ERR = TRUE). A valid operating mode also has to be set in the low word of the MODE (Page 599) input in simulation mode. Otherwise, QBAD = 1 is output. Simulation takes highest priority. The simulation value is converted into a raw value, based on the operating mode and the input parameters VHRANGE and VLRANGE. This value is verified in the same way as a raw value from the process image. This allows simulation of the QBAD, QCHF_LL and QCHF_HL states.

If VLRANGE > VHRANGE, the state QBAD = TRUE can not be simulated. The QCHF_LL and QCHF_HL outputs are set according to the value of SIM_V.

If a QBAD is to be formed in the negative range with a unipolar measuring range, the value must be set to -119%.

If PA_ON = TRUE, no conversion to a raw value is performed. QBAD = FALSE is always set when SIM_ON = TRUE.

If the block is in the simulation state, QSIM = TRUE.

Note

Remember that the simulation value is always output in simulation mode regardless of whether one of the parameters LAST_ON (substitute value) or SUBS_ON (last valid value).

Substitute value

When input parameter SUBS_ON = TRUE and the raw value is invalid, the value at input parameter SUBS_V will be output as a substitute. The quality code will be set to QUALITY = 16#48 and QBAD = 1.

Hold last value

With input parameter LAST_ON = TRUE, the last valid output value (V_LAST) is output if the delta value is invalid. If V_DELTA > 0, the following applies:

- ABS (V - V_LAST) > V_DELTA: V = V_LAST1 (second to last valid output value)
- ABS (V - V_LAST) ≤ V_DELTA: V = V_LAST (last valid output value)

The quality code will be set to QUALITY = 16#44, DELTA_ON and QBAD = 1.

If valid raw values are available, V_DELTA > 0 and ABS (V - V_LAST) > V_DELTA, the last valid output value (V_LAST) with QUALITY = 16#44 (QBAD = 0) is held for the duration of one cycle.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Delayed Value Acceptance

Only if PA_ON = FALSE: After a restart or if the quality code is changed from "BAD" to "GOOD", the quality code and value are not updated unless the number of cycles specified in CNT_LIM has expired. When CNT_LIM = 0 (default setting), this function is disabled. During this delay time, the Quality Code = 16#00 and QBAD = 1 and the last value will be retained.

Value limiting

When PA_ON = TRUE, you can set a limiting filter for process values of the process image (partition).

If the switch LIMIT_ON = TRUE, the process values (V) are limited as follows:

- To V_HL, if V > V_HL
- To LL_V, if V < V_LL

Error handling

The plausibility of input parameters is not checked. If an invalid mode is set in the low word of the MODE (Page 599) input parameter, it is assumed that the raw value is invalid.

Startup characteristics

The accept value delay is started when CNT_LIM is # 0.

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.11.2 I/Os of CH_U_AI

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
CH_F_HL	Overshoot high limit of the input value (mA)	REAL	0	IO
CH_F_LL	Undershoot low limit of the input value (mA)	REAL	0	IO
CH_F_ON	1 = activate limit monitoring	BOOL	0	IO
CNT_LIM	Limits of the startup counter	INT	0	IO
CNT_RES	Startup counter	INT	0	IO
DELTA_ON	Last delta process value exceeded	BOOL	0	IO
LAST_BAD	Last invalid process value	BOOL	0	IO
LAST_ON	1 = last valid value: Enable injection	BOOL	0	IO
LIMIT_ON	1 = enable limiting of the process value at PA field device	BOOL	0	IO
LL_V	Process value, if V < V_LL	REAL	0	IO
MODE	Value status and mode	DWORD	0	IO
OUT	Process-image input value	REAL	0	IO
OVHRANGE	High limit of the process value (copy)	REAL	0	O
OVLRANGE	Low limit of the process value (copy)	REAL	0	O
PA_ON	1 = use PA field device in the process image	BOOL	0	IO
PQC	1 = use value status in the process image	BOOL	0	IO
QBAD	1 = invalid process value	BOOL	0	O
QC	Status of the input process value	BYTE	0	IO
QCHF_HL	1 = input value high limit	BOOL	0	O
QCHF_LL	1 = input value low limit	BOOL	0	O
QLAST	1 = last valid value: Injection active	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QSUBS	1 = substitution active	BOOL	0	O
QUALITY	Process-value status	BYTE	0	O
SIM_ON	1 = activate simulation	BOOL	0	IO
SIM_V	Simulation value	REAL	0	IO
STATUS	Process-value status	BYTE	0	O
SUBS_ON	1 = enable substitution	BOOL	0	IO
SUBS_V	Substitute value	REAL	0	IO
V	Process value	REAL	0	O
V_DELTA	Delta (V - V_LAST) of the process value	REAL	0	IO
V_HL	High limit	REAL	0	IO
V_LAST	Last valid process value	REAL	0	IO
V_LAST1	Second to last valid process value	REAL	0	IO
V_LL	Low limit	REAL	0	IO

I/O (parameter)	Meaning	Data type	Default	Type
VALUE	Input value	WORD	0	IO
VHRANGE	High limit of the process value	REAL	100	IO
VLRANGE	Low limit of the process value	REAL	0	IO

4.12 CH_U_AO: Analog value output (universal)

4.12.1 Description of CH_U_AO

Object name (type + number)

FC 284

- CH_U_AO block I/Os (Page 343)

Area of application

Block CH_U_AO processes analog output signals of S7-300/400 SM analog output modules or of PA field devices (PA profile 3.0 Analog Output, only REAL values [e.g. SP] with quality code are output).

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32) and the restart OB 100.

NOTICE
With PCS 7 it is not intended that other blocks will be inserted between the plant block and the output driver. If you deviate from this principle, ensure when interconnecting the block that from the outputs of the plant block until the output driver all blocks that form the output signal are installed in the same OBs.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The PA_ON input is configured in accordance with the I/O devices used (S7 signal modules [= 0] or PA field devices [= 1]).
- The icon for the quality code of the analog output channel is interconnected with the QC_SP output (for PA devices).
- The MODE input is interconnected with the corresponding OMODE_xx output of the MOD block (or of the PADP block).
- The CH_U_AO block is installed downstream of the MOD/PADP block that is assigned to it in OB 100.

Note

If you do not use the CFC function "Generate module drivers", you have to ensure that the CH_U_AO block is installed downstream of the MOD/PADP block assigned to it in OB 100.

Function and operating principle

Block CH_U_AO cyclically processes all channel-specific signal functions/REAL values with quality code.

Block CH_U_AO uses a tag (input parameter PA_ON) to distinguish between an analog raw value and a REAL value of a PA field device with quality code. Further information is available in the "Addressing" section.

- PA_ON = TRUE

The block writes the REAL value (SP) with quality code (ST_SP) of a PA field device to the process image (partition).

- PA_ON = FALSE

The block writes the process value as an analog raw value to a process image (partition). The MODE input parameter determines the form in which the raw value is to be generated.

If the high byte of the MODE (Page 599) input parameter = 0 (value status), the raw value is still written to the process image (partition), but with the quality code "invalid value".

Quality code

The quality code may assume the following states:

State	Quality code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, process related	16#28
Uncertain, device related	16#68
Uncertain, process related	16#78
Last valid value	16#44
Substitute value	16#48
Range overshoot	16#54
High value limited	16#56
Low value limited	16#55
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error, or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

Addressing

The symbol (symbol table) generated in HW Config for the analog output channel must be interconnected with the VALUE output parameter.

For PA field devices (PA_ON = TRUE), you must interconnect the icon for the REAL value with the O_SP output parameter.

Normal value

- Only if PA_ON = FALSE: Parameters ULRANGE and UHRANGE map the process value U to the raw VALUE (quantization steps) of the analog output module according to the MODE (Page 599). For example, in mode 4 mA to 20 mA (16#0203), the raw value for 4 mA is output if U = ULRANGE and the raw value for 20 mA is output if U = UHRANGE.
- The block switches the parameters UHRANGE and ULRANGE through to the outputs OVHRANGE and OVLRANGE. You can interconnect the outputs with the manipulated-variable limits NM_LMNHR and NM_LMNLR of the controller CTRL_PID, for example.
- PHYS_LIM can be used to set the limits for the raw VALUE. The default setting of PHYS_LIM = 0 limits the value at output VALUE to the default limits of the module. According to the example above, the block calculates the raw value for 20 mA if U > UHRANGE and the raw value for 4 mA if U < ULRANGE. As a result, the quality codes 16#56 (high value limited) and 16#55 (low value limited) are applied at the QUALITY output instead of 16#80 (valid value).

- If you want to output analog values outside the default limits up to the physical limits of the module, set PHYS_LIM = 1. The output values are limited only if, taking the example above, you exceed the module limit values by specifying U = 200 (36 mA) or U = -50 (-4 mA) when ULRANGE = 0 and UHRANGE = 100. The output values are then limited to the physical limits specified in the data sheets of the modules and the corresponding quality codes are output.
- The QCHF_HL and QCHF_LL outputs also provide information on whether output-value limits have been set.

Simulation

If the input parameter SIM_ON = TRUE is set, the value of SIM_U is output with quality code (QUALITY) = 16#60. QBAD is always reset. Simulation takes highest priority. If the block is in the simulation state, QSIM = TRUE.

I/O fault

If the high byte of the MODE input parameter is set to 0 (value status), the quality code QUALITY = 16#00 is set. The actual raw value is always written to the process image (partition).

Value limiting

Only if PA_ON = FALSE: You can limit very low or very high process values that would lead to an error (QBAD = TRUE) before they were entered in the process image (partition).

If the switch LIMIT_ON = TRUE, the process values (U) are limited as follows:

- To V_HL, if U > V_HL and
- To LL_V, if U < V_LL.

Error handling

The plausibility of input parameters is not checked. If an invalid mode has been set in the low word of the MODE (Page 599) input, the digitized output value will be set to 0 and QUALITY = 16#00 is output.

Startup characteristics

The MOD blocks set the LSB in byte 2 of their OMODE_xx (Page 606) output parameters in OB 100. If the block detects this code, it responds with an acknowledgement and:

- If START_ON is not set, computes the process value U and writes the result to the process image.

  If START_ON is set, the raw value corresponding to the START_U process value is written to the process image.
- When PA field devices are enabled (PA_ON = TRUE), the actual REAL value with quality code is written to the process image.

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.12.2 I/Os of CH_U_AO

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
LIMIT_ON	1 = enable limiting of the process value	BOOL	0	IO
LL_V	Process value, if U < V_LL	REAL	0	IO
MODE	Value status and mode	DWORD	0	IO
O_SP	Process image setpoint	REAL	0	O
OVHRANGE	Output high limit of the process value	REAL	100	O
OVLRANGE	Output low limit of the process value	REAL	0	O
PA_ON	1 = PA field device, 0 = signal module	BOOL	0	IO
PHYS_LIM	1 = enable physical limits of the module	BOOL	0	IO
QBAD	1 = invalid output value	BOOL	0	O
QCHF_HL	1 = overshoot of process value	BOOL	0	O
QCHF_LL	1 = undershoot of process value	BOOL	0	O
QC_SP	Process image quality code setpoint	BYTE	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QUALITY	Value status of the output value	BYTE	0	O
SIM_ON	1 = activate simulation	BOOL	0	IO
SIM_U	Simulation value	REAL	0	IO
START_ON	1 = substitution at startup	BOOL	0	IO
START_U	Substitute value at startup	REAL	0	IO
ST_SP	Setpoint status	BYTE	0	IO
U	Process value	REAL	0	IO
UHRANGE	High limit of the process value	REAL	100	IO
ULRANGE	Low limit of the process value	REAL	0	IO
V_HL	High limit	REAL	0	IO
V_LL	Low limit	REAL	0	IO
VALUE	Output value	WORD	0	O

4.13 CH_U_DI: Digital value input (universal)

4.13.1 Description of CH_U_DI (Universal)

Object name (type + number)

FC285

- CH_U_DI block I/Os (Page 348)

Area of application

Block CH_U_DI processes digital input signals of S7-300/400 SM digital input modules or of PA field devices (PA profile 3.0 Discrete Input).

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32).

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The PA_ON input is configured in accordance with the I/O devices used (S7 signal modules [= 0] or PA field devices [= 1]).
- The icon for the quality code of the digital input channel is interconnected with input QC (for PA devices).
- Input MODE is interconnected with the corresponding output OMODE_xx of the MOD block (or of the PADP block).

Function and operating principle

Block CH_U_DI cyclically processes all channel-specific signal functions/process values with quality code of a PA field device.

The block uses a tag (PA_ON input parameter) to distinguish between a digital value with or without quality code of the data type BOOL and a digital value with quality code of the data type BYTE of a PA field device. Further information is available in the "Addressing" section.

- PA_ON = TRUE

  The block cyclically reads the process value (OUT_D) and the status byte (QUALITY, see "Addressing") of the PROFIBUS PA field device (structure in accordance with the discrete input of the PA profiles) from the process image (partition). The status byte contains information about the measured value and the status of the PROFIBUS field device. The block transfers the process value to output Q as shown below:

  – Q = FALSE, if the process value = 0

  – Q = TRUE, if the process value <> 0
    If the high byte of the MODE (Page 599) input parameter = 16#40 (value status = higher-level error), the process value and quality code are treated as if PA_ON = FALSE.

- PA_ON = FALSE

  The block reads a digital value of the data type BOOL from the process image (partition). If the high byte of the MODE input parameter = 16#40 (value status = higher-level error, QMOD_ERR = TRUE), the digital value is treated as invalid. If input parameter PQC = TRUE, it reads the value status of the digital value from the process image (partition).

Quality code

The program generates a quality code of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, process related	16#28
Uncertain, device related	16#68
Uncertain, process related	16#78
Last valid value	16#44
Substitute value	16#48
Range overshoot	16#54
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error, or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

Addressing

The symbol generated in HW Config (symbol table) for the digital input channel must be interconnected to the VALUE input. If the process image (partition) also contains the value status of the digital input channel, interconnect the corresponding symbol with input VALUE_QC and set input PQC = TRUE.

When using PA field devices (PA_ON = TRUE), interconnect the symbol generated in HW Config (symbol table) for the digital input channel with input I_OUT_D.

Normal value

The digital value of the process image (partition) and the quality code QUALITY = 16#80 are applied to output Q.

Simulation

If the SIM_ON input parameter = TRUE, the value of the SIM_I (PA_ON = FALSE) or SIM_OUT (PA_ON = TRUE) input parameter is output with quality code QUALITY = 16#60 to the output parameter Q. QBAD is always reset. Simulation takes highest priority.

If the block is in the simulation state, QSIM = TRUE.

Note

Remember that the simulation value is always output in simulation mode regardless of whether one of the parameters LAST_ON (substitute value) or SUBS_ON (last valid value).

Substitute value

When input parameter SUBS_ON = TRUE and the digital value of the process image (partition) is invalid, the function outputs the signal QBAD = 1 and the value at input parameter SUBS_I with quality code QUALITY = 16#48 to output parameter Q.

Hold last value

If the input parameter SUBS_ON = FALSE, the function outputs the last valid output value when the raw value is invalid. The quality code is set to QUALITY = 16#44 and QBAD = 1.

Last valid output value = Q_LAST.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.13.2 I/Os of CH_U_DI

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
I_OUT_D	Input value of the process image	BYTE	0	IO
LAST_ON	1 = last valid value: Enable injection	BOOL	0	IO
MODE	Value status and mode	DWORD	0	IO
OUT_D	Process value	BYTE	0	O
PA_ON	1 = PA field device	BOOL	0	IO
PQC	1 = use value status in the process image	BOOL	0	IO
Q	Process value	BOOL	0	O
Q_LAST	Last valid process value	BOOL	0	IO
QBAD	1 = invalid process value	BOOL	0	O
QC	Value status in the process image (icon)	BYTE	0	IO
QLAST	1 = last valid value: Injection active	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QSUBS	1 = substitution active	BOOL	0	O
QUALITY	Process-value status	BYTE	0	O
SIM_I	Input simulation value	BOOL	0	IO
SIM_ON	1 = activate simulation	BOOL	0	IO
SIM_OUT	Output simulation value	BYTE	0	IO
STATUS	Process-value status	BYTE	0	O
SUBS_I	Input substitute value	BOOL	0	IO
SUBS_ON	1 = enable substitution	BOOL	0	IO
SUBS_OUT	Output substitute value	BYTE	0	IO
V_LAST	Last valid process value	BYTE	0	IO
VALUE	Input value	BOOL	0	IO
VALUE_QC	Value status in the process image	BOOL	0	IO

4.14 CH_U_DO: Digital-Value Output (Universal)

4.14.1 Description of CH_U_DO (Universal)

Object name (type + number)

FC 286

- CH_U_DO block I/Os (Page 353)

Area of application

Block CH_U_DO processes digital output signals of S7-300/400 SM digital output modules or of PA field devices (PA profile 3.0 Discrete Output, only SP or RCAS_IN).

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32) and the restart OB 100.

NOTICE
With PCS 7 it is not intended that other blocks will be inserted between the plant block and the output driver. If you deviate from this principle, ensure when interconnecting the block that from the outputs of the plant block until the output driver all blocks that form the output signal are installed in the same OBs.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The PA_ON input is configured in accordance with the I/O devices used (S7 signal modules [= 0] or PA field devices [= 1]).
- The icon for the quality code of the digital output channel is interconnected with the QC_SP output (for PA devices).
- The MODE input is interconnected with the corresponding OMODE_xx output of the MOD block (or of the PADP block).
- The CH_U_DO block is installed downstream of the MOD/PADP block that is assigned to it in OB 100.

Note

If you do not use the CFC function "Generate module drivers", you have to ensure that the CH_U_DO block is installed downstream of the MOD/PADP block assigned to it in OB 100.

Function and operating principle

Block CH_U_DO cyclically processes all channel-specific signal functions/process values with quality code of a PA field device.

The block uses a tag (PA_ON input parameter) to distinguish between a digital output value without quality code of the data type BOOL and a digital output value with quality code of the data type BYTE of a PA field device. Further information is available in the "Addressing" section.

- PA_ON = TRUE

  The block writes the process value with quality code to a process image (partition) (structure of the process value corresponds with the digital output of the PA profiles, 1 byte (SP) with 1 byte (ST_SP) quality code). The quality code contains information on the process-value state. The coding of the quality code is described in PROFIBUS 3.0 "General Requirements".

  If the high byte of the MODE (Page 599) input parameter = 16#40 (value status = higher-level error, QMOD_ERR = TRUE), the function continues to write the process value with quality code to the process image (partition), but sets the quality code at the QUALITY block output to "invalid value".

- PA_ON = FALSE

  The block writes a digital value to a process image (partition). If the high byte of the MODE input parameter = 0 (value status), this digital value will still be written to the process image (partition), but an "invalid value" quality code is set.

Quality code

The quality code may assume the following states:

State	Quality code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, process related	16#28
Uncertain, device related	16#68
Uncertain, process related	16#78
Last valid value	16#44
Substitute value	16#48
Range overshoot	16#54
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error, or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

Addressing

The symbol generated with HW Config in the symbol table for the digital output channel must be interconnected with the VALUE output parameter.

When PA field devices (PA_ON = TRUE) are used, the symbol generated in the symbol table for the process value with quality code must be interconnected with output variable O_SP.

Normal value

The digital value is written to the process image (partition) and quality code (QUALITY) = 16#80.

Simulation

If the input parameter SIM_ON = TRUE, the value of input parameter SIM_I (PA_ON = FALSE) or SIM_SP (PA_ON = TRUE) is written to the process image (partition) and the quality code QUALITY = 16#60 is set. QBAD is always reset. Simulation takes highest priority. If the block is in the simulation state, QSIM = TRUE.

I/O fault

If the high byte of the MODE (Page 599) input parameter = 0 (value status), the quality code QUALITY = 16#00 is set. The function always writes the current digital value to the process image (partition).

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

The MOD blocks set the LSB in byte 2 of their OMODE_xx (Page 606) output parameters in OB 100. If the block detects this code, it responds with an acknowledgement and:

- If START_ON is not set, writes process value I to the process image.
- If START_ON is set, START_I is used instead of process value I.

When PA field devices (PA_ON = TRUE) are used, the current BYTE value and the quality code are written to the process image.

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.14.2 I/Os of CH_U_DO

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
I	Process value	BOOL	0	IO
MODE	Value status and mode	DWORD	0	IO
O_SP	Process image setpoint	BYTE	0	O
PA_ON	1 = PA field device	BOOL	0	IO
QBAD	1 = invalid output value	BOOL	0	O
QC_SP	Process image quality code setpoint	BYTE	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QUALITY	Value status of the output value	BYTE	0	O
SIM_I	Simulation value	BOOL	0	IO
SIM_ON	1 = activate simulation	BOOL	0	IO
SIM_SP	Simulation setpoint	BYTE	0	IO
SP	Setpoint	BYTE	0	IO
ST_SP	Setpoint status	BYTE	0	IO
START_I	Substitute value at startup	BOOL	0	IO
START_ON	1 = substitution at startup	BOOL	0	IO
VALUE	Output value	BOOL	0	O

4.15 FF_A_AI: Working with the PA Profile Transmitter

4.15.1 Description of FF_A_AI

Object name (type + number)

FB 410

- FF_A_AI block I/Os (Page 357)

Area of application

The block FF_A_AI processes (cyclic service) the PA profile "Transmitter" of an FF field device.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32).

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The icon for the quality code of the analog input channel is interconnected with input QC_W.
- The MODE input is interconnected with the OMODE_00 output of the MOD_PAL0 block or MOD_PAX0 block.

Function and operating principle

The block FF_A_AI cyclically reads the process value (Quality Code) of the FF field device from the process image partition. The process variable is available in a physical unit. The status byte (STATUS) contains information about the status of the FF field device.

Linking in ASSET

The block FF_A_AI can be linked in ASSET.

For further information, refer to the section:

Integrating FF devices in ASSET (Page 612)

Quality code

The program generates a quality code (QUALITY output) of the resultant value which may assume the states shown below:

State	Quality Code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, due to process	16#28
Uncertain, due to device	16#68
Uncertain, due to process	16#78
Last valid value	16#44
Substitute Value	16#48
Range overshoot	16#54
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error or simulation and from a quality code, that comes directly from the device (parameter QC-W).

The quality code that comes directly from the device can have values from 16#00 – 16#FF. The quality codes 16#84-16#87 and 16#90 – 16#93 of the FF field device are analyzed like the quality codes 16#80 – 16#83. If the quality code 16#80 (valid value) and a process value (input VALUE) with the bit pattern 16#7FFFFFFF (invalid value) from the FF device are transferred, then this is handled the same as quality code 16#00 (invalid value) by the FF field device.

In addition to the status byte and to improve interconnectability, the output interface provides further Boolean (BOOL) values, which contain important details. They conform to bit combinations specifications in PROFIBUS PA 3.0 "General Requirements".

The block recognizes a higher-level error, for example, failure of a DP/PA link, via the MODE (Page 611) input parameter.

- If the MODE high byte = 16#80, then the values in the process image (partition) are valid.
- If the MODE high byte = 16#40 (value status = higher-level error), the analog value is treated as invalid.

The mode of the input parameter MODE_LW is not considered.

Addressing

You must configure an icon for the analog input channel in the icon table and connect it with the input VALUE. The CFC function "Generate module drivers" interconnects the icon for the quality code of the analog input channel with the input parameter QC_W.

Simulation

When input parameter SIM_ON = TRUE, the value of the input parameter SIM_V is output with quality code (QUALITY =) 16#60. Simulation takes highest priority. QBAD is always set to FALSE. If the block is in the simulation status, QSIM = TRUE is set.

Substitute Value

If input parameter SUBS_ON = TRUE, the value of input parameter SUBS_V is output as a value if the values are invalid. The quality code (QUALITY) is set to 16#48 and QBAD = 1.

Output invalid value

If the input parameter SUBS_ON = FALSE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Value Limiting

You can limit the maximum low and high range of process variables in the process image (partition).

LIMIT_ON = TRUE limits process variables (V):

- To V_HL, if V > V_HL
- To LL_V, if V < V_LL

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

The block has no faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.15.2 I/Os of FF_A_AI

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You can find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
LIMIT_ON	1 = limit input value ON	BOOL	0	I
LL_V	Value if V < V_LL	REAL	0	I
MODE	Value status	DWORD	0	I
MODE_LW	Mode	WORD	1	I
QA_1	1 = alarm/warning 1	BOOL	0	O
QA_2	1 = alarm/warning 2	BOOL	0	O
QBAD	1 = group event QBAD_X	BOOL	0	O
QBAD_0	1 = non-specific	BOOL	0	O
QBAD_1	1 = configuration error	BOOL	0	O
QBAD_2	1 = not connected	BOOL	0	O
QBAD_3	1 = device failure	BOOL	0	O
QBAD_4	1 = sensor failure	BOOL	0	O
QBAD_5	1 = no communication (last usable value)	BOOL	0	O
QBAD_6	1 = no communication (last usable value)	BOOL	0	O
QBAD_7	1 = out of service	BOOL	0	O
QBAD_HL	1 = high limit of physical range of sensor has been reached	BOOL	0	O
QBAD_LL	1 = low limit of physical range of sensor has been reached	BOOL	0	O
QC_W	Status of the input process value	WORD	0	I
QCASCAD0	1 = OK (cascade)	BOOL	0	O
QCASCAD1	1 = initialization acknowledged	BOOL	0	O
QCASCAD2	1 = initialization request	BOOL	0	O
QCASCAD3	1 = not required	BOOL	0	O
QCASCAD4	1 = reserved	BOOL	0	O
QCASCAD5	1 = do not select	BOOL	0	O
QCASCAD6	1 = local overwrite	BOOL	0	O
QCASCAD7	1 = reserved	BOOL	0	O
QCASCAD8	1 = initiate fail-safe	BOOL	0	O
QCONST	1 = constant	BOOL	0	O
QERR	1 = output error (inverted value of ENO)	BOOL	1	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QOUT_HHL	1 = active critical alarm, high limit of OUT has been exceeded	BOOL	0	O
QOUT_HL	1 = active warning, high limit of OUT has been exceeded	BOOL	0	O
QOUT_LL	1 = active warning, low limit of OUT has been exceeded	BOOL	0	O
QOUT_LLL	1 = active critical alarm, low limit of OUT has been undershot	BOOL	0	O
QNONCAS0	1 = OK (non-cascade)	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QNONCAS1	1 = update event	BOOL	0	O
QNONCAS2	1 = active warning (priority <8)	BOOL	0	O
QNONCAS3	1 = active critical alarm (priority >8)	BOOL	0	O
QNONCAS4	1 = unacknowledged updated event	BOOL	0	O
QNONCAS5	1 = unacknowledged warning	BOOL	0	O
QNONCAS6	1 = unacknowledged critical alarm	BOOL	0	O
QNONCAS7	1 = initiate fail-safe	BOOL	0	O
QNONCAS8	1 = maintenance required	BOOL	0	O
QNONCAS9	1 = function test/local overwrite; usable value	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QSUBS	1 = substitution active	BOOL	0	O
QUALITY	Value status of the process variable	BYTE	0	O
QUNCERT	1 = group message QUNCERTx	BOOL	0	O
QUNCERT0	1 = non-specific	BOOL	0	O
QUNCERT1	1 = last usable value	BOOL	0	O
QUNCERT2	1 = set of substitute values	BOOL	0	O
QUNCERT3	1 = initial value	BOOL	0	O
QUNCERT4	1 = unacknowledged updated event	BOOL	0	O
QUNCERT5	1 = engineering unit violation (unit not in the valid range)	BOOL	0	O
QUNCERT6	1 = below normal	BOOL	0	O
QUNCERT7	1 = configuration error	BOOL	0	O
QUNCERT8	1 = sensor calibration	BOOL	0	O
QUNCERT9	1 = simulated value	BOOL	0	O
SIM_ON	1 = activate simulation	BOOL	0	I
SIM_V	Simulation value	REAL	0	I
STATUS	Process value status	BYTE	0	O
SUBS_ON	1 = enable substitution	BOOL	0	I
SUBS_V	Substitute value	REAL	0	I
V	Process value	REAL	0	O
V_HL	High limit input value	REAL	0	I
V_LL	Low limit input value	REAL	0	I
VALUE	Process Image Input Value	REAL	0	I

4.16 FF_A_AO: Working with PA Profile Actuator

4.16.1 Description of FF_A_AI

Object name (type + number)

FB 411

- FF_A_AO block I/Os (Page 363)

Area of application

The block FF_A_AO processes (cyclic service) the PA profile "Actuator" of an FF field device.

Calling OBs

The calling OB is cyclic interrupt OB3x in which you install the block (for example, OB32) and the restart OB100.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The MODE input is interconnected with the OMODE_00 output of the MOD_PAL0 block or MOD_PAX0 block.

Function and operating principle

Depending on the selection (configuration of the input MODE_LW) of the user data configuration the block reads the user data from the partition process image and writes it in the partition process image.

The coding of the set user data configuration must be set in the input variable MODE_LW according to the MODE (Page 611) for FF field devices. And specifies which variables are to be read and written in the process image (partition).

The block cyclically writes the setpoint (SP) with Quality Code (configuration of the setpoints and process values in accordance with the Analog Output of the PA profiles, REAL with 1 byte Quality Code) into the process image (partition). The PA profile contains the setpoint and other analog values as physical units. The Quality Code contains the information on the setpoint status. The coding of the quality codes is described in PROFIBUS 3.0 "General Requirements". The reference variable (RCAS_IN) can be transferred with the quality code to the process image (partition) in the same cycle.

Linking in ASSET

The block FF_A_AO can be linked in ASSET.

For further information, refer to the section:

Integrating FF devices in ASSET (Page 612)

Quality code

The program generates a quality code (QUALITY output) of the resultant value which may assume the states shown below:

State	Quality Code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, due to process	16#28
Uncertain, due to device	16#68
Uncertain, due to process	16#78
Last valid value	16#44
Substitute Value	16#48
Range overshoot	16#54
Range overshoot low	16#55
Range overshoot high	16#56
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error or simulation and from a quality code, that comes directly from the device (parameter QC_POS_D_W, QC_RCAS_OUT_W, QC_READBACK_W).

The quality code that comes directly from the device can have values from 16#00 – 16#FF. The quality codes 16#84-16#87 and 16#90 – 16#93 of the FF field device are analyzed like the quality codes 16#80 – 16#83. If the Quality Code 16#80 (valid value) and an associated process value (inputs I_RCAS_OUT or I_READBACK) with the bit pattern 16#7FFFFFFF (invalid value) from the FF device are transferred, then this is handled the same as Quality Code 16#00 (invalid value) by the FF field device.

The data of the FF field device, as well as the process variable (READBACK) and the discrete position feedback (READBACK) are read cyclically from the process image (partition). Optionally you can also read the active reference variable (RCAS_OUT) and the detailed device information (CHECKBACK). The device information is available bit-granular at the block output. The data is read from the process image (partition). In order to improve the interconnectability, important detailed information is provided by the status bytes read (ST_READBACK) or ST_RCAS_OUT (if READBACK is not available) as Boolean (BOOL) values at the output interface. These conform to the bit combinations specified in PROFIBUS 3.0 "General Requirements".

If a higher-level error has occurred (QMOD_ERR = TRUE), the data continue to be written to the process image (partition) and no data are read from the process image (partition). As long as the higher-priority error is active the last values are held with QBAD = TRUE.

Note

The status byte of the reference variable (ST_RCAS_IN) is pre-defined with zero. The reference variable only becomes active in the FF field device if you set the quality code to 16#80.

Configuring the MODE_LW input

In addition to the MODE input the MODE_LW (mode low word) input must be configured according to the parameters used:

Parameters used	MODE_LW
SP	16#0100
SP, READBACK, POS_D	16#0103
SP, CHECK_BACK	16#0104
SP, READBACK, POS_D, CHECK_BACK	16#0105
RCAS_IN, RCAS_OUT	16#0206
RCAS_IN, RCAS_OUT, CHECK_BACK	16#0207
SP, RCAS_IN, READBACK, RCAS_OUT, POS_D, CHECK_BACK	16#0308

Addressing

You must interconnect each used connection of the block with the icons configured in the HW Config or in the icon table according to its user data configuration:

I/O	Data type
I_READBACK	REAL
I_RCAS_OUT	REAL
I_POS_D_W	WORD
O_SP	REAL
O_RCAS_IN	REAL

In addition the quality code of the used I/Os must be interconnected with the icon configured in the HW Config:

I/O	Data type
CHECK_0_W	WORD
CHECK_1_W	WORD
CHECK_2_W	WORD
QC_READBACK_W	WORD
QC_RCAS_OUT_W	WORD
QC_POS_D_W	WORD

I/O	Data type
QC_SP_W	WORD
QC_RCAS_IN_W	WORD

Simulation

If the input parameter SIM_ON = TRUE, the value of the input SIM_SP (and the option SIM_RCAS_IN) is output with Quality Code (QUALITY) = 16#60. Simulation takes highest priority. QBAD is set to FALSE. If the block is in the simulation status, QSIM = TRUE is set.

Value Limiting

You can limit the maximum low and high range of process variables in the process image (partition). The process variables "V" (READBACK and RCAS_OUT) are limited when LIMIT_ON = TRUE:

- To V_HL, if V > V_HL.
- To LL_V, if V < V_LL.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

The block will run through one time in OB 100 at system start. The output and in/out parameters are calculated.

Time response

Not available

Message response

Not available

Operating and monitoring

The block has no faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.16.2 I/Os of FF_A_AO

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
CHECK_0_W	Additional information field device	WORD	0	I
CHECK_1_W	Additional information field device	WORD	0	I
CHECK_2_W	Additional information field device	WORD	0	I
I_POS_D_W	Current position of the valve (discrete) (symbol)	WORD	0	I
I_RCAS_OUT	Function block setpoint	REAL	0	I
I_READBACK	Current position of the final control element during runtime (between OPEN and CLOSE positions) in PV units (icon)	REAL	0	I
LIMIT_ON	1 = limit input value ON	BOOL	0	I
LL_V	High limit input value. Set value if V < V_LL	REAL	0	I
MODE	Value status	DWORD	0	I
MODE_LW	Mode	WORD	0	I
O_RCAS_IN	Process image target setpoint provided by a monitoring host to the analog control or output block (symbol)	REAL	0	O
O_SP	Setpoint (symbol)	REAL	0	O
POS_D	Current position of the valve (discrete). The coding of the POS bytes is as follows: 0 = not initialized, 1 = closed, 2 = open, 3 = intermediate	BYTE	0	O
QA_1	1 = alarm/warning 1	BOOL	0	O
QA_2	1 = alarm/warning 2	BOOL	0	O
QBAD	1 = group event QBAD_X	BOOL	0	O
QBAD_0	1 = non-specific	BOOL	0	O
QBAD_1	1 = configuration error	BOOL	0	O
QBAD_2	1 = not connected	BOOL	0	O
QBAD_3	1 = device failure	BOOL	0	O
QBAD_4	1 = sensor failure	BOOL	0	O
QBAD_5	1 = no communication (last usable value)	BOOL	0	O
QBAD_6	1 = no communication (no usable value)	BOOL	0	O
QBAD_7	1 = out of service	BOOL	0	O
QCASCAD0	1 = OK (cascade)	BOOL	0	O
QCASCAD1	1 = initialization acknowledged	BOOL	0	O
QCASCAD2	1 = initialization request	BOOL	0	O
QCASCAD3	1 = not required	BOOL	0	O
QCASCAD4	1 = reserved	BOOL	0	O
QCASCAD5	1 = do not select	BOOL	0	O
QCASCAD6	1 = local overwrite	BOOL	0	O
QCASCAD7	1 = reserved	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QCASCAD8	1 = initiate fail-safe	BOOL	0	O
QCB_0	1 = field device in fail-safe position active	BOOL	0	O
QCB_1	1 = request for manual mode	BOOL	0	O
QCB_2	1 = field device in manual mode, LOCKED OUT switch is active	BOOL	0	O
QCB_3	1 = emergency control active	BOOL	0	O
QCB_4	1 = actual position feedback differs from expected position	BOOL	0	O
QCB_5	1 = torque limit in OPEN direction has been exceeded	BOOL	0	O
QCB_6	1 = torque limit in CLOSE direction is exceeded	BOOL	0	O
QCB_7	1 = status of travel monitoring equipment; if YES, travel time for actuator has been exceeded	BOOL	0	O
QCB_8	1 = actuator is opening	BOOL	0	O
QCB_9	1 = actuator is closing	BOOL	0	O
QCB_10	1 = the interrupt generated by any change to the static data (function and transducer block)	BOOL	0	O
QCB_11	1 = simulation of process values is enabled	BOOL	0	O
QCB_12	Not used	BOOL	0	O
QCB_13	1 = internal control loop interrupted	BOOL	0	O
QCB_14	1 = positioner inactive (OUT status = BAD)	BOOL	0	O
QCB_15	1 = device under self test	BOOL	0	O
QCB_16	1 = valve stroke limit is exceeded	BOOL	0	O
QCB_17	1 = additional input (for diagnostics, for example) is activated	BOOL	0	O
QCONST	1 = constant	BOOL	0	O
QC_POS_D_O	Quality code POS output	BYTE	0	O
QC_POS_D_W	Quality code POS	WORD	16#80	I
QC_RCAS_IN	Quality code RCAS_IN	BYTE	0	O
QC_RCAS_IN_W	Quality code RCAS_IN (symbol)	WORD	0	O
QC_RCAS_OUT_W	Quality-code function-block setpoint	WORD	16#80	I
QC_READBACK_O	Quality-code function-block setpoint output	BYTE	0	O
QC_READBACK_W	Quality-code function-block setpoint (symbol)	WORD	16#80	I
QC_SP	Quality code setpoint	BYTE	0	O
QC_SP_W	Quality code setpoint (symbol)	WORD	0	O
QERR	1 = output error (inverted value of ENO)	BOOL	1	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QNONCAS0	1 = OK (non-cascade)	BOOL	0	O
QNONCAS1	1 = update event	BOOL	0	O
QNONCAS2	1 = active warning (priority <8)	BOOL	0	O
QNONCAS3	1 = active critical alarm (priority >8)	BOOL	0	O
QNONCAS4	1 = unacknowledged updated event	BOOL	0	O
QNONCAS5	1 = unacknowledged warning	BOOL	0	O
QNONCAS6	1 = unacknowledged critical alarm	BOOL	0	O
QNONCAS7	1 = initiate fail-safe	BOOL	0	O
QNONCAS8	1 = maintenance required	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QNONCAS9	1 = function test/local overwrite; usable value	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QUNCERT	1 = group event QUNCERTx	BOOL	0	O
QUNCERT0	1 = non-specific	BOOL	0	O
QUNCERT1	1 = last usable value	BOOL	0	O
QUNCERT2	1 = set of substitute values	BOOL	0	O
QUNCERT3	1 = initial value	BOOL	0	O
QUNCERT4	1 = imprecise sensor conversion	BOOL	0	O
QUNCERT5	1 = engineering unit violation (unit not in the valid range)	BOOL	0	O
QUNCERT6	1 = below normal	BOOL	0	O
QUNCERT7	1 = configuration error	BOOL	0	O
QUNCERT8	1 = sensor calibration	BOOL	0	O
QUNCERT9	1 = simulated value	BOOL	0	O
RCAS_IN	Target setpoint provided by a monitoring host to the analog control or output block	REAL	0	I
RCAS_OUT	Function block setpoint	REAL	0	O
READBACK	Current position of the final control element within the travel span (between OPEN and CLOSE positions) in PV units.	REAL	0	O
SIM_ON	1 = simulation active	BOOL	0	I
SIM_POS_D	Current position of the valve (discrete)	BYTE	0	I
SIM_RCAS_IN	Simulation RCAS_IN	REAL	0	I
SIM_READBACK	Simulation of the current position of the final control element during runtime (between OPEN and CLOSE positions) in PV units.	REAL	0	I
SIM_SP	Simulation setpoint	REAL	0	I
SP	Setpoint	REAL	0	I
ST_POS_D	Status POS	BYTE	0	O
ST_READBACK	Function-block readback-value status	BYTE	0	O
ST_RCAS_IN	RCAS_IN status	BYTE	0	I
ST_RCAS_OUT	Function-block setpoint status	BYTE	0	O
ST_SP	Setpoint status	BYTE	16#80	I
V_HL	High limit input value	REAL	0	I
V_LL	Low limit input value	REAL	0	I

4.17 FF_A_DI: Reading digital values

4.17.1 Description of FF_A_DI

Object name (type + number)

FB 412

- FF_A_DI block I/Os (Page 369)

Area of application

The block FF_A_DI is for reading in (cyclical service) Digital values (Discrete Input) from an FF device.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32).

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The symbol for the quality code of the digital input channel is interconnected with the input QC_W.
- The MODE input is interconnected with the appropriate OMODE_00 output of the MOD_PAL0 block or MOD_PAX0 block.

Function and operating principle

Block FF_A_DI interfaces an FF field device and the blocks of the SIMATIC PCS 7 libraries. It can also be interconnected with other SIMATIC S7 blocks.

The block cyclically reads the process value (I_OUT_D_W) with status byte of the FF field device (structure in accordance with the discrete input of the PA profiles). The status byte (STATUS) contains information about the status of the FF field device. The process value and important status byte information are made available bit-granular for better interconnectability on the output interface. These conform to the bit combinations specified in PROFIBUS "General Requirements".

Linking in ASSET

The block FF_A_DI can be linked in ASSET.

For further information, refer to the section:

Integrating FF devices in ASSET (Page 612)

Quality Code

A quality code (QUALITY output) is generated for the result value that can adopt the following values:

State	Quality Code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, due to process	16#28
Uncertain, due to device	16#68
Uncertain, due to process	16#78
Last valid value	16#44
Substitute Value	16#48
Range overshoot	16#54
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error or simulation and from a quality code, that comes directly from the device (parameter QC-W).

The quality code that comes directly from the device can have values from 16#00 – 16#FF. The quality codes 16#84-16#87 and 16#90 – 16#93 of the FF field device are analyzed like the quality codes 16#80 – 16#83.

The block recognizes a higher-level error, for example, failure of a DP/PA link, via the MODE (Page 611) input parameter.

- If the MODE high byte = 16#80, then the values in the process image (partition) are valid.
- If the MODE high byte = 16#40 (value status = higher-level error, QMOD_ERR = TRUE), the process value is treated as invalid.

The mode of the input parameter MODE_LW is not considered.

Addressing

In HW Config, you must configure an icon for the digital input channel and connect it with the input I_OUT_D_W. The CFC function "Generate module drivers" interconnects the icon for the quality code of the digital input channel with the input parameter QC_W.

Simulation

When input parameter SIM_ON = TRUE, the value of input SIM_I is output with quality code (QUALITY =) 16#60. Simulation takes highest priority. QBAD is always set to FALSE. If the block is in the simulation status, QSIM = TRUE is set.

Substitute Value

If input parameter SUBS_ON = TRUE, the value of input parameter SUBS_I is output as a value if the process values are invalid. The quality code (QUALITY) is set to 16#48 and QBAD = 1.

Output invalid value

If the input parameter SUBS_ON = FALSE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

The block has no faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.17.2 I/Os of FF_A_DI

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You can find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
I_OUT_D_W	Process value of the process image	WORD	0	I
MODE	Value status	DWORD	0	I
MODE_LW	Mode	WORD	2	I
OUT_D	Process value	BYTE	0	O
QA_1	1 = alarm/warning 1	BOOL	0	O
QA_2	1 = alarm/warning 2	BOOL	0	O
QBAD	1 = group event QBAD_X	BOOL	0	O
QBAD_0	1 = non-specific	BOOL	0	O
QBAD_1	1 = configuration error	BOOL	0	O
QBAD_2	1 = not connected	BOOL	0	O
QBAD_3	1 = device failure	BOOL	0	O
QBAD_4	1 = sensor failure	BOOL	0	O
QBAD_5	1 = no communication (last usable value)	BOOL	0	O
QBAD_6	1 = no communication (no usable value)	BOOL	0	O
QBAD_7	1 = out of service	BOOL	0	O
QC_W	Quality code value of the process image	WORD	0	I
QCONST	1 = constant	BOOL	0	O
QERR	1 = output error (inverted value of ENO)	BOOL	1	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QNONCAS0	1 = OK (non-cascade)	BOOL	0	O
QNONCAS1	1 = update event	BOOL	0	O
QNONCAS4	1 = unacknowledged updated event	BOOL	0	O
QNONCAS7	1 = initiate fail-safe	BOOL	0	O
QNONCAS8	1 = maintenance required	BOOL	0	O
QNONCAS9	1 = function test/local overwrite; usable value	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QSUBS	1 = substitution active	BOOL	0	O
QUALITY	Value status of the process variable	BYTE	0	O
QUNCERT	1 = group event QUNCERTx	BOOL	0	O
QUNCERT0	1 = non-specific	BOOL	0	O
QUNCERT1	1 = last usable value	BOOL	0	O
QUNCERT2	1 = set of substitute values	BOOL	0	O
QUNCERT3	1 = initial value	BOOL	0	O
QUNCERT4	1 = imprecise sensor conversion	BOOL	0	O
QUNCERT5	1 = engineering unit violation (unit not in the valid range)	BOOL	0	O
QUNCERT6	1 = below normal	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QUNCERT7	1 = configuration error	BOOL	0	O
QUNCERT8	1 = sensor calibration	BOOL	0	O
QUNCERT9	1 = simulated value	BOOL	0	O
Q0	Process value bit 0	BOOL	0	O
Q1 .... Q7	Process value bit 1 ... 7	BOOL	0	O
SIM_I	Simulation value	BYTE	0	I
SIM_ON	1 = activate simulation	BOOL	0	I
STATUS	Process value status	BYTE	0	O
SUBS_I	Substitute value	BYTE	0	I
SUBS_ON	1 = substitution active	BOOL	0	I

4.18 FF_A_DO: Output of digital values

4.18.1 Description of FF_A_DO

Object name (type + number)

FB 413

- FF_A_DO block I/Os (Page 375)

Area of application

Block FF_A_DO (cyclic service) is used for the output of digital values (SP or RCAS_IN, max. 8) via an FF field device.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32) and the restart OB 100.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The MODE input is interconnected with the OMODE_00 output of the MOD_PAL0 block or MOD_PAX0 block.

Function and operating principle

Depending on the selection (configuration of the input MODE_LW) of the user data configuration the block FF_A_DO reads the user data from the (partition) process image and writes it in the (partition) process image.

The coding of the set user-data configuration must be set for FF devices in the input variable MODE_LW according to the MODE_LW settings for FF devices (Page 611). This specifies which variables are to be read and written in the process image (partition).

The block writes the setpoint (O_SP_W) with quality code (QC_SP_W) to the process image (partition). The Quality Code contains the information on the setpoint status. The coding of the quality codes is described in PROFIBUS 3.0 "General Requirements". As an option, the setpoint in the RCAS (Remote Cascade) status (RCAS_IN) can be transferred with the quality code to the process image (partition) in the same cycle.

Linking in ASSET

The block FF_A_DO can be linked in ASSET.

For further information, refer to the section:

Integrating FF devices in ASSET (Page 612)

Quality code

The program generates a quality code (QUALITY output) of the resultant value which may assume the states shown below:

State	Quality Code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, due to process	16#28
Uncertain, due to device	16#68
Uncertain, due to process	16#78
Last valid value	16#44
Substitute Value	16#48
Range overshoot	16#54
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error or simulation and from a quality code, that comes directly from the device (parameter QC_READBACK_W, QC_RCAS_OUT_W).

The quality code that comes directly from the device can have values from 16#00 – 16#FF. The quality codes 16#84-16#87 and 16#90 – 16#93 of the FF field device are analyzed like the quality codes 16#80 – 16#83. As an option, the data of the FF field device, the status of the valve (READBACK), the process value of the valve setting in the state RCAS (RCAS_OUT), and the detailed device information (CHECKBACK) are read cyclically from the process image (partition). The device information is available bit-granular at the block output.

To improve the interconnectability, important detailed information is provided from the read status bytes in the form of Boolean (BOOL) values on the output interface. These conform to the bit combinations specified in PROFIBUS "General Requirements". If READBACK and RCAS_OUT exist simultaneously, the detailed information is derived from the READBACK status byte.

If a higher-level error has occurred (QMOD_ERR = TRUE), the data continue to be written to the process image (partition) and no data are read from the process image (partition). As long as the higher-priority error is active the last values are held with QBAD = TRUE.

Note

The default value of the setpoint status byte (ST_SP) and of the reference variable (ST_RCAS_IN) is zero. The setpoint and the reference variable become active in the PROFIBUS PA field device only if you set the corresponding status byte to 16#80.

Configuring the MODE_LW input

In addition to the MODE input the MODE_LW (mode low word) input must be configured according to the parameters used:

Parameters used	MODE_LW
SP_D	16#0400
SP_D, READBACK_D	16#0409
SP_D, CHECKBACK_D	16#040A
SP_D, READBACK_D, CHECK_BACK_D	16#040B
RCAS_IN_D, RCAS_OUT_D	16#050C
RCAS_IN_D, RCAS_OUT_D, CHECK_BACK_D	16#050D
SP_D, RCAS_IN_D , READBACK_D, RCAS_OUT_D, CHECK_BACK_D	16#060E

Addressing

You must interconnect each used connection of the block with the icons configured in the HW Config:

I/O	Data type
I_READBACK_W	WORD
I_RCAS_OUT_W	WORD
O_SP_W	WORD
O_RCAS_IN_W	WORD

In addition the quality code of the used I/Os must be interconnected with the icon configured in the HW Config:

I/O	Data type
CHECK_0_W	WORD
CHECK_1_W	WORD
CHECK_2_W	WORD
QC_READBACK_W	WORD
QC_RCAS_OUT_W	WORD
QC_SP_W	WORD
QC_RCAS_IN_W	WORD

Simulation

If the input parameter SIM_ON = TRUE, the value of the input parameter SIM_SP (and option SIM_RCAS_IN, if selected) is output with the quality code (QUALITY) = 16#60. Simulation takes highest priority. QBAD is set to FALSE. If the block is in the simulation status, QSIM = TRUE is set.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

The block will run through one time in OB 100 at system start. The output and in/out parameters are calculated.

Time response

Not available

Message response

Not available

Operating and monitoring

The block has no faceplate.

4.18.2 I/Os of FF_A_DO

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
CHECK_0_W	Additional information field device	WORD	0	I
CHECK_1_W	Additional information field device	WORD	0	I
CHECK_2_W	Additional information field device	WORD	0	I
I_RCAS_OUT_W	Function block setpoint	WORD	0	I
I_READBACK_W	Process value (READBACK). (symbol)	WORD	0	I
MODE	Value status	DWORD	0	I
MODE_LW	Mode	WORD	0	I
O_RCAS_IN_W	Target setpoint provided by a monitoring host to the analog control or output block (symbol)	WORD	0	O
O_SP_W	Setpoint (symbol)	WORD	0	O
QA_1	1 = alarm/warning 1	BOOL	0	O
QA_2	1 = alarm/warning 2	BOOL	0	O
QBAD	1 = group event QBAD_X	BOOL	0	O
QBAD_0	1 = non-specific	BOOL	0	O
QBAD_1	1 = configuration error	BOOL	0	O
QBAD_2	1 = not connected	BOOL	0	O
QBAD_3	1 = device failure	BOOL	0	O
QBAD_4	1 = sensor failure	BOOL	0	O
QBAD_5	1 = no communication (last usable value)	BOOL	0	O
QBAD_6	1 = no communication (no usable value)	BOOL	0	O
QBAD_7	1 = out of service	BOOL	0	O
QCB_0	1 = field device in fail-safe position	BOOL	0	O
QCB_1	1 = request for manual mode at device	BOOL	0	O
QCB_2	1 = field device in manual mode	BOOL	0	O
QCB_3	1 = emergency control active	BOOL	0	O
QCB_4	1 = the actuator left the end position it had already reached	BOOL	0	O
QCB_5	1 = valve connection break	BOOL	0	O
QCB_6	1 = indicates a short-circuit at the valve connection	BOOL	0	O
QCB_7	Not used	BOOL	0	O
QCB_8	1 = actuator is opening	BOOL	0	O
QCB_9	1 = actuator is closing	BOOL	0	O
QCB_10	1 = alarm generated by a change in the static data of FB and TB	BOOL	0	O
QCB_11	1 = simulation of process values is enabled	BOOL	0	O
QCB_12	Not used	BOOL	0	O
QCB_13	1 = internal control loop interrupted	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QCB_14	1 = valve inactive (status OUT_D bad)	BOOL	0	O
QCB_15	1 = device under self test	BOOL	0	O
QCB_16	1 = valve stroke limit is exceeded	BOOL	0	O
QCB_17	1 = break time exceeded when changing from OPEN to CLOSE	BOOL	0	O
QCB_18	1 = break time exceeded when changing from CLOSE to OPEN	BOOL	0	O
QCB_19	1 = error occurred in the internal cycle test	BOOL	0	O
QCB_20	1 = timeout during the transition from OPEN to CLOSE	BOOL	0	O
QCB_21	1 = timeout during the transition from CLOSE to OPEN	BOOL	0	O
QCB_22	1 = valve blocked mechanically	BOOL	0	O
QCB_23	Not used	BOOL	0	O
QCONST	1 = constant	BOOL	1	O
QC_RCAS_IN	Quality code target setpoint	BYTE	0	O
QC_RCAS_IN_W	Quality code target setpoint (symbol)	WORD	0	O
QC_RCAS_OUT_W	Quality-code function-block setpoint (symbol)	WORD	0	I
QC_READBACK_O	Quality-code function-block setpoint output	BYTE	0	O
QC_READBACK_W	Quality-code function-block setpoint (symbol)	WORD	0	I
QC_SP	Quality code setpoint	BYTE	0	O
QC_SP_W	Quality code setpoint (symbol)	WORD	0	O
QERR	1 = output error (inverted value of ENO)	BOOL	1	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QNONCAS0	1 = OK (non-cascade)	BOOL	0	O
QNONCAS1	1 = update event	BOOL	0	O
QNONCAS4	1 = unacknowledged updated event	BOOL	0	O
QNONCAS7	1 = initiate fail-safe	BOOL	0	O
QNONCAS8	1 = maintenance required	BOOL	0	O
QNONCAS9	1 = function test/local overwrite; usable value	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QUNCERT	1 = group event QUNCERTx	BOOL	0	O
QUNCERT0	1 = non-specific	BOOL	0	O
QUNCERT1	1 = last usable value	BOOL	0	O
QUNCERT2	1 = set of substitute values	BOOL	0	O
QUNCERT3	1 = initial value	BOOL	0	O
QUNCERT4	1 = imprecise sensor conversion	BOOL	0	O
QUNCERT5	1 = engineering unit violation (unit not in the valid range)	BOOL	0	O
QUNCERT6	1 = below normal	BOOL	0	O
QUNCERT7	1 = configuration error	BOOL	0	O
QUNCERT8	1 = sensor calibration	BOOL	0	O
QUNCERT9	1 = simulated value	BOOL	0	O
RCAS_IN	Target setpoint	BYTE	0	I
RCAS_OUT	Function block setpoint	BYTE	0	O
READBACK	Process value (READBACK)	BYTE	0	O
SIM_ON	1 = activate simulation	BOOL	0	I

I/O (parameter)	Meaning	Data type	Default	Type
SIM_RCAS_IN	Simulation target setpoint	BYTE	0	I
SIM_READBACK	Simulation of the current position of the final control element during runtime (between OPEN and CLOSE positions) in PV units.	BYTE	0	I
SIM_SP	Simulation value	BYTE	0	I
SP	Setpoint	BYTE	0	I
ST_READBACK	Function-block setpoint status	BYTE	0	O
ST_RCAS_IN	RCAS_IN status	BYTE	0	I
ST_RCAS_OUT	Function-block setpoint status	BYTE	0	O
ST_SP	Setpoint status	BYTE	0	I

4.19 MSG_TS: Generation of time-stamped process values

4.19.1 Description of MSG_TS

Object name (type + number)

FB131

- MSG_TS block I/Os (Page 381)

Function

The MSG_TS message block is used to generate time-stamped process messages. It forms the interface between the outputs of the IMDRV_TS block and the ALARM_8P block, whose time stamps for its 8 messages are entered in the 1st associated value in an ARRAY of BYTE.

How it works

The block has inputs (VALUE_xx) whose information is used by the "Generate module drivers" function to interconnect the input structures/parameters TS_xx, TS_C_xx with the output structures/parameters TS_xxx, TS_C_xxx of IMDRV_TS. The information of the individual process alarms with time stamp (TS_xx) is assigned to the corresponding message number (EV_IDxx) and channel number (1 to 16) of the two ALARM_8P blocks. OR_32_TS is always interconnected between MSG_TS and IMDRV_TS.

ALARM_8P with EV_ID_01:

- SIG_1 = TS_00.MSG_SIG
- SIG_2 = TS_01.MSG_SIG

  ....
- SIG_8 = TS_07.MSG_SIG

The time stamps from TS_00.TS to TS_07.TS are entered in an ARRAY of BYTE at the first associated value SD_1.

ALARM_8P with EV_ID_02:

- SIG_1 = TS_08.MSG_SIG
- SIG_2 = TS_09.MSG_SIG

  ....
- SIG_8 = TS_15.MSG_SIG

The time stamps from TS_08.TS to TS_15.TS are entered in an ARRAY of BYTE at the first associated value SD_1.

After all the messages have been assigned, the block calls the ALARM_8P blocks in OB 1 and transfers the new messages to the OS. Errors that may occur during data communication between the block and the interface module (IM) are reported in IMDRV_TS with ALARM_8P (for example, an I/O access error). The feedback from the ALARM_8P blocks (STAT_xx, M_ACK_xx) is available at the block output. If STAT_xx = 11 (the previous job is not yet completed), it calls ALARM_8P again in the next cycle.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32) and OB1.

Use in CFC

The symbol generated with HW Config (symbol table) for the digital input channels must be interconnected with the VALUE inputs.

Based on the settings in HW Config that decide which channels will be time stamped and the address information in the VALUE_xx input, the "Generate module drivers" function interconnects the I/Os with the relevant ones of the driver block IMDRV_TS.

OR_32_TS is always interconnected between MSG_TS and IMDRV_TS.

Message response

The block uses two ALARM_8P blocks, which are called as multiple instances. The 8 time-stamp values per ALARM_8P call are transferred by means of an ARRAY [0..65] of BYTE with the 1st associated value. The structure of the ARRAY is as follows:

BYTE 0	Format identifier of the following date/time stamp Always = 0
BYTE 1	Format identifier of the following date/time stamp Always = 1 (ISP format)
BYTE 2 - 9:	Date/time stamp for signal_1
BYTE 10 - 17:	Date/time stamp for signal_2
. .	. .
BYTE 58 - 65:	Date/time stamp for signal_8

The format identifier of bytes 0 - 1 specifies the bit coding of the time stamp structure (8 bytes are assigned to one time stamp value). The ISP format is always supported.

Time stamp in ISP format

Complete time to ISP conventions (time since 1.1.1900; 00:00 h). Due to the 4 bytes for the seconds, the time expired since 1.1.1900; 0:00 h can be expressed in seconds.

Byte	Content	Area
0 - 3	Seconds as of 1.1.1900 0:00.00,000	Corresponding to 1.1.1900 ... 6.2.2036
4 - 7	Fractions of seconds in multiples of 1/2 ^32 s	0 .. <1
The time conversion is valid in the date range covered by the time formats together, in other words from 1.1.1990 up to and including 6.2.2036 (Feb 6th).		

The driver block outputs the time stamps provided by the IM in ISP format and without any changes.

Additional information

For more information, refer to the section:

Message texts of MSG_TS (Page 382)

4.19.2 I/Os of MSG_TS

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
EV_ID_01	Message ID 1	DWORD	16#20	I
EV_ID_02	Message ID 2	DWORD	16#21	I
M_ACK_01	Message acknowledgment 1	WORD	0	O
M_ACK_02	Message acknowledgment 2	WORD	0	O
Q_ERR_01	Message error 1	BOOL	0	O
Q_ERR_02	Message error 2	BOOL	0	O
QBAD	1 = invalid process value	BOOL	0	O
QERR	1 = processing error	BOOL	1	O
STAT_01	Status output 1	WORD	0	O
STAT_02	Status output 2	WORD	0	O
TS_xx	Time stamp (xx = 00 – 15) Byte 0: Bit 0: Message signal state (MsgSig) Bit 1: Edge change information (TriInf) Bit 2: Handshake (Hd) Byte 1: Quality code of the time stamp (ST) DWORD TS0: Date/time stamp in ISP format (seconds) DWORD TS1: Date/time stamp in ISP format (fractions of seconds)	STRUCT		I
TS_C_xx	TS communication (xx = 00 - 15) Bit 0: Acknowledgment of transfer (HS) Bit 1: Interconnection check (LI)	BYTE	0	IO
VALUE_xx	Input value (xx = 00 – 15)	BOOL	0	I

4.19.3 Message texts of MSG_TS

Assignment of message text and message class

You will find more information in Message classes

Message block	Message no.	Default message text	Message class
EV_ID_01 (ALARM_8P)			
	1	TEXT S_CH_00	S
	2	TEXT S_CH_01	S
	3	TEXT S_CH_02	S
	4	TEXT S_CH_03	S
	5	TEXT S_CH_04	S
	6	TEXT S_CH_05	S
	7	TEXT S_CH_06	S
	8	TEXT S_CH_07	S
EV_ID_02 (ALARM_8P)			
	1	TEXT S_CH_08	S
	2	TEXT S_CH_09	S
	3	TEXT S_CH_10	S
	4	TEXT S_CH_11	S
	5	TEXT S_CH_12	S
	6	TEXT S_CH_13	S
	7	TEXT S_CH_14	S
	8	TEXT S_CH_15	S

4.20 PA_AI: PROFIBUS PA Analog-Value Input

4.20.1 Description of PA_AI

Object name (type + number)

FB101

- PA_AI block I/Os (Page 387)

Area of application

The PA_AI block processes (cyclic service) the "Transmitter" PA profile of a PROFIBUS 3.0 class A and B PA field device or a primary or secondary variable of a HART field device.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32).

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The icon for the quality code of the analog input channel is interconnected with input QC.
- The MODE input is interconnected with the corresponding output OMODE_xx of the PADP_L0x block.

Function and operating principle

The PA_AI block reads the process value with status byte (quality code) of the PROFIBUS or HART field device (structure in accordance with the analog input of the PA profiles) cyclically from the process image (partition). The process value is available as a physical unit. The status byte (STATUS) contains information about the status of the PROFIBUS field device.

Quality code

The program generates a quality code (QUALITY output) of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, process related	16#28
Uncertain, device related	16#68
Uncertain, process related	16#78
Last valid value	16#44
Substitute value	16#48
Range overshoot	16#54
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

In addition to the status byte and to improve interconnectability, the output interface provides further Boolean (BOOL) values, which contain important details. They conform to bit combinations specifications in PROFIBUS PA 3.0 "General Requirements". Device-specific control system messages are generated via ALARM_8P on the basis of the QC.

The block recognizes a higher-level error, for example, failure of a DP/PA link, via the MODE input parameter.

- If the MODE high byte = 16#80, then the values in the process image (partition) are valid.
- If the MODE high byte = 16#40 (value status = higher-level error), the analog value is treated as invalid.

The operating mode set in the low word of the MODE input parameter will be ignored.

Addressing

The symbol of the analog input channel is generated and entered in the symbol table via HW Config. You must interconnect it with the VALUE input. The CFC function "Generate module drivers" interconnects the icon for the quality code of the analog input channel with the input parameter QC.

Simulation

When input parameter SIM_ON = TRUE, the value of the input parameter SIM_V is output with quality code (QUALITY =) 16#60. Simulation takes highest priority. QBAD is always set to FALSE. If the block is in the simulation state, QSIM = TRUE is set.

Note

Remember that the simulation value is always output in simulation mode regardless of whether one of the parameters LAST_ON (substitute value) or SUBS_ON (last valid value).

Substitute value

If input parameter SUBS_ON = TRUE, the value of input parameter SUBS_V is output as a value if the values are invalid. The quality code (QUALITY) is set to 16#48 and QBAD = 1.

Hold last value

If input parameter LAST_ON = TRUE, the last valid output value is output if the process values are invalid. The quality code (QUALITY) is set to 16#44 and QBAD = 1.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Value limiting

You can limit the maximum low and high range of process values of the process image (partition).

If the switch LIMIT_ON = TRUE, the process values (V) are limited:

- To V_HL, if V > V_HL
- To LL_V, if V < V_LL

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.20.2 I/Os of PA_AI

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You can find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
LAST_ON	1 = last valid value: Enable injection	BOOL	0	IO
LIMIT_ON	1 = limit input value ON	BOOL	0	I
LL_V	Value if V < V_LL	REAL	0	I
MODE	Value status and mode	DWORD	0	I
QA_1	1 = alarm/warning 1	BOOL	0	O
QA_2	1 = alarm/warning 2	BOOL	0	O
QBAD	1 = group event QBAD_X	BOOL	0	O
QBAD_0	1 = non-specific	BOOL	0	O
QBAD_1	1 = configuration error	BOOL	0	O
QBAD_2	1 = not connected	BOOL	0	O
QBAD_3	1 = device failure	BOOL	0	O
QBAD_4	1 = sensor failure	BOOL	0	O
QBAD_5	1 = no communication (last usable value)	BOOL	0	O
QBAD_6	1 = no communication (last usable value)	BOOL	0	O
QBAD_7	1 = out of service	BOOL	0	O
QBAD_HL	1 = high limit of physical range of sensor has been reached	BOOL	0	O
QBAD_LL	1 = low limit of physical range of sensor has been reached	BOOL	0	O
QC	Status of the input process value	BYTE	0	I
QCASCAD0	1 = OK (cascade)	BOOL	0	O
QCASCAD1	1 = initialization acknowledged	BOOL	0	O
QCASCAD2	1 = initialization request	BOOL	0	O
QCASCAD3	1 = not required	BOOL	0	O
QCASCAD4	1 = reserved	BOOL	0	O
QCASCAD5	1 = do not select	BOOL	0	O
QCASCAD6	1 = local overwrite	BOOL	0	O
QCASCAD7	1 = reserved	BOOL	0	O
QCASCAD8	1 = initiate fail-safe	BOOL	0	O
QCONST	1 = constant	BOOL	0	O
QERR	1 = output error (inverted value of ENO)	BOOL	1	O
QLAST	1 = last valid value: Injection active	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QOUT_HHL	1 = active critical alarm, high limit of OUT has been exceeded	BOOL	0	O
QOUT_HL	1 = active warning, high limit of OUT has been exceeded	BOOL	0	O
QOUT_LL	1 = active warning, low limit of OUT has been undershot	BOOL	0	O
QOUT_LLL	1 = active critical alarm, low limit of OUT has been undershot	BOOL	0	O
QNONCAS0	1 = OK (non-cascade)	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QNONCAS1	1 = update event	BOOL	0	O
QNONCAS2	1 = active warning (priority <8)	BOOL	0	O
QNONCAS3	1 = active critical alarm (priority >8)	BOOL	0	O
QNONCAS4	1 = unacknowledged updated event	BOOL	0	O
QNONCAS5	1 = unacknowledged warning	BOOL	0	O
QNONCAS6	1 = unacknowledged critical alarm	BOOL	0	O
QNONCAS7	1 = initiate fail-safe	BOOL	0	O
QNONCAS8	1 = maintenance required	BOOL	0	O
QNONCAS9	1 = function test/local overwrite; usable value	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QSUBS	1 = substitution active	BOOL	0	O
QUALITY	Quality-code process value	BYTE	0	O
QUNCERT	1 = group message QUNCERTx	BOOL	0	O
QUNCERT0	1 = non-specific	BOOL	0	O
QUNCERT1	1 = last usable value	BOOL	0	O
QUNCERT2	1 = set of substitute values	BOOL	0	O
QUNCERT3	1 = initial value	BOOL	0	O
QUNCERT4	1 = unacknowledged updated event	BOOL	0	O
QUNCERT5	1 = engineering unit violation (unit not in the valid range)	BOOL	0	O
QUNCERT6	1 = below normal	BOOL	0	O
QUNCERT7	1 = configuration error	BOOL	0	O
QUNCERT8	1 = sensor calibration	BOOL	0	O
QUNCERT9	1 = simulated value	BOOL	0	O
SIM_ON	1 = activate simulation	BOOL	0	I
SIM_V	Simulation value	REAL	0	I
STATUS	Process-value status	BYTE	0	O
SUBS_ON	1 = enable substitution	BOOL	0	I
SUBS_V	Substitute value	REAL	0	I
V	Process value	REAL	0	O
V_HL	High limit input value	REAL	0	I
V_LL	Low limit input value	REAL	0	I
VALUE	Process image input value	REAL	0	I

4.21 PA_AO: PROFIBUS PA analog value output

4.21.1 Description of PA_AO

Object name (type + number)

FB 103

- PA_AO block I/Os (Page 393)

Area of application

Block PA_AO processes (cyclic service) the "Actuator" PA profile of a PROFIBUS PA 3.0 class A and B PA field device.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32) and the restart OB 100.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The icon for the quality code of the analog output channel O_SP is interconnected with the output QC_SP and other selected options.
- The MODE input is interconnected with the corresponding output OMODE_xx of the PADP_L0x block.

Function and operating principle

The block reads the user data from the process image (partition) and writes them to the process image (partition), depending on the selection (with HW Config or SIMATIC PDM) of the user-data configuration of the "Analog Output" PA profile in accordance with PROFIBUS PA 3.0.

The low word of the MODE input variable contains the coding of the user data configuration set in the PROFIBUS PA 3.0 "Analog Output" profile. This specifies which tags of the process image (partition) are to be read and written.

The block cyclically writes the setpoint (SP) with Quality Code (configuration of the setpoints and process values in accordance with the Analog Output of the PA profiles, REAL with 1 byte Quality Code) into the process image (partition). The PA profile contains the setpoint and other analog values as physical units. The quality code contains information on the setpoint state. The coding of the quality code is described in PROFIBUS 3.0 "General Requirements". The reference variable (RCAS_IN) can be transferred with the quality code to the process image (partition) in the same cycle.

Quality code

The program generates a quality code (QUALITY output) of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, process related	16#28
Uncertain, device related	16#68
Uncertain, due to process	16#78
Last valid value	16#44
Substitute value	16#48
Range overshoot	16#54
Range overshoot low	16#55
Range overshoot high	16#56
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error or simulation and from a quality code that comes directly from the device (parameter QC_X, X = parameter name).

The QC returned directly from the device may assume values from 00 - FF in accordance with PROFIBUS requirements.

The data of the PROFIBUS PA field device, as well as the process value (READBACK) and the discrete position feedback (POS_D) are read cyclically from the process image (partition). Optionally you can also read the active reference variable (RCAS_OUT) and the detailed device information (CHECKBACK). The device information is available per bit at the block output. The data is read from the process image (partition). In order to improve the interconnectability, important detailed information is provided by the status bytes read (ST_READBACK) or ST_RCAS_OUT (if READBACK is not available) as Boolean (BOOL) values at the output interface. These conform to the bit combinations specified in PROFIBUS 3.0 "General Requirements".

If a higher-level error has occurred (QMOD_ERR = TRUE), the data continue to be written to the process image (partition) and no data are read from the process image (partition). As long as the higher-level error is active the last values are retained with QBAD = TRUE.

Note

The status byte (ST_RCAS_IN) is pre-defined with zero. The reference variable only becomes active in the PROFIBUS field device if you set the quality code to 16#80.

Addressing

You have to interconnect one of the icons configured using HW Config (for example, SP) for the analog output channel (PROFIBUS PA 3.0 profile "Analog Output") with the corresponding I/O:

I/O	Data type
I_READBACK	REAL
I_RCAS_OUT	REAL
I_POS_D	BYTE
O_SP	REAL
O_RCAS_IN	REAL

In CFC, the "Generate module drivers" function automatically interconnects the icon for the corresponding quality code of the I/O and the remaining configured icons of the analog output channel (with quality code).

Simulation

If the input parameter SIM_ON = TRUE, the value of the input SIM_SP (and the option SIM_RCAS_IN) is output with Quality Code (QUALITY) = 16#60. Simulation takes highest priority. QBAD is set to FALSE. If the block is in the simulation state, QSIM = TRUE is set.

Value limiting

You can limit the maximum low and high range of process values of the process image (partition). The process values "V" (READBACK and RCAS_OUT) are limited if LIMIT_ON = TRUE:

- To V_HL, if V > V_HL.
- To LL_V, if V < V_LL.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

The block will run through one time in OB 100 at system start. The output and in/out parameters are calculated.

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.21.2 I/Os of PA_AO

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
CHECK_0	Additional information field device	BYTE	0	I
CHECK_1	Additional information field device	BYTE	0	I
CHECK_2	Additional information field device	BYTE	0	I
I_POS_D	Current position of the valve (discrete) (symbol)	BYTE	0	I
I_RCAS_OUT	Function block setpoint	REAL	0	I
I_READBACK	Current position of the final control element during runtime (between OPEN and CLOSE positions) in PV units (symbol)	REAL	0	I
LIMIT_ON	1 = limit input value ON	BOOL	0	I
LL_V	High limit input value. Set value if V < V_LL	REAL		I
MODE	MODE input parameter	DWORD	0	I
O_RCAS_IN	Target setpoint provided by a monitoring host to the analog control or output block (symbol)	REAL	0	O
O_SP	Setpoint (symbol)	REAL	0	O
POS_D	Current position of the valve (discrete). The coding of the POS bytes is as follows: 0 = not initialized, 1 = closed, 2 = open, 3 = intermediate	BYTE	0	O
QA_1	1 = alarm/warning 1	BOOL	0	O
QA_2	1 = alarm/warning 2	BOOL	0	O
QBAD	1 = group event QBAD_X	BOOL	0	O
QBAD_0	1 = non-specific	BOOL	0	O
QBAD_1	1 = configuration error	BOOL	0	O
QBAD_2	1 = not connected	BOOL	0	O
QBAD_3	1 = device failure	BOOL	0	O
QBAD_4	1 = sensor failure	BOOL	0	O
QBAD_5	1 = no communication (last usable value)	BOOL	0	O
QBAD_6	1 = no communication (no usable value)	BOOL	0	O
QBAD_7	1 = out of service	BOOL	0	O
QCASCAD0	1 = OK (cascade)	BOOL	0	O
QCASCAD1	1 = initialization acknowledged	BOOL	0	O
QCASCAD2	1 = initialization request	BOOL	0	O
QCASCAD3	1 = not required	BOOL	0	O
QCASCAD4	1 = reserved	BOOL	0	O
QCASCAD5	1 = do not select	BOOL	0	O
QCASCAD6	1 = local overwrite	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QCASCAD7	1 = reserved	BOOL	0	O
QCASCAD8	1 = initiate fail-safe	BOOL	0	O
QNONCAS9	1 = function test/local overwrite; usable value	BOOL	0	O
QCB_0	1 = field device in fail-safe position active	BOOL	0	O
QCB_1	1 = request for manual mode	BOOL	0	O
QCB_2	1 = field device in manual mode, LOCKED OUT switch is active	BOOL	0	O
QCB_3	1 = emergency control active	BOOL	0	O
QCB_4	1 = actual position feedback differs from expected position	BOOL	0	O
QCB_5	1 = torque limit in OPEN direction has been exceeded	BOOL	0	O
QCB_6	1 = torque limit in CLOSE direction has been exceeded	BOOL	0	O
QCB_7	1 = status of travel monitoring equipment; if YES, travel time for actuator has been exceeded	BOOL	0	O
QCB_8	1 = actuator is opening	BOOL	0	O
QCB_9	1 = actuator is closing	BOOL	0	O
QCB_10	1 = the interrupt generated by any change to the static data (function and transducer block)	BOOL	0	O
QCB_11	1 = simulation of process values is enabled	BOOL	0	O
QCB_12	Not used	BOOL	0	O
QCB_13	1 = internal control loop interrupted	BOOL	0	O
QCB_14	1 = positioner inactive (OUT status = BAD)	BOOL	0	O
QCB_15	1 = device under self test	BOOL	0	O
QCB_15	1 = device under self test (MODE = out of service)	BOOL	0	O
QCB_16	1 = valve travel limit has been exceeded	BOOL	0	O
QCB_17	1 = additional input (for diagnostics, for example) is activated	BOOL	0	O
QCONST	1 = constant	BOOL	0	O
QC_POS_D	Quality code POS	BYTE	0	I
QC_POS_D_O	Quality code POS output	BYTE	0	O
QC_RCAS_IN	Quality code RCAS_IN (symbol)	BYTE	0	O
QC_RCAS_OUT	Quality-code function-block setpoint	BYTE	0	I
QC_READBACK	Quality-code function-block setpoint (symbol)	BYTE	0	I
QC_READBACK_O	Quality-code function-block setpoint output	BYTE	0	O
QC_SP	Quality code setpoint (symbol)	BYTE	0	O
QERR	1 = output error (inverted value of ENO)	BOOL	1	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QNONCAS0	1 = OK (non-cascade)	BOOL	0	O
QNONCAS1	1 = update event	BOOL	0	O
QNONCAS2	1 = active warning (priority <8)	BOOL	0	O
QNONCAS3	1 = active critical alarm (priority >8)	BOOL	0	O
QNONCAS4	1 = unacknowledged updated event	BOOL	0	O
QNONCAS5	1 = unacknowledged warning	BOOL	0	O
QNONCAS6	1 = unacknowledged critical alarm	BOOL	0	O
QNONCAS7	1 = initiate fail-safe	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QNONCAS8	1 = maintenance required	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QUNCERT	1 = group event QUNCERTx	BOOL	0	O
QUNCERT0	1 = non-specific	BOOL	0	O
QUNCERT1	1 = last usable value	BOOL	0	O
QUNCERT2	1 = set of substitute values	BOOL	0	O
QUNCERT3	1 = initial value	BOOL	0	O
QUNCERT4	1 = imprecise sensor conversion	BOOL	0	O
QUNCERT5	1 = engineering unit violation (unit not in the valid range)	BOOL	0	O
QUNCERT6	1 = below normal	BOOL	0	O
QUNCERT7	1 = configuration error	BOOL	0	O
QUNCERT8	1 = sensor calibration	BOOL	0	O
QUNCERT9	1 = simulated value	BOOL	0	O
RCAS_IN	Target setpoint provided by a monitoring host to the analog control or output block	REAL	0	I
RCAS_OUT	Function block setpoint	REAL	0	O
READBACK	Current position of the final control element within the travel span (between OPEN and CLOSE positions) in PV units.	REAL	0	O
SIM_ON	1 = simulation active	BOOL	0	I
SIM_POS_D	Current position of the valve (discrete)	BYTE	0	I
SIM_RCAS_IN	Simulation RCAS_IN	REAL	0	I
SIM_READBACK	Simulation of the current position of the final control element during runtime (between OPEN and CLOSE positions) in PV units.	REAL	0	I
SIM_SP	Simulation setpoint	REAL	0	I
SP	Setpoint	REAL	0	I
ST_POS_D	Status POS	BYTE	0	O
ST_READBACK	Function-block readback-value status	BYTE	0	O
ST_RCAS_IN	RCAS_IN status	BYTE	0	I
ST_RCAS_OUT	Function-block setpoint status	BYTE	0	O
ST_SP	Setpoint status	BYTE	0	I
V_HL	High limit input value	REAL	0	I
V_LL	Low limit input value	REAL	0	I

4.22 PA_DI: PROFIBUS PA digital value input

4.22.1 Description of PA_DI

Object name (type + number)

FB104

- PA_DI block I/Os (Page 399)

Area of application

The PA_DI block (cyclic service) is used for reading in digital values (discrete input) via a PROFIBUS field device class A and B.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32).

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The icon for the quality code of the digital input channel is interconnected with input QC.
- The MODE input is interconnected with the corresponding output OMODE_xx of the PADP_L0x block.

Function and operating principle

The PA_DI block interfaces a PROFIBUS field device and the blocks of the SIMATIC PCS 7 libraries. It can also be interconnected with other SIMATIC S7 blocks. The block requires a PROFIBUS DP interface. This can either be integrated in the CPU or can be external DP interface (CP). The changeover from PROFIBUS PD to PROFIBUS PA is implemented over a SIMATIC segment coupler DP/PA link (or DP/PA coupler) or similar segment couplers that comply with the standards EN 50170, Vol.2, EN 61158-2, IEC 1158-2.

The block cyclically reads the process value (OUT_D) with status byte of the PROFIBUS PA field device (structure in accordance with the discrete input of the PA profiles). The status byte (STATUS) contains information about the status of the PROFIBUS field device. The process value and status bytes (2 bytes) are read directly and consistently as a WORD. The process value and important status byte information are made available bit-granular for better interconnectability on the output interface. These conform to the bit combinations specified in PROFIBUS "General Requirements".

Quality code

The program generates a quality code (QUALITY output) of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, process related	16#28
Uncertain, device related	16#68
Uncertain, process related	16#78
Last valid value	16#44
Substitute value	16#48
Range overshoot	16#54
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error, or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

The block recognizes a higher-level error, for example, failure of a DP/PA link, via the MODE input parameter.

- If the MODE high byte = 16#80, then the values in the process image (partition) are valid.
- If the MODE high byte = 16#40 (value status = higher-level error, QMOD_ERR = TRUE), the process value is treated as invalid.

The operating mode set in the low word of the MODE input parameter will be ignored.

Addressing

The symbol generated with HW Config in the symbol table for the digital input channel has to be interconnected to the I_OUT_D input. The CFC function "Generate module drivers" interconnects the icon for the quality code of the digital input channel with the input parameter QC.

Note

Remember that the simulation value is always output in simulation mode regardless of whether one of the parameters LAST_ON (substitute value) or SUBS_ON (last valid value).

Simulation

When input parameter SIM_ON = TRUE, the value of input SIM_I is output with quality code (QUALITY =) 16#60. Simulation takes highest priority. QBAD is always set to FALSE. If the block is in the simulation state, QSIM = TRUE is set.

Substitute value

If input parameter SUBS_ON = TRUE, the value of input parameter SUBS_I is output as a value if the process values are invalid. The quality code (QUALITY) is set to 16#48 and QBAD = 1.

Hold last value

If input parameter LAST_ON = TRUE, the last valid output value is output if the process values are invalid. and sets QBAD = 1 and the quality code to QUALITY = 16#44.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.22.2 I/Os of PA_DI

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You can find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
I_OUT_D	Process value of the process image	BYTE	0	I
LAST_ON	1 = last valid value: Enable injection	BOOL	0	IO
MODE	MODE input parameter	DWORD	0	I
OUT_D	Process value	BYTE	0	O
QA_1	1 = alarm/warning 1	BOOL	0	O
QA_2	1 = alarm/warning 2	BOOL	0	O
QBAD	1 = group event QBAD_X	BOOL	0	O
QBAD_0	1 = non-specific	BOOL	0	O
QBAD_1	1 = configuration error	BOOL	0	O
QBAD_2	1 = not connected	BOOL	0	O
QBAD_3	1 = device failure	BOOL	0	O
QBAD_4	1 = sensor failure	BOOL	0	O
QBAD_5	1 = no communication (last usable value)	BOOL	0	O
QBAD_6	1 = no communication (no usable value)	BOOL	0	O
QBAD_7	1 = out of service	BOOL	0	O
QC	Quality-code value of the process image	BYTE	0	I
QCONST	1 = constant	BOOL	0	O
QERR	1 = output error (inverted value of ENO)	BOOL	1	O
QLAST	1 = last valid value: Injection active	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QNONCAS0	1 = OK (non-cascade)	BOOL	0	O
QNONCAS1	1 = update event	BOOL	0	O
QNONCAS4	1 = unacknowledged updated event	BOOL	0	O
QNONCAS7	1 = initiate fail-safe	BOOL	0	O
QNONCAS8	1 = maintenance required	BOOL	0	O
QNONCAS9	1 = function test/local overwrite; usable value	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QSUBS	1 = substitution active	BOOL	0	O
QUALITY	Quality-code process value	BYTE	0	O
QUNCERT	1 = group event QUNCERTx	BOOL	0	O
QUNCERT0	1 = non-specific	BOOL	0	O
QUNCERT1	1 = last usable value	BOOL	0	O
QUNCERT2	1 = set of substitute values	BOOL	0	O
QUNCERT3	1 = initial value	BOOL	0	O
QUNCERT4	1 = imprecise sensor conversion	BOOL	0	O
QUNCERT5	1 = engineering unit violation (unit not in the valid range)	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QUNCERT6	1 = below normal	BOOL	0	O
QUNCERT7	1 = configuration error	BOOL	0	O
QUNCERT8	1 = sensor calibration	BOOL	0	O
QUNCERT9	1 = simulated value	BOOL	0	O
Q0	Process value bit 0	BOOL	0	O
Q1 .... Q7	Process value bit 1 ... bit 7	BOOL	0	O
SIM_I	Simulation value	BYTE	0	I
SIM_ON	1 = activate simulation	BOOL	0	I
STATUS	Process-value status	BYTE	0	O
SUBS_I	Substitute value	BYTE	0	I
SUBS_ON	1 = substitution active	BOOL	0	I

4.23 PA_DO: PROFIBUS PA digital value output

4.23.1 Description of PA_DO

Object name (type + number)

FB 105

- PA_DO block I/Os (Page 404)

Area of application

Block PA_DO (cyclic service) is used for the output of digital values (SP/RCAS_IN, max. 8) to a PROFIBUS field device class A and B.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32) and the restart OB 100.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The icon for the quality code of the digital output channel O_SP is interconnected with the output QC_SP and other selected options.
- The MODE input is interconnected with the corresponding output OMODE_xx of the PADP_L0x block.

Function and operating principle

Block PA_DO reads the user data from the process image (partition) and writes them to the process image (partition), depending on the selection (with HW Config or SIMATIC PDM) of the user-data configuration of the "Digital Output" PA profile in accordance with PROFIBUS 3.0.

The low word of the MODE input variable contains the coding of the user-data configuration set in the PROFIBUS 3.0 "Discrete Output" profile. This specifies which variables are to be read and written in the process image (partition).

The block writes the setpoint (SP) with quality code to the process image (partition) (structure of the setpoints and process values corresponds with the digital output of the PA profiles, 1 byte with 1 byte quality code). The quality code contains information on the setpoint state. The coding of the quality code is described in PROFIBUS 3.0 "General Requirements". As an option, the setpoint in the RCAS (remote cascade) state (RCAS_IN) can be transferred with the quality code to the process image (partition) in the same cycle.

Quality code

The program generates a quality code (QUALITY output) of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, process related	16#28
Uncertain, device related	16#68
Uncertain, process related	16#78
Last valid value	16#44
Substitute value	16#48
Range overshoot	16#54
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error, or simulation and from a quality code that comes directly from the device (parameter QC_X, X = parameter name).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

The data of the PROFIBUS field device (state of the READBACK valve), the process value of the valve setting in the state RCAS (RCAS_OUT), and the detailed device information (CHECKBACK) can be read cyclically from the process image (partition) as an option. The device information is available per bit at the block output.

To improve interconnectability, important detailed information is provided from the read status bytes in the form of Boolean (BOOL) values on the output interface. These conform to the bit combinations specified in PROFIBUS "General Requirements". If READBACK and RCAS_OUT exist simultaneously, the detailed information is derived from the ST_READBACK status byte.

If a higher-level error has occurred (QMOD_ERR = TRUE), the data continue to be written to the process image (partition) and no data are read from the process image (partition). As long as the higher-level error is active the last values are retained with QBAD = TRUE.

Note

The default value of the setpoint status byte (ST_SP) and of the reference variable (ST_RCAS_IN) is zero. The setpoint and the reference variable become active in the PROFIBUS field device only if you set the corresponding status byte to 16#80.

Addressing

You have to interconnect one of the icons configured using HW Config (for example, SP) for the digital output channel (PROFIBUS 3.0 "Analog Output" profile) with the corresponding I/O:

I/O	Data type
I_READBACK	BYTE
I_RCAS_OUT	BYTE
O_SP	BYTE
O_RCAS_IN	BYTE

In CFC the "Generate module drivers" function automatically interconnects the icon for the corresponding quality code (if it exists) of the I/O with the remaining configured icons of the digital output channel (with quality code).

Simulation

If the input parameter SIM_ON = TRUE, the value of the input parameter SIM_SP (and option SIM_RCAS_IN, if selected) is output with the quality code (QUALITY) = 16#60. Simulation takes highest priority. QBAD is set to FALSE. If the block is in the simulation state, QSIM = TRUE is set.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

The block will run through one time in OB 100 at system start. The output and in/out parameters are calculated.

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.23.2 I/Os of PA_DO

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
CHECK_0	Additional information field device	BYTE	0	I
CHECK_1	Additional information field device	BYTE	0	I
CHECK_2	Additional information field device	BYTE	0	I
I_RCAS_OUT	Function block setpoint	BYTE	0	I
I_READBACK	Process value (READBACK). (symbol)	BYTE	0	I
MODE	MODE input parameter	DWORD	0	I
O_RCAS_IN	Target setpoint provided by a monitoring host to the analog control or output block (symbol)	BYTE	0	O
O_SP	Setpoint (symbol)	BYTE	0	OI
QA_1	1 = alarm/warning 1	BOOL	0	O
QA_2	1 = alarm/warning 2	BOOL	0	O
QBAD	1 = group event QBAD_X	BOOL	0	O
QBAD_0	1 = non-specific	BOOL	0	O
QBAD_1	1 = configuration error	BOOL	0	O
QBAD_2	1 = not connected	BOOL	0	O
QBAD_3	1 = device failure	BOOL	0	O
QBAD_4	1 = sensor failure	BOOL	0	O
QBAD_5	1 = no communication (last usable value)	BOOL	0	O
QBAD_6	1 = no communication (no usable value)	BOOL	0	O
QBAD_7	1 = out of service	BOOL	0	O
QCONST	1 = constant	BOOL	0	O
QCB_0	1 = field device in fail-safe position	BOOL	0	O
QCB_1	1 = request for manual mode at device	BOOL	0	O
QCB_2	1 = field device in manual mode	BOOL	0	O
QCB_3	1 = emergency control active	BOOL	0	O
QCB_4	1 = the actuator left the end position it had already reached	BOOL	0	O
QCB_5	1 = valve connection break	BOOL	0	O
QCB_6	1 = indicates a short-circuit at the valve connection	BOOL	0	O
QCB_7	Not used	BOOL	0	O
QCB_8	1 = actuator is opening	BOOL	0	O
QCB_9	1 = actuator is closing	BOOL	0	O
QCB_10	1 = alarm generated by a change to the static data of FB and TB	BOOL	0	O
QCB_11	1 = simulation of process values is enabled	BOOL	0	O
QCB_12	Not used	BOOL	0	O
QCB_13	1 = internal control loop interrupted	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QCB_14	1 = valve inactive (status OUT_D bad)	BOOL	0	O
QCB_15	1 = device under self test	BOOL	0	O
QCB_16	1 = valve travel limit has been exceeded	BOOL	0	O
QCB_17	1 = break time exceeded when changing from OPEN to CLOSE	BOOL	0	O
QCB_18	1 = break time exceeded when changing from CLOSE to OPEN	BOOL	0	O
QCB_19	1 = error occurred in the internal cycle test	BOOL	0	O
QCB_20	1 = timeout during the transition from OPEN to CLOSE	BOOL	0	O
QCB_21	1 = timeout during the transition from CLOSE to OPEN	BOOL	0	O
QCB_22	1 = valve blocked mechanically	BOOL	0	O
QCB_23	Not used	BOOL	0	O
QC_RCAS_IN	Quality-code target setpoint (symbol)	BYTE	0	O
QC_RCAS_OUT	Quality-code function-block setpoint (symbol)	BYTE	0	I
QC_READBACK	Quality-code function-block setpoint (symbol)	BYTE	0	I
QC_READBACK_O	Quality-code function-block setpoint output	BYTE	0	O
QC_SP	Quality-code setpoint (symbol)	BYTE	0	O
QERR	1 = output error (inverted value of ENO)	BOOL	1	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QNONCAS0	1 = OK (non-cascade)	BOOL	0	O
QNONCAS1	1 = update event	BOOL	0	O
QNONCAS4	1 = unacknowledged updated event	BOOL	0	O
QNONCAS7	1 = initiate fail-safe	BOOL	0	O
QNONCAS8	1 = maintenance required	BOOL	0	O
QNONCAS9	1 = function test/local overwrite; usable value	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QSUBS	1 = substitution active	BOOL	0	O
QUNCERT	1 = group event QUNCERTx	BOOL	0	O
QUNCERT0	1 = non-specific	BOOL	0	O
QUNCERT1	1 = last usable value	BOOL	0	O
QUNCERT2	1 = set of substitute values	BOOL	0	O
QUNCERT3	1 = initial value	BOOL	0	O
QUNCERT4	1 = imprecise sensor conversion	BOOL	0	O
QUNCERT5	1 = engineering unit violation (unit not in the valid range)	BOOL	0	O
QUNCERT6	1 = below normal	BOOL	0	O
QUNCERT7	1 = configuration error	BOOL	0	O
QUNCERT8	1 = sensor calibration	BOOL	0	O
QUNCERT9	1 = simulated value	BOOL	0	O
RCAS_IN	Target setpoint	BYTE	0	I
RCAS_OUT	Function block setpoint	BYTE	0	O
READBACK	Process value (READBACK)	BYTE	0	O
SIM_ON	1 = activate simulation	BOOL	0	I
SIM_RCAS_IN	Simulation target setpoint	BYTE	0	I

I/O (parameter)	Meaning	Data type	Default	Type
SIM_READBACK	Simulation of the current position of the final control element during runtime (between OPEN and CLOSE positions) in PV units	REAL	0	I
SIM_SP	Simulation value	BYTE	0	I
SP	Setpoint	BYTE	0	I
ST_READBACK	Function-block setpoint status	BYTE	0	O
ST_RCAS_IN	RCAS_IN status	BYTE	0	I
ST_RCAS_OUT	Function-block setpoint status	BYTE	0	O
ST_SP	Setpoint status	BYTE	0	I
V_HL	High limit input value	REAL	0	I
V_LL	Low limit input value	REAL	0	I

4.24 PA_TOT: PROFIBUS PA Totalizer

4.24.1 Description of PA_TOT

Object name (type + number)

FB 102

- PA_TOT block I/Os (Page 410)

Area of application

Block PA_TOT processes the cyclic parameters of the "Totalizer" PA profile of a PA field device, in accordance with PROFIBUS 3.0 class A and B.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32).

Use in CFC

If the CFC function "**Generate Module Drivers**" is used, the following actions are executed automatically:

- The symbol for the quality code of the analog input channel is interconnected with the input QC and other selected options.
- The MODE input is interconnected with the corresponding output OMODE_xx of the PADP_L0x block.

Function and operating principle

Block PA_TOT reads the process value (TOTAL) with the status byte (quality code) of the PROFIBUS field device (structure corresponds with the totalizer of the PA profiles) cyclically from the process image (partition). The process value is available as a physical unit. The status byte (STATUS) contains information about the state of the PROFIBUS field device.

In addition to the status byte and to improve interconnectability, the output interface provides further Boolean (BOOL) values, which contain important details. These conform to the bit combinations specified in PROFIBUS 3.0 "General Requirements".

The input variables SET_TOT and MODE_TOT can also be written to the process image (partition) as an option.

The block detects a higher-level error, for example, failure of a DP/PA link, via the MODE input parameter.

- If the MODE high byte = 16#80, then the values in the process image (partition) are valid.
- If the MODE high byte = 16#40 (value status = higher-level error, QMOD_ERR = TRUE), the analog value is treated as invalid.

Quality code

The program generates a quality code (QUALITY output) of the resultant value which may assume the states shown below:

State	Quality code
Valid value	16#80
Simulation	16#60
Higher-level error, last valid value	16#14
Higher-level error, substitute value	16#18
Bad, process related	16#28
Uncertain, device related	16#68
Uncertain, process related	16#78
Last valid value	16#44
Substitute value	16#48
Range overshoot	16#54
Maintenance request present	16#A4
Invalid value	16#00

The quality code is formed from internal events, such as channel error, higher-level error, or simulation and from a quality code that comes directly from the device (parameter QC).

The quality code that comes directly from the device can take on values from 16#00 – 16#FF in accordance with PROFIBUS requirements.

Addressing

The symbol generated with HW Config in the symbol table for the analog input channel must be interconnected with the TOTAL input parameter. The CFC function "Generate module drivers" interconnects the quality-code symbol of the analog input channel with input parameter QC and, if they exist, with output parameters O_SET_TOT and O_MODE_TOT.

Simulation

If input parameter SIM_ON = TRUE, the value of input parameter SIM_V is output with quality code (QUALITY =) 16#60. Simulation takes highest priority. QBAD is always set to FALSE. If the block is in the simulation state, QSIM = TRUE is set.

Substitute value

If input parameter SUBS_ON = TRUE, the value of input parameter SUBS_V is output as a value if the process values are invalid. The quality code (QUALITY) is set to 16#48 and QBAD = 1.

Hold last value

If input parameter LAST_ON = TRUE, the last valid output value is output if the process values are invalid. The quality code (QUALITY) is set to 16#44 and QBAD = 1.

Output invalid value

If the input parameters SUBS_ON and LAST_ON both = FALSE or both = TRUE and an invalid process value is present, then this will be output and QBAD will be set to 1.

Value limiting

You can limit the maximum low and high range of process values of the process image (partition).

If the switch LIMIT_ON = TRUE, the process values (V) are limited:

- To V_HL, if V > V_HL
- To LL_V, if V < V_LL

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

The block does not have a faceplate.

Additional information

For more information, refer to the section:

Notes on using driver blocks (Page 281)

4.24.2 I/Os of PA_TOT

The factory setting of the block display in CFC is identified in the "I/O" column: I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You can find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
LAST_ON	1 = last valid value: Enable injection	BOOL	0	IO
LIMIT_ON	1 = limit input value ON	BOOL	0	I
LL_V	Input value if V < V_LL	REAL	0	I
MODE	MODE input parameter	DWORD	0	I
MODE_TOT	Totalizer mode: 0 = balanced, 1 = pos_only, 2 = neg_only, 3 = hold	BYTE	0	I
O_MODE_TOT	Totalizer mode	BYTE	0	O
O_SET_TOT	Totalizer algorithm	BYTE	0	O
QA_1	1 = alarm/warning 1	BOOL	0	O
QA_2	1 = alarm/warning 2	BOOL	0	O
QBAD	1 = Group event QBAD_X	BOOL	0	O
QBAD_0	1 = non-specific	BOOL	0	O
QBAD_1	1 = configuration error	BOOL	0	O
QBAD_2	1 = not connected	BOOL	0	O
QBAD_3	1 = device failure	BOOL	0	O
QBAD_4	1 = sensor failure	BOOL	0	O
QBAD_5	1 = no communication (last usable value)	BOOL	0	O
QBAD_6	1 = no communication (no usable value)	BOOL	0	O
QBAD_7	1 = out of service	BOOL	0	O
QBAD_HL	1 = physical high limit of sensor has been reached	BOOL	0	O
QBAD_LL	1 = physical low limit of sensor has been reached	BOOL	0	O
QC	Input process value quality code	BYTE	O	I
QCASCAD0	1 = OK (cascade)	BOOL	0	O
QCASCAD1	1 = initialization acknowledged	BOOL	0	O
QCASCAD2	1 = initialization request	BOOL	0	O
QCASCAD3	1 = not required	BOOL	0	O
QCASCAD4	1 = reserved	BOOL	0	O
QCASCAD5	1 = do not select	BOOL	0	O
QCASCAD6	1 = local overwrite	BOOL	0	O
QCASCAD7	1 = reserved	BOOL	0	O
QCASCAD8	1 = initiate fail-safe	BOOL	0	O
QCONST	1 = constant	BOOL	0	O
QERR	1 = output error (inverted value of ENO)	BOOL	1	O
QLAST	1 = last valid value: Injection active	BOOL	0	O
QMOD_ERR	1 = higher-level error	BOOL	0	O
QNONCAS0	1 = OK (non-cascade)	BOOL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
QNONCAS1	1 = update event	BOOL	0	O
QNONCAS2	1 = active warning (priority <8)	BOOL	0	O
QNONCAS3	1 = active critical alarm (priority >8)	BOOL	0	O
QNONCAS4	1 = unacknowledged updated event	BOOL	0	O
QNONCAS5	1 = unacknowledged warning	BOOL	0	O
QNONCAS6	1 = unacknowledged critical alarm	BOOL	0	O
QNONCAS7	1 = initiate fail-safe	BOOL	0	O
QNONCAS8	1 = maintenance required	BOOL	0	O
QNONCAS9	1 = function test/local overwrite; usable value	BOOL	0	O
QOUT_HHL	1 = active critical alarm, high limit of OUT has been exceeded	BOOL	0	O
QOUT_HL	1 = active warning, high limit of OUT has been exceeded	BOOL	0	O
QOUT_LL	1 = active warning, low limit of OUT has been undershot	BOOL	0	O
QOUT_LLL	1 = active critical alarm, low limit of OUT has been undershot	BOOL	0	O
QSIM	1 = simulation active	BOOL	0	O
QSUBS	1 = substitution active	BOOL	0	O
QUALITY	Quality-code process value	BYTE	0	O
QUNCERT	1 = group event QUNCERTx	BOOL	0	O
QUNCERT0	1 = non-specific	BOOL	0	O
QUNCERT1	1 = last usable value	BOOL	0	O
QUNCERT2	1 = set of substitute values	BOOL	0	O
QUNCERT3	1 = initial value	BOOL	0	O
QUNCERT4	1 = imprecise sensor conversion	BOOL	0	O
QUNCERT5	1 = engineering unit violation (unit not in the valid range)	BOOL	0	O
QUNCERT6	1 = below normal	BOOL	0	O
QUNCERT7	1 = configuration error	BOOL	0	O
QUNCERT8	1 = sensor calibration	BOOL	0	O
QUNCERT9	1 = simulated value	BOOL	0	O
SET_TOT	Algorithm: 0 = totalizer, 1 = reset 0, 2 = preset PRESET_TOT	BYTE	0	I
SIM_ON	1 = activate simulation	BOOL	0	I
SIM_V	Simulation value	REAL	0	I
STATUS	Process-value status	BYTE	0	O
SUBS_ON	1 = enable substitution	BOOL	0	I
SUBS_V	Substitute value	REAL	0	I
TOTAL	Input value	REAL	0	I
V	Process value	REAL	0	O

I/O (parameter)	Meaning	Data type	Default	Type
V_HL	High limit input value	REAL	0	I
V_LL	Low limit input value	REAL	0	I

4.25 RCV_341: Receiving data in serial mode with CP 341

4.25.1 Description of RCV_341

Object name (type + number)

FB 121

- RCV_341 block I/Os (Page 417)

Area of application

Block RCV_341 is used for receiving serial data via the CP 341 module.

Calling OBs

OB 100 and the cyclic OB (recommendation 100 ms) used for receiving data.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The RACK_NO, SUBN1_ID, SUBN2_ID, and SLOT_NO inputs are configured.
- The MODE input is interconnected with the OMODE_00 output of the MOD_CP block.

Instructions for creating a user block for receiving serial data

Requirements: The optional package for configuring point-to-point connections (CP PtP Param) must be installed.

In HW Config you can also set the following transmission types (protocols):

- DK 3964R
- RK 512
- ASCII
- MODBUS master
- MODBUS slave

Procedure

- Insert the RCV_341 block into your CFC chart.
- Set the logical base address of the CP 341 module at the LADDR block input.
- Define the input buffer for user data in a block (referred to as RCV_DATA in the following).
- Install RCV_DATA downstream of the RCV_341 block in the same cyclic OB. The input buffer definition can be a simple variable or an array of variables. All S7 data types except ANY are permitted in the variable definitions.
- Apply the input buffer to the output of the RCV_DATA block.
- Interconnect this output with the R_DATA input of the RCV_341 block.
- To control and evaluate received data, define the following I/Os at the RCV_DATA block and interconnect them with the corresponding I/O of the RCV_341 block:

I/O	Data type	Meaning
Inputs:		
NDR	BOOL	New data received
ERROR	BOOL	Error when receiving new data
STATUS	WORD	Error status
LEN	INT	Length in bytes of received data
Outputs:		
EN_R	BOOL	Enable receipt of data
R	BOOL	Reset receipt of data

Block RCV_341 is ready to receive data when EN_R = TRUE. If NDR = TRUE, new data are stored in the data area of the RCV_DATA block you have interconnected with R_DATA. Variable LEN indicates the length of data received. You must save the new data received in your block or set EN_R = FALSE, since all data could be overwritten in the next cycle.

If the variable ERROR = TRUE, an error event number is entered in STATUS. (The meanings of these event numbers are described in the *CP 341* manual.) Event class 8 should be evaluated according to the selected transmission type in order to determine how to handle data received with errors.

For error cases (ERROR = TRUE), you should not set a reset (R = TRUE) for STATUS = 16#1E0D.
In other cases, we recommended that you set R = TRUE for one cycle.

- **Procedure DK 3964R**

  This procedure does not require the assignment or evaluation of further variables of the RCV_341 block.

  You can find additional information in the *CP 341* manual.

- **Remote coupler RK 512**

  The variables (hidden outputs) of the RCV_341 block indicate the origin of user data.

I/O	Data type	Meaning
L_TYP	CHAR	Area type on the remote CPU
L_NO	INT	DB number of the remote CPU
L_OFFSET	INT	DB offset of the remote CPU
L_CF_BYT	INT	Comm. flag byte number, remote CPU
L_CF_BIT	INT	Comm. flag bit number, remote CPU

- You can find additional information in the *CP 341* manual.
  This manual also explains how to evaluate the variables if RCV_341 is to provide data for the communication partner (see "Provide data").

- **ASCII driver**

  Does not require assignment or evaluation of further variables of RCV_341.

  You can find additional information in the *CP 341* manual.
  If you set the "Delimiter" mode in HW Config, please note that the following applies:
  Length of input buffer = user data + delimiter.

- **MODBUS master**

  Does not require assignment or evaluation of further variables of RCV_341.

  User data received from the partner will be entered into the input buffer according to the selected function code.

  You can find additional information in the manual titled *Loadable Drivers for Point-to-Point CPs MODBUS Protocol RTU Format; S7 Is Master*.

- **MODBUS slave**

  In MODBUS SLAVE mode, the driver on the CP 341 module controls data exchange.
  You can find additional information in the manual titled *Loadable Drivers for Point-to-Point CPs MODBUS Protocol RTU Format; S7 Is Slave*.

  You must set input MODB_SL = TRUE at the RCV_341 block.

Function and operating principle

The block receives data from a partner connected to a CP 341 by means of the P_RCV_RK (FB 122) block that is identical to the P_RCV_RK (FB 7) block in the CP PtP library. Diagnostic events detected by P_RCV_RK will be reported via ALARM_8P if no higher-level error is queued. The message function can be disabled.

- New data are received by setting output NDR = TRUE. NDR will be reset in the next cycle. Received data must be cleared by the user program during this cycle and are entered in the user-program structure that is interconnected with the R_DATA input.
- The data at the P_RCV_RK outputs are transferred 1:1 to the outputs of the RCV_341 block. As long as there are no higher-level errors pending (MODE = 16#40xxxxxx), ALARM_8P will generate an error message if the receive operation was aborted due to an error.
- In MODBUS slave mode, the MODB_341 (FB 80) block controls data exchange between the CP 341 by operating as a MODBUS slave and a MODBUS master. MODB_341 is identical to MODB_341 of the MODBUS library.

Redundancy

In H systems, the higher-level MOD_CP block evaluates redundancy of the DP master systems. Redundant serial communication is not supported and must thus be controlled manually by the user, separately from this block.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

A restart (OB 100) is reported via the LSB in byte 2 of the OMODE output.

ALARM_8P will be initialized.

Overload behavior

Not available

Time response

Not available

Message response

The block uses ALARM_8P to report diagnostic information of P_RCV_RK. The message function can be disabled by setting EN_MSG = FALSE.

Operating and monitoring

The block does not have a faceplate.

Additional information

You will find more information in:

Message texts and associated values of RCV_341 (Page 418)

4.25.2 I/Os of RCV_341

The factory setting of the block display in CFC is identified in the "I/O" column:
I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
ACC_MODE	1 = accept MODE settings	BOOL	1	IO	
EN_MSG	1 = enable message	BOOL	1	I	
EN_R	Enable receive/fetch	BOOL	0	I	
ERROR	1 = error when receiving data	BOOL	0	O	
EV_ID	Message number	DWORD	0	I	
L_CF_BIT	Protocol RK512: Comm. flag bit number, remote CPU	INT	0	O	
L_CF_BYT	Protocol RK512: Comm. flag byte number, remote CPU	INT	255	O	
L_NO	Protocol RK512: DB number of the remote CPU	INT	0	O	
L_OFFSET	Protocol RK512: DB offset of the remote CPU	INT	0	O	
L_TYP	Protocol RK512: Area type on the remote CPU	CHAR	' '	O	
LADDR	Logical address CP 341	INT	0	I	
LEN	Length of received data	INT	0	O	
MODB_SL	1 = MODBUS slave active	BOOL	0	I	
MODE	Module mode (xx = 00 - 06/00 - 15/00 - 31)	DWORD	0	I	
MSG_ACK	Message acknowledgement	WORD	0	O	
MSG_STAT	Message error information	WORD	0	O	
NDR	1 = no error when receiving new data	BOOL	0	O	
QERR	1 = program error	BOOL	1	O	
QMODF	1 = CP 341 error	BOOL	0	O	
R	Reset	BOOL	0	I	
R_DATA	Received data	ANY		I	
RACK_NO	Rack number	BYTE	0	I	
SLOT_NO	Slot number (0 in DP/PA link)	BYTE	0	I	
STATUS	Error status	WORD	0	O	
SUBN1_ID	ID of the primary DP master system	BYTE	16#FF	I	
SUBN2_ID	ID of the redundant DP master system	BYTE	16#FF	I	

Additional information

For more information, refer to the section:

Message texts and associated values of RCV_341 (Page 418)

4.25.3 Message Texts and Associated Values of RCV_341

Assignment of message texts and message classes

Message block ALARM_8P	Message no.	Default message text	Message class
EV_ID	1	CP 341 @1%d@/@2%d@/@3%d @event class@4%d@ no. @5%d@	S
	2		No message
	3		No message
	4		No message
	5		No message
	6		No message
	7		No message
	8		No message

Assignment of associated values

Message block ALARM_8P	Associated value	Block parameters	Meaning
EV_ID	1	SUBN_ID	Number of DP master system (byte)
	2	RACK_NO	Rack/station number (byte)
	3	SLOT_NO	Slot number (byte)
	4	STATUS(EV_CLAS)	Event class
	5	STATUS(EV_NO)	Event number

4.26 SND_341: Sending serial data with CP 341

4.26.1 Description of SND_341

Object name (type + number)

FB 120

- SND_341 block I/Os (Page 423)

Area of application

Block SND_341 is used for transmitting serial data via the CP 341 module.

Calling OBs

OB 100 and the cyclic OB (recommendation 100 ms) used for transmitting data.

Use in CFC

If the CFC function "**Generate module drivers**" is used, the following actions are executed automatically:

- The RACK_NO, SUBN1_ID, SUBN2_ID, and SLOT_NO inputs are configured.
- The MODE input is interconnected with the OMODE_00 output of the MOD_CP block.

Instructions for creating a user block for transmitting serial data

Requirements: The optional package for configuring point-to-point connections (CP PtP Param) must be installed.

In HW Config you can also set the following transmission types (protocols):

- DK 3964R
- RK 512
- ASCII
- MODBUS master
- MODBUS slave

Procedure

- Install the SND_341 block into your CFC chart.
- Set the logical base address of the CP 341 module at the LADDR block input.
- Define the send buffer for user data in a block (referred to as SND_DATA in the following).
- Install SND_DATA upstream of SND_341 in the same cyclic OB. The send buffer definition can be a simple variable or an array of variables. All S7 data types except ANY are permitted in the variable definitions.
- Apply the send buffer to the output of SND_DATA.
- Interconnect this output with the S_DATA input of the SND_341 block. The length of the interconnected data structure determines the length of the protocol to be transmitted.
- To control and evaluate transmitted data, define the following I/Os at SND_DATA and interconnect them with the corresponding I/O of SND_341:

I/O	Data type	Meaning
Inputs:		
DONE	BOOL	Send request end without error
ERROR	BOOL	Send request end with error
STATUS	WORD	Error status
Outputs:		
REQ	BOOL	Initialize send request
R	BOOL	Reset data transmission

Block SND_341 initializes a send request when input REQ = TRUE. If the send request is completed successfully (DONE = TRUE), a new send request is initialized automatically, provided that REQ = TRUE. If the data to be sent are not yet available, you have to set REQ = FALSE. If the variable ERROR = TRUE, an error event number is entered in STATUS. The meanings of these event numbers are described in the *CP 341* manual. Event class 7 should be evaluated according to the selected transmission type in order to determine how to handle data sent with errors or data not transmitted.

- **Procedure DK 3964R**

  This procedure does not require any evaluation or assignment of further variables of SND_341.

  You can find additional information in the *CP 341* manual.

- **Remote coupler RK 512**

  These variables (hidden inputs) of SND_341 are to be set:

I/O	Data type	Meaning
R_CPU_NO	INT	Number of the remote CPU
R_TYP	CHAR	Area type on the remote CPU
R_NO	INT	DB number of the remote CPU
R_OFFSET	INT	DB offset of the remote CPU
R_CF_BYT	INT	Comm. flag byte number, remote CPU
R_CF_BIT	INT	Comm. flag bit number, remote CPU
You can find additional information in the *CP 341* manual. This manual also explains how to set the variables if SND_341 is to fetch data from the communication partner (see "Fetching data"). In this case, the input variable is hidden. SF (send or fetch, data type CHAR) = "F" must be set.		

- **ASCII driver**

  Does not require assignment or evaluation of further variables of SND_341.

  You can find additional information in the *CP 341* manual.

- **MODBUS master**

The variable (hidden input) R_TYP of SND_341 must be set equal to 'X'. This table shows how the send buffer for the request frame should be structured, according to the function code in the transmission protocol:	
Byte	**Meaning**
1	MODBUS slave address
2	MODBUS function code
3	See function code x
4	See function code x
:	
x	CRC check (message-frame checksum)
x+1	CRC check (message-frame checksum)
"Master-slave" data transfer starts at the slave address, followed by the function code and the transfer of the data. The structure of the data field is determined by the function code used.	
You can find additional information in the manual titled *Loadable Drivers for Point-to-Point CPs MODBUS Protocol RTU Format; S7 Is Master*. The CRC check at the end of the message frame is formed by the MODBUS master driver on the CP 341 module.	

- **MODBUS slave**

  In MODBUS SLAVE mode, the driver on the CP 341 module controls data exchange. You can find additional information in the manual titled *Loadable Drivers for Point-to-Point CPs MODBUS Protocol RTU Format; S7 Is Slave*.

  You are **not** required to insert an SND_341 block into your chart.

  You can find additional information in "RCV_341 MODBUS slave".

Function and operating principle

The block uses the P_SND_RK (FB 123) block to transfer data to a communication partner that is connected to a CP 341. P_SND_RK is identical to P_SND_RK (FB 8) in the CP PtP library. Diagnostic events detected by P_SND_RK are reported via ALARM_8P as long as no higher-level errors (MODE = 16#40xxxxxx) are pending. The message function can be disabled.

- Data transfer begins as soon as input REQ = TRUE. A new send request is only possible after DONE = TRUE or ERROR = TRUE has been set by P_SND_RK. Data at the outputs of P_SND_RK are transferred 1:1 to the outputs of SND_341.
- The length of data to be transferred is determined by the length of the transmit data structure in the user block, which is interconnected with the input S_DATA.

Redundancy

In H systems, the higher-level MOD_CP block evaluates redundancy of the DP master systems. Redundant serial communication is not supported and must thus be controlled manually by the user, separately from this block.

Error handling

The plausibility of input parameters is not checked.

Startup characteristics

A restart (OB 100) is reported via the LSB in byte 2 of the OMODE output.

ALARM_8P will be initialized.

Overload behavior

Not available

Time response

Not available

Message response

The block uses ALARM_8P to report diagnostic information of P_SND_RK.
The message function can be disabled by setting EN_MSG = FALSE.

Operating and monitoring

The block does not have a faceplate.

Note: If you have selected the "Enable operator control and monitoring" option in the block object properties in the CFC, the variables transferred to the OS are identified under "I/Os of ..." (OCM column, "+"). Default: Option not activated.

Additional information

You will find more information in:

Message texts and associated values of SND_341 (Page 424)

4.26.2 I/Os of SND_341

The factory setting of the block display in CFC is identified in the "I/O" column:
I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
ACC_MODE	1 = accept MODE settings	BOOL	1	IO	
DONE	1 = request end without error	BOOL	0	O	
EN_MSG	1 = enable message	BOOL	1	I	
ERROR	1 = request end with error	BOOL	0	O	
EV_ID	Message number	DWORD	0	I	
LADDR	Logical address CP 341	INT	0	I	
MODE	Module mode (xx = 00 - 06/00 - 15/00 - 31)	DWORD	0	I	
MSG_ACK	Message acknowledgement	WORD	0	O	
MSG_STAT	Message error information	WORD	0	O	
QERR	1 = program error	BOOL	1	O	
QMODF	1 = CP 341 error	BOOL	0	O	
R	Reset	BOOL	0	I	
R_CF_BIT	Protocol RK512: Number of the remote CPU	INT	0	I	
R_CF_BYT	Protocol RK512: Number of the remote CPU	INT	255	I	
R_CPU_NO	Protocol RK512: Number of the remote CPU	INT	1	I	
R_NO	Protocol RK512: Number of the remote CPU	INT	0	I	
R_OFFSET	Protocol RK512: Number of the remote CPU	INT	0	I	
R_TYP	Protocol RK512: Number of the remote CPU/MODBUS master = X	CHAR	'X'	I	
RACK_NO	Rack number	BYTE	16#FF	I	
REQ	Initialize request	BOOL	0	I	
S_DATA	Send data	ANY		I	
SF	Send (S) or receive/fetch (F)	CHAR	'S'	I	
SLOT_NO	Slot number	BYTE	16#FF	I	
STATUS	Error status	WORD	0	O	
SUBN1_ID	ID of the primary DP master system	BYTE	16#FF	I	
SUBN2_ID	ID of the redundant DP master system	BYTE	16#FF	I	

Additional information

For more information, refer to the section:

Message texts and associated values of SND_341 (Page 424)

4.26.3 Message Texts and Associated Values of SND_341

Assignment of message texts and message classes

Message block ALARM_8P	Message no.	Default message text	Message class
EV_ID	1	CP 341 @1%d@/@2%d@/@3%d @event class@4%d@ no. @5%d@	S
	2		No message
	3		No message
	4		No message
	5		No message
	6		No message
	7		No message
	8		No message

Assignment of associated values

Message block ALARM_8P	Associated value	Block parameters	Meaning
EV_ID	1	SUBN_ID	Number of DP master system (byte)
	2	RACK_NO	Rack/station number (byte)
	3	SLOT_NO	Slot number (byte)
	4	STATUS(EV_CLAS)	Event class
	5	STATUS(EV_NO)	Event number

4.27 Internal blocks

4.27.1 MODB_341: Internal block

Object name (type + number)

FB 80

This block is used for the MODBUS SLAVE mode.

Note

In MODBUS SLAVE mode, the driver on the CP 341 module control data exchange.
You can find additional information in the manual titled "Loadable Drivers for Point-to-Point CPs MODBUS Protocol RTU Format; S7 Is Slave".

Family: MAINT

5

5.1 ASSETMON: Process variable monitoring for violation of limits

5.1.1 Description of ASSETMON

Object name (type + number)

FB86

- ASSETMON block I/Os (Page 432)
- Asset management block icons (Page 594)
- ASSETMON faceplate (Page 553)

Area of application

The block is used to monitor three analog process variables for exceeding three limits. It reports the wait status of the process variables:

- When overshooting a limit
- Via the device-based quality code or
- Via binary message inputs.

Note

The block does not contain any technology or device based processing parts and consequently is not taken into account by the driver generator.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and OB100.

Function and operating principle

If input MONITOR = 1 (default setting), the block monitors the inputs PV_0, PV_1 and PV_2 for either undershooting or overshooting three limits. When reaching or overshooting a limit the respective output QLR_PV_x (maintenance demand), QLD_PV_x (maintenance request) and QLA_PV_x (maintenance alarm) (x = 0, 1, 2) is set to TRUE and the applicable message is output.

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. A separate message is sent for each of the three process variables.

The labeling of process values in the monitoring view is entered in the parameter data of the EDD for the relevant instance (see also section PLT ID).

Monitoring of the individual limits can be switched off with SUP_MR_x, SUP_MD_x or SUP_MA_x = 1.

In addition to limit value monitoring, the quality code of the individual process variables is also analyzed. The device-specific quality code of a process value triggers the corresponding message that is also used by the limit value monitoring.

In addition seven message inputs (MESSAGE1, ..., MESSEGE7) are available that generate a message if status = 1 is set.

Binary message inputs

MESSAGE1	Maintenance alarm	Message (S) requires acknowledgement
MESSAGE2	Maintenance request	Message (F) requires acknowledgement
MESSAGE3	Maintenance required	Message (M) requires acknowledgement
MESSAGE4	Local operator control	Message (SA) does not require acknowledgement
MESSAGE5	Simulation	Message (SA) does not require acknowledgement
MESSAGE6	Out of service	Message (S) requires acknowledgement
MESSAGE7	Passivated	Message (SA) does not require acknowledgement

For each PV_x, an input (RESETx), and an output (QRESETx) are available that can be used to reset the count value of the PV_x in the technology block (e.g. interconnection of output QRESETx with input RESET of the COUNT_P block).

If MONITOR = 0, then the three process variables and their quality code are not analyzed, rather just the 7 inputs (MESSAGE1, ... , MESSAGE7).

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 EN_MSG = 0. In this case MS = 8 is set.

The detailed diagnostics is shown in the diagnostic view of the faceplate via the Boolean 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 tests for the relevant inputs DIAG1 through 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 TRUE, then if an internal alarm is triggered through PV_0, PV_1 or PV_2, the message, the explanatory text "Additional status available" is output.

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. Select **Insert > SIMATIC PDM > TAG**.
3. Highlight the inserted TAG object and select the context menu command SIMATIC PDM > Device Selection...
4. In the tree structure CFC > DATA_OBJECTS > CFC >, select AssetMon and close the dialog with "OK".
5. In the context menu select **Open Object** and enter all necessary data in the parameter assignment screen form.
6. Select **File> Save** .
   The parameter assignment screen form is closed.
7. Select the TAG object and then **Tools > SIMATIC PDM > Create PLT-ID**.

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:

- The quality-code inputs QC_0, QC_1, and QC_2. Only device-based errors have an effect, process-relevant ones do not.
- The limit monitoring of PV_0, PV_1, and PV_2.
- The binary message inputs (external MS).
- The interconnectable input MS_IN (external MaintenanceState).
- The interconnectable input STATUS (unchanged quality code of a device).

Of all these events the highest priority event will be displayed in the MS.

The 16 DIAGx do not have any influence on the MS, rather they are used only for visualization of the detail diagnostics in the diagnostics view of the faceplate.

Possible quality codes for creating the maintenance status (MS)

Note

The table applies exclusively for the MS and not for the quality code displays of the individual process values, these are created exclusively by the QC_x parameters.

The priority is analogous to the MS coding, Consequently the following applies: The greater the value of the MS the higher the priority.

Signal	Signal Detail	Quality code	MS
PV_x QualityCode Input	Failure, due to device	0x00	7
PV_x QualityCode Input	Failure, due to device	0x14	7
PV_x QualityCode Input	Failure, due to device	0x18	7
PV_x QualityCode Input	Uncertain, device-specific	0x44	7
PV_x QualityCode Input	Uncertain, device-specific	0x48	7
PV_x QualityCode Input	Uncertain, device-specific	0x68	6
PV_x QualityCode Input	Uncertain, device-specific	0x54	6
PV_x QualityCode Input	Maintenance required	0xA4	5
PV_x QualityCode Input	min. 1 PV simulated	0x60	3
PV_x HighLimit exceeded	Maintenance alarm	0x24	7
PV_x HighLimit exceeded	Maintenance request	0xA8	6
PV_x HighLimit exceeded	Maintenance required	0xA4	5
MESSAGE1	Maintenance alarm	0x24	7
MESSAGE2	Maintenance request	0xA8	6
MESSAGE3	Maintenance required	0xA4	5
MESSAGE4	Local operator control	0x3C	4
MESSAGE5	Simulation	0x60	3
MESSAGE6	Out of service	0x1C	2
MESSAGE7	Passivated	0x23	1
Messages disabled			8
MS_IN	External maintenance state		0 – 9
STATUS	All quality codes from 0 to 255 are possible, MS creation according to QC_MS_STAT table	0x00 … 0xFF	0 – 9

Error handling

In the event of arithmetical errors the outputs ENO = 0 and QERR = 1 will be set.

Startup characteristics

After startup, the messages will be suppressed during the number of cycles set at RUNUPCYC.

Time response

Not applicable.

Message response

The block reports by means of ALARM_8P and NOTIFY_8P.

Additional information

You will find more information on this subject in the following sections:

Operating and monitoring ASSETMON (Page 435)

Message texts and associated values of ASSETMON (Page 434)

Maintenance status of MS (Page 616)

Notes on using driver blocks (Page 281)

5.1.2 Description of ASSETMON

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Type	Default	Type	OCM
DIAGx	Asset detail diagnosis (x = 1 to 16)	BOOL	0	I	+
DIFFALMx	Differential value to the next expected alarm (x = 0, 1, 2)	REAL	0	O	+
EN_MSG	1 = enable message	BOOL	1	I	
EV_IDx	Message number (x = 1, 2, 3)	DWORD	0	I	
LA_PV_x	Limit PV_x (x = 0, 1, 2) maintenance alarm	REAL	100	i	+
LD_PV_x	Limit PV_x (x = 0, 1, 2) maintenance request	REAL	100	i	+
LR_PV_x	Limit PV_x (x = 0, 1, 2) maintenance required	REAL	100	i	+
MESSAGE1	1 = message: bad, maintenance alarm	BOOL	0	I	
MESSAGE2	1 = message: uncertain, maintenance request	BOOL	0	I	
MESSAGE3	1 = message: good, maintenance required	BOOL	0	I	
MESSAGE4	1 = message: passivated	BOOL	0	I	
MESSAGE5	1 = message: bad, out of service	BOOL	0	I	
MESSAGE6	1 = message: bad, local operation/function control	BOOL	0	I	
MESSAGE7	1 = message: uncertain, simulation	BOOL	0	I	
MONITOR	1 = monitor PV_x (x = 0, 1, 2) switched on	BOOL	0	I	
MS	Maintenance status	DWORD	0	I	+
MS_IN	External interconnectable MS	DWORD	0	I	
STATUS	Unchanged QC of a device	BYTE	16#80	I	
MSG_ACKx	Message acknowledgement (x = 1, 2)	WORD	0	O	
MSGSTATx	Message error information (x = 1, 2, 3)	WORD	0	O	
O_MS	Maintenance status	DWORD	0	O	+
PLT_ID	Asset ID of EDD	DWORD	0	I	+
PV_x	Process value PV_x (x = 0, 1, 2)	REAL	0	I	+
QC_x	Quality code PV_x (x = 0, 1, 2)	BYTE	16#80	I	
QERR	1 = program error	BOOL	1	O	
QLA_PV_x	1 = limit PV_x (x = 0, 1, 2), maintenance alarm exceeded	BOOL	0	O	+
QLD_PV_x	1 = limit PV_x (x = 0, 1, 2), maintenance request exceeded	BOOL	0	O	+
QLR_PV_x	1 = limit PV_x (x = 0, 1, 2), maintenance required exceeded	BOOL	0	O	+
QRESET_x	1 = reset PV_x (x = 0, 1, 2)	BOOL	0	O	
RESET_x	1 = reset PV_x (x = 0, 1, 2)	BOOL	0	I	+
RUNUPCYC	Number of run-up cycles	INT	3	I	
SUP_MA_x	1 = suppress monitoring of the limit PVx (x = 0, 1, 2), maintenance alarm exceeded	BOOL	0	I	+

I/O (parameter)	Meaning	Type	Default	Type	OCM
SUP_MD_x	1 = suppress monitoring of the limit PVx (x = 0, 1, 2), maintenance request exceeded	BOOL	0	I	+
SUP_MR_x	1 = suppress monitoring of the limit PVx (x = 0, 1, 2), maintenance required exceeded	BOOL	0	I	+

5.1.3 Message Texts and Associated Values of ASSETMON

Assignment of message texts and message classes

Message block	Message no.	Default message text	Message class
EV_ID1 (ALARM_8P)	1	bad, maintenance alarm, PV_0 @4W%t#ASSETMON_TXT@	S
	2	uncertain, maintenance request, PV_0 @4W%t#ASSETMON_TXT@	F
	3	good, maintenance required, PV_0 @4W%t#ASSETMON_TXT@	M
	4	bad, maintenance alarm, PV_1 @4W%t#ASSETMON_TXT@	S
	5	uncertain, maintenance request, PV_1 @4W%t#ASSETMON_TXT@	F
	6	good, maintenance required, PV_1 @4W%t#ASSETMON_TXT@	M
	7	bad, maintenance alarm, PV_2 @4W%t#ASSETMON_TXT@	S
	8	uncertain, maintenance request, PV_2 @4W%t#ASSETMON_TXT@	F
EV_ID2 (ALARM_8P)	1	good, maintenance required, PV_2 @2W%t#ASSETMON_TXT@	M
	2	bad, maintenance alarm, external @2W%t#ASSETMON_TXT@	S
	3	uncertain, maintenance request, external @2W%t#ASSETMON_TXT@	F
	4	good, maintenance required, external @2W%t#ASSETMON_TXT@	M
	5	bad, device out of service	S
EV_ID3 (NOTIFY_8P)	1	bad, passivated	SA
	3	bad, local operation/function control	SA
	4	uncertain, simulation	SA
	5	Configuration change	SA
	6	process-related fault	SA

Associated values

Message block ALARM 8P	Associated value	Meaning
EV_ID1	4	Text number from ASSETMON_TXT
EV_ID2	2	Text number from ASSETMON_TXT

You can find the message texts and their text numbers in "Text library for ASSETMON (Page 613)".

5.1.4 Operator Control and Monitoring of ASSETMON

Additional information

Additional information is available in the following sections:

- Asset management block icons (Page 594)
- ASSETMON faceplate (Page 553)

5.2 MS_MUX: Determination of the worst individual status

5.2.1 Description of MS_MUX

Object name (type + number)

FC 288

- MS_MUX block I/Os (Page 437)

Area of application

The block MS_MUX determines the maintenance status (MS) of a functional unit consisting of multiple FF field devices, by selecting the worst individual status of the FF field devices.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32).

Function and operating principle

The inputs MS_x (0<=x <= 9) are interconnected with the outputs Q_MS of ASSETMON blocks that each represent an FF field device.

Among the inputs MS_x the worst maintenance status (MS) is determined and made available at the output QMS for further interconnection with the input MS_IN of an ASSETMON block of a functional unit.

Installation in ASSET

You can find information about installing the MS_MUX block in Linking FF Devices in ASSET (Page 612).

Startup characteristics

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

Not available

5.2.2 I/Os of MS_MUX

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
MS_x	Maintenance State _x (x = 0- 3)	DWORD	0	I
MS_x	Maintenance State _x (x = 4- 9)	DWORD	0	I
QMS	Worst Maintenance State	DWORD	0	O

5.3 ST_MUX: Determination of the status value for FF_Field devices

5.3.1 Description of ST_MUX

Object name (type + number)

FC 287

- ST_MUX block I/Os (Page 439)

Area of application

The block ST_MUX determines the status of an FF field device by selecting the worst of the signal processing blocks FF_A_DI, FF_A_AI, FF_A_DO and FF_A_AO of the field device.

Calling OBs

The calling OB is cyclic interrupt OB 3x in which you install the block (for example, OB 32).

Function and operating principle

The inputs ST_x (0<=x <= 9) are interconnected with the outputs STATUS (FF_A_DI and FF_A_AI) or ST_xx or QC_xx (FF_A_DO and FF_A_AO) of all signal processing blocks of an FF field device. Among the ST_x inputs, ST_MUX determines the worst status and makes it available at the output QST for further interconnection with an ASSETMON component.

If different statuses with highest priority are present then the value of the first ST_x input is used.

Installation in ASSET

You can find information about installing the ST_MUX block in Linking FF Devices in ASSET (Page 612).

Startup characteristics

Not available

Time response

Not available

Message response

Not available

Operating and monitoring

Not available

5.3.2 I/Os of ST_MUX

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
QST	Worst status	BYTE	16#80	O
ST_x	Status_x (x = 0- 3)	BYTE	16#80	I
ST_X	Status_x (x = 4- 9)	BYTE	16#80	I

5.4 STATEREP: Status display of block groups

5.4.1 Description of STATEREP

Object name (type + number)

FB 87

- STATEREP block I/Os (Page 441)

Area of application

The block STATEREP is used for displaying the status of a group of blocks that is designed for hiding messages automatically.

Function and operating principle

The block has 32 BOOL inputs that describe defined states. Depending on which state is set, it will be output at the INT output.

Error handling

If more than one parameter is set, then the block outputs the messages QSTATE=0 as well as QERR=1.

Startup characteristics

Not available.

Time response

Not available.

Message response

Not available.

5.4.2 I/Os of STATEREP

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O	Meaning	Data type	Default	Type	OCM
State1 .. 10	Process status 1 – 10	BOOL	0	I	
State11 .. 32	Process status 11 – 32	BOOL	0	I	
QERR	1 = error, more than one status is active	BOOL	1	O	
QSTATE	Process status	INT	0	O	+

Family: @SYSTEM

6

6.1 AL_DELAY - alarm delay

6.1.1 Description of AL_DELAY

Object name (type + number) and family

FC 290

Family: @SYSTEM

How it works

This block is used by all blocks with message capability and does not, therefore, have a comprehensive help system. It is used to delay the triggering of alarms, in other words, the alarm is triggered only when the cause of the alarm exists for longer than a specified period. The alarm exiting the state can also be delayed.

6.2 Internal blocks

6.2.1 P_RCV_RK: Internal Block

Object name (type + number)

FB 122

This block is a system block and is only used internally. Therefore, there is no help available for it.

6.2.2 P_SND_RK: Internal Block

Object name (type + number)

FB 123

This block is a system block and is only used internally. Therefore, there is no help available for it.

Family: TIME

7

7.1 OB1_TIME: Determining the Degree of CPU Utilization

7.1.1 Description of OB1_TIME

Object name (type + number)

FB 69

- OB1_TIME block I/Os (Page 446)

Function

The OB1_TIME block provides information relating to the degree of CPU utilization.

How it works

Block OB1_TIME is installed in OB 1.

- The block is reset (i.e., CNT, MAX, MIN, MEAN and the internal ACT_TME are reset, MIN = 2147483000) and started by a negative edge (1 → 0) at input STOP_RES. The current system time is determined and saved internally under L_TME.
- In each execution cycle, the block determines the system time of day in ms, saves it internally in ACT_TIME and calculates the maximum value since the reset time (MAX), the root mean square value (MEAN) and the minimum value (MIN) of the time that has passed since its last execution (OB_1_TIME = ACT_TIME-L_TIME). Counter CNT is then incremented by 1 and L_TIME = ACT_TIME is reset. The root mean square value is calculated as follows:

$$MEAN = \sqrt{\frac{1}{CNT+1}\left(CNT * MEAN^2 + OB1\_TIME^2\right)}$$

- The calculated values must be interpreted by the commissioning personnel in order to derive the degree of CPU utilization.
- A 1 at the input STOP_RES causes the block algorithm not to be processed further (processing is "halted). ENO is reset during this time to 0.

Calling OBs

OB 1

Startup characteristics

Reset to default values.

Message response

Not available

Error handling

Only by means of the operating system.

Operating and monitoring

Not available

7.1.2 I/Os of OB1_TIME

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
CNT	Counter	DINT	0	O
MAX	Maximum time value in milliseconds	DINT	0	O
MAX_CNT	Maximum CNT value in milliseconds	DINT	10000	I
MEAN	Quadratic mean value in milliseconds	DINT	0	O
MIN	Minimum time value in milliseconds	DINT	0	O
OB1_TIME	Cyclic execution time: ACT_TIME - L_TIME	DINT	0	O
QERR	Inverted value of ENO	BOOL	1	O
STOP_MAX	1 = STOP if CNT = MAX_CNT	BOOL	0	I
STOP_RES	STOP/reset: 1 = STOP, 0 = reset	BOOL	1	I

Family: MATH

8

8.1 ADD4_P: Adder for a maximum of 4 values

8.1.1 Description of ADD4_P

Object name (type + number)

FC256

- ADD4_P block I/Os (Page 447)

Function

Block ADD4_P calculates the sum of up to 4 values:
V = U1+...+ Un (n ≤ 4)

Calling OBs

Only the OB in which you install the block.

Error handling

In case of an overflow/underflow, the REAL value of the high/low limit is set in the result V. ENO will be set low and QERR high.

8.1.2 I/Os of ADD4_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
U1	Addend 1	REAL	0.0	I
U2	Addend 2	REAL	0.0	I
U3	Addend 3	REAL	0.0	I
U4	Addend 4	REAL	0.0	I
V	Result	REAL	0.0	O

8.2 ADD8_P: Adder for a maximum of 8 values

8.2.1 Description of ADD8_P

Object name (type + number)

FC 257

- ADD8_P block I/Os (Page 448)

Function

The ADD8_P block calculates the sum of up to 8 values:

V = U1 + U2 + U3 +...+ Un (n ≤ 8)

Error handling

In case of an overflow/underflow, the REAL value of the high/low limit that has been violated is set in the result V and ENO is set to 0.

Calling OBs

Only the OB in which you install the block.

8.2.2 I/Os of ADD8_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
U1	Addend 1	REAL	0.0	I
....	....	....	....	....
U8	Addend 8	REAL	0.0	I
V	Result	REAL	0.0	O

8.3 AVER_P: Mean time value

8.3.1 Description of AVER_P

Object name (type + number)

FB34

- AVER_P block I/Os (Page 450)

Function

Block AVER_P calculates the mean time value of an active parameter based on the time which has passed since its start. The equation below is used:

V = (N ∗ Vold + U)/(N+1)

Explanations:

- U = Applied parameter
- V = Current mean
- Vold: Mean value of cycles executed since startup
- N = Number of cycles used for mean

How it works

Operating principle of the block:

- Calculation is started with a positive edge at the RUN input. An existing result V is overwritten by the input value U. You will find more information in the section "Startup characteristics".
- In the subsequent cycles, the result is recalculated in output V and the cycle counter N is incremented.
- The calculation is terminated by resetting the RUN input and the actual values of the results V and N are saved.

Calling OBs

The cyclic interrupt OB in which you install the block (for example, OB32) and also OB100.

Error handling

Arithmetic errors are indicated by ENO = 0 or QERR = 1. Arithmetic errors occur when the range limits of the REAL data type exceed the results of the formula described above. The value of V from the previous cycle is retained in this case. If V takes the value = #+INF or #-INF because of the corresponding value of U, there is also an arithmetic error in the next cycle.

Startup characteristics

During the initial run and during a CPU startup, the following takes place:

- The input value U is written to output V.
- The cycle counter N is reset.

This is done by calling the block in the startup OB.

Time response

To allow it to perform its required function, the block is called in a cyclic interrupt OB. The user can use the following formula to calculate the Taverage time:

Taverage = N * Tsampling

where Tsampling is the sampling time of the block.

When configuring with CFC, the higher-level runtime group of the block and its sampling parameter may have to be taken into consideration.

8.3.2 I/Os of AVER_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
N	Number of cycles used in the calculation	REAL	0	O
QERR	1 = error output (inverted ENO)	BOOL	1	O
RUN	Averaging: 0 = OF, 1 = ON	BOOL	0	I
U	Input	REAL	0	I
V	Mean value	REAL	0	O

8.4 COUNT_P: Counter

8.4.1 Description of COUNT_P

Object name (type + number)

FB36

- COUNT_P block I/Os (Page 452)

Function

A positive edge of the binary input signal I0 increments or decrements the counter value V, according to the set mode.

How it works

Block COUNT_P operates according to the following principle:

- The operating mode can be set at the MODE parameter:
  - MODE = 0 = Up counter
  - MODE = 1 = Down counter
- Block behavior when used as up an counter:
  - Every positive edge at the I0 input increments the counter.
  - When the high limit V = V_HL is reached, the counter is not incremented further and output QVHL is set high.
  - When the mode is reversed to "Down count", output V is decremented at the next positive edge of I0 and QVHL is reset.
  - RESET = 1 sets V = V_LL, QVLL = 1, QVHL = 0, and the internal edge flag is corrected to the input value.
- Block behavior when operating as down counter:
  - Every positive edge at the I0 input decrements the counter.
  - When the lower limit V = V_LL has been reached, the counter is not decremented further and output QVLL is set high.
  - When the mode is reversed to "Up count", the output V is incremented at the next positive edge of I0 and QVLL is reset.
  - RESET = 1 sets V = V_HL, QVHL = 1, QVLL = 0, and the internal edge flag is corrected to the input value.

Calling OBs

Only the OB in which you install the block (for example, OB32).

Error handling

Arithmetic errors are indicated by ENO = 0 or QERR = 1.

Startup characteristics

During the initial run and during a CPU startup, the block performs one RESET, according to the operating mode set. You can find additional information in “Operating Principle, RESET”.

Time response

Not applicable. However, it is advisable to install the block in the OB that also contains the edge triggering block.

8.4.2 I/Os of COUNT_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	Permissible values
I0	Input	BOOL	0	I	
MODE	Mode: 0 = count up, 1 = count down	BOOL	0	I	
QERR	1 = Error	BOOL	1	O	
QVHL	1 = V_HL reached	BOOL	0	O	
QVLL	1 = V_LL reached	BOOL	0	O	
RESET	1 = reset	BOOL	0	I	
V	Counter value	DINT	0	O	
V_HL	High limit of V	DINT	100	I	V_HL ≥ V_LL
V_LL	Low limit of V	DINT	0	I	V_LL ≤ V_HL

8.5 MEANTM_P: Calculating the mean time value

8.5.1 Description of MEANTM_P

Object name (type + number)

FB42

- MEANTM_P block I/Os (Page 454)

Function

Block MEANTM_P is used to calculate a mean time value of an analog input signal across a configurable past time period, in accordance with the equation:

Vn = (U1 + U2 +...+Un)/n

where U1...Un are the detected values used for averaging.

How it works

During every execution of the block the arithmetic mean value is calculated from the current input value U and the values saved during the time T_WINDOW. This is then updated at the output V. The current input value then overwrites the oldest previous value.

- The time window across which averaging is to be carried out is entered in the parameter T_WINDOW.
- The block determines the number n of values to be saved from the integer part of the quotient T_WINDOW / SAMPLE_T.
- The block can save up to 20 previous values in its internal memory. Data is reduced if the time window is longer.
- The STOP_RES input has the following effect:
  - If STOP_RES = "1", the calculation process is stopped. The output value is retained for the duration of this period.
  - The mean time value calculation is reset by a falling edge 1 → 0.
- If SAMPLE_T or T_WINDOW is changed, the mean time value is reset.

Note

The T_WINDOW and SAMPLE_T parameters must be set in such a way that the quotient T_WINDOW / SAMPLE_T is not greater than 32748.

Calling OBs

This is the cyclic interrupt OB in which you install the block (for example, OB32).

Error handling

Only by means of the operating system

Startup characteristics

Not available

If the block was active before the CPU stop, and continues to calculate afterwards, the CPU out-time relative to T_WINDOW has to be taken into consideration. This allows you to decide whether the result can still be used or whether the calculation process has to be reset via the input STOP_RES.

Time response

The block must be called from a cyclic interrupt OB.

8.5.2 I/Os of MEANTM_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	Permissible values
QERR	1 = error	BOOL	1	O	
SAMPLE_T	Sampling time in seconds	REAL	1.0	I	>0
STOP_RES	Stop/reset mean value calculation	BOOL	0	I	
T_WINDOW	Size of the time window in seconds	REAL	20	I	
U	Input value	REAL	0	I	
V	Output value	REAL	0	O	

8.6 MUL4_P: Multiplier for a maximum of 4 values

8.6.1 Description of MUL4_P

Object name (type + number)

FC262

- MUL4_P block I/Os (Page 455)

Function

Block MUL4_P multiplies up to 4 values

V:= U1*...*Un (n ≤ 4)

Parameter Value	Type	No.	Meaning
MUL4_P	FC	262	Multiplier with 4 inputs

Calling OBs

The OB in which you install the block.

Error handling

In case of an overflow/underflow, the REAL value of the high/low limit is set in the result V. ENO will be set low and QERR high.

8.6.2 I/Os of MUL4_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
U1	Input 1	REAL	1.0	I
....	....	....	....	....
U4	Input 4	REAL	1.0	I
V	Result	REAL	1.0	O

8.7 MUL8_P: Multiplier for a maximum of 8 values

8.7.1 Description of MUL8_P

Object name (type + number)

FC263

- MUL8_P block I/Os (Page 457)

Function

Block MUL8_P multiplies up to 8 values

V:= U1*U2*U3*...*Un (n ≤ 8)

Note

If you want to multiply a maximum of 4 values, use MUL4_P instead of MUL8_P in order to reduce computation time.

Parameter Value	Type	No.	Meaning
MUL8_P	FC	263	Multiplier with 8 inputs

Calling OBs

The OB in which you install the block.

Error handling

In case of an overflow/underflow, the REAL value of the high/low limit is set in the result V. ENO will be set low and QERR high.

8.7.2 I/Os of MUL8_P

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type
U1	Input 1	REAL	1.0	I
....		....	....	....
U8	Input 8	REAL	1.0	I
V	Result	REAL	1.0	O

Family: CONVERT

9

9.1 General information about conversion blocks

Purpose of the conversion blocks

In CFC you can only connect block outputs (source type) to inputs (target type) of the same data type (for example, REAL output with REAL input). Conversion blocks must be used to allow the interconnection of different data types. The input and output data of the block are of a different type, and it thus converts the input data type according to the data type set at the output. The CFC block library ELEMENTA contains the conversion blocks required for these interconnections. There is also an additional R_TO_DW block with expanded properties for process engineering applications.

Calling OBs

The conversion block must be installed in the OB before the block that evaluates the conversion result.

9.2 R_TO_DW: Conversion

9.2.1 Description of R_TO_DW

Object name (type + number)

FC 282

- R_TO_DW block I/Os (Page 460)

Function

Block R_TO_DW converts a REAL number to a double word (DW). REAL numbers between 0 and 4,294,967,000 are accepted.

Error handling

If the values are outside the limits specified above, ENO = 0 and the low limit (= 0) or high limit (= 4,294,967,000) will be set.

9.2.2 I/Os of R_TO_DW

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 459)".

I/O (parameter)	Meaning	Data type	Default	Type
U	Value to convert	REAL	0.0	I
V	Converted value	DWORD	0	O

Family: OPERATE

10

10.1 Overview of Operator-Control Blocks

Introduction

In this chapter we shall introduce the operator control blocks and show how they can be used to influence block parameters.

Purpose of the operator control blocks

An operator control block represents the operator control interface between blocks on the AS and on the OS. It provides the following standard solutions:

How can the input parameter "W" of the function block "FB_yyy" be controlled by the operator and also be modified by the AS program?

The basic solution is shown in the figure, using an operator control block with its two parts OP_AS and OP_OS.

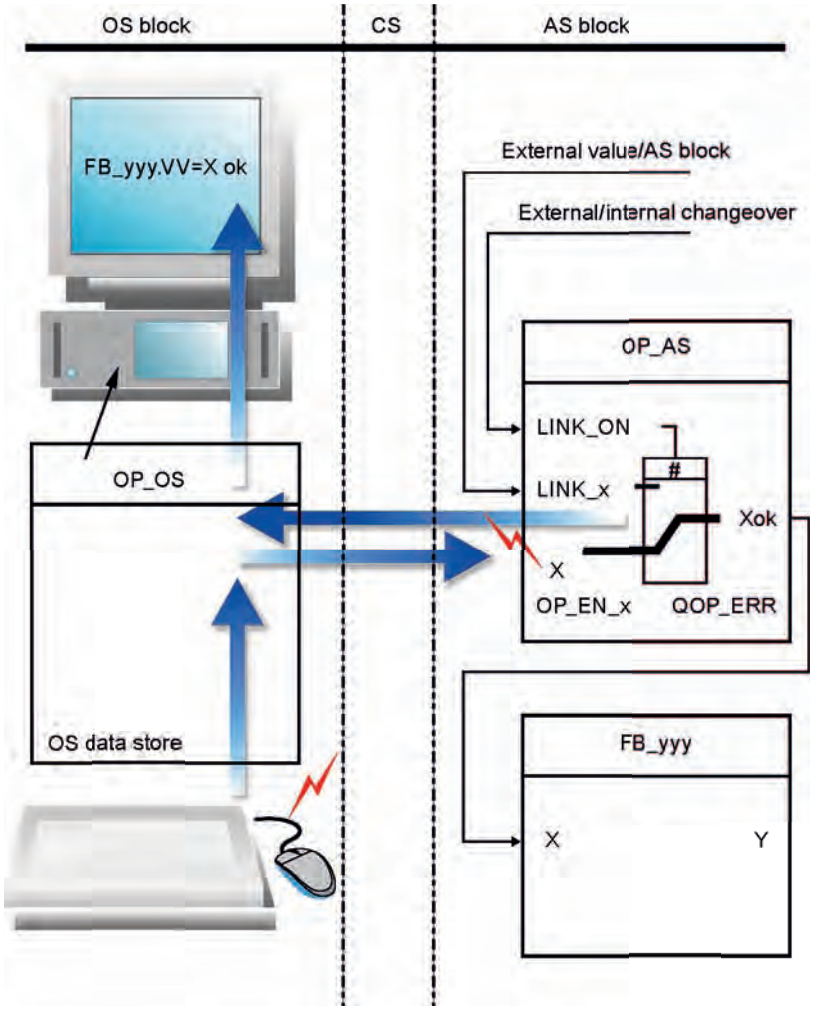


Figure 10-1 Concept of operator control

Concept of operator control

The operator control block must be used at two locations simultaneously:

- In the AS (AS block, abbreviated as OP_AS)
- In the OS (OS block, abbreviated as OP_OS)

The valid value supplied to output Xok by OP_AS can be obtained in two different ways:

- Externally, in other words it is provided by another AS block (via the LINK_x input)
- Internally, set by the operator using the OS block OP_OS

General operating procedure:

- The OP_OS selected on the OS queries the values of OP_AS or the enable/disable status of operator control. The display keeps the operator constantly informed of the current status of OP_AS. The result is displayed asynchronously to the OP_AS in an OS-specific runtime cycle.
- The operator controls/modifies an operator-controllable I/O of OP_OS. Its algorithm checks the validity of input data:
  - Invalid entries (block-specific) are corrected or rejected, depending on the situation. A corresponding message is output to the operator.
  - The corrected or valid value is transferred to OP_AS and logged on the OS.
- OP_AS receives the value and performs a validity check, since it could well be that the current AS status has changed since the last change made to OP_OS.
  - Entries that are now invalid (block-specific) are corrected or rejected, depending on the situation. OP_AS reports the result at the Boolean output QOP_ERR, in other words it outputs a pulse of a duration equal to the sampling time of OP_AS.
  - The corrected or valid value (or the old valid value if new values were rejected) is output for further use at the corresponding Xok output of OP_AS.

Overview

The table below provides an overview of the operator control blocks. These are implemented as FBs that require an instance DB for each application.

Block name	Meaning	Operating method	FB no.
OP_A	Operator control of analog value		45
OP_A_LIM	Operator control of analog value	Limiting	46
OP_A_RJC	Operator control of analog value	Rejecting	47
OP_D	Digital value control, 2 buttons		48
OP_D3	Digital value control, 3 buttons		49
OP_TRIG	Digital value control, button function		50

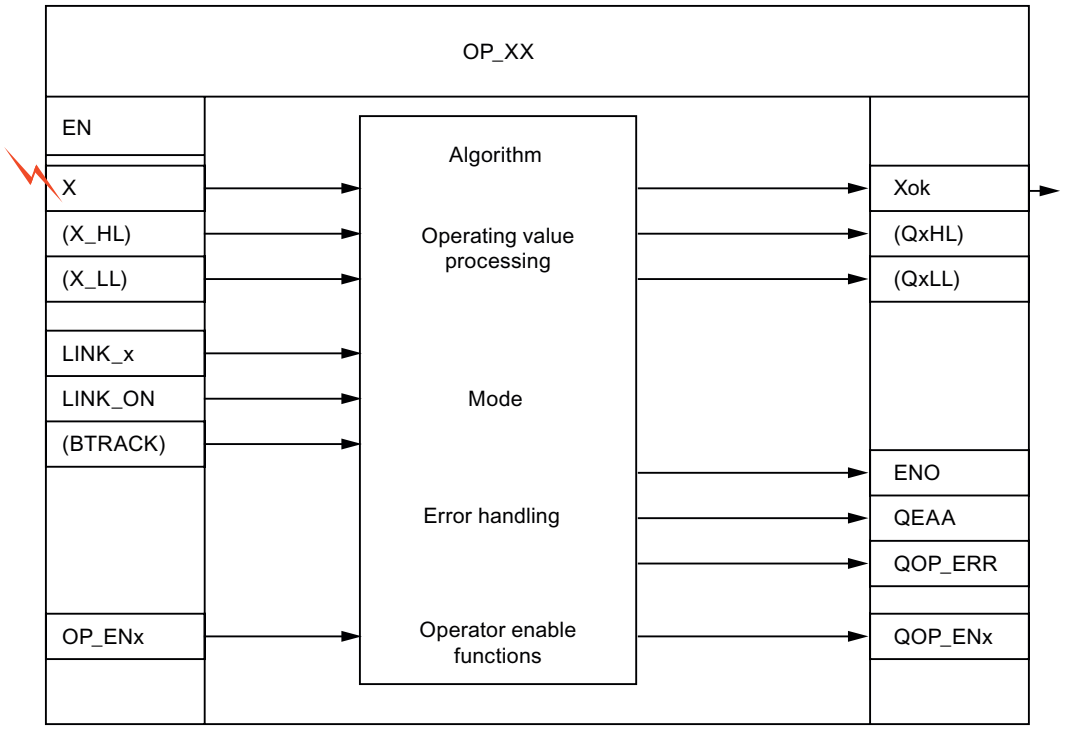
General I/Os

The operator control blocks (refer to the figure above) have precisely defined I/Os for control of binary and analog values. The significance of their I/Os is the same for all operator control blocks as well for operator control blocks that process only analog values (in the picture indicated by brackets).

The function of these I/Os is described briefly below.

Inputs

- EN is used to set/reset the block algorithm.
  - EN = 1: The block is called from the OB in which it is installed.
  - EN = 0: The block call is skipped in the OB.
- X (representative identifier for the operator controlled input) is written as an IO type by the OS, sampled by the AS block and overwritten, if appropriate. This input is retroactive. It may not be interconnected.
- X_HL and X_LL define the high and low adjustment limits of X (only for analog value operation). Values set by the operator that violate these limits will either be limited or rejected, depending on the type of the operator control block.



Operator control block structure (general)

- LINK_x is interconnected with an external value. This provides the external alternative to input X that is supplied internally (by the operator) (see also LINK_ON).
- LINK_ON changes the mode that selects the value to be processed:
  - LINK_ON = 1: The value at input LINK_x, which is usually received from another interconnected block, is treated as an external default value.
  - LINK_ON = 0: The value at input X, which is usually entered on the OS, is treated as an internal default value (from its own OP_XX).
- BTRACK (if it exists) is used for bumpless changeover during the transition of LINK_ON = 1 to LINK_ON = 0.
  - BTRACK = 1: When LINK_ON = 1, the algorithm ensures that the operator controlled inputs X track the LINK_x inputs. The object of this operation is to ensure that the block does not process any old operating values of the X inputs and thus change active values (that have been output) during the changeover to manual mode (LINK_ON = 0).
  - BTRACK = 0: When LINK_ON = 1, the operator controlled inputs X are not overwritten and normally therefore remain different from the LINK_x value. During the changeover LINK_ON = 0 these old values become valid again and lead to corresponding changes of the active Xok output values (referred to as bump).
- The OP_EN_x parameter is used to enable/disable operator control of the assigned input X:
  - OP_EN_x = 1: Input X is enabled for operator control.
  - OP_EN_x = 0: Operator control is blocked or rejected.

Outputs

- ENO indicates the validity of the result Xok (1 = OK, 0 = invalid).
- QERR = Inverted ENO (stored in the block instance).
- A logic "1" at the QOP_ERR parameter indicates an operator input error. The output will be reset again after one cycle (sampling time).
- Xok (representative identifier for the effective active output). This output contains the valid output value, depending on the operator control block type and mode. It is made available by interconnecting to the AS block whose input is to be adjusted. Depending on the operator control block, the specific identifier is then V or Q_1, for example.
- QxHL and QxLL indicate violation of the high/low adjustment limits for X (analog value adjustments only). Values entered by the operator that violate these limits will either be limited or rejected, depending on the type of the operator control block.
- QOP_ENx contains the output value that is passed on to OP_ENx. It can be queried by other AS blocks, and provides information on the current enable/disable status of operator control of OP_AS.

10.2 OP_A: Analog value operation

10.2.1 Description of OP_A

Object name (type + number)

FB45

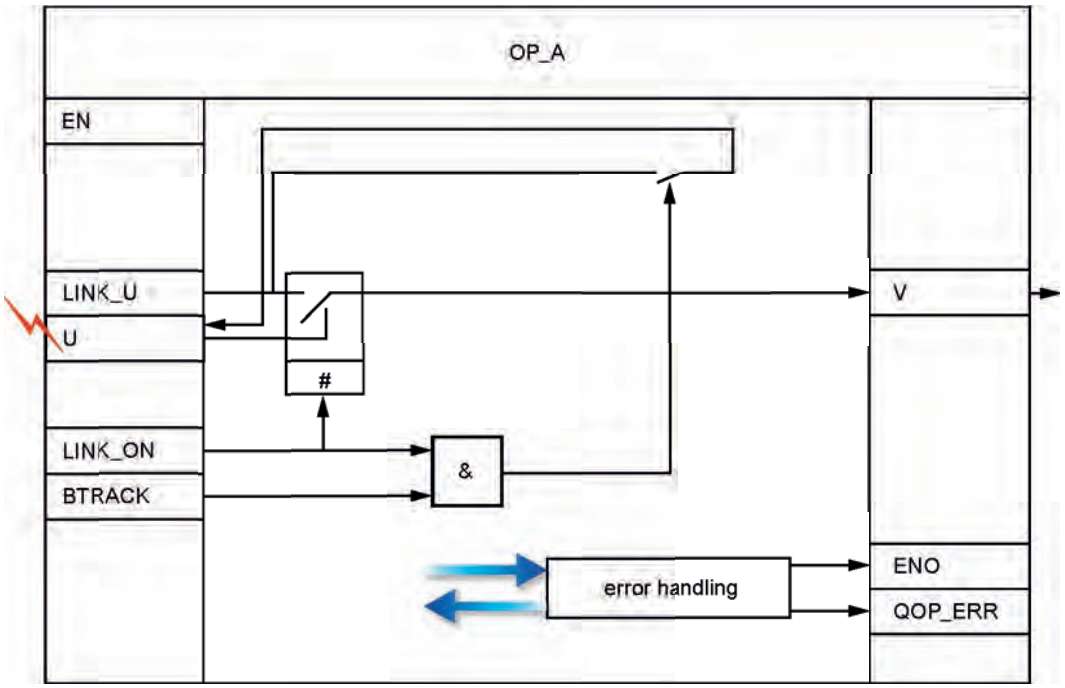
- OP_A block I/Os (Page 468)
- OP_A block icon (Page 597)
- OP_A faceplate (Page 572)

Function

The OP_A block is a basic operator control block for adjusting an analog value of an AS block, without limit monitoring and operator control enable.

How it works

The OP_A block operates according to the following principle:



OP_A structure

- The value of U is written by the operator on the OS.
- LINK_U is supplied with an external value (configured or interconnected).
- LINK_ON enables the external/internal value:
  - LINK_ON = 1: LINK_U is passed to V.
  - LINK_ON = 0: U is passed to V.
- BTRACK allows tracking of an operator controlled input U (only if LINK_ON = 1).
  - BTRACK = 1: U tracks the value of LINK_U. This ensures that a bump will not occur at output V during a transition to LINK_ON = 0.
  - BTRACK = 0: U retains its previous (operator entered) value. This becomes active again after the transition to LINK_ON = 0.

Calling OBs

The operator control block must be installed in the same OB before the block that evaluates operator input.

Error handling

The following error indications exist:

- ENO = 0, system indication only (no particular handling in the block)
- QOP_ERR = 1 is set for the duration of one cycle, if a change is made at input U while an active external value (LINK_ON = 1) is set. Input U retains its previous value (prior to operator intervention).

Error indications of OP_A

ENO	QOP_ERR	Cause, reaction (if applicable)
0	X	Errors detected by the system (no special handling routine in the block)
1	1 (π)	Operator input not permitted if LINK_ON=1. Input U retains its value.

π: Pulse with sampling time duration

X: Any value

Time response

Not applicable.

Message response

Not applicable. The output QOP_ERR can be interconnected with a message block in order to report operator errors. You will find more information in "Message blocks (Page 487)".

Additional information

You will find more information in:

I/Os of OP_A (Page 468)

Operating and monitoring OP_A (Page 468)

10.2.2 I/Os of OP_A

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
BTRACK	1 = Bumpless changeover	BOOL	1	I	+
LINK_ON	0 = operator control active, 1 = interconnection active	BOOL	0	I	+
LINK_U	Interconnectable input for U	REAL	0.0	I	
QOP_ERR	1 = operator error	BOOL	1	O	
U	Operator input, analog value	REAL	0.0	IO	+
V	Analog value	REAL	0.0	O	+

10.2.3 Operator Control and Monitoring of OP_A

Additional information

Additional information is available in the following sections:

- OP_A block icon (Page 597)
- OP_A faceplate (Page 572)

10.3 OP_A_LIM: Analog value operation (limiting)

10.3.1 Description of OP_A_LIM

Object name (type + number)

FB46

- OP_A_LIM block I/Os (Page 471)
- OP_A_LIM block icon (Page 597)
- OP_A_LIM faceplate (Page 572)

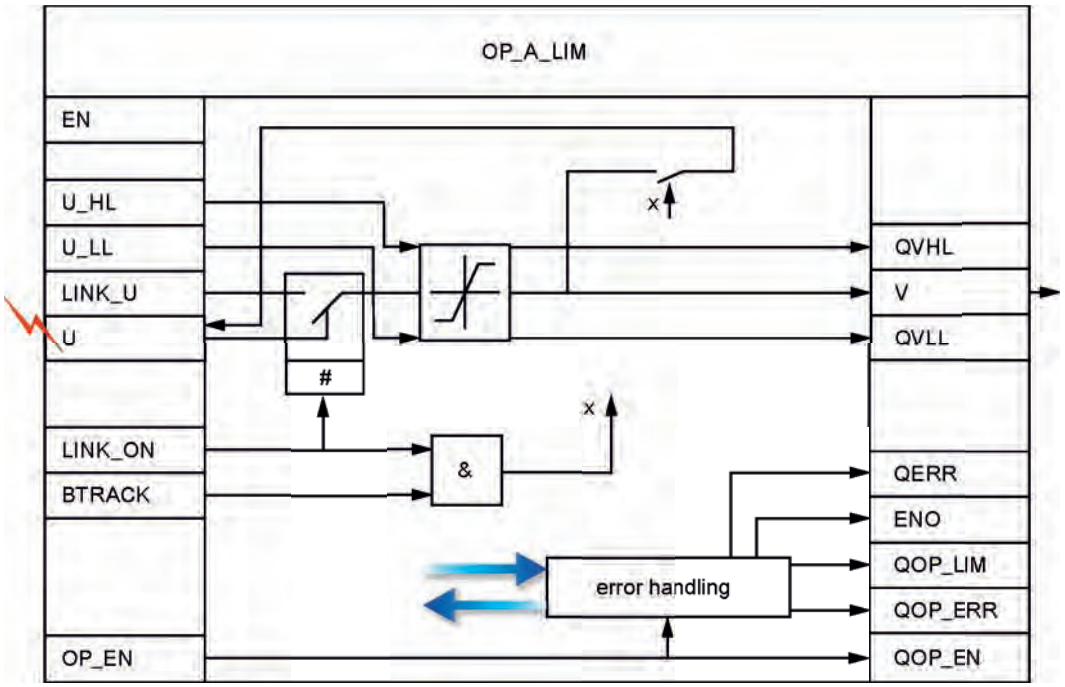
Function

The OP_A_LIM (**op**eration **a**nalog **lim**ited) block is used to control an analog value of a block.

Operator control outside the operating limits is limited to the relevant violated limit value. Instead of the value (U) set by the operator, an interconnected or configured value (LINK_U) can be checked (LINK_ON = 1).

How it works

Operating principle of the block:



OP_A_LIM structure

- The value of U is written by the operator on the OS. Operator control is set as follows:
  - enabled if OP_EN = 1,
  - disabled if OP_EN = 0
- LINK_U is supplied with an external value (configured or interconnected).
- LINK_ON passes the limited external/internal value to U_LL or U_HL:
  - LINK_ON = 1: The limited value LINK_U is passed to V
  - LINK_ON = 0: The limited value U is passed to V and written back to input U, in other words, input U may change due to a modification of the limits, without operator input.
- BTRACK allows the operator controlled input U to track LINK_U (only if LINK_ON = 1)
  - BTRACK = 1: The operator input U tracks the limited value LINK_U. This ensures that a bump will not occur at output V during a transition to LINK_ON = 0.
  - BTRACK = 0: U retains its previous (operator entered) value. It is passed to output V again after the transition to LINK_ON = 0.

Calling OBs

The operator control block must be installed in the same OB and before the block that uses the operator input.

Error handling

The following errors are displayed:

ENO	QOP_ERR	QOP_LIM	Cause, reaction (if applicable)
0	X	X	Errors recognized by the system
1	1 (π)	0	Operator input not permitted if LINK_ON=1. Input U retains its value.
1	1 (π)	0	Operator input was enabled on the OS (OP_EN=1), but has been disabled on the AS in the meantime (OP_EN=0).
1	1 (π)	1 (π)	Enabled operator control outside of limits. Input U is limited.

π: Pulse with sampling time duration
X: Any value

Error displays of OP_A_LIM (limitations)

QVLL	QVHL	Cause
1	0	Input value < U_LL (input value = U or LINK_U)
0	1	Input value > U_HL (input value = U or LINK_U)

Time response

Not applicable.

Message response

Not applicable. Outputs QOP_ERR or QOP_LIM can be interconnected with a message block in order to report operator errors. You will find more information in "Message blocks (Page 487)".

Additional information

You will find more information in:

I/Os of OP_A_LIM (Page 471)

Operating and monitoring OP_A_LIM (Page 472)

10.3.2 I/Os OP_A_LIM

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
BTRACK	1 = bumpless changeover	BOOL	1	I	+
LINK_ON	0 = operator control active, 1 = interconnection active	BOOL	0	I	+
LINK_U	Interconnectable input for U	REAL	0.0	I	
OP_EN	1 = operator input enabled	BOOL	1	I	
QERR	1 = processing error	BOOL	1	O	+
QOP_EN	1 = operator-control enable	BOOL	0	O	+
QOP_ERR	1 = operator error	BOOL	0	O	
QOP_LIM	1 = operator error, limiting	BOOL	0	O	
QVHL	1 = high limit active	BOOL	0	O	
QVLL	1 = low limit active	BOOL	0	O	
U	Operator input, analog value	REAL	0.0	IO	+
U_HL	High limit	REAL	100.0	I	+
U_LL	Low limit	REAL	0.0	I	+
V	Analog value	REAL	0.0	O	+

10.3.3 Operator Control and Monitoring of OP_A_LIM

Additional information

Additional information is available in the following sections:

- OP_A_LIM block icon (Page 597)
- OP_A_LIM faceplate (Page 572)

10.4 OP_A_RJC: Analog value operation (rejecting)

10.4.1 Description of OP_A_RJC

Object name (type + number)

FB47

- OP_A_RJC block I/Os (Page 475)
- OP_A_RJC block icon (Page 597)
- OP_A_RJC faceplate (Page 572)

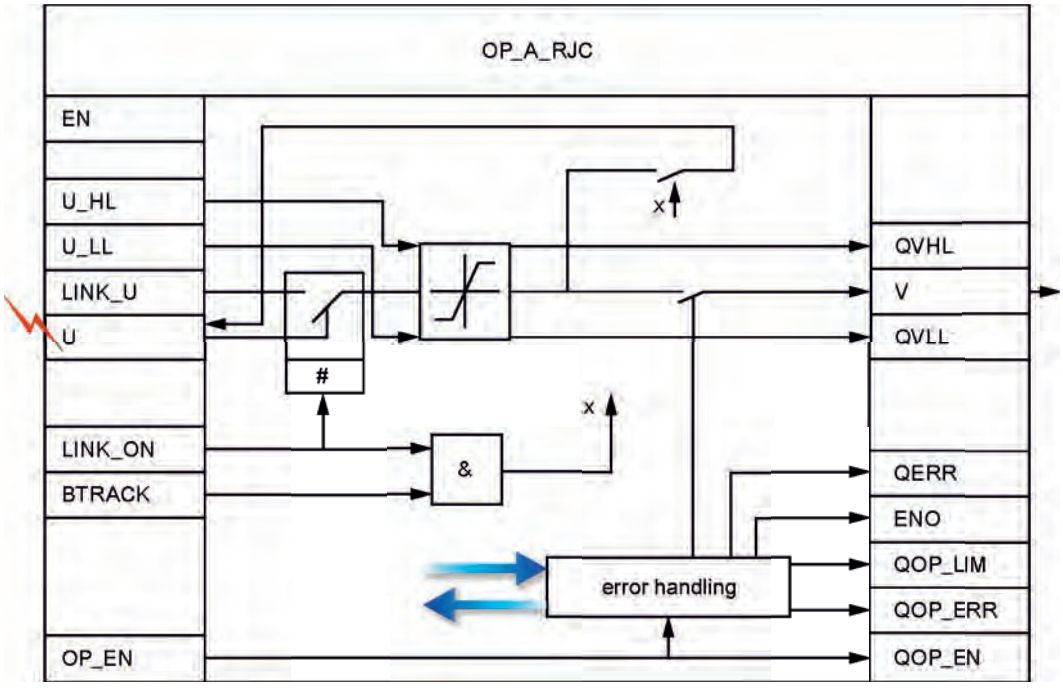
Function

The OP_A_RJC (**op**eration **a**nalog **rejec**ted) block is used to control an analog value of a block.

Any operation outside the operating limits will be discarded. Instead of the entered value (U), an interconnected or configured value (LINK_U) can be checked (LINK_ON = 1). In this case, the block limits the value according to OP_A_LIM.

How it works

Operating principle of the block:



OP_A_RJC structure

- The value of U is written by the operator on the OS. Operator control is set as follows:
  - enabled if OP_EN = 1,
  - disabled if OP_EN = 0
- LINK_U is assigned an external value (configured or interconnected).
- LINK_ON passes the limited external/internal value to U_LL or U_HL:
  - LINK_ON = 1: Limited LINK_U value is passed to V.
  - LINK_ON = 0: The old (limited) U value is passed to V and written back to input U, in other words input U may change due to a change of operator input limits and without operator intervention.
- BTRACK allows tracking of an operator controlled input U (only if LINK_ON = 1).
  - BTRACK = 1: The operator input U tracks the limited value LINK_U. This ensures that a bump will not occur at output V during a transition to LINK_ON = 0.
  - BTRACK = 0: U retains its previous (operator entered) value. It is passed to output V again after the transition to LINK_ON = 0.

Calling OBs

The operator control block must be installed in the same OB and before the block that uses the operator input.

Error handling

The following errors are displayed:

ENO	QOP_ERR	QOP_RJC	Cause, reaction (if applicable)
0	X	X	Errors recognized by the system
1	1 (π)	0	Operator input not permitted if LINK_ON=1. Input U retains its value.
1	1 (π)	0	Operator input was enabled on the OS (OP_EN=1), but has been disabled on the AS in the meantime (OP_EN=0).
1	1 (π)	1 (π)	Enabled operator control outside of limits. Operator input is discarded.

π : Pulse with sampling time duration

X: Any value

Error displays of OP_A_RJC (limitation only if LINK_ON=1)

QVLL	QVHL	Cause
1	0	Input value < U_LL (input value = LINK_U)
0	1	Input value > U_HL (input value = LINK_U)

Time response

Not applicable.

Message response

Not applicable. Outputs QOP_ERR or QOP_RJC can be interconnected with a message block in order to report operator errors. You will find more information in "Message blocks (Page 487)".

Additional information

You will find more information in:

I/Os of OP_A_RJC (Page 475)

Operating and monitoring OP_A_RJC (Page 476)

10.4.2 I/Os of OP_A_RJC

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
BTRACK	1 = bumpless changeover	BOOL	1	I	+
LINK_ON	0 = operator control active, 1 = interconnection active	BOOL	0	I	+
LINK_U	Interconnectable input for U	REAL	0.0	I	
OP_EN	1 = operator-control enable	BOOL	1	I	
QERR	1 = processing error	BOOL	1	O	+
QOP_EN	1 = operator-control enable	BOOL	0	O	+
QOP_ERR	1 = Operator error, rejecting	BOOL	0	O	
QOP_RJC	1 = operator error rejecting	BOOL	0	O	
QVHL	1 = high limit active	BOOL	0	O	
QVLL	1 = low limit active	BOOL	0	O	
U	Operator input, analog value	REAL	0.0	IO	+
U_HL	High limit	REAL	100.0	I	+
U_LL	Low limit	REAL	0.0	I	+
V	Analog value	REAL	0.0	O	+

10.4.3 Operator Control and Monitoring of OP_A_RJC

Additional information

You will find additional information in the following sections:

- OP_A_RJC block icon (Page 597)
- OP_A_RJC faceplate (Page 572)

10.5 OP_D: Operator input of digital values (two-button control)

10.5.1 Description of OP_D

Object name (type + number)

FB48

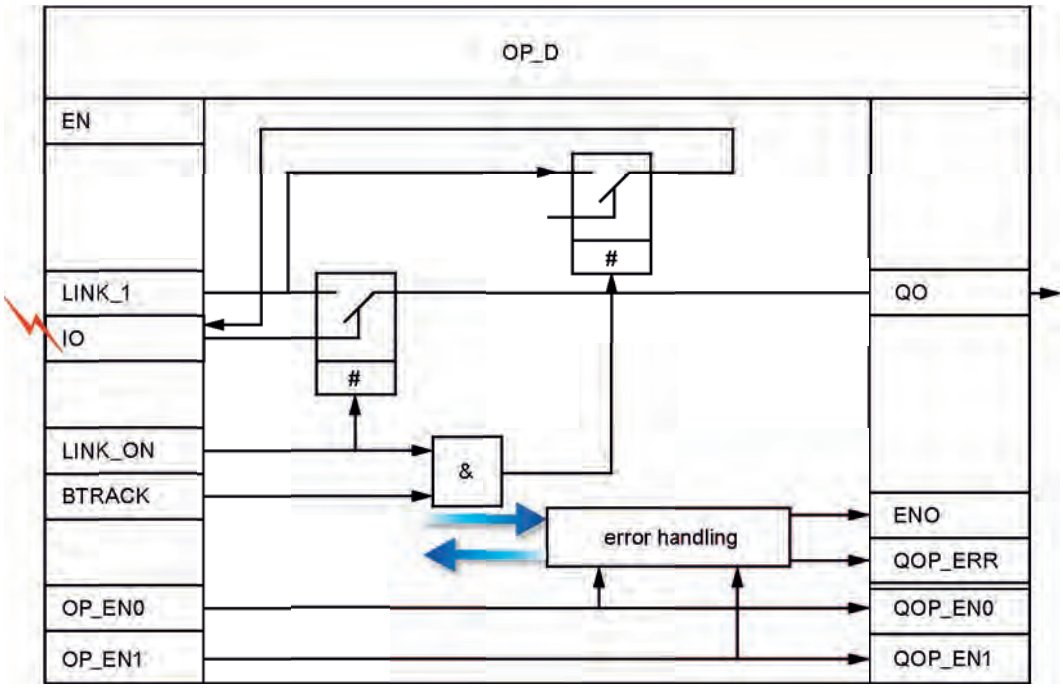
- OP_D block I/Os (Page 479)
- OP_D block icon (Page 598)
- OP_D faceplate (Page 573)

Function

The operator control block OP_D is used to control a digital value of a block by means of two buttons. If the operator enters a valid value, it is output to the Q output.

How it works

Operating principle of the block:



OP_D structure

- I0 is written via by the operator at the OS. This is enabled by two separate inputs:
  - OP_EN0 = 1 for setting "0",
  - OP_EN1 = 1 for setting "1".
- LINK_I is supplied with a configured or interconnected external value.
- LINK_ON enables the external/internal value:
  - LINK_ON = 1: LINK_U is passed to Q0,
  - LINK_ON = 0: The entered I0 value is passed to Q0.
- BTRACK allows the operator controlled input I0 to track LINK_I (only if LINK_ON = 1):
  - BTRACK = 1: The operator input I0 tracks LINK_I. This ensures that a bump does not occur at output Q0 during the transition of LINK_ON = 0.
  - BTRACK = 0: I0 retains its last (entered) value. This is passed to output Q0 again after the transition to LINK_ON = 0.

Calling OBs

The operator control block must be installed in the same OB and before the block that uses the operator input.

Error handling

The following errors are displayed:

ENO	QOP_ERR	Cause, reaction (if applicable)
0	X	Errors detected by the system (no special handling routine in the block)
1	1 (π)	Operator control was not enabled or performed while LINK_ON=1. Input I0 retains its value.

π : Pulse with sampling time duration

X: Any value

Time response

Not applicable.

Message response

Not applicable. The output QOP_ERR can be interconnected with a message block in order to report operator errors. You will find more information in "Message blocks (Page 487)".

Additional information

You will find more information in:

I/Os of OP_D (Page 479)

Operating and monitoring OP_D (Page 479)

10.5.2 I/Os of OP_D

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible. You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
BTRACK	1 = bumpless changeover	BOOL	1	I	+
I0	Operator input 0	BOOL	0	IO	+
LINK_I	Interconnectable input for I	BOOL	0	I	
LINK_ON	0 = operator control active, 1 = interconnection active	BOOL	0	I	+
OP_EN0	Deactivate operator control enable input	BOOL	1	I	
OP_EN1	Activate operator control enable input	BOOL	1	I	
Q0	Binary output	BOOL	0	O	+
QOP_EN0	Deactivate operator control enable output	BOOL	0	O	+
QOP_EN1	Activate operator control enable output	BOOL	0	O	+
QOP_ERR	1 = operator error	BOOL	0	O	

10.5.3 Operator Control and Monitoring of OP_D

Additional information

You will find additional information in the following sections:

- OP_D block icon (Page 598)
- OP_D faceplate (Page 573)

10.6 OP_D3: Operator input of digital values (3-button control)

10.6.1 Description of OP_D3

Object name (type + number)

FB49

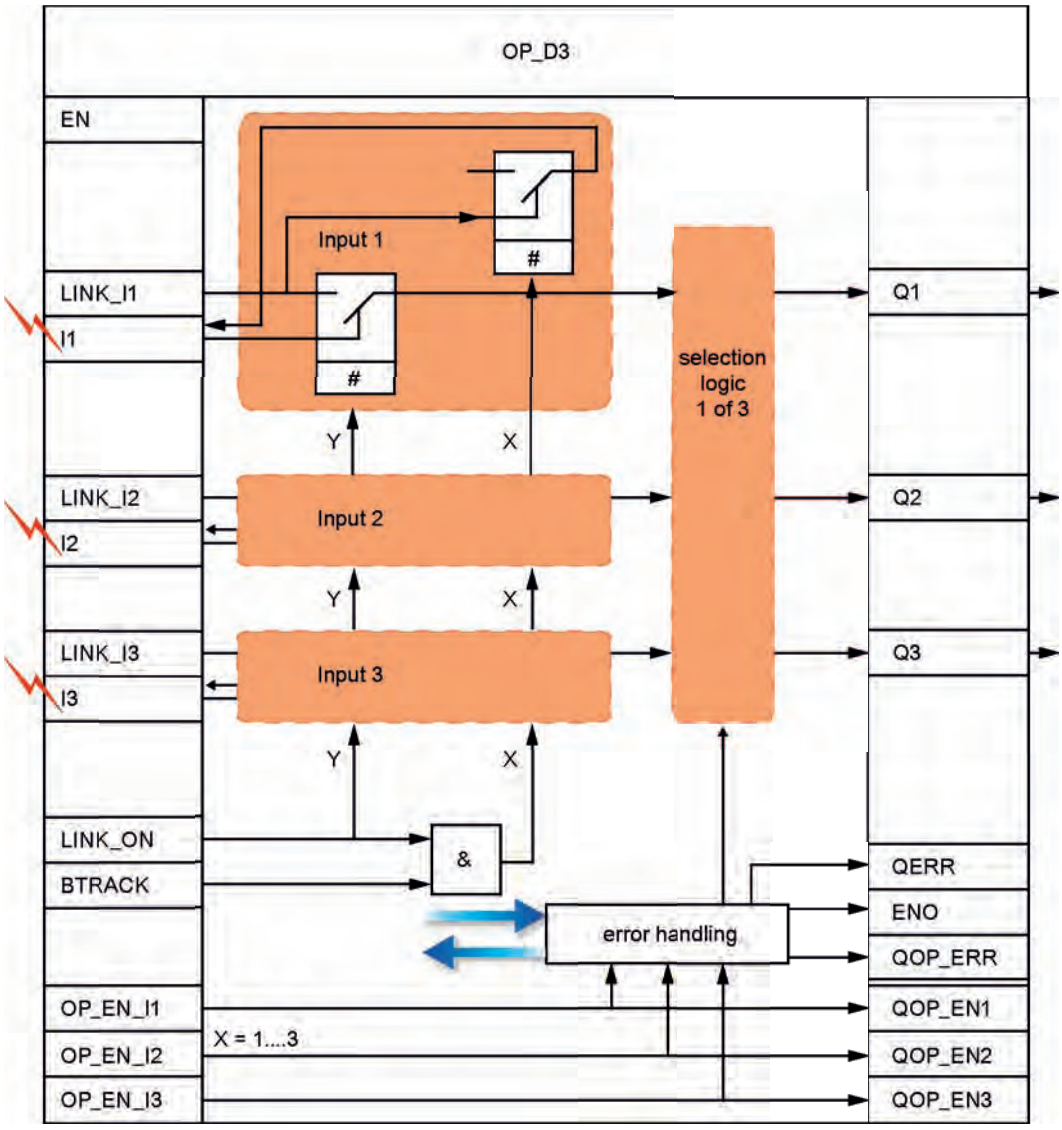
- OP_D3 block I/Os (Page 483)
- OP_D3 block icon (Page 598)
- OP_D3 faceplate (Page 574)

Function

The operator control block OP_D3 is used to perform a logical one-out-of-three digital value operation. If one of the three operator inputs I1, I2 or I3 is set, the corresponding output is set to 1 and the other outputs are reset, assuming operator input is permissible.

How it works

The block operates according to the following principle. The expression x = 1..3 is used here as an index for the relevant three inputs/outputs:



OP_D3 structure

- The OS control system sets the inputs I1, I2 and I3 simultaneously ("1" for the input to be activated and "0" for the other two). Three separate inputs are used for enabling/disabling operator control:
  - OP_EN_Ix = 1: Enables operator control of input Ix
  - OP_EN_Ix = 0: Disables operator control of input Ix
- Each LINK_Ix is supplied with an external configured or interconnected value.

- LINK_ON enables the external/internal values:
  - LINK_ON = 1: LINK_Ix are processed and passed to Qx.
  - LINK_ON = 0: Operator controlled Ix inputs are processed and passed to Qx.
- BTRACK allows tracking of the operator controlled inputs Ix (only if LINK_ON = 1):
  - BTRACK = 1: The operator controlled inputs Ix are tracked to LINK_Ix. This ensures that a bump does not occur at output Qx during the transition of LINK_ON = 0.
  - BTRACK = 0: Ix retains its last (operated) value. This is passed to output Qx again after the transition to LINK_ON = 0.
- The selection logic takes the three input values (Ix or LINK_Ix) in the order x = 1, 2, 3 and notes the highest index "x" of the input with logical "1" status. The output Qx specified by the index will be set ("1"), and the remaining two outputs Qx will be reset ("0"). If all three inputs I1 = I2 = I3 = 0, the outputs are not changed.

Calling OBs

The operator control block must be installed in the same OB and before the block that uses the operator input.

Error handling

The following errors are displayed:

ENO	QOP_ERR	Cause, reaction (if applicable)
0	X	Errors recognized by the system
0	0	All inputs are "0" or more than one input is "1". The output is set according to the highest set input. You will find more information in the sections "How it works" and "Selection logic".
0	1	More than one input is "1". The output is set according to the highest set input. You will find more information in the sections "How it works" and "Selection logic". The error handling routine also changes the inputs Ix according to the rule: "The Ix input with the highest index "x" remains set, the other will be reset".

Time response

Not applicable.

Message response

Not applicable. The output QOP_ERR can be interconnected with a message block in order to report operator errors. You will find more information in "Message blocks (Page 487)".

Additional information

You will find more information in:

I/Os of OP_D3 (Page 483)

Operating and monitoring OP_D3 (Page 483)

10.6.2 I/Os of OP_D3

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
BTRACK	1 = bumpless changeover	BOOL	1	I	+
I1	Operator input 1	BOOL	0	IO	+
I2	Operator input 2	BOOL	0	IO	+
I3	Operator input 3	BOOL	1	IO	+
LINK_I1	Interconnectable input for I1	BOOL	0	I	
LINK_I2	Interconnectable input for I2	BOOL	0	I	
LINK_I3	Interconnectable input for I3	BOOL	0	I	
LINK_ON	0 = operator control active, 1 = interconnection active	BOOL	0	I	+
OP_EN_I1	Operator control enable input 1	BOOL	1	I	
OP_EN_I2	Operator control enable input 2	BOOL	1	I	
OP_EN_I3	Operator control enable input 3	BOOL	1	I	
Q1	Binary output 1 (SWITCH1)	BOOL	0	O	+
Q2	Binary output 2 (SWITCH2)	BOOL	0	O	+
Q3	Binary output 3 (SWITCH3)	BOOL	1	O	+
QERR	1 = processing error	BOOL	1	O	+
QOP_EN1	Operator control enable output 1	BOOL	0	O	+
QOP_EN2	Operator control enable output 2	BOOL	0	O	+
QOP_EN3	Operator control enable output 3	BOOL	0	O	+
QOP_ERR	1 = operator error	BOOL	0	O	

10.6.3 Operator Control and Monitoring of OP_D3

Additional information

You will find additional information in the following sections:

- OP_D3 block icon (Page 598)
- OP_D3 faceplate (Page 574)

10.7 OP_TRIG: Operator input of digital values (single-button control)

10.7.1 Description of OP_TRIG

Object name (type + number)

FB50

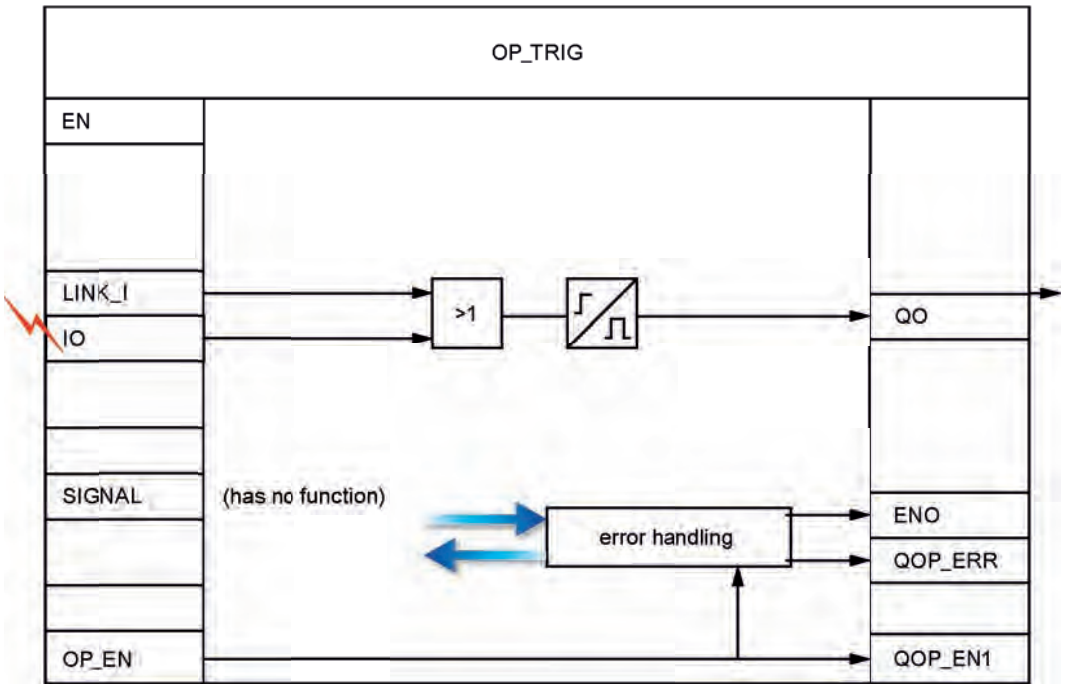
- Block I/Os (Page 486)
- OP_TRIG block icon (Page 598)
- OP_TRIG faceplate (Page 575)

Function

Operator control block OP_TRIG is used to implement single pushbutton control (comparable with RESET pushbutton).

How it works

Operating principle of the block:



OP_TRIG structure

- The value 1 is written to I0 by the operator if this was permitted by OP_EN = 1. Output Q0 is set to 1 for one cycle (sampling time) and then reset again. The operator input I0 is reset by the operator control block after processing.
- The interconnectable input (LINK_I) is redundant to the operator input. On its positive edge a logical "1" is set at output Q0 for the duration of one cycle (sampling time) and is then reset. LINK_I does not have any influence on the operator control enable QOP_EN.
- The block has an interconnectable input (SIGNAL) that is displayed on the OS. It does not have any function and is only used for the OS display. Here, it is advisable to interconnect the signal to be reset, since it does not make any sense to use of the output signal Q0 of the block that is set for only one cycle.

Calling OBs

The operator control block must be installed in the same OB and before the block that uses the operator input.

Error handling

The following errors are displayed:

ENO	QOP_ERR	Cause, reaction (if applicable)
0	X	Errors detected by the system (no special handling routine in the block)
1	1 (π)	Operator control is not enabled. Input I0 is set to "0".

π: Pulse with sampling time duration
X: Any value

Time response

Not applicable.

Message response

Not applicable. The output QOP_ERR can be interconnected with a message block in order to report operator errors. You will find more information in "Message blocks (Page 487)".

Additional information

You will find more information in:

I/Os of OP_TRIG (Page 486)

Operating and monitoring OP_TRIG (Page 486)

10.7.2 I/Os of OP_TRIG

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameters)	Meaning	Data type	Default	Type	OCM	Permissible values
I0	Operator input	BOOL	0	IO	+	
LINK_I	Input interconnectable with I0	BOOL	0	I		
OP_EN	1 = operator-control enable	BOOL	1	I		
Q0	Binary output	BOOL	0	O		
QOP_EN	1 = operator-control enable	BOOL	0	O	+	
QOP_ERR	1 = operator error	BOOL	0	O		
SIGNAL	Interconnectable input for display on OS	BOOL	0	I	+	

10.7.3 Operator Control and Monitoring of OP_TRIG

Additional information

Additional information is available in the following sections:

- OP_TRIG block icon (Page 598)
- OP_TRIG faceplate (Page 575)

Family: MESSAGE

11

11.1 Overview of Message Blocks

What Message Blocks Are Used For

The operator may require information on events involving changes of a digital value or status in the AS. Due to the message blocks implemented in the AS, there is not need for the OS system to poll the AS in order to obtain this information. These blocks monitor the digital values and report changes to the OS (including additional and configurable information). The OS system can visualize, log and archive this information.

The table shows an overview of the message blocks, which are implemented as FBs.

Overview of Message Blocks

Object Name	Type Name	Meaning	Operation Method
43	MESSAGE	Generation of configurable messages	SIMATIC Process Control - Standard
59	MSG_NACK	Generation of user-specific messages which do not require acknowledgement	SIMATIC Process Control - Standard

The adaptation of messages for individual blocks is described under "Message Response".

11.2 MSG_NACK: User-specific messages (which do not require acknowledgment)

11.2.1 Description of MSG_NACK

Object name (type + number)

FB 78

- MSG_NACK block I/Os (Page 490)

Area of application

The MSG_NACK block is used to generate user-specific messages, which do not require an acknowledgement (operating messages).

Function

The block can generate up to eight user-specific messages of this type.

How it works

Messages not requiring acknowledgement are output via NOTIFY_8P. The output of individual or all messages can be blocked.

- I_1 to I_8: Changes at these monitored signals are reported. A configurable message text is assigned to each of these signals. You can adapt this text and re-use it subsequently in the OS configuration.
  Each change at these inputs causes a message to be sent to the OS (unless the function is blocked).
- MSG_LOCK: Allows process-specific locking of messages output from this block. At the positive edge of the lock signal, all active process messages are reset, and thus sent to the OS as exited state.
- AUX_PR01 to AUX_PR10: These inputs can be interconnected with any values of any data type. These values are referred to as associated values, are limited to 16 characters, and are included in the message sent to the OS, thus providing more detailed information on the event triggering the message.
- The operator can lock the messages of a process tag on the OS. The OS reports this status to the corresponding message block, which returns a confirmation (via its NOTIFY_8P) to the OS. The message is then entered as acknowledged and exited state in the message event log of the OS.
- QMSG_SUP indicates that message suppression is enabled.
- The OS evaluates MSG_STAT, QMSG_ERR, and MSG_ACK.

Calling OBs

The message block must be installed in the OB where reporting occurs (e.g. OB 35) and also in OB 100.

Error handling

Error handling of block MSG_NACK is limited to the evaluation of the error information of ALARM_8P. You will find more information in the manual *System Software for S7-300/400 - System and Standard Functions*.

You can find information about the error messages associated with the MSG_STAT parameter in the online help under *NOTIFY_8P; STATUS Parameter*.

Startup characteristics

During startup, the message block suppresses all messages. The duration (number of cycles) of message suppression is set in the RUNUPCYC parameter. An internal counter that is initialized with this parameter value during restart decrements each time the block is executed. Messages are not generated if the counter value does not equal zero.

Messages

Messages are generated via NOTIFY_8P (SFB 31). NOTIFY_8P is assigned 8 digital inputs and 10 associated values. Every edge transition detected at one or more digital inputs triggers a message. The associated values are consistently assigned to the message at the time of edge evaluation. All eight signals are assigned a common message number (MSG_ID), which is subdivided at the OS into 8 messages. The ES assigns the message number automatically by calling the message server.

Message text

Each block message has a default message text and is assigned to an internal or external block parameter and to a message class (operating message – without acknowledgment). You can change the message texts as part of configuration.

Associated values

The associated values can be assigned in differing numbers and sequences to every message. Associated values not used in the block algorithm can be interconnected freely as input parameters AUX_PRxx at the block.

Permitted data types for associated values: BOOL, BYTE, WORD, DWORD, CHAR, INT, DINT, REAL, and ARRAY OF BYTE

11.2.2 I/Os of MSG_NACK

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
AUX_PRxx	(xx = 01 - 10) Associated value 1 - 10	*1)	0	IO	
EV_ID	Message number NOTIFY_8P (assigned by the ES)	DWORD	0	I	
I_x	(x = 1 - 8) Input 1 - 8	BOOL	0	I	
EN_I_x	(x = 1 - 8) 1 = enable message 1 - 8	BOOL	1	I	
MSG_LOCK	Lock all messages	BOOL	0	I	
MSG_STAT	STATUS output of NOTIFY_8P	WORD	0	O	
OOS	Reserve	BOOL	0	I	+
QERR	1 = processing error	BOOL	1	O	
QMSG_ERR	ERROR output of NOTIFY_8P	BOOL	0	O	
QEN_I_x	(x = 1 - 8) 1 = enable message 1 - 8	BOOL	0	O	
QMSG_SUP	Process-message suppression via operator control is active	BOOL	0	O	
RUNUPCYC	Number of run-up cycles	INT	3	I	

*1) The associated values are assigned to NOTIFY_8P. Permitted data types for associated values: BOOL, BYTE, WORD, DWORD, CHAR, INT, DINT, REAL, and ARRAY OF BYTE

11.3 MESSAGE: Message block (configurable messages)

11.3.1 Description of MESSAGE

Object name (type + number)

FB43

- MESSAGE block I/Os (Page 493)

Function

The MESSAGE block is used to generate configurable (requiring acknowledgement) messages. It forms the interface between the block outputs whose changes are to be reported and the S7 block ALARM_8P.

How it works

The block inputs can be used to assign individual messages to the monitored signals and also to enable or block messages depending on the process status.

- I_1 to I_8: Changes at these monitored signals are reported. A configurable message text (24 characters) is assigned to each of these signals. You can adapt this text and re-use it subsequently in the OS configuration.
  Each change at these inputs causes a message to be sent to the OS (unless the function is blocked).
- I_1ISCSF to I_8ISCSF: By setting the parameter to "1", the associated message within the block is handled like a control system message (CSF). With this parameter, you specify whether the message of the corresponding input (I_1...I_8) is locked by the input parameter MSG_LOCK=1 (I_x = 0) or not locked (I_x = 1).
- MSG_LOCK: Allows process-specific locking of messages output from this block. On the positive edge of the lock signal, all active process messages (not control system messages) are reset, and reported as exited state to the OS.
- AUX_PR01 to AUX_PR10: These inputs can be interconnected to any values of any data type. These values are referred to as associated values, are limited to 16 characters and included in the message to the OS. You can use these to describe the cause of the message in detail.
- The operator can lock the messages of a process tag on the OS. The OS reports this status to the corresponding message block. This returns a confirmation of the lock (via its ALARM_8P) to the OS. The message is then entered as acknowledgement and exited state in the message event log of the OS.
- QMSG_SUP indicates that messages are being suppressed.
- The OS system evaluates MSG_STAT, QMSG_ERR and MSG_ACK.

Calling OBs

The block must be installed in the OB (for example OB32) in which the events to be monitored are detected and also in OB100.

Error handling

Error handling of the MESSAGE block is limited to the evaluation of the error information of ALARM_8P. You will find more information in the manual *System Software for S7-300/400 - System and Standard Functions*.

You can find information about the error messages associated with the MSG_STAT parameter in the online help under *NOTIFY_8P; STATUS Parameter*.

Startup characteristics

The message block suppresses all messages during startup, including control system messages. The duration (number of cycles) of message suppression is set in the RUNUPCYC parameter. An internal counter that is initialized with this parameter value during restart decrements each time the block is executed. Messages are not generated if the counter value does not equal zero.

Messages

Messages are generated by means of ALARM_8P (SFB35). All blocks use the PMC communication channel. ALARM_8P is assigned 8 digital inputs and 10 associated values. Every detected edge transition at one or more digital inputs triggers a message, irrespective of an acknowledgment. The associated values are assigned consistently to the message at the time of edge evaluation. All 8 signals are assigned a common message number (MSG_ID), which is subdivided on the OS into 8 messages. The ES assigns the message number automatically by calling the message server.

Message text

Each block message has a default message text and is assigned to an internal or external block parameter and to a specific message class. You can change the message texts and message class during configuration. The block algorithm is not affected by a change in the message class.

Additional information

You will find more information in:

I/Os of MESSAGE (Page 493)

Message texts and associated values of MESSAGE (Page 493)

11.3.2 I/Os of MESSAGE

The factory setting of the block display in CFC is identified in the "I/O" column:

I/O name **bold** = I/O visible, I/O name normal = I/O not visible.

You will find explanations of and information about the abbreviations used in "General information about the block description (Page 15)".

I/O (parameter)	Meaning	Data type	Default	Type	OCM
AUX_PRxx	(xx = 01 - 10) Associated value 1 - 10	*1)	0	IO	
EV_ID	Message number ALARM_8P (assigned by the ES)	DWORD	0	I	
I_x	(x = 1 - 8) Input 1 - 8	BOOL	0	I	
I_xISCSF	(x = 1 - 8) 1 = control-system message	BOOL	0	I	
MSG_ACK	ACK_STATE output of ALARM_8P	WORD	0	O	
MSG_LOCK	Lock messages dependent on process	BOOL	0	I	
MSG_STAT	STATUS output of ALARM_8P	WORD	0	O	
OOS	Reserve	BOOL	0	I	+
QERR	1 = processing error	BOOL	1	O	
QMSG_ERR	ERROR output of ALARM_8P	BOOL	0	O	
QMSG_SUP	Process-message suppression via operator control is active	BOOL	0	O	
RUNUPCYC	Number of run-up cycles	INT	3	I	

*1) The following data types are permitted for associated values: BOOL, BYTE, WORD, DWORD, CHAR, INT, DINT, REAL and ARRAY OF BYTE

11.3.3 Message Texts and Associated Values of MESSAGE

Assignment of message texts and message classes to the block parameters

Message no.	Block parameters	Default message text	Message class
1	I1	TEXT 1	F
:	:	:	:
8	I8	TEXT 8	F

Associated values

The associated values can be assigned in differing numbers and sequences to every message. Associated values not used in the block algorithm can be interconnected freely as input parameters AUX_PRx at the block.

Permitted data types for associated values: BOOL, BYTE, WORD, DWORD, CHAR, INT, DINT, REAL, and ARRAY OF BYTE

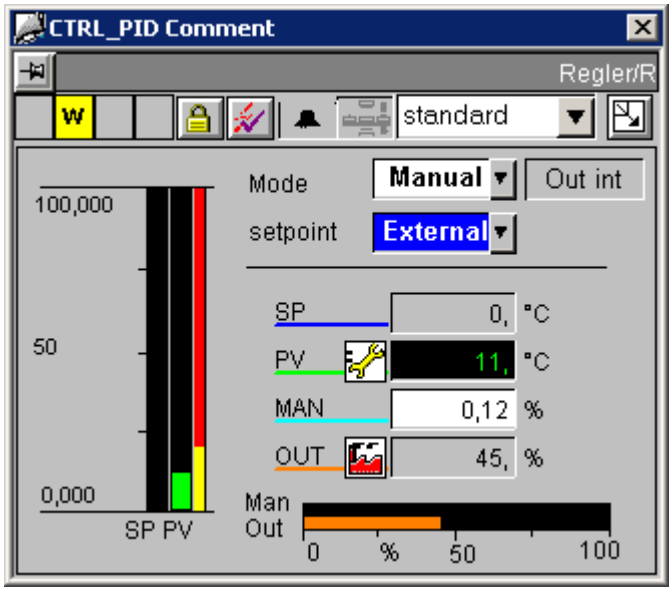
12 Faceplates

12.1 Faceplates: Plant blocks

12.1.1 CTRL_PID (All Views)

Overview

12.1.2 CTRL_PID: Standard view



Analog displays and number formats

All analog displays are implemented using the "AdvancedAnalogDisplay" object. The number format is defined via the "Format_InputValue" and "Format_OutputValue" properties of the block symbol.

Control permissions

This view has the following 2 "permission" objects for the input of setpoints and manipulated variables, since control permissions for these variables depend upon various factors:

- "Permission_Setpoint"
- "Permission_Manual"

In addition to the WinCC authorization levels, the permission objects also evaluate the parameters listed below:

Permission object	Parameters
"Permission_Setpoint"	"Q_SP_OP = TRUE"
"Permission_Manual"	"QLMNOP = TRUE"

Operating the PID tuner and tuning

The PID tuner is operated in the parameter view (Tuning On/Off).

When tuning is turned on in the parameter view, the standard view displays a combo box above the "Manual/Auto" mode combo box. This combo box can also be used to turn off tuning again from the standard view. All other control functions of the controller are disabled when "Tuning On" is set.

Sequence and positioning of direct connections to controllable objects

@Level5	-->	**Operator-control enable**		
Manual_COMBOBOX	-->	Operator-control enable		
External_COMBOBOX	-->	Operator-control enable		
Permission_Setpoint	-->	Level_Source	-->	Level_Target
Permission_Manual	-->	Level_Source		
Permission_Setpoint	-->	**Target_Operator-ControlEnable**		
Setpoint_AnalogValue	-->	Operator-control enable		
Permission_Manual	-->	**Target_Operator-ControlEnable**		
Manual_AnalogValue	-->	Operator-control enable		
Format	-->	**Format_InputValue**		
Setpoint_AnalogValue	-->	Format		
ProcessValue_AnalogValue	-->	Format		
Format	-->	**Format_OutputValue**		
Manual_AnalogValue	-->	Format		
Output_AnalogValue	-->	Format		

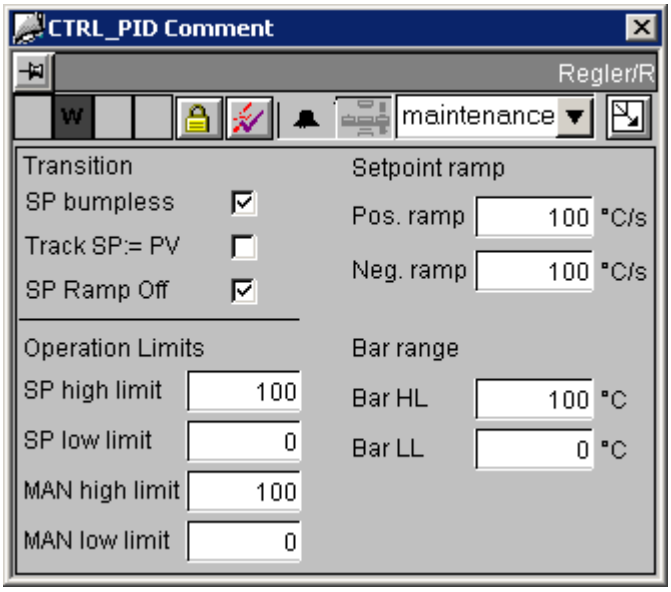
Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.3 CTRL_PID: Maintenance view



If a block of the type CPM is instantiated in the chart of the controller and this has the name if the controller block with the extension "_cpm", an additional button CPM is displayed with which the CPM faceplate can be called.

Control permissions

The "Permission_SP_Bumpless" object evaluates the WinCC permission levels and the "OPTI_EN = FALSE" parameter.

Sequence and positioning of direct connections to controllable objects

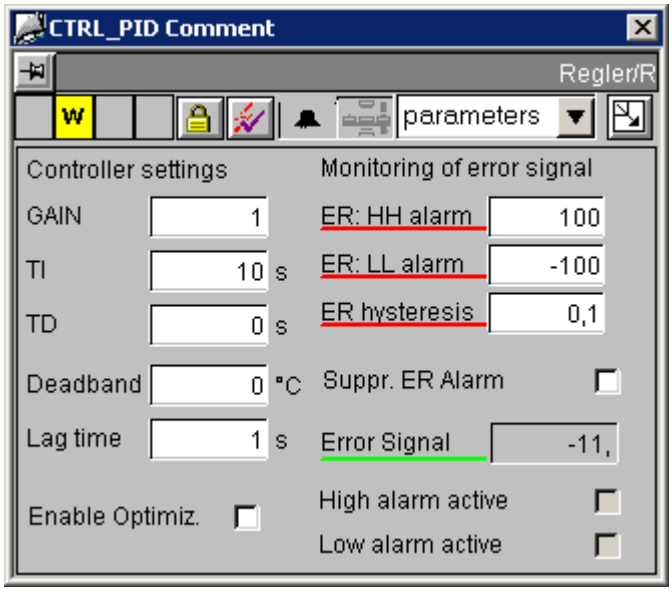
@Level6	-->	**Operator control enable**
Permission_SP_Bumpless	-->	Level_Source
Permission_SP_Bumpless	-->	**Target_Operator-ControlEnable**
Bumpless_CHECKBOX_L	-->	Operator control enable
SP_TRK_ON_CHECKBOX_L	-->	Operator control enable
SPRAMP_OFF_CHECKBOX_L	-->	Operator control enable
SPHighLimit_AnalogValue	-->	Operator control enable
SPLowLimit_AnalogValue	-->	Operator control enable
ManHighLimit_AnalogValue	-->	Operator control enable
ManLowLimit_AnalogValue	-->	Operator control enable
SPURLM_AnalogValue	-->	Operator control enable
SPDRLM_AnalogValue	-->	Operator control enable
MO_PVHR_AnalogValue	-->	Operator control enable
MO_PVLR_AnalogValue	-->	Operator control enable
Permission_SP_Bumpless	-->	**Target_BackColor**
SPHighLimit_AnalogValue	-->	BackColor_Value
SPLowLimit_AnalogValue	-->	BackColor_Value
ManHighLimit_AnalogValue	-->	BackColor_Value
ManLowLimit_AnalogValue	-->	BackColor_Value
SPURLM_AnalogValue	-->	BackColor_Value
SPDRLM_AnalogValue	-->	BackColor_Value
MO_PVHR_AnalogValue	-->	BackColor_Value
MO_PVLR_AnalogValue	-->	BackColor_Value

Additional information

You will find more information in:

Objects in the overview (Page 555)

12.1.4 CTRL_PID: Parameter view



If a block of the type GAIN_SHD is instantiated in the chart of the controller and this has the name of the controller block with the extension "_gsc", an additional button GAIN_SHD is displayed with which the GAIN_SHD faceplate can be called.

Analog displays and number formats

The "ControlError_AnalogValue" process value is implemented using the "AdvancedAnalogDisplay" object. The number format is defined at the "Format_InputValue" property of the block icon.

All other analog displays are implemented by means of the conventional "Floating-point format" I/O field.

Control permissions

The "Permission_Gain" object evaluates the WinCC permission levels and the "OPTI_EN = FALSE" parameter.

Sequence and positioning of direct connections to controllable objects

@Level6	-->	**Operator control enable**
Permission_Gain	-->	Level_Source
OPTI_EN_CHECKBOX_L	-->	Operator control enable
Permission_Gain	-->	**Target_Operator-ControlEnable**
Gain_AnalogValue	-->	Operator control enable
TN_AnalogValue	-->	Operator control enable
TV_AnalogValue	-->	Operator control enable
DEADB_W_AnalogValue	-->	Operator control enable
TM_LAG_AnalogValue	-->	Operator control enable
ERH_ALM_AnalogValue	-->	Operator control enable
ERL_ALM_AnalogValue	-->	Operator control enable
ER_HYS_AnalogValue3	-->	Operator control enable
M_SUP_ER_CHECKBOX_L	-->	Operator control enable
Permission_Gain	-->	**Target_BackColor**
Gain_AnalogValue	-->	BackColor_Value
TN_AnalogValue	-->	BackColor_Value
TV_AnalogValue	-->	BackColor_Value
DEADB_W_AnalogValue	-->	BackColor_Value
TM_LAG_AnalogValue	-->	BackColor_Value
ERH_ALM_AnalogValue	-->	BackColor_Value
ERL_ALM_AnalogValue	-->	BackColor_Value
ER_HYS_AnalogValue3	-->	BackColor_Value
Format	-->	**Format_InputValue**
ControlError_AnalogValue	-->	Format

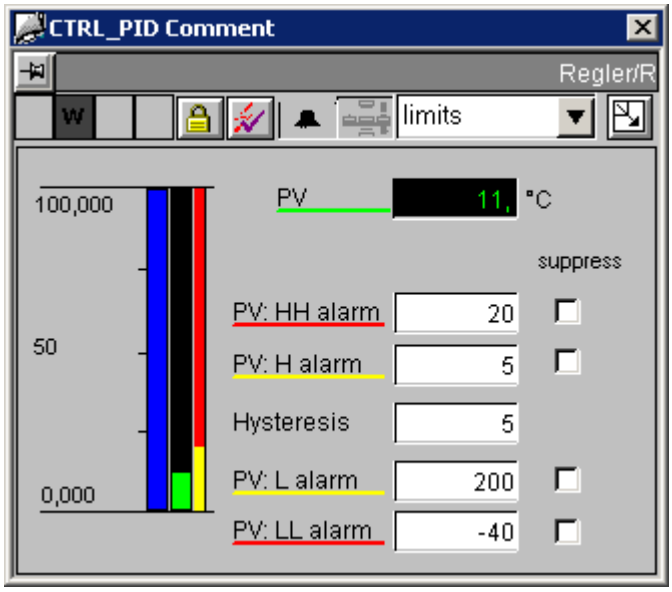
Additional information

You will find more information in:

Objects in the overview (Page 555)

12.1.5 CTRL_PID: Limits view

Setpoint Operation Limits



The setpoint bar in this view shows the setpoint input limits relative to bar limits.

Setpoint input limits are set in the maintenance view.

Analog displays and number formats

The "ProcessValue_AnalogValue" process value is implemented using the "AdvancedAnalogDisplay" object. The number format is defined via the "Format_InputValue" property of the block icon.

All other analog displays are implemented by means of the conventional "Floating-point format" I/O field.

Control permissions

In addition to the WinCC authorization levels, the "Permission_AlarmHigh_AnalogValue" permission object also evaluates the "OPTI_EN = FALSE" parameter.

Sequence and positioning of direct connections to controllable objects

@Level6	-->	**Operator-control enable**
Permission_AlarmHigh_AnalogValue	-->	Level_Source
Permission_AlarmHigh_AnalogValue	-->	**Target_Operator-ControlEnable**
AlarmHigh_AnalogValue	-->	Operator-control enable
WarningHigh_AnalogValue	-->	Operator-control enable
Hysteresis_AnalogValue	-->	Operator-control enable
WarningLow_AnalogValue	-->	Operator-control enable
AlarmLow_AnalogValue	-->	Operator-control enable
AlarmHigh_CHECKBOX_R	-->	Operator-control enable
WarningHigh_CHECKBOX_R	-->	Operator-control enable
WarningLow_CHECKBOX_R	-->	Operator-control enable
AlarmLow_CHECKBOX_R	-->	Operator-control enable
Permission_AlarmHigh_AnalogValue	-->	**Target_BackColor**
AlarmHigh_AnalogValue	-->	BackColor_Value
WarningHigh_AnalogValue	-->	BackColor_Value
Hysteresis_AnalogValue	-->	BackColor_Value
WarningLow_AnalogValue	-->	BackColor_Value
AlarmLow_AnalogValue	-->	BackColor_Value
Format	-->	**Format_InputValue**
ProcessValue_AnalogValue	-->	Format_InputValue

Additional information

You will find more information in:

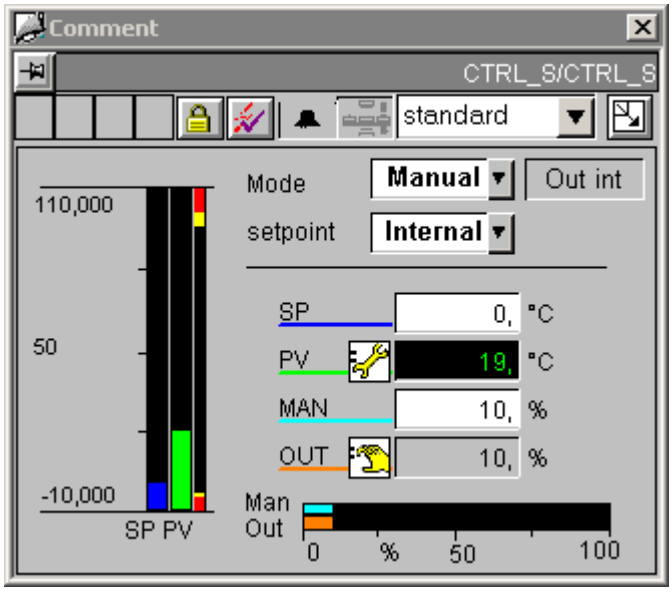
Objects in the overview (Page 555)

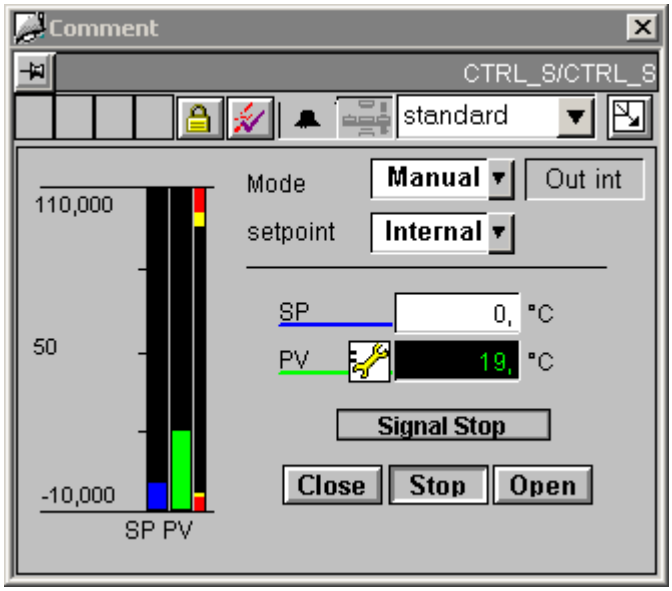
12.1.6 CTRL_S (All Views)

Overview

CTRL_S: Standard view (Page 504)
CTRL_S: Maintenance view (Page 506)
CTRL_S: Parameter view (Page 506)
CTRL_S: Limits view (Page 507)
CTRL_S: StandardS view (Page 507)
Global view: Message view (Page 557)
Global view: Batch view (Page 558)
Global view: Trend view (Page 559)

12.1.7 CTRL_S: Standard view





Key display

In contrast to the **CTRL_PID** standard view, in the **CTRL_S** standard view the bar display and the manual and manipulated-variable analog values are displayed as specified in the LMNR_ON parameter. The keys for operating the LMNDN_OP and LMNUP_OP parameters are displayed when "LMNR_ON = FALSE".

The visibility of the LMNDN_OP_BinOp, LMNUP_OP_BinOp, and LMN_OP_Stop_BinOp keys and the QLMNUP_QLMNDN status display is controlled by a script. This script is called when the "Other/Display" properties of the "Output_BarStandard_3" object change.

The visibility of these objects is also controlled via the X position of the geometry, since the "Visible" property is already used for internal object functions.

There is only one "Stop" key for LMNDN_OP and LMNUP_OP. A script determines to which of these two parameters the value "0" is written. If the "Display_Variable" property changes, these scripts are activated in the "LMNDN_OP_BinOp" and "LMNUP_OP_BinOp" objects.

Control permissions

This view has three permission objects:

- "Permission_Setpoint"
- "Permission_Manual"
- "Permission_LMNDN_OP"

In addition to the WinCC authorization levels, the permission objects also evaluate the parameters listed below:

Permission object	Parameters
"Permission_Setpoint"	"Q_SP_OP = TRUE"
"Permission_Manual"	"QLMNVOP = TRUE"
"Permission_LMNDN_OP"	"QLMNSOP = TRUE"

Operating the PID tuner and tuning

The PID tuner is operated in the parameter view (Tuning On/Off).

When tuning is turned on, the standard view displays a combo box above the "Manual/Auto" mode combo box. This combo box can be used to turn off tuning again.

Sequence and positioning of direct connections to controllable objects

@Level5	-->	**Operator-control enable**		
Manual_COMBOBOX	-->	Operator-control enable		
External_COMBOBOX	-->	Operator-control enable		
Permission_Setpoint	-->	Level_Source	-->	Level_Target
Permission_Manual	-->	Level_Source	-->	Level_Target
Permission_LMNDN_OP	-->	Level_Source		
Permission_Setpoint	-->	**Target_Operator-ControlEnable**		
Setpoint_AnalogValue	-->	Operator-control enable		
Permission_Manual	-->	**Target_Operator-ControlEnable**		
Manual_AnalogValue	-->	Operator-control enable		
Permission_LMNDN_OP	-->	**Target_Operator-ControlEnable**		
LMN_OP_Stop_BinOp	-->	Operator-control enable		
LMNDN_OP_BinOp	-->	Operator-control enable		
LMNUP_OP_BinOp	-->	Operator-control enable		
Format	-->	**Format_InputValue**		
Setpoint_AnalogValue	-->	Format		
ProcessValue_AnalogValue	-->	Format		
Format	-->	**Format_OutputValue**		
Manual_AnalogValue	-->	Format		
Output_AnalogValue	-->	Format		

Additional information

You will find more information in:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.8 CTRL_S: Maintenance view

See: CTRL_PID: Maintenance view (Page 498)

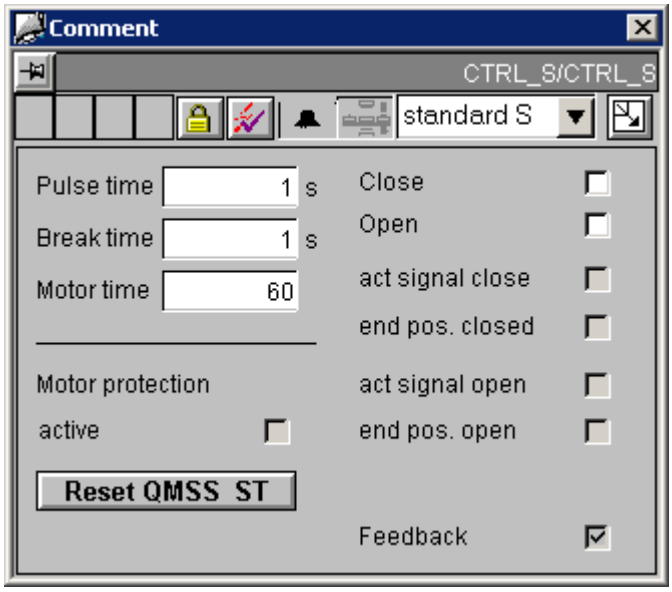
12.1.9 CTRL_S: Parameter view

See: CTRL_PID: Parameter view (Page 500)

12.1.10 CTRL_S: Limits view

See: CTRL_PID: Limits view (Page 502)

12.1.11 CTRL_S: StandardS View



Control permissions

This view has two permission objects:

- "Permission_LMNDN_OP"
- "Permission_PulseTime_AnalogValue"

In addition to the WinCC authorization levels, the permission objects also evaluate the parameters listed below:

Permission object	Parameters
"Permission_LMNDN_OP"	"QLMNSOP = TRUE"
"Permission_PulseTime_AnalogValue"	"OPTI_EN = FALSE"

Sequence and positioning of direct connections to controllable objects

@Level5	-->	**Operator-control enable**		
Permission_LMNDN_OP	-->	Level_Source	-->	Level_Target
@Level6	-->	**Operator-control enable**		
Permission_PulseTime_AnalogValue	-->	Level_Source		
Reset_ButtonBit	-->	Operator-control enable		
Permission_LMNDN_OP	-->	**Target_Operator-ControlEnable**		
LMNDN_OP_Checkbox	-->	Operator-control enable		
LMNUP_OP_Checkbox	-->	Operator-control enable		
Permission_PulseTime_AnalogValue	-->	**Target_Operator-ControlEnable**		
PulseTime_Analogvalue	-->	Operator-control enable		
BreakTime_Analogvalue	-->	Operator-control enable		
MTR_TM_AnalogValue	-->	Operator-control enable		
Permission_PulseTime_AnalogValue	-->	**Target_BackColor**		
PulseTime_Analogvalue	-->	BackColor_Value		
BreakTime_Analogvalue	-->	BackColor_Value		
MTR_TM_AnalogValue	-->	BackColor_Value		

Additional information

You will find more information in:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.12 DIG_MON (All Views)

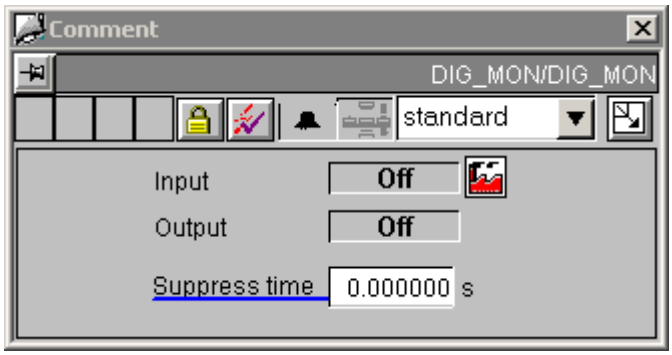
Overview

DIG_MON: Standard view (Page 509)

Global View: Message View (Page 557)

Global View: Batch view (Page 558)

12.1.13 DIG_MON: Standard view



Sequence and positioning of direct connections to controllable objects

@Level5	-->	**Operator-control enable**
SuppressTime_PCS7_AnalogValue	-->	Operator-control enable
@Level6	-->	**Back color**
SuppressTime_PCS7_AnalogValue	-->	Back color

Additional information

You will find more information in the following sections:

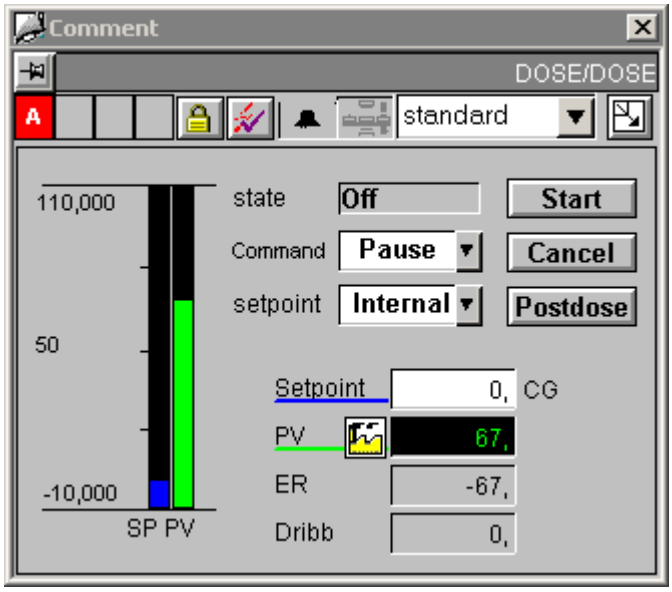
Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.14 DOSE (All Views)

Overview

12.1.15 DOSE: Standard view



Setpoint displays

The "Setpoint_AnalogValue" and "BarStandard_2" setpoint displays must display the "SP_OP" variable for "Internal setpoint" and the "SP_EXT" variable for "External setpoint". A script controls the variable link of these two objects using the "SetLink" command.

This script is called when the "Display_Variable1" property of the "External_COMBOBOX" object is changed.

Control permissions

This view has four permission objects:

- "Permission_Setpoint"
- "Permission_Start"
- "Permission_Cancel"
- "Permission_Post_Dose"

In addition to the WinCC authorization levels, the permission objects also evaluate the parameters listed below:

Permission object	Parameters
"Permission_Setpoint"	"Q_SP_OP = TRUE"
"Permission_Start"	"QSTRT_OP = TRUE"
"Permission_Cancel"	"QCN_OP = TRUE"
"Permission_Post_Dose"	"QPD_OP = TRUE"

Sequence and positioning of direct connections to controllable objects

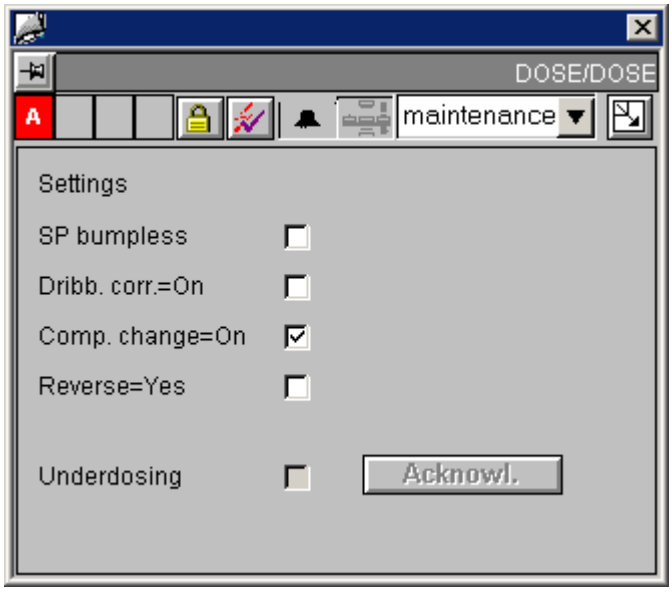
@Level5	-->	**Operator-control enable**		
Pause_COMBOBOX	-->	Operator-control enable		
External_COMBOBOX	-->	Operator-control enable		
Permission_Setpoint	-->	Level_Source	-->	Level_Target
Permission_Start	-->	Level_Source	-->	Level_Target
Permission_Cancel	-->	Level_Source	-->	Level_Target
Permission_Post_Dose	-->	Level_Source		
Permission_Setpoint	-->	**Target_Operator-ControlEnable**		
Setpoint_AnalogValue	-->	Operator-control enable		
Permission_Start	-->	**Target_Operator-ControlEnable**		
Start_ButtonBit	-->	Operator-control enable		
Permission_Cancel	-->	**Target_Operator-ControlEnable**		
Cancel_ButtonBit	-->	Operator-control enable		
Permission_Post_Dose	-->	**Target_Operator-ControlEnable**		
Post_Dose_ButtonBit	-->	Operator-control enable		
Format	-->	**Format_InputValue**		
Setpoint_AnalogValue	-->	Format		
ProcessValue_AnalogValue	-->	Format		

Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)
The quality code display (Page 614)

12.1.16 DOSE: Maintenance view



Control permissions

In addition to the WinCC authorization levels, the "Permission_ACK_TOL_OP" permission object also evaluates the "QTOL_N = TRUE" and "QAK_OP= TRUE" parameters.

Sequence and positioning of direct connections to controllable objects

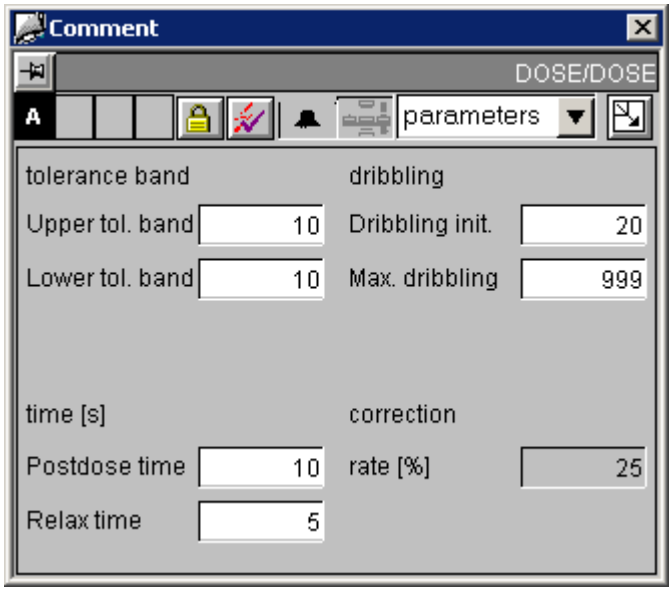
@Level6	-->	**Operator-control enable**
SPBUMPON_CHECKBOX_L	-->	Operator-control enable
DRIB_COR_CHECKBOX_L	-->	Operator-control enable
COMP_CHG_CHECKBOX_L2	-->	Operator-control enable
REVERSE_CHECKBOX_L	-->	Operator-control enable
Permission_ACK_TOL_OP	-->	**Level_Source**
Permission_ACK_TOL_OP	-->	**Target_Operator-ControlEnable**
ACK_TOL_OP_ButtonBit	-->	Operator-control enable

Additional information

You will find more information in:

Objects in the overview (Page 555)

12.1.17 DOSE: Parameter view



Sequence and positioning of direct connections to controllable objects

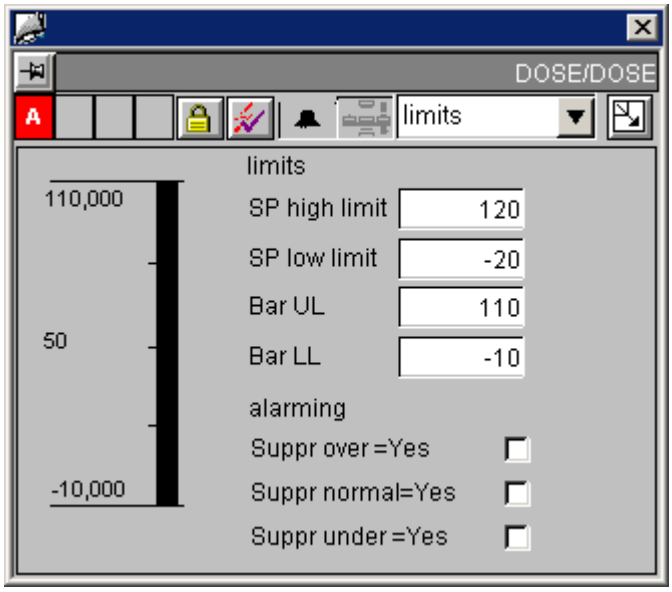
@Level6	-->	**Operator-control enable**
TOL_P_PCS7_AnalogValue	-->	Operator-control enable
TOL_N_PCS7_AnalogValue	-->	Operator-control enable
PDOS_TME_PCS7_AnalogValue	-->	Operator-control enable
RELAXTME_PCS7_AnalogValue	-->	Operator-control enable
DRIBB_PCS7_AnalogValue	-->	Operator-control enable
DRIBBMAX_PCS7_AnalogValue	-->	Operator-control enable
@Level6	-->	**Back color**
TOL_P_PCS7_AnalogValue	-->	BackColor_Value
TOL_N_PCS7_AnalogValue	-->	BackColor_Value
PDOS_TME_PCS7_AnalogValue	-->	BackColor_Value
RELAXTME_PCS7_AnalogValue	-->	BackColor_Value
DRIBB_PCS7_AnalogValue	-->	BackColor_Value
DRIBBMAX_PCS7_AnalogValue	-->	BackColor_Value

Additional information

You will find more information in:

Objects in the overview (Page 555)

12.1.18 DOSE: Limits view



Sequence and positioning of direct connections to controllable objects

@Level6	-->	**Operator-control enable**
SP_HLM_PCS7_AnalogValue	-->	Operator-control enable
SP_LLM_PCS7_AnalogValue	-->	Operator-control enable
MO_PVHR_PCS7_AnalogValue	-->	Operator-control enable
MO_PVLR_PCS7_AnalogValue	-->	Operator-control enable
M_SUP_2_UEBERDOS_CHECKBOX_L	-->	Operator-control enable
M_SUP_1_Dos_ok_CHECKBOX_L	-->	Operator-control enable
M_SUP_3_UNTERDOS_CHECKBOX_L	-->	Operator-control enable
@Level6	-->	**Back color**
SP_HLM_PCS7_AnalogValue	-->	BackColor_Value
SP_LLM_PCS7_AnalogValue	-->	BackColor_Value
MO_PVHR_PCS7_AnalogValue	-->	BackColor_Value
MO_PVLR_PCS7_AnalogValue	-->	BackColor_Value

Additional information

You will find more information in:

Objects in the overview (Page 555)

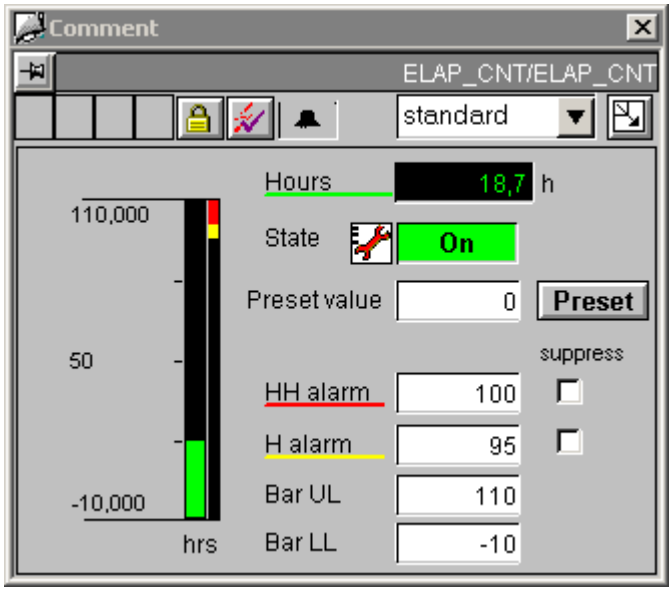
12.1.19 ELAP_CNT (All Views)

Overview

ELAP_CNT: Standard view (Page 516)

Global view: Message view (Page 557)

12.1.20 ELAP_CNT: Standard view



Sequence and positioning of direct connections to controllable objects

@Level5	-->	**Operator-control enable**
HOURS_OP_AnalogValue	-->	Operator-control enable
TRACK_OP_ButtonBit	-->	Operator-control enable
@Level5	-->	**Back color**
HOURS_OP_AnalogValue	-->	BackColor_Value
@Level6	-->	**Operator-control enable**
AlarmHigh_AnalogValue	-->	Operator-control enable
WarningHigh_AnalogValue	-->	Operator-control enable
MO_HOUHR_PCS7_AnalogValue	-->	Operator-control enable
MO_HOULR_PCS7_AnalogValue	-->	Operator-control enable
AlarmHigh_CHECKBOX_R	-->	Operator-control enable
WarningHigh_CHECKBOX_R	-->	Operator-control enable
@Level6	-->	**Back color**
AlarmHigh_AnalogValue	-->	BackColor_Value
WarningHigh_AnalogValue	-->	BackColor_Value
MO_HOUHR_PCS7_AnalogValue	-->	BackColor_Value
MO_HOULR_PCS7_AnalogValue	-->	BackColor_Value
Format	-->	**Format_InputValue**
HOURS_AnalogValue	-->	Format

Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

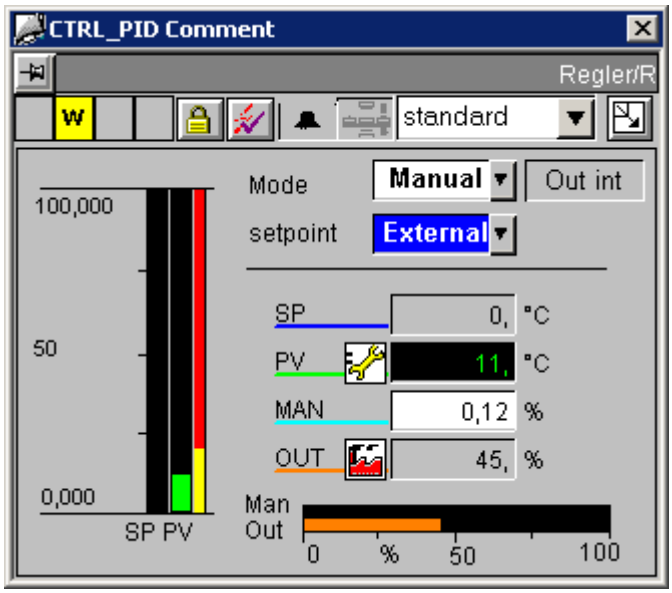
The quality code display (Page 614)

12.1.21 FMCS_PID (All Views)

Overview

FMCS_PID: Standard view (Page 518)
FMCS_PID: Maintenance view (Page 520)
FMCS_PID: Parameter view (Page 522)
FMCS_PID: Limits view (Page 524)
FMCS_PID: StandardS view (Page 525)
Global view: Message view (Page 557)
Global view: Batch view (Page 558)
Global view: Trend view (Page 559)

12.1.22 FMCS_PID: Standard view



This FMCS_PID standard view looks identical to the CTRL_PID standard view (Page 496).

Analog displays and number formats

All analog displays are implemented using the "AdvancedAnalogDisplay" object. The value format is set via the "Format_InputValue" and "Format_OutputValue" block-icon properties.

Control permissions

This view has three permission objects:

- "Permission_Manual_COMBOBOX"
- "Permission_Setpoint"
- "Permission_Manual"

In addition to the WinCC authorization levels, the permission objects also evaluate the parameters listed below:

Permission object	Parameters
"Permission_Manual_COMBOBOX"	• "QMODF = FALSE"
"Permission_Setpoint"	• "Q_SP_OP = TRUE" • "QMODF = FALSE"
"Permission_Manual"	• "QLMNOP = TRUE" • "QMODF = FALSE"

Sequence and positioning of direct connections to controllable objects

@Level5	-->	**Operator-control enable**		
Permission_Manual_COMBOBOX	-->	Level_Source	-->	Level_Target
Permission_Setpoint	-->	Level_Source	-->	Level_Target
Permission_Manual	-->	Level_Source		
Permission_Manual_COMBOBOX	-->	**Target_Operator-ControlEnable**		
Manual_COMBOBOX	-->	Operator-control enable		
External_COMBOBOX	-->	Operator-control enable		
Permission_Setpoint	-->	**Target_Operator-ControlEnable**		
Setpoint_AnalogValue	-->	Operator-control enable		
Permission_Manual	-->	**Target_Operator-ControlEnable**		
Manual_AnalogValue	-->	Operator-control enable		
Format	-->	**Format_InputValue**		
Setpoint_AnalogValue	-->	Format		
ProcessValue_AnalogValue	-->	Format		
Format	-->	**Format_OutputValue**		
Manual_AnalogValue	-->	Format		
Output_AnalogValue	-->	Format		

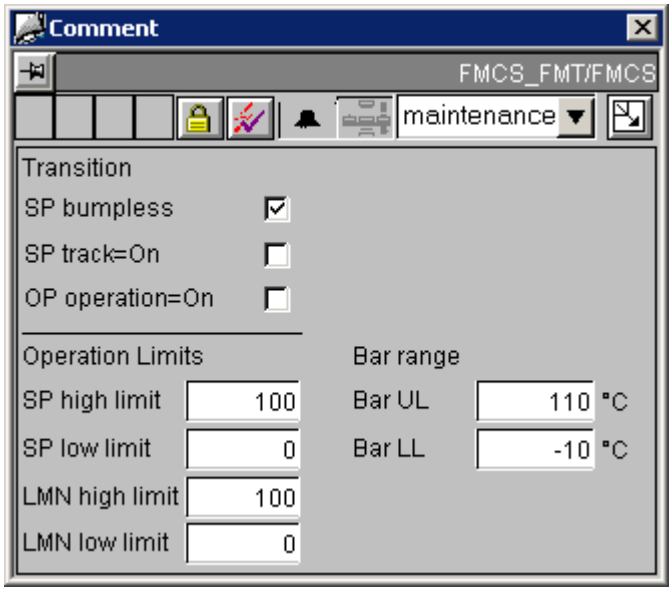
Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.23 FMCS_PID: Maintenance view



If a block of the type CPM is instantiated in the chart of the controller and this has the name of the controller block with the extension "_cpm", an additional button CPM is displayed with which the CPM faceplate can be called.

Control permissions

As well as the WinCC authorization levels, the "Permission_all" permission object also evaluates the "QMODF = FALSE" parameter.

Sequence and positioning of direct connections to controllable objects

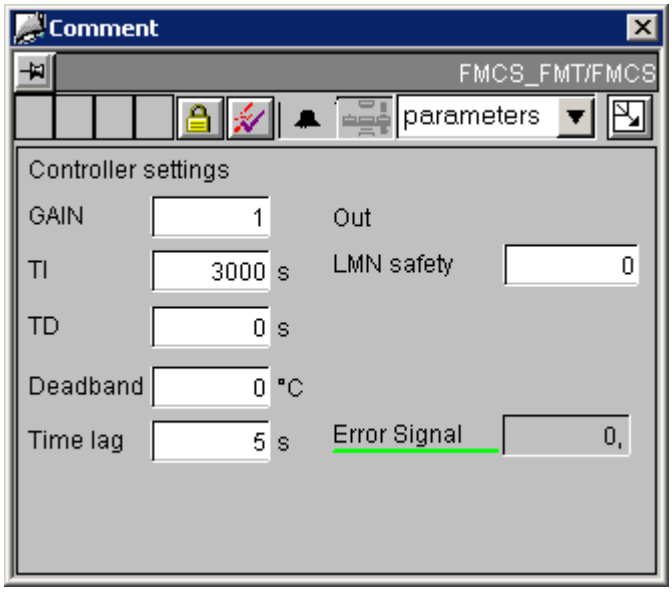
@Level6	-->	**Operator control enable**
Permission_all	-->	Level_Source
Permission_all	-->	**Target_Operator-ControlEnable**
Bumpless_CHECKBOX_L	-->	Operator control enable
SP_TRK_ON_CHECKBOX_L	-->	Operator control enable
Oper_OP_CHECKBOX	-->	Operator control enable
SPHighLimit_AnalogValue	-->	Operator control enable
SPLowLimit_AnalogValue	-->	Operator control enable
ManHighLimit_AnalogValue	-->	Operator control enable
ManLowLimit_AnalogValue	-->	Operator control enable
MO_PVHR_AnalogValue	-->	Operator control enable
MO_PVLR_AnalogValue	-->	Operator control enable
Permission_all	-->	**Target_BackColor**
SPHighLimit_AnalogValue	-->	BackgroundColor_Value
SPLowLimit_AnalogValue	-->	BackgroundColor_Value
ManHighLimit_AnalogValue	-->	BackgroundColor_Value
ManLowLimit_AnalogValue	-->	BackgroundColor_Value
MO_PVHR_AnalogValue	-->	BackgroundColor_Value
MO_PVLR_AnalogValue	-->	BackgroundColor_Value

Additional information

You will find more information in:

Objects in the overview (Page 555)

12.1.24 FMCS_PID: Parameter view



If a block of the type GAIN_SHD is instantiated in the chart of the controller and this has the name of the controller block with the extension "_gsc", an additional button GAIN_SHD is displayed with which the GAIN_SHD faceplate can be called.

Analog displays and number formats

The "ControlError_AnalogValue" process value is implemented using the "AdvancedAnalogDisplay" object. The number format is defined at the "Format_InputValue" property of the block icon.

All other analog displays are implemented by means of the conventional "Floating-point format" I/O field.

Control permissions

As well as the WinCC authorization levels, the "Permission_all" permission object also evaluates the "QMODF = FALSE" parameter.

Sequence and positioning of direct connections to controllable objects

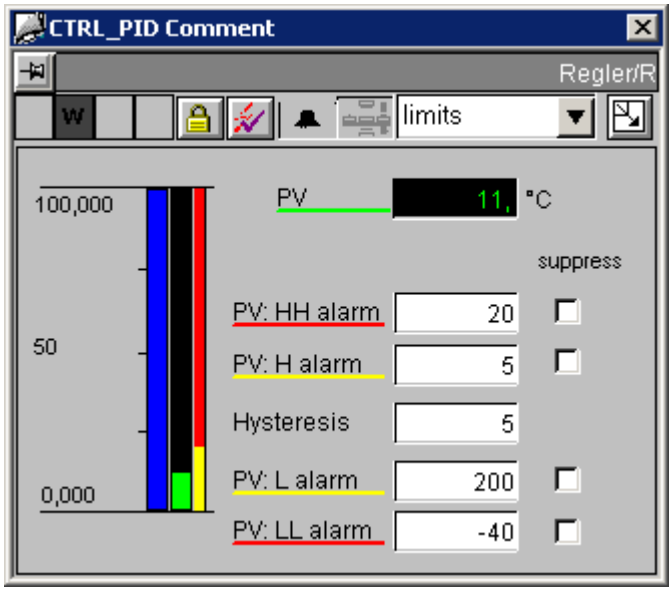
@Level6	-->	**Operator control enable**		
Permission_all	-->	Level_Source	-->	Level_Target
OPTI_EN_CHECKBOX_L	-->	Operator control enable		
Permission_all	-->	**Target_Operator-ControlEnable**		
Gain_AnalogValue	-->	Operator control enable		
TN_AnalogValue	-->	Operator control enable		
TV_AnalogValue	-->	Operator control enable		
DEADB_W_AnalogValue	-->	Operator control enable		
TM_LAG_AnalogValue	-->	Operator control enable		
LMN_SAVE_PCS7_AnalogValue2	-->	Operator control enable		
Permission_all	-->	**Target_BackColor**		
Gain_AnalogValue	-->	BackgroundColor_Value		
TN_AnalogValue	-->	BackgroundColor_Value		
TV_AnalogValue	-->	BackgroundColor_Value		
DEADB_W_AnalogValue	-->	BackgroundColor_Value		
TM_LAG_AnalogValue	-->	BackgroundColor_Value		
LMN_SAVE_PCS7_AnalogValue2	-->	BackgroundColor_Value		
Format	-->	**Format_InputValue**		
ErrorSignal_AnalogValue	-->	Format		

Additional information

You will find more information in:

Objects in the overview (Page 555)

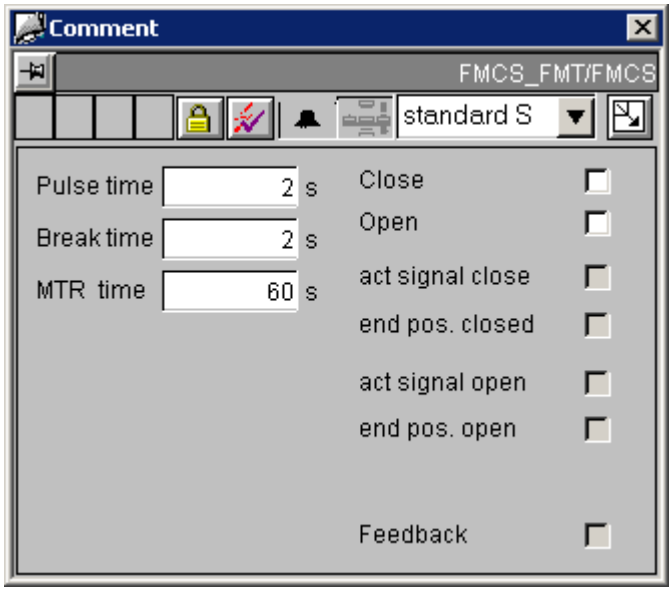
12.1.25 FMCS_PID: Limits view



This FMCS_PID limits view and the CTRL_PID limits view (Page 502) are almost identical.

The only difference is that here, "QMODF = FALSE" is also queried for the control-permission objects.

12.1.26 FMCS_PID: StandardS View



Control permissions

This view has two permission objects:

- "Permission_LMNDN_OP"
- "Permission_PulseTime_AnalogValue"

In addition to the WinCC authorization levels, the permission objects also evaluate the parameters listed below:

Permission object	Parameters
"Permission_LMNDN_OP"	• "QLMNSOP = TRUE" • "QMODF = FALSE"
"Permission_Pulse_TM"	• "QMODF = FALSE"

Sequence and positioning of direct connections to controllable objects

@Level5	-->	**Operator-control enable**		
Permission_LMNDN_OP	-->	Level_Source		
@Level6	-->	**Operator-control enable**		
Permission_Pulse_TM	-->	Level_Source	-->	Level_Target
Permission_LMNDN_OP	-->	**Target_Operator-ControlEnable**		
LMNDN_OP_Checkbox	-->	Operator-control enable		
LMNUP_OP_Checkbox	-->	Operator-control enable		
Permission_Pulse_TM	-->	**Target_Operator-ControlEnable**		
PulseTime_Analogvalue	-->	Operator-control enable		
BreakTime_Analogvalue	-->	Operator-control enable		
MTR_TM_AnalogValue	-->	Operator-control enable		
Permission_Pulse_TM	-->	**Target_BackColor**		
PulseTime_Analogvalue	-->	BackColor_Value		
BreakTime_Analogvalue	-->	BackColor_Value		
MTR_TM_AnalogValue	-->	BackColor_Value		

Additional information

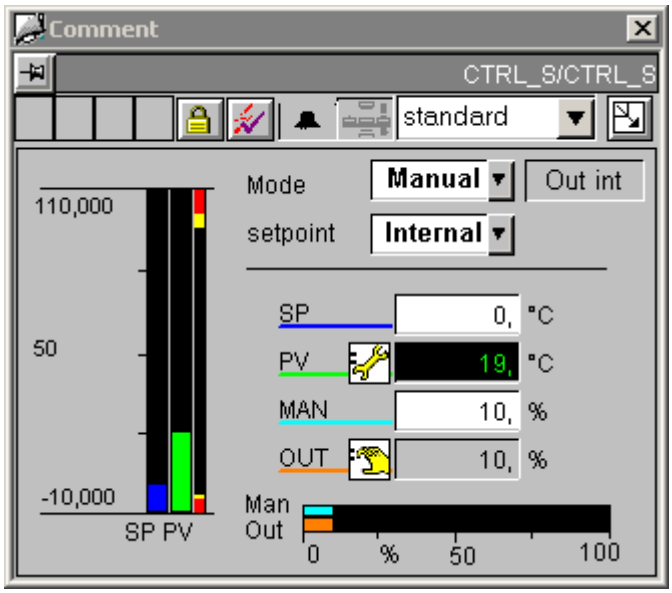
You will find more information in:

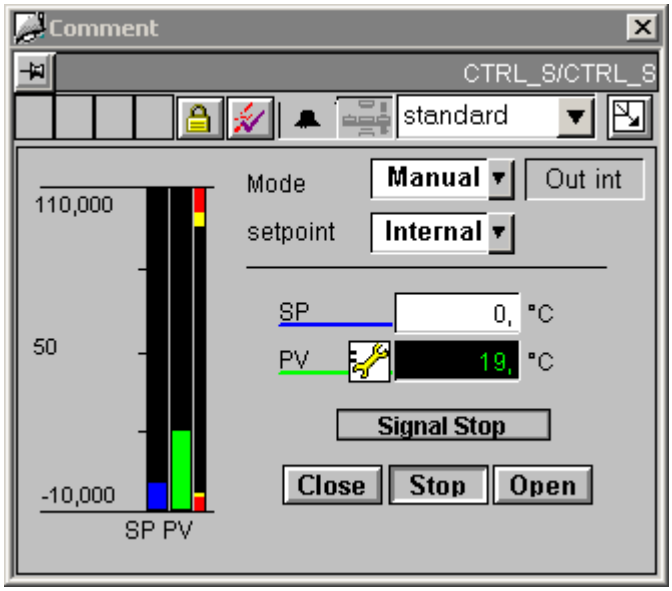
Objects in the overview (Page 555)

12.1.27 FMT_PID (All Views)

Overview

12.1.28 FMT_PID: Standard view





This FMT_PID standard view looks almost identical to the CTRL_S standard view (Page 504).

In contrast to the CTRL_S standard view, in the FMT_PID standard view the bar display and the manual and manipulated-variable analog values are displayed as specified in the LMNR_ON and QSTEPCON parameters.

Display of bar, analog values, and keys

When "LMNR_ON = FALSE" and "QSTEPCON = TRUE", keys for operating the LMNDN_OP and LMNUP_OP parameters are shown.

The visibility of the LMNDN_OP_BinOp, LMNUP_OP_BinOp, and LMN_OP_Stop_BinOp keys and the QLMNUP_QLMNDN status display is controlled by a script. This script is called when the "Links/QSTEPCON" and "Links/QLMNR_ON" properties of the "VISIBLE_Analog_Output" object are changed.

The visibility of these objects is also controlled via the X position of the geometry, since the "Visible" property is already used for internal object functions.

There is only one "Stop" key for LMNDN_OP and LMNUP_OP. A script determines to which of these two parameters the value "0" is written. If the "Display_Variable" property changes, these scripts are activated in the "LMNDN_OP_BinOp" and "LMNUP_OP_BinOp" objects.

Control permissions

This view has four permission objects:

- "Permission_Manual_COMBOBOX"
- "Permission_Setpoint"
- "Permission_Manual"
- "Permission_LMNDN_OP"

In addition to the WinCC authorization levels, the permission objects also evaluate the parameters listed below:

Permission object	Parameters
"Permission_Manual_COMBOBOX"	• "QMODF = FALSE"
"Permission_Setpoint"	• "Q_SP_OP = TRUE" • "QMODF = FALSE"
"Permission_Manual"	• "QLMNVOP = TRUE" • "QMODF = FALSE"
"Permission_LMNDN_OP"	• "QLMNSOP = TRUE" • "QMODF = FALSE"

Sequence and positioning of direct connections to controllable objects

@Level5	-->	**Operator-control enable**		
Permission_Manual_COMBOBOX	-->	Level_Source	-->	Level_Target
Permission_Setpoint	-->	Level_Source	-->	Level_Target
Permission_Manual	-->	Level_Source	-->	Level_Target
Permission_LMNDN_OP	-->	Level_Source		
Permission_Manual_COMBOBOX	-->	**Target_Operator-ControlEnable**		
Manual_COMBOBOX	-->	Operator-control enable		
External_COMBOBOX	-->	Operator-control enable		
Permission_Setpoint	-->	**Target_Operator-ControlEnable**		
Setpoint_AnalogValue	-->	Operator-control enable		
Permission_Manual	-->	**Target_Operator-ControlEnable**		
Manual_AnalogValue	-->	Operator-control enable		
Permission_LMNDN_OP	-->	**Operator-control enable**		
LMN_OP_Stop_BinOp	-->	Operator-control enable		
LMNDN_OP_BinOp	-->	Operator-control enable		
LMNUP_OP_BinOp	-->	Operator-control enable		

Format	-->	**Format_InputValue**		
Setpoint_AnalogValue	-->	Format		
ProcessValue_AnalogValue	-->	Format		
Format	-->	**Format_OutputValue**		
Manual_AnalogValue	-->	Format		
Output_AnalogValue	-->	Format		

Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.29 FMT_PID: Maintenance view

See: FMCS_PID: Maintenance view (Page 520)

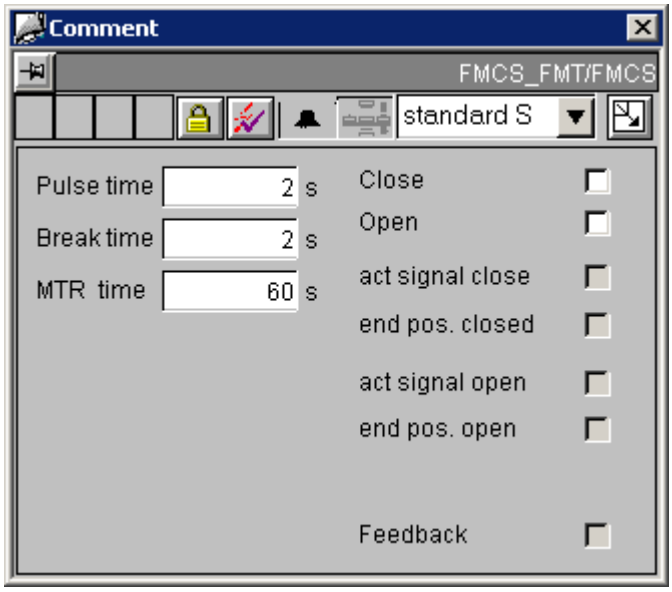
12.1.30 FMT_PID: Parameter view

See: FMCS_PID: Parameter view (Page 522)

12.1.31 FMT_PID: Limits view

See: FMCS_PID: Limits view (Page 524)

12.1.32 FMT_PID: StandardS View



The FMT_PID standardS view looks almost identical to the FMCS_PID standardS view (Page 525).

Differences to FMCS_PID

The FMT_PID standardS view differs from the FMCS_PID standardS view in terms of control permissions and display:

- In addition to the WinCC authorization levels, the "Permission_LMNDN_OP" permission object also evaluates the "QLMNSOP = TRUE" and "QMODF = FALSE" parameters.
- The standardS view is only displayed if parameter "QSTEPCON = TRUE". Otherwise, all elements in this view are invisible.

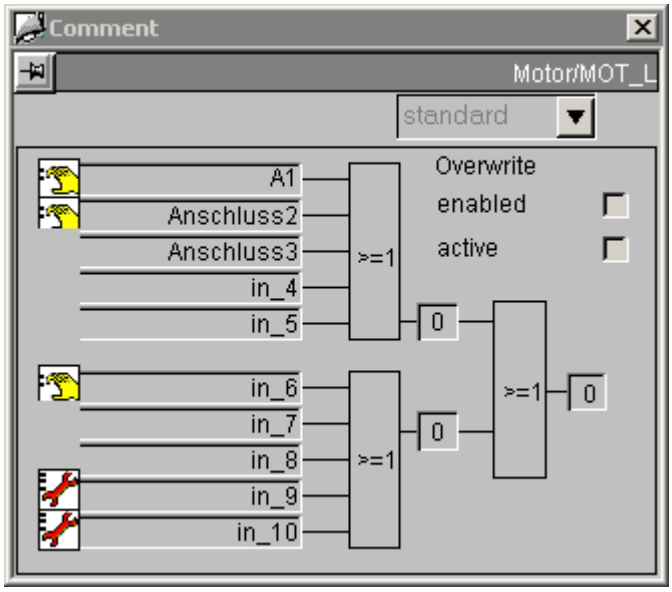
Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.33 INTERLOK: Standard view



Sequence and positioning of direct connections to controllable objects

@Level6	-->	Permission
overwrite_active_CHECKBOX_L2	-->	Permission

The "I1_1" to "I2_5" inputs always show the "log 1" text (string_1).

The basic state of inputs is:

- The "log 0" state is displayed with black characters on a gray background.
- The "log 1" state is displayed with white characters on a red background.

If the input is inverted with NEG1_1, the display colors are also inverted.

The same goes for NEGRES_1 = 1. Here, the colors of all five inputs of the first logic element are inverted. In this way, the inputs marked by a red background indicate summing output "Q" with reference to the error state.

In order to avoid misinterpretations in such situations, it is recommended that the NEGRES_1 and NEGRES_2 inputs are not used.

Furthermore, it is advisable to use logic OR operations, since this is the only method to ensure the colors are correct and that inputs with red text will lead to an interlock.

Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.34 MEAS_MON (All Views)

Overview

MEAS_MON: Standard view (Page 533)

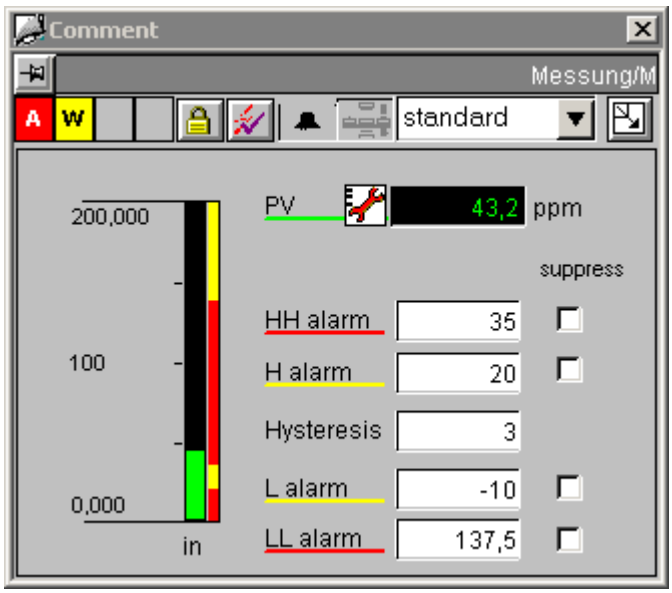
MEAS_MON: Limits view (Page 535)

Global view: Message view (Page 557)

Global view: Batch view (Page 558)

Global view: Trend view (Page 559)

12.1.35 MEAS_MON: Standard view



Analog displays and number formats

The "PV_AnalogValue" process value is set via the "AdvancedAnalogDisplay". The number format is defined via the "Format_InputValue" property of the block icon.

The other analog displays are implemented by means of the conventional "Floating-point format" I/O field.

Sequence and positioning of direct connections to controllable objects

@Level6	-->	**Operator-control enable**
AlarmHigh_AnalogValue	-->	Operator-control enable
WarningHigh_AnalogValue	-->	Operator-control enable
Hysteresis_AnalogValue	-->	Operator-control enable
WarningLow_AnalogValue	-->	Operator-control enable
WarningHigh_AnalogValue	-->	Operator-control enable
AlarmHigh_CHECKBOX_R	-->	Operator-control enable
WarningHigh_CHECKBOX_R	-->	Operator-control enable
WarningLow_CHECKBOX_R	-->	Operator-control enable
AlarmLow_CHECKBOX_R	-->	Operator-control enable
@Level6	-->	**Back color**
AlarmHigh_AnalogValue	-->	Back color
WarningHigh_AnalogValue	-->	Back color
Hysteresis_AnalogValue	-->	Back color
WarningLow_AnalogValue	-->	Back color
WarningHigh_AnalogValue	-->	Back color
Format	-->	**Format_InputValue**
PV_AnalogValue	-->	Format

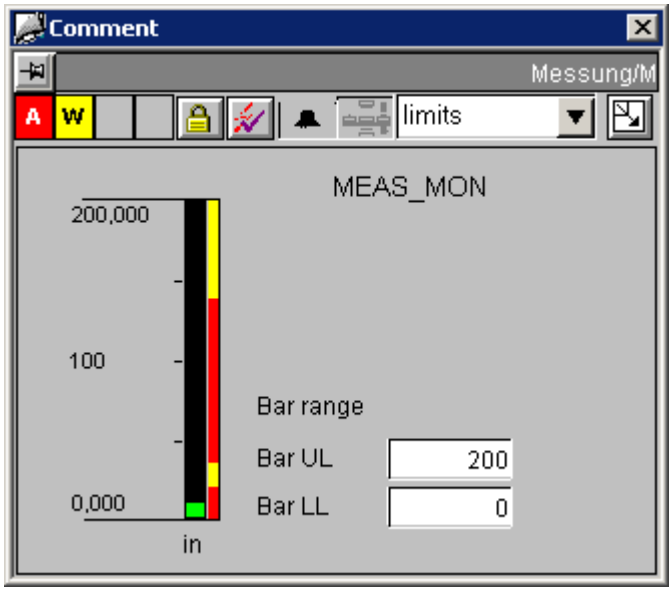
Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.36 MEAS_MON: Limits view



Analog displays

Both analog displays are implemented by means of the conventional "Floating-point format" I/O field.

Sequence and positioning of direct connections to controllable objects

@Level6	-->	**Operator-control enable**
MO_PVHR_AnalogValue	-->	Operator-control enable
MO_PVLR_AnalogValue	-->	Operator-control enable
@Level6	-->	**Back color**
MO_PVHR_AnalogValue	-->	Back color
MO_PVLR_AnalogValue	-->	Back color

Additional information

You will find more information in:

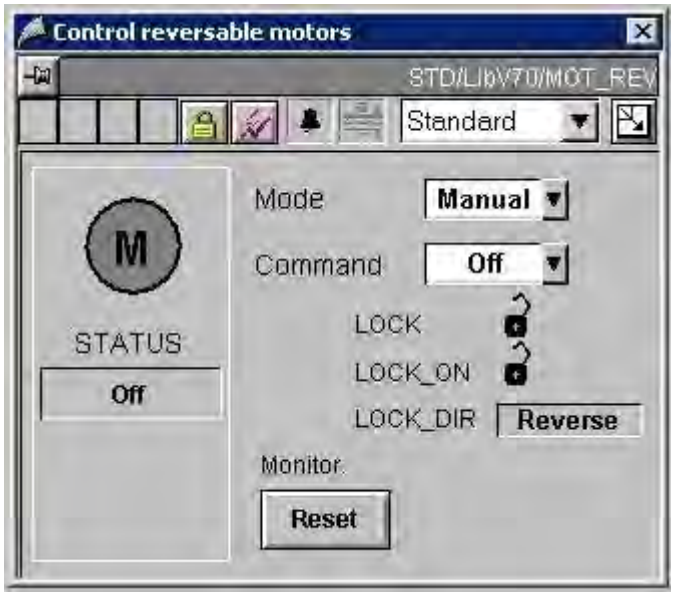
Objects in the overview (Page 555)

12.1.37 MOT_REV (All Views)

Overview

MOT_REV: Standard view (Page 536)
MOT_REV: Maintenance view (Page 537)
Global view: Message view (Page 557)
Global view: Batch view (Page 558)

12.1.38 MOT_REV: Standard view



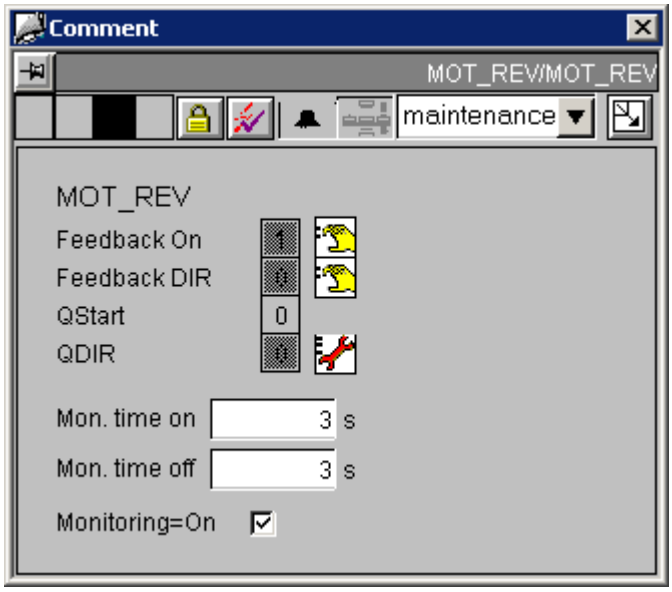
Sequence and positioning of direct connections to controllable objects

@Level5	-->	**Operator-control enable**
Auto_Manual_COMBOBOX	-->	Operator-control enable
Open_Close_Stop_3COMBOBOX	-->	Operator-control enable
Reset_ButtonBit	-->	Operator-control enable

Additional information

You will find more information in:
Objects in the overview (Page 555)

12.1.39 MOT_REV: Maintenance view



Sequence and positioning of direct connections to controllable objects

@Level6	-->	**Operator-control enable**
TIME_ON_AnalogValue	-->	Operator-control enable
TIME_OFF_AnalogValue	-->	Operator-control enable
Monitoring_ON_CHECKBOX_L1	-->	Operator-control enable
@Level6	-->	**Back color**
TIME_MON_PCS7_AnalogValue	-->	Back color

Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.40 MOT_SPED (All Views)

Overview

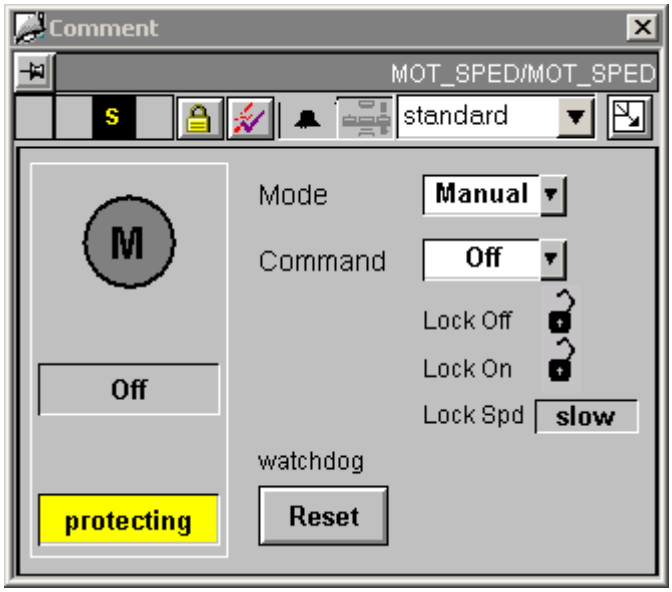
MOT_SPED: Standard view (Page 538)

MOT_SPED: Maintenance view (Page 539)

Global view: Message view (Page 557)

Global view: Batch view (Page 558)

12.1.41 MOT_SPED: Standard view



Sequence and positioning of direct connections to controllable objects

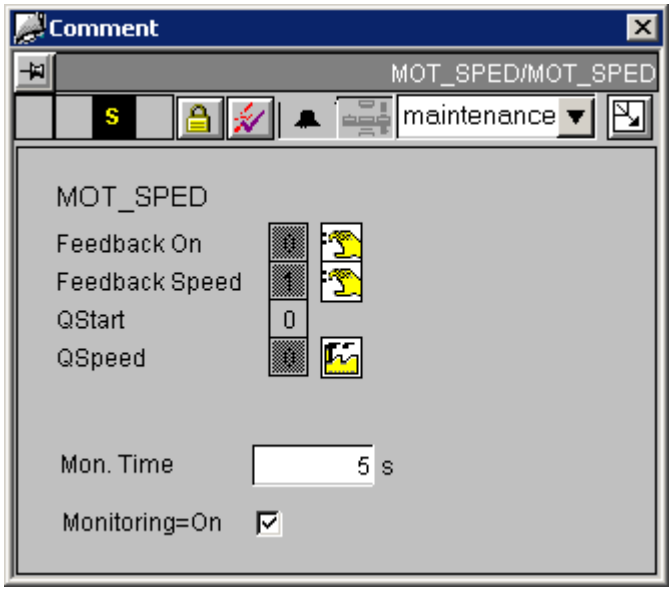
@Level5	-->	**Operator-control enable**
Auto_Manual_COMBOBOX	-->	Operator-control enable
Open_Close_Stop_3COMBOBOX	-->	Operator-control enable
Reset_ButtonBit	-->	Operator-control enable

Additional information

You will find more information in:

Objects in the overview (Page 555)

12.1.42 MOT_SPED: Maintenance view



Sequence and positioning of direct connections to controllable objects

@Level6	-->	**Operator-control enable**
TIME_MON_PCS7_AnalogValue	-->	Operator-control enable
Monitoring_ON_CHECKBOX_L1	-->	Operator-control enable
@Level6	-->	**Back color**
TIME_OFF_PCS7_AnalogValue	-->	Back color

Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.43 MOTOR (All Views)

Overview

MOTOR: Standard view (Page 540)
MOTOR: Maintenance view (Page 542)
Global view: Message view (Page 557)
Global view: Batch view (Page 558)

12.1.44 MOTOR: Standard view

Sequence and positioning of direct connections to controllable objects

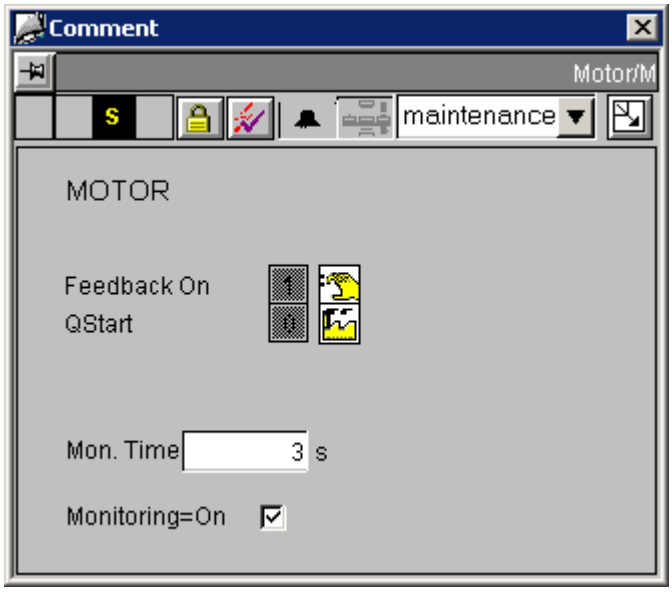
@Level5	-->	**Operator-control enable**		
Permission_Manual_BinOp0	-->	Level_Source	-->	Level_Target
Permission_Automatic_BinOp1	-->	Level_Source	-->	Level_Target
Permission_Off_BinOp2	-->	Level_Source	-->	Level_Target
Permission_On_BinOp1	-->	Level_Source	-->	Level_Target
Reset_ButtonBit	-->	Operator-control enable		
Permission_Manual_BinOp0	-->	**Target_Operator-ControlEnable**		
Manual_BinOp0	-->	Operator-control enable		
Permission_Automatic_BinOp1	-->	**Target_Operator-ControlEnable**		
Automatic_BinOp1	-->	Operator-control enable		
Permission_Off_BinOp2	-->	**Format_InputValue**		
Off_BinOp2	-->	Format		
Permission_On_BinOp1	-->	**Format_OutputValue**		
On_BinOp1	-->	Format		
In addition to the WinCC authorization levels, the permission objects also evaluate the following AS-block parameters:				
Permission_Manual_BinOp0	-->	"QMANOP = TRUE"		
Permission_Automatic _BinOp1	-->	"QAUTOP = TRUE"		
Permission_Off_BinOp2	-->	"QOFF_OP = TRUE"		
Permission_On_BinOp1	-->	"QON_OP = TRUE"		

Additional information

You will find more information in:

Objects in the overview (Page 555)

12.1.45 MOTOR: Maintenance view



Sequence and positioning of direct connections to controllable objects

@Level6	-->	**Operator-control enable**
Monitoring_ON_CHECKBOX_L1	-->	Operator-control enable
Monitoring_PCS7_AnalogValue	-->	Operator-control enable

Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

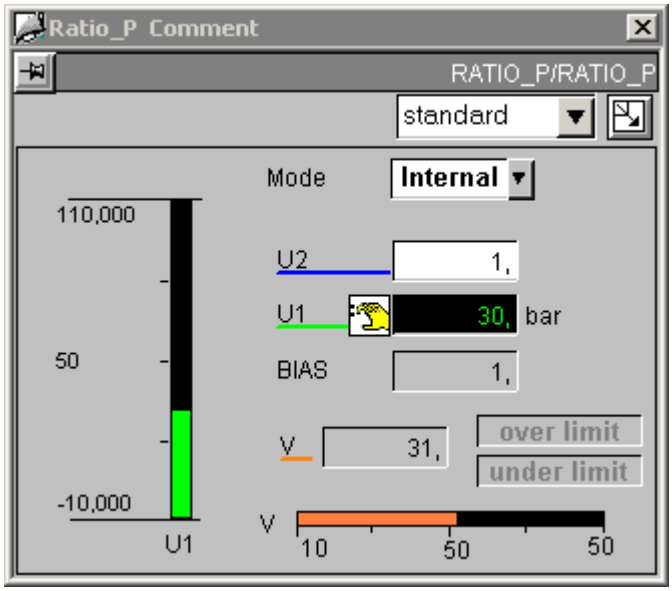
The quality code display (Page 614)

12.1.46 RATIO_P (All Views)

Overview

RATIO_P: Standard view (Page 543)
RATIO_P: Limits view (Page 545)

12.1.47 RATIO_P: Standard view



Sequence and positioning of direct connections to controllable objects

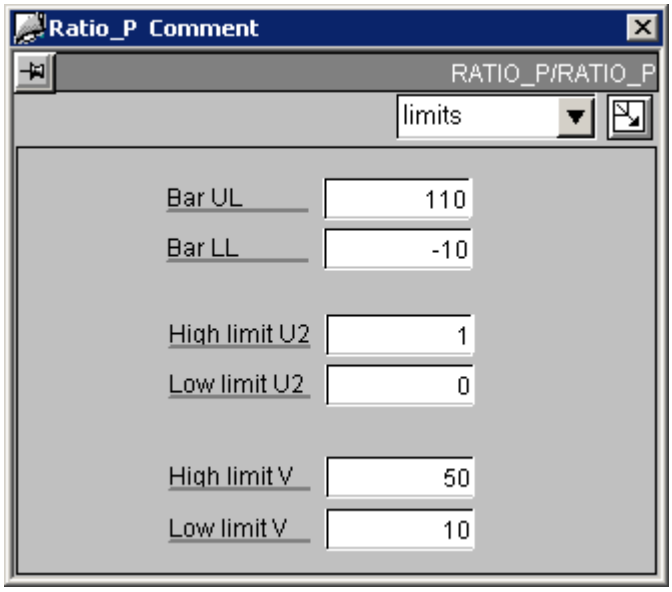
@Level5	-->	**Operator-control enable**		
Permission_ExternalComboBox	-->	Level_Source	-->	Level_Target
Permission_U2	-->	Level_Source		
Permission_ExternalComboBox	-->	**Target_Operator-ControlEnable**		
External_COMBOBOX	-->	Operator-control enable		
Permission_U2	-->	**Target_Operator-ControlEnable**		
U2_AnalogValue	-->	Operator-control enable		
Format	-->	**Format_InputValue**		
U2_AnalogValue	-->	Format		
Format	-->	**Format_OutputValue**		
V_AnalogValue	-->	Format		
Format	-->	**Format_xx**		
BIAS_AnalogValue	-->	Format		

Additional information

You will find more information in:

The quality code display (Page 614)

12.1.48 RATIO_P: Limits view



Sequence and positioning of direct connections to control objects

@Level6	-->	**Operator-control enable**
MO_U1HR_AnalogValue	-->	Operator-control enable
MO_U1LR_AnalogValue	-->	Operator-control enable
U2_HL_AnalogValue1	-->	Operator-control enable
U2_LL_AnalogValue2	-->	Operator-control enable
V_HL_AnalogValue3	-->	Operator-control enable
V_LL_AnalogValue4	-->	Operator-control enable
@Level6	-->	**BackColor**
MO_U1HR_AnalogValue	-->	BackColor_Value
MO_U1LR_AnalogValue	-->	BackColor_Value
U2_HL_AnalogValue1	-->	BackColor_Value
U2_LL_AnalogValue2	-->	BackColor_Value
V_HL_AnalogValue3	-->	BackColor_Value
V_LL_AnalogValue4	-->	BackColor_Value

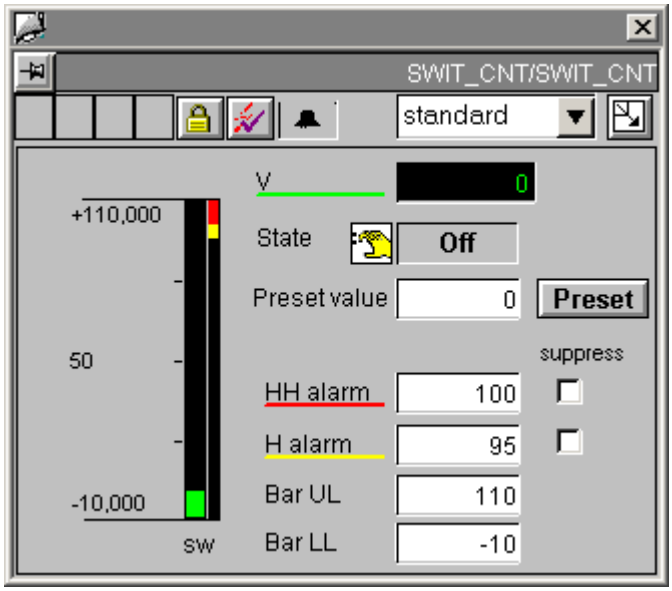
12.1.49 SWIT_CNT (All Views)

Overview

SWIT_CNT: Standard view (Page 546)

Global view: Message view (Page 557)

12.1.50 SWIT_CNT: Standard view



Sequence and positioning of direct connections to controllable objects

@Level5	-->	**Operator-control enable**
VTRACK_OP_AnalogValue	-->	Operator-control enable
TRACK_OP_ButtonBit	-->	Operator-control enable
@Level5	-->	**Back color**
VTRACK_OP_AnalogValue	-->	BackColor_Value
@Level6	-->	**Operator-control enable**
AlarmHigh_AnalogValue	-->	Operator-control enable
WarningHigh_AnalogValue	-->	Operator-control enable
MO_HOUHR_PCS7_AnalogValue	-->	Operator-control enable
MO_HOULR_PCS7_AnalogValue	-->	Operator-control enable
AlarmHigh_CHECKBOX_R	-->	Operator-control enable
WarningHigh_CHECKBOX_R	-->	Operator-control enable

@Level6	-->	**Back color**
AlarmHigh_AnalogValue	-->	BackColor_Value
WarningHigh_AnalogValue	-->	BackColor_Value
MO_HOUHR_PCS7_AnalogValue	-->	BackColor_Value
MO_HOULR_PCS7_AnalogValue	-->	BackColor_Value

Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.1.51 VAL_MOT (All Views)

Overview

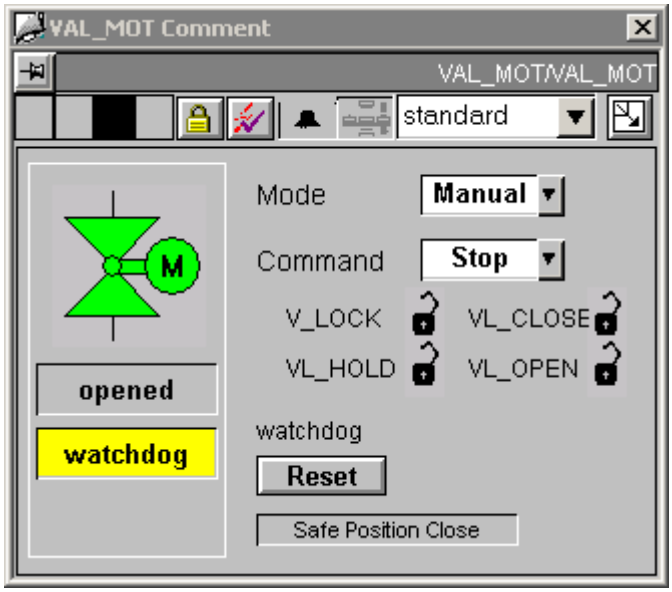
VAL_MOT: Standard view (Page 548)

VAL_MOT: Maintenance view (Page 549)

Global view: Message view (Page 557)

Global view: Batch view (Page 558)

12.1.52 VAL_MOT: Standard view



Sequence and positioning of direct connections to controllable objects

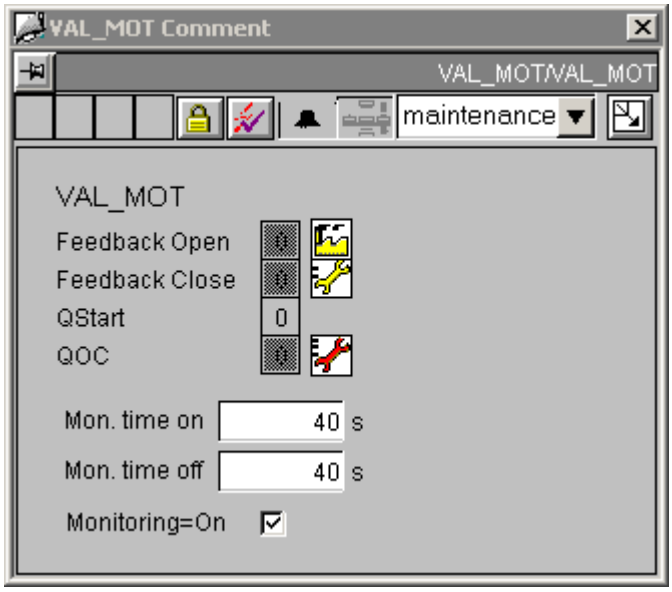
@Level5	-->	**Operator-control enable**
Auto_Manual_COMBOBOX	-->	Operator-control enable
Open_Close_Stop_3COMBOBOX	-->	Operator-control enable
Reset_ButtonBit	-->	Operator-control enable

Additional information

You will find more information in:

Objects in the overview (Page 555)

12.1.53 VAL_MOT: Maintenance view



Sequence and positioning of direct connections to controllable objects

@Level6	-->	**Operator-control enable**
Time_ON_AnalogValue	-->	Operator-control enable
TIME_OFF_PCS7_AnalogValue	-->	Operator-control enable
@Level6	-->	**Back color**
Time_ON_AnalogValue	-->	Back color
TIME_OFF_PCS7_AnalogValue	-->	Back color
Monitoring_ON_CHECKBOX_L1	-->	Back color

Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

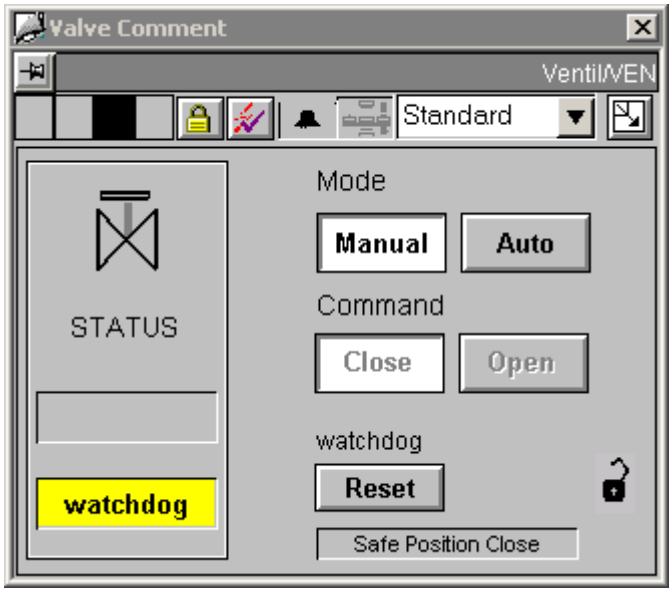
The quality code display (Page 614)

12.1.54 VALVE (All Views)

Overview

VALVE: Standard view (Page 550)
VALVE: Maintenance view (Page 552)
Global view: Message view (Page 557)
Global view: Batch view (Page 558)

12.1.55 VALVE: Standard view



The two interlock icons adjacent to "Close" and "Open" belong to the "Lock to Close" and "Lock to Open" interlocks. They are set to invisible by default, since the "Lock to Safe Position" interlock icon at the bottom right is used primarily.

The icons can always be switched to visible for all instances of the type VALVE, if required.

Sequence and positioning of direct connections to controllable objects

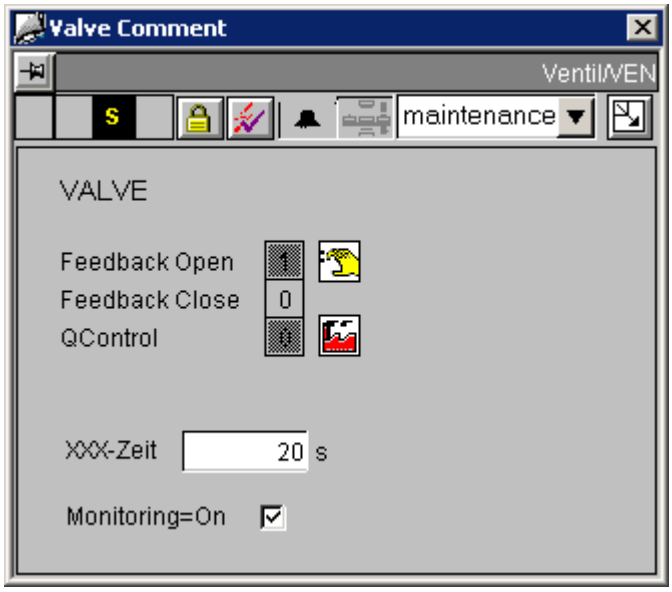
@Level5	-->	**Operator-control enable**		
Permission_Manual_BinOp0	-->	Level_Source	-->	Level_Target
Permission_Automatic_BinOp1	-->	Level_Source	-->	Level_Target
Permission_Off_BinOp2	-->	Level_Source	-->	Level_Target
Permission_On_BinOp1	-->	Level_Source	-->	Level_Target
Reset_ButtonBit	-->	Operator-control enable		
Permission_Manual_BinOp0	-->	**Target_Operator-ControlEnable**		
Manual_BinOp0	-->	Operator-control enable		
Permission_Automatic_BinOp1	-->	**Target_Operator-ControlEnable**		
Automatic_BinOp1	-->	Operator-control enable		
Permission_Close_BinOp2	-->	**Target_Operator-ControlEnable**		
Close_BinOp2	-->	Operator-control enable		
Permission_Open_BinOp1	-->	**Target_Operator-ControlEnable**		
Open_BinOp1	-->	Operator-control enable		
In addition to the WinCC authorization levels, the permission objects also evaluate the following AS-block parameters:				
Permission_Manual_BinOp0	-->	"QMANOP = TRUE"		
Permission_Automatic_BinOp1	-->	"QAUTOP = TRUE"		
Permission_Off_BinOp2	-->	"QOFF_OP = TRUE"		
Permission_On_BinOp1	-->	"QON_OP = TRUE"		

Additional information

You will find more information in:

Objects in the overview (Page 555)

12.1.56 VALVE: Maintenance view



Sequence and positioning of direct connections to controllable objects

@Level6	-->	**Operator-control enable**
Monitoring_ON_CHECKBOX_L1	-->	Operator-control enable
Monitoring_PCS7_AnalogValue	-->	Operator-control enable

Additional information

You will find more information in the following sections:

Objects in the overview (Page 555)

The quality code display (Page 614)

12.2 Faceplates: Asset management

12.2.1 Views of the ASSETMON Faceplate [Asset]

Views

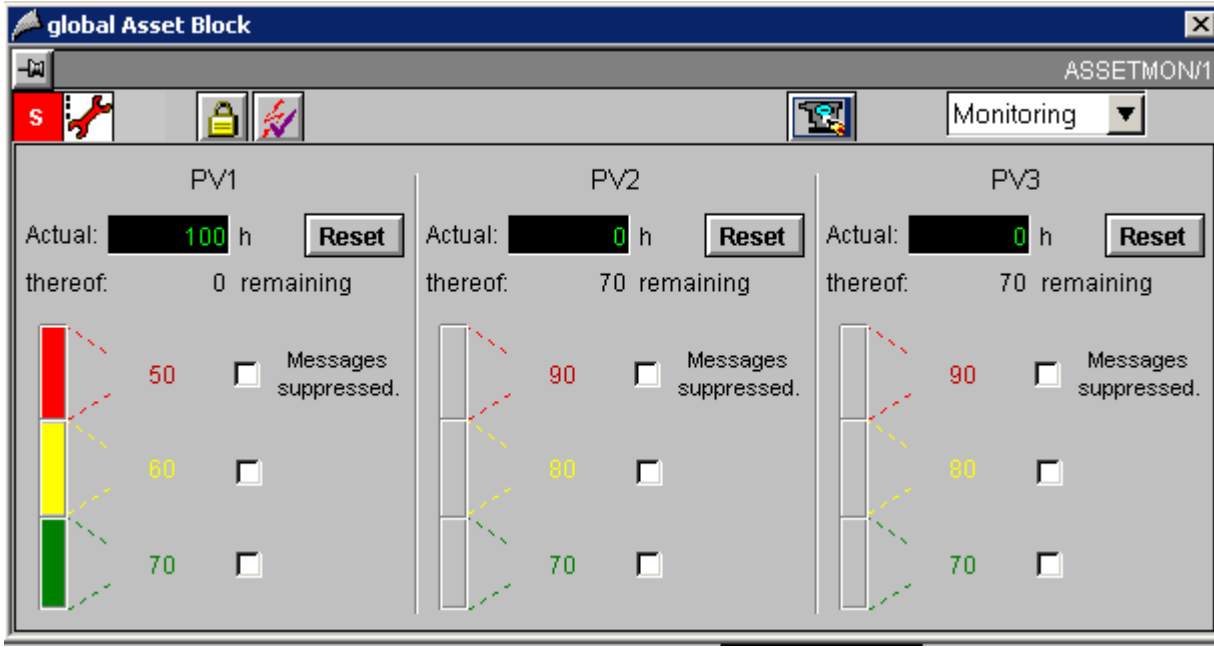
The faceplate has five views:

- Monitoring view
- Diagnostic View
- Ident View
- Maintenance view
- Message View

Monitoring view

In this view, up to three process values are displayed. The texts for this are read from the EDD.

If there are no texts for the process value, the monitoring view is not active (grayed out).

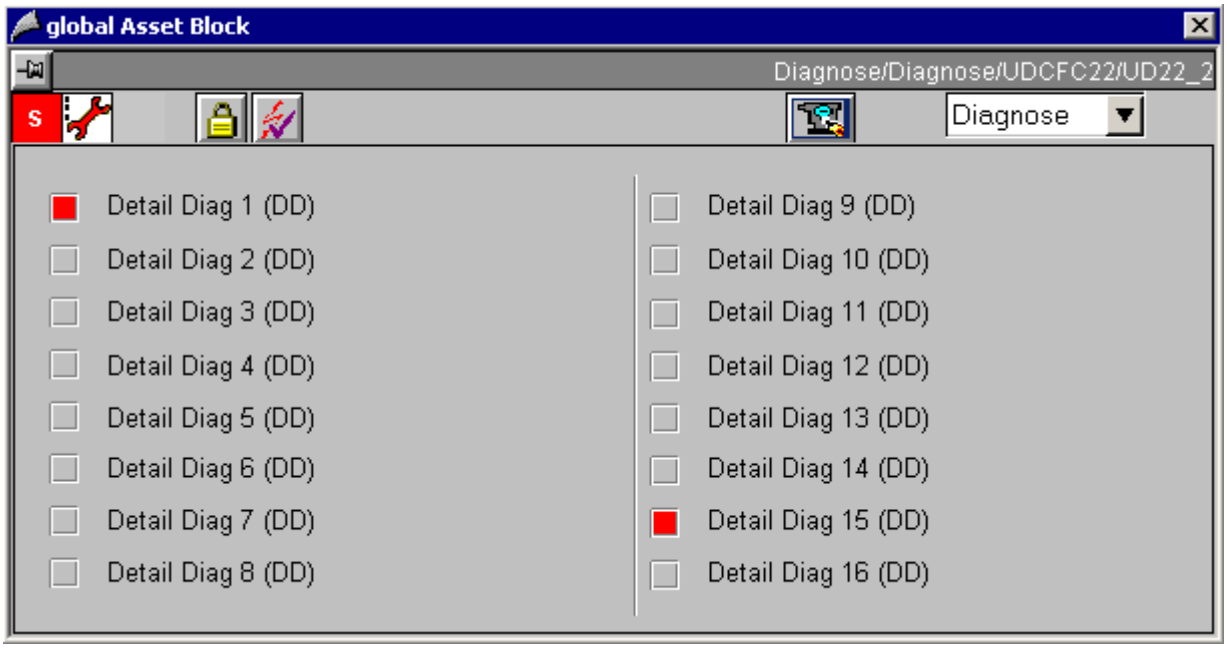


The actual value is displayed in the upper display. In the box under it, the difference between the actual value and the next limit value is displayed.

The limit value display is displayed in the GOOD status in the form of a gray bar. If a limit is reached, a color frame appears at the corresponding position. The bar does not represent an analog value but rather the status. Using the Options box next to the limit display, you can suppress the message (by checking the box).

Diagnostic View

In the diagnostic view, the texts for 16 detailed diagnoses are displayed. The texts correspond to the entries assigned in the EDD in PDM.



In the diagnostic view, the text of the detailed diagnosis is identified by a status display and is therefore relevant if the block input DIAGx = 1 is set.

Ident View

See Ident view [asset] (Page 564)

Maintenance view

See Maintenance view [asset] (Page 562)

Message View

See Message view [asset] (Page 561)

Additional information

You will find more information in:

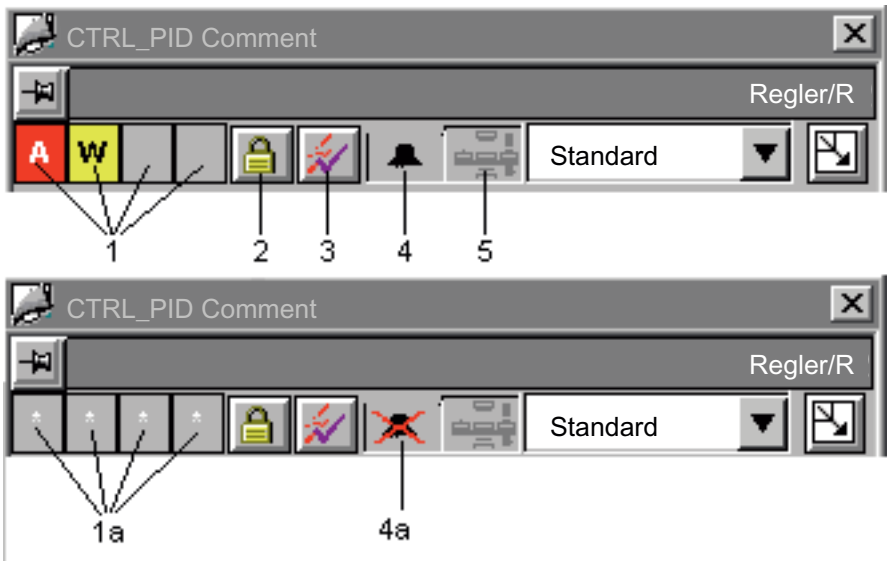
Global representations and views of asset faceplates (Page 568)

12.3 Global views and representations

12.3.1 Overview objects

Toolbar Icons

Each faceplate has a toolbar containing the following icons:



Pinning the faceplate

In the left-hand corner above the overview row, there is an icon that can be used to "pin" a faceplate so that it remains during an area change. The key appears as follows:

- Not pinned  (once the faceplate has been called)
- Pinned  (once the button has been pressed)

The faceplate remains pinned until it is closed again. Pressing the key again has no effect.

Group display

The group display [1] shows the information that is transferred from ALARM_8P of the block instance to WinCC. The group display is linked to the "EventState" of the variable.

Lock/Unlock Messages

The "Lock/unlock messages" function is implemented in the overview with a key [**2**].

The group displays shows whether all the messages of a block instance are locked or unlocked. If a white * character [**1a**] appears in all 4 of the group-display fields, all messages are locked.

The button is only visible to operators assigned the permission level defined in the block symbol property "Processcontrolling_backup".

Acknowledge messages

With key [**3**], all messages of the block instance can be acknowledged.

The button is only visible to operators assigned the permission level defined in the block symbol property "Processcontrolling_backup".

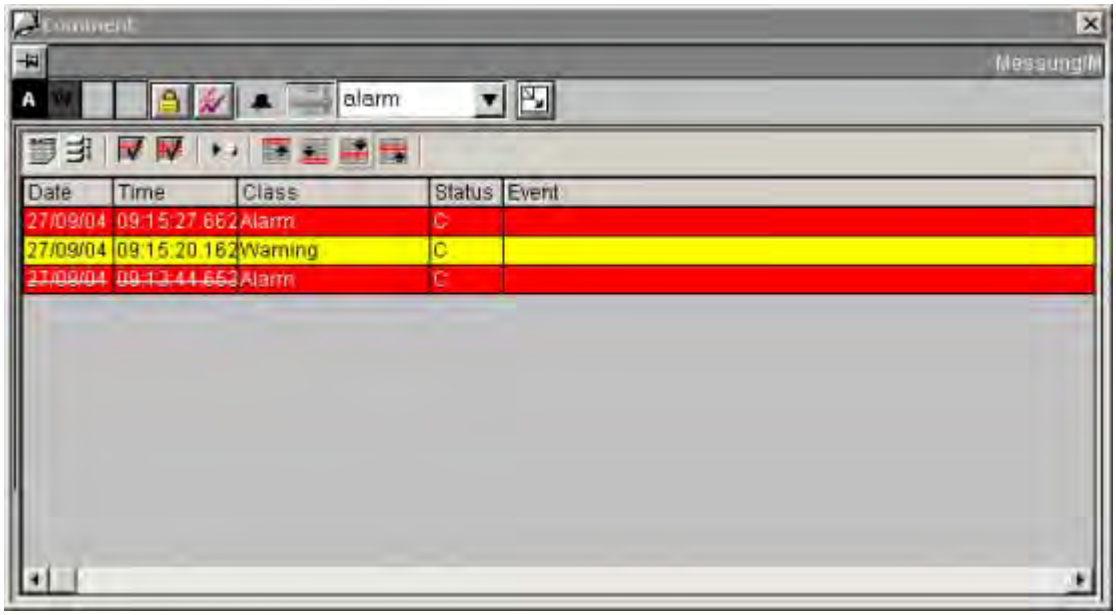
Suppress Messages

Message suppression indicates whether the "Suppress process messages" function in the AS block is deactivated [**4**] or activated [**4a**] using the "MSG_LOCK" parameter. If message suppression is activated, all messages in this block instance are suppressed – except for process control messages.

Occupied Display

The occupied display [**5**] indicates whether the SIMATIC BATCH block instance is occupied ("OCCUPIED" parameter). Additional information is then displayed in the batch view.

12.3.2 Global View: Message View



Faceplates containing message view

The following faceplates contain a message view:

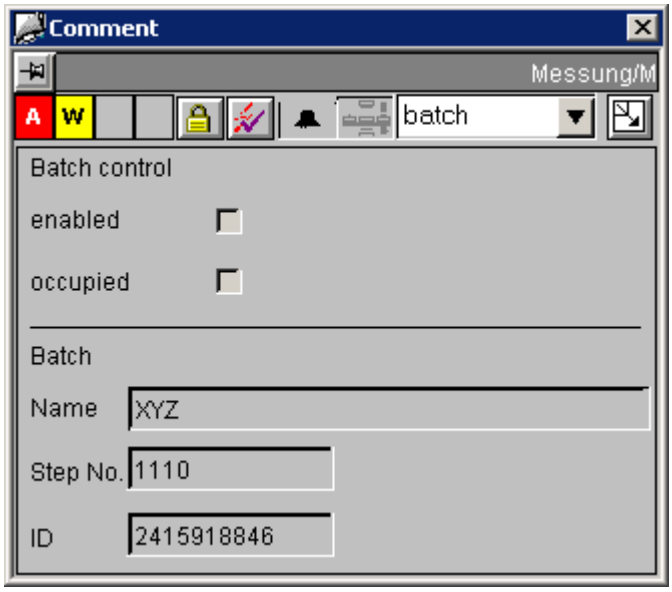
- CTRL_PID
- CTRL_S
- ELAP_CNT
- FMCS_PID
- FMT_PID
- DIG_MON
- DOSE
- MEAS_MON
- MOT_REV
- MOT_SPED
- MOTOR
- SWIT_CNT
- VAL_MOT
- VALVE

Additional information

You will find more information in:

Objects in the overview (Page 555)

12.3.3 Global View: Batch view



Faceplates containing the batch view

The following faceplates contain a batch view:

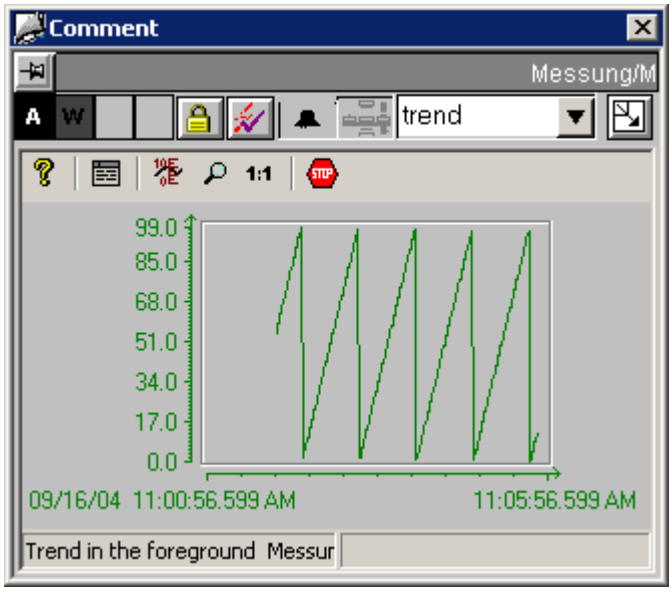
- CTRL_PID
- CTRL_S
- FMCS_PID
- FMT_PID
- DIG_MON
- DOSE
- MEAS_MON
- MOT_REV
- MOT_SPED
- MOTOR
- VAL_MOT
- VALVE

Additional information

You will find more information in:

Objects in the overview (Page 555)

12.3.4 Global View: Trend view



Faceplates containing trend view

The following faceplates contain a trend view:

- CTRL_PID
- CTRL_S
- FMCS_PID
- FMT_PID
- DOSE
- MEAS_MON

Note

You will find more information on configuring trend variables in the manual titled "PCS 7 - Programming Instructions for Blocks", chapter "Configuring the trend view".

Additional information

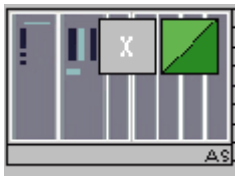
You will find more information in:

Objects in the overview (Page 555)

12.3.5 Display for avoiding stop without asset management

"OB_BEGIN" block icon

If your system does not have ASSET diagnostics, a separate block icon is provided to on the OS display avoidance of stop in the template @Template.pdl. The block icon is stored in the "Diagnostics" section under the name "OB_BEGIN".



Configuration

You configure the OB_BEGIN block icon for each AS. You then interconnect each block icon with the corresponding OB_BEGIN structure variable.

To achieve all the required interconnections to the block icon, it is best to use the PCS 7 WinCC Wizard for interconnecting faceplates to process tags. In the tag dialog "List of all structure variables", you can select the relevant OB_BEGIN instance.

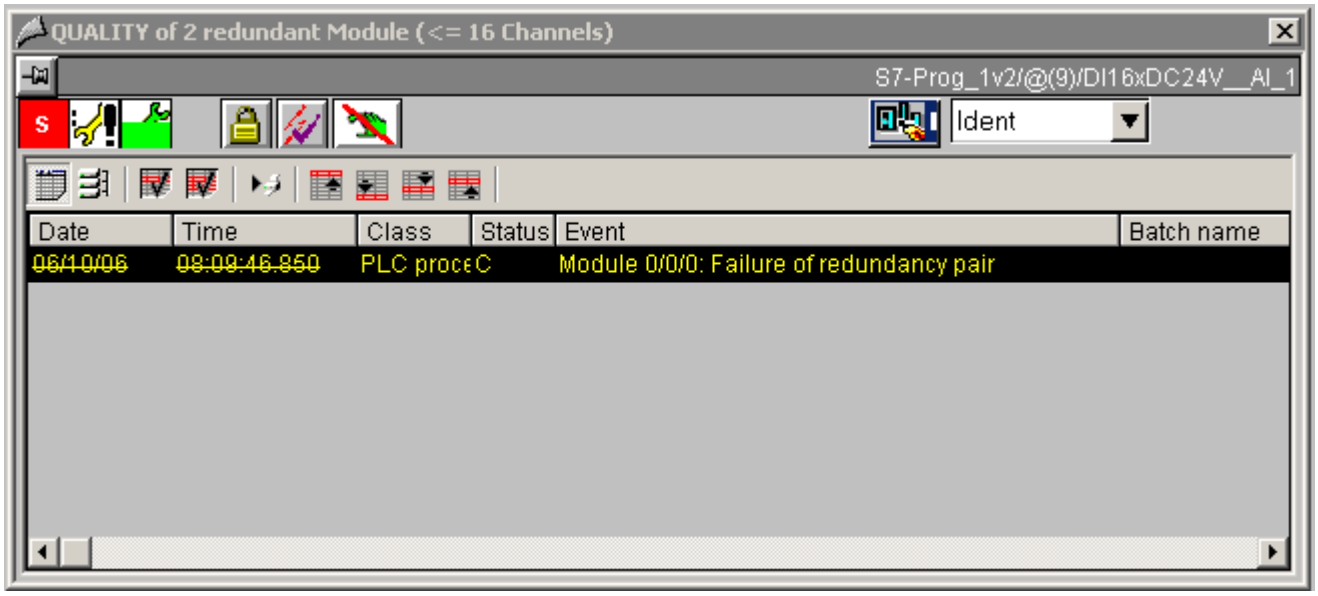
Note on the "OB_BEGIN" faceplate

In the OB_BEGIN faceplate for the OB_BEGIN and CPU_RT blocks without asset management, the message view, the performance view and the detailed views (OB3x and OB8x/OB1) are displayed if SFC78 is supported on the AS. If SFC78 is not supported, only the message view of the faceplate is displayed.

The identification view and parameter view are not shown.

12.3.6 Message View [Asset]

Layout



Differences to the global message view:

Current pending messages are displayed, regardless of their acknowledgment status.

In addition to the "tagname" filter, only messages relating to "Diagnostics" are displayed.

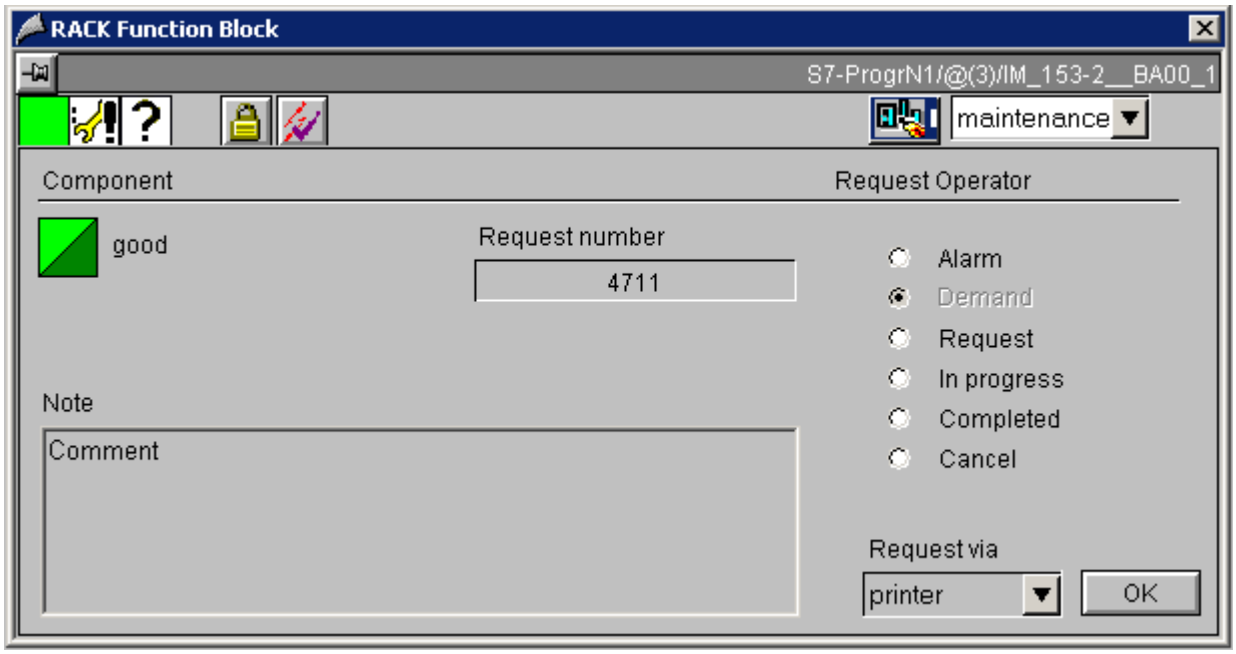
Additional information

You will find more information in:

Global representations and views of asset faceplates (Page 568)

12.3.7 Maintenance View [Asset]

Layout



The maintenance view of an asset faceplate has the following display and operator control elements:

- **Request number**: The job number assigned for the maintenance job is displayed here.
- **Request operator**: The maintenance requests from the different components are defined here. The maintenance operator determines the status that is set for maintenance. The following statuses are available:
  - Alarm
  - Demand
  - Request
  - In progress
  - Completed
  - Cancel

  If one of the listed radio buttons is clicked, a dialog box opens, in which a comment and a job number can be entered. The request status icon then changes.

- Note: The comments entered for the maintenance job are displayed here.
- Request via: (At present, only "Printer" is supported) Printout of a report for the selected faceplate

Note

A second print job may not be initiated until the previous job is complete.

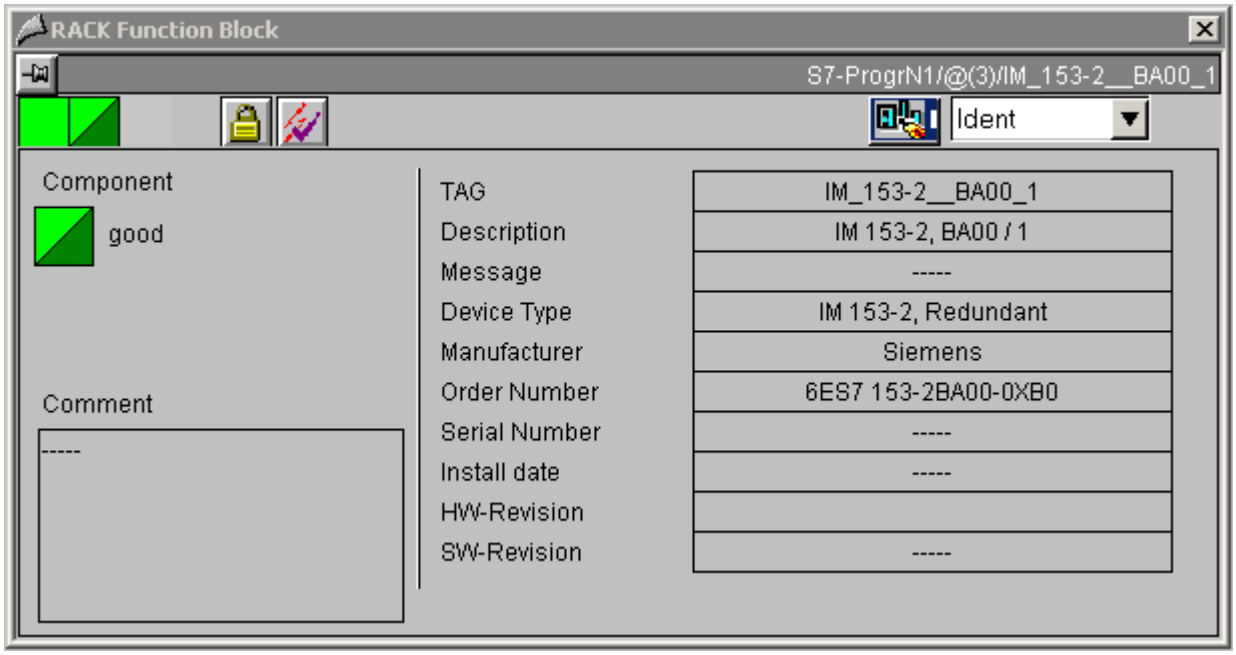
Additional information

You will find more information in:

Global representations and views of asset faceplates (Page 568)

12.3.8 Ident View [Asset]

@PG_ASSETAS_Ident



The layout of the Ident page is the same for all faceplates; only the dynamic sampling of the ident data is different.

Exception: For devices containing a Web server, a button is available to call a viewer that displays the HTML pages on the devices.

This button is displayed in the overview row of the faceplate if

- the device was configured for SNMP
- the property "Web-compliant" is stored in the block icon.

This button is used in the OSM faceplate. The button is also displayed when the network components are unreachable for certain reasons.

Note that with the MS Windows 2003 Server operating system, the IP addresses of the network components to be reached must be included in the "Trusted sites" security settings.

AS faceplate

On the AS faceplate, the ident data are read out of the properties of the calling block icon. The information is determined from the HW Config data when the block icons are created/updated.

Ident Data	Property in block icon	Origin
TAG	"tag"	CFC "block name"
Description	"HWComment"	HW Config "Designation"/"Name"
Notification	- - -	Not available
Device type	"HWType"	HW Config "Type"
Manufacturer	"HWVendor"	Not accessible
Order number	"HWOrderNo"	HW Config "Order number"
Serial number	- - -	Not available
Install date	- - -	Not available
HW revision	- - -	Not available
SW revision	"HWSWVersion"	Not accessible
Comment	"HWComment"	HW Config "Comment"
"Not accessible" means that the user cannot make an entry or modification here.		

PC faceplate

No identification data is currently displayed in the PC faceplate.

IPC faceplate

For IPC faceplates, the identification data of the variables is read from the data manager. These variables are created using the "Export variables for WinCC" function in the OPC server properties or with the "Create/update diagnostic screens" function followed by compiling the OS.

Ident Data	Variable / property in the block icon	Origin
TAG	"sysName"	Windows computer name
Description	"sysLocation"	DiagMonitor
Notification	"&ipAddress()"	Windows IP address
Device type	"ProductName"	DiagMonitor
Manufacturer	"Manufacturer"	DiagMonitor
Order number	---	Not available
Serial number	"SerialNumber"	DiagMonitor
Install date	---	Not available
HW revision	"HwVersion"	DiagMonitor
SW revision	"SwVersion"	DiagMonitor
Comment	"&comment"	Object properties PC station "Comment"

OSM faceplate

For OSM faceplates, the Ident data of the variables are read from the data manager. These variables are created using the "Export variables for WinCC" function in the OPC server properties or with the "Create/update diagnostic screens" function followed by compiling the OS.

Ident Data	Variable / property in the block icon	Origin
TAG	"sysName"	WBM *) "System Name"
Description	"sysLocation"	WBM *) "System Location"
Notification	"&ipAddress()"	HW Config/OPC server "IP address"
Device type	"sysDescr"	WBM "Device Type"
Manufacturer	- - -	Not available
Order number	- - - / "snInfoOrderNr" **)	Not available / WBM "Order Number"
Serial number	- - - / "snInfoSerialNr" **)	Not available / WBM "MAC Address"
Install date	- - -	Not available
HW revision	- - - / "snHWVersion" **)	Not available / WBM "Hardware"
SW revision	- - - / "snSWVersion" **)	Not available / WBM "Firmware"
Comment	HWComment"	HW Config/OPC server "Comment"
*) WBM = Web based management (Web interface of the network object)		
**) If the SNMP "MIB-II" profile is used, this data is not available		

PDM faceplate

On the PDM faceplate, the ident data is read out and displayed in XML format via of a COM interface to PDM. The data is stored under the following labels:

Ident Data	Label / property in the block icon	Origin
TAG	"Device.App.Ident.TAG"	EDD PDM "TAG"
Description	"Device.App.Ident.Description"	EDD PDM "Descriptor"
Notification	"Device.App.Ident.Message"	EDD PDM "Message"
Device type	"Device.Type.Ident.Type"	EDD PDM "Product designation"
Manufacturer	"Device.Type.Ident.Manufacturer"	EDD PDM "Manufacturer"
Order number	"Device.Type.Ident.OrderNumber"	EDD PDM "Device order number"
Serial number	"Device.Ident.SerialNumber"	EDD PDM "Device serial number"
Install date	"Device.App.Ident.InstallDate"	EDD PDM "Install date"
HW revision	"Device.Type.Ident.HwRevision"	EDD PDM "Hardware revision"
SW revision	"Device.Type.Ident.SwRevision"	EDD PDM "Software revision"
Comment	"Device.Type.Ident.Comment"	HW Config "Comment"

ASSETMON faceplate

On the ASSETMON faceplate, the ident data is read out and displayed in XML format via of a COM interface to PDM. The data is stored under the following labels:

Ident Data	Label / variable	Origin
TAG	"Device.App.Ident.TAG"	EDD PDM "TAG"
Description	"Device.App.Ident.Description"	EDD PDM "Descriptor"
Notification	"Device.App.Ident.Message"	EDD PDM "Install location"
Device type	"Device.Type.Ident.Type"	Not available
Manufacturer	"Device.Type.Ident.Manufacturer"	EDD PDM "Manufacturer"
Order number	"Device.Type.Ident.OrderNumber"	EDD PDM "Device order number"
Serial number	"Device.Ident.SerialNumber"	EDD PDM "Serial number"
Install date	"Device.App.Ident.InstallDate"	EDD PDM "Install date"
HW revision	"Device.Type.Ident.HwRevision"	EDD PDM "Hardware revision"
SW revision	"Device.Type.Ident.SwRevision"	EDD PDM "Software revision"
Comment	"&comment"	CFC "Block comment"

Additional information

You will find more information in:

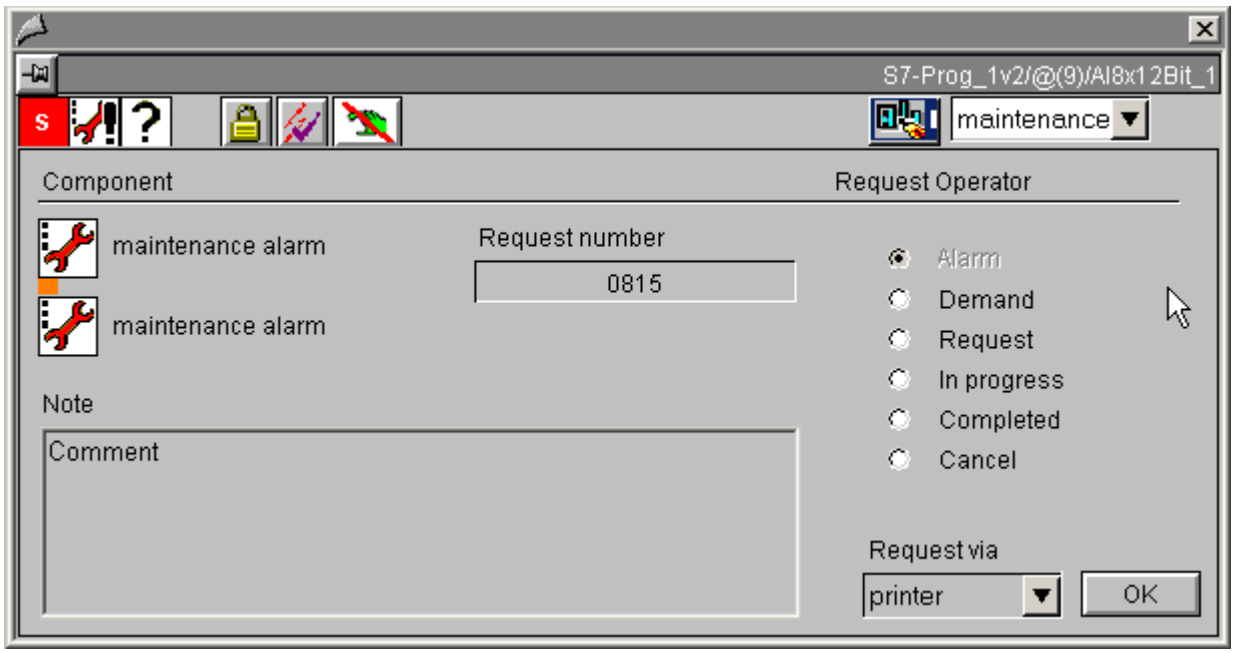
Global representations and views of asset faceplates (Page 568)

12.3.9 Global Representations and Views of Asset Faceplates

Introduction

The following representations and views appear in all asset faceplates.

Icons for diagnostics and maintenance



Icons in the faceplate	Meaning	Icons
S	**Group display with a maintenance alarm** Indicates the group request status that represents the component in the hierarchical group display.	**Possible group display statuses** S Maintenance alarm F Maintenance request M Maintenance required Good

Icons in the faceplate	Meaning	Icons
	**Maintenance status display with a maintenance alarm** Indicates the current total status of the component. It is formed from the sum of the individual statuses of a redundant component and the response/action of the operator. If the operator has not made any settings, the 8 icons listed under "Before operator action" can be displayed in the status display. If the operator performs a status change, the 3 icons listed under "After operator action" can be displayed in the status display. The operator changes the maintenance status in the faceplate "Maintenance" view.	**Before operator action:**  Maintenance alarm  Maintenance request  Maintenance required  Component good  Local operation  At least one process value is being simulated  Out of service  Component passivated  Untested/unknown  Configuration changed **After operator action:**  State change to Maintenance alarm  State change to Maintenance required  State change to Maintenance request

Icons in the faceplate	Meaning	Icons
	**Status display with icon** Question mark = status unclear or not yet clear. Indicates the current processing status of the component. 1. Normal status (no diagnostics pending) --> no icon. 2. Pending diagnostics with no operator response --> no icon. 3. Action triggered by operator as confirmation/reclassification/job --> "?" icon; no activity on the component. 4. Action as in progress --> "In progress" icon	 Repairing component (in progress)
	**Component status display** If 2 icons are displayed under "Component", these indicate the component itself and its redundant component. If the components are not redundant, only one icon with text appears under "Component". The icons listed under "Component" indicate the original status of the component, as reported by the AS.	**Possible statuses in the status display**  Maintenance alarm  Maintenance request  Maintenance required  Component good  Local operation/local Overwrite  At least one process value is being simulated  Out of service  Component passivated  Untested/unknown  Configuration changed

Note

If the maintenance status is "Untested/unknown", all the other dynamic displays in the asset-management faceplates are not relevant for this instance.

Overview

The following keys are added to the overview of asset faceplates, which always looks the same:

- The key for calling HW Config is added to **@PG_ASSETAS_Overview**:

  The icon only appears if STEP 7 is also installed on the computer on which the faceplate was opened.

- The keys for calling PDM and HW Config are added to **@PG_ASSETPDM_Overview**:

  These keys only appear if PDM and STEP 7 are also installed on the computer, on which the faceplate was opened.

- The key for depassivation is added to **@PG_ASSETAS_Overview**:

  The icon only appears if STEP 7 is also installed on the computer on which the faceplate was opened.

Additional information

You will find more information in:

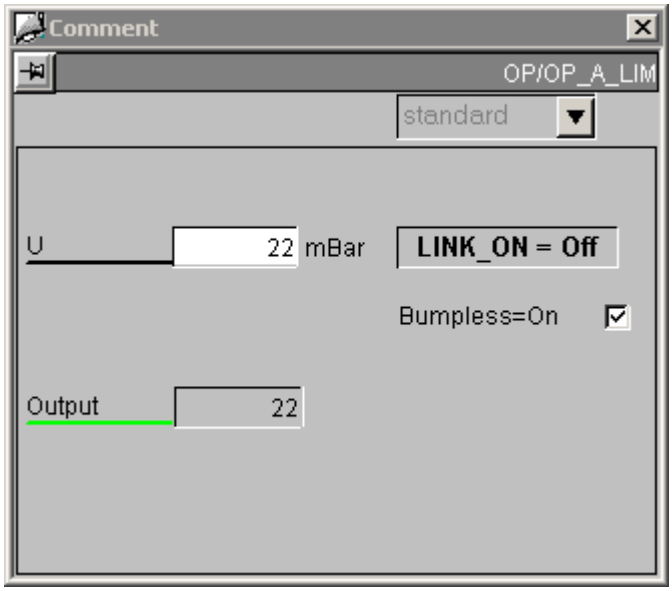
Objects in the overview (Page 555)

12.4 Faceplates: Operator-control blocks

12.4.1 OP_A: Standard view

See: OP_A_LIM: Standard view (Page 572).

12.4.2 OP_A_LIM: Standard view



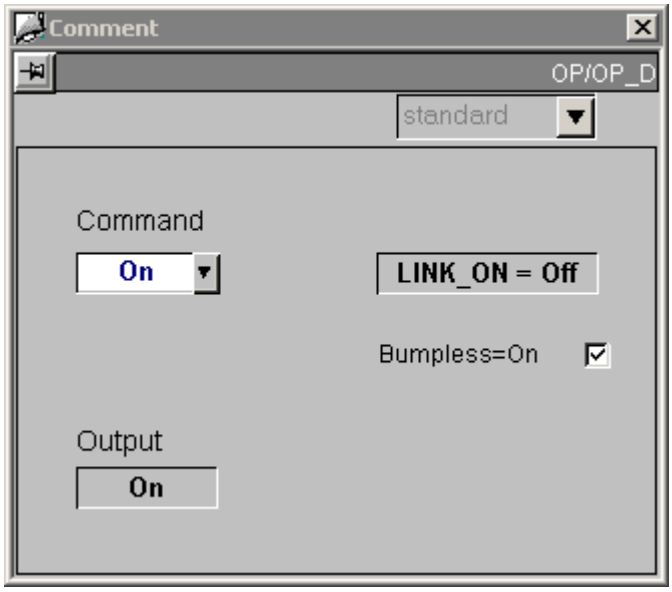
Sequence and positioning of direct connections to control objects

@Level5	-->	**Operator-control enable**
Permission Setpoint	-->	LevelSource
Permission Setpoint	-->	**Target_Operator-ControlEnable**
Setpoint_PCS7_AnalogValue	-->	Operator-control enable
@Level6	-->	**Operator-control enable**
BumplessOn_CHECKBOX_L	-->	Operator-control enable

12.4.3 OP_A_RJC: Standard view

See: OP_A_LIM: Standard view (Page 572).

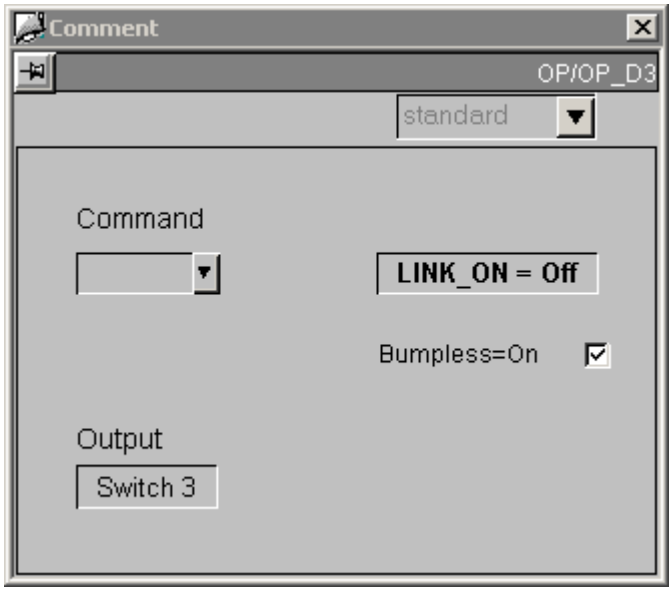
12.4.4 OP_D: Standard view



Sequence and positioning of direct connections to control objects

@Level5	-->	**Operator-control enable**
I0_PCS7_COMBOBOX	-->	Operator-control enable
@Level5	-->	**BackColor**
I0_PCS7_COMBOBOX	-->	BackColor Text1
I0_PCS7_COMBOBOX	-->	BackColor Text2
@Level6	-->	**Operator-control enable**
BumplessOn_CHECKBOX_L	-->	Operator-control enable

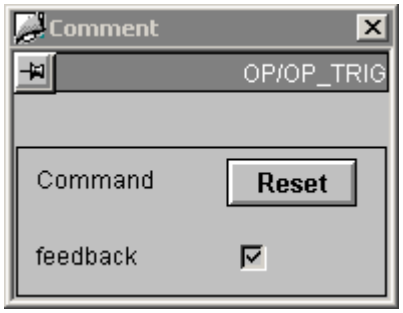
12.4.5 OP_D3: Standard view



Sequence and positioning of direct connections to control objects

@Level5	-->	**Operator-control enable**
Command_PCS7_3COMBOBOX	-->	Operator-control enable
@Level5	-->	**BackColor**
Command_PCS7_3COMBOBOX	-->	BackColor Text1
Command_PCS7_3COMBOBOX	-->	BackColor Text2
Command_PCS7_3COMBOBOX	-->	BackColor Text3
@Level6	-->	**Operator-control enable**
BumplessOn_CHECKBOX_L	-->	Operator-control enable

12.4.6 OP_TRIG: Standard view



The key labeled "Reset" writes a logical 1 to the "I0" parameter of the OP_TRIG block.

The "Reset" text is read from the "s7_shortcut" attribute of the parameter and can be adapted to specific instances.

If the "Feedback" check box is checked, the "SIGNAL" parameter of the OP_TRIG block is read.

The "Feedback" text is configured in the faceplate and is thus type-specific.

Access control

As well as the WinCC authorization levels, the "Permission_Reset" permission object also evaluates the "QOP_EN = TRUE" parameter.

Sequence and positioning of direct connections to control objects

@Level5	-->	**Operator-control enable**
Permission_Reset	-->	Level_Source
Permission_Reset	-->	**Target_Operator-ControlEnable**
BumplessOn_CHECKBOX_L1	-->	Operator-control enable

Block icons

13

13.1 General Properties of Block Icons

Basic properties

The following properties of all "@@PCS7Typicals" picture block icons should never be modified:

- Geometry/Width
- Geometry/Height
- Other/OP_enabled
- Other/Password
- Other/Display
- General/Servername
- Styles/Group-relevant (only for blocks with Alarm_8P messages)

Properties which exist in all block symbols:

Properties	Element and property in user object	Object	Description
Other/Processcontrolling _backup	POP.Permission	I/O box	Instance-spec. control permission Default = 5
Other/ HigherProcesscontrolling _backup	HIPOP.Permission	I/O box	Instance-spec. control permission Default = 6
General/tag	NameOfTag.OutputValue	I/O box	Text displayed in the icon
General/type	Type.OutputValue	I/O box	Reference for the generation of icons from the PH and for wizards
General/tagname	Tagname.OutputValue	I/O box	Actual variable name that is passed to the variable prefixes of the picture windows
General/Servername	Servername.OutputValue	I/O box	Block/faceplate type
General/Version	Version.OutputValue	I/O box	Version number
Styles/View_Tag	NameOfTag.Display Rectangle17.Display (if it exists)	Rectangle I/O box	Can be used to hide the variable name
MouseClick left	PCS7_OpenGroupDisplay _V6 (lpszPictureName, lpszObjectName)		Calls the faceplate

Properties of block icons that are not overwritten when the plant hierarchy is updated:

Properties that exist in all block icons:

- HigherProcesscontrolling_backup
- Processcontrolling_backup
- View_Tag

Properties that do not exist in all block icons:

- ReturnPath
- StandardTrend
- Format_InputValue
- Format_OutputValue
- Format_xx

Additional information

You will find more information in the following sections:

Position of faceplates (Page 578)

Highlighting the block icon for "Loop in alarm" and "Select picture via process tag" (Page 579)

13.2 Position of Faceplates

Specifying the Position

The position of the faceplate can be specified when the block icon is opened.

Each block icon possesses three properties for this purpose:

- **DefaultPos**. If this property is set to TRUE, the faceplate call behaves as before; this is also the default setting. If this property is set to FALSE, the faceplate is called at the position specified with the properties "leftPos/topPos".
- **LeftPos** = Horizontal position at which the upper left point of the faceplate is situated.
- **TopPos** = Vertical position at which the upper left point of the faceplate is situated.

13.3 Highlighting the Block Icon for "Loop in Alarm" and "Select Picture via Process Tag"

Setting Highlighting

The functions “Loop in alarm” and “Select picture via process tag” enable the program to jump to the corresponding process picture of the process tag.

The associated block icon of the process tag is highlighted using the "HighlightBlockIcon" function. The "tagname" variable is highlighted in cyan blue.

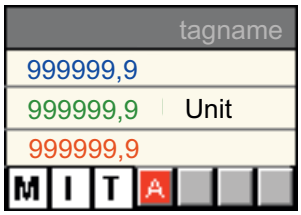
The block icon has two properties for this purpose:

- **HighLightBlockIcon** – Set by the functions "Loop in alarm" and "Select picture via process tag"; highlights the "tagname" variable in cyan blue.
- **HighLightBlockIconBackColor** – Color for highlighting The default setting is "cyan".

13.4 Block icons: Plant blocks

13.4.1 Block icon: CTRL_S

Properties



For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 110/Height = 77		
General/UnitPV	Unit.Text/.PV_IN#unit	Stat.Text	Display: PV unit
General/Unit_MAN_OP	Unit.Text/.MAN_OP#unit	Stat.Text	Display: Manipulated-variable unit
Links/CollectValue	GroupDisplay.CollectValue/.Event State	Group display	
Links/SetpointValue	SetpointValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Setpoint
Links/ProcessValue	ProcessValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Process value
Links/OutputValue	OutputValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Manipulated variable
Links/Mode_MAN_AUT	Manual_AdvancedStatus Display.Status	AdvancedStatusDis.	Display: Manual/Auto
Links/Mode_INT_EXT	External_AdvancedStatus Display.Status	AdvancedStatusDis.	Display: External/Internal
Links/LMN_SEL	Tracking_AdvancedStatus Display.Status	AdvancedStatusDis.	Display: Manipulated-variable correction
Links/QLMNR_ON	OutputValue_AdvancedAnalog Display.Display Unit_MAN_OP.Display	AdvancedAnalogDis. Stat.Text	See below for description
Links/QLMNUP	LMNUP_StatusDisplay	Stat.Text	Display: QLMNUP
Links/QLMNDN	LMNDN_StatusDisplay	Stat.Text	Display: QLMNDN
Styles/ReturnPath	TrendFunctions2 .OutputValue	I/O box	
Styles/StandardTrend	TrendFunctions2 .FontSize	I/O box	

Properties	Element and property in user object	Object	Description
Styles/Format_InputValue	ProcessValue_AdvancedAnalog Display.Format SetpointValue_AdvancedAnalog Display.Format	AdvancedAnalogDis. AdvancedAnalogDis.	Formats the numbers for process value and setpoint
Styles/Format_OutputValue	OutputValue_AdvancedAnalog Display.Format	AdvancedAnalogDis.	Formats the numbers for the manipulated variable
Styles/Format_xx	Format_xx.OutputValue	I/O box	Further format

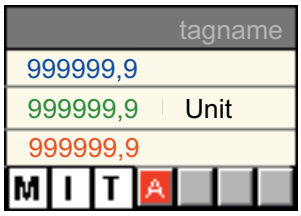
The CTRL_S block icon differs from that of CTRL PID as follows: If position feedback (LMNR_ON = 0) is not available, the program displays the binary control signals QLMNUP and QLMNDN instead of the manipulated variable.

The visibility of these texts is also controlled by scripts. The scripts are called when changes are made to the QLMNUP and QLMNDN properties.

Note: The "OutputValue_AdvancedAnalogDisplay" and "Unit_MAN_OP" objects must always be brought to the foreground in the user object in order to ensure proper functioning of the visualization control.

13.4.2 Block icon: CTRL_PID

Properties



For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 110/Height = 77		
General/UnitPV	UnitPV.Text	Stat.Text	Display: PV unit
General/Unit_MAN_OP	Unit_MAN_OP.Text	Stat.Text	Display: MAN_OP unit
Links/CollectValue	GroupDisplay.CollectValue	GroupDisplay.	
Links/SetpointValue	SetpointValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Setpoint
Links/ProcessValue	ProcessValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Process value
Links/OutputValue	OutputValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Manipulated variable
Links/LMN_SEL	Tracking_AdvancedStatus Display.Status	AdvancedStatusDis.	Display: Manipulated-variable correction
Links/Mode_MAN_AUT	Manual_AdvancedStatus Display.Status	AdvancedStatusDis.	Display: Manual/Auto
Links/Mode_INT_EXT	External_AdvancedStatus Display.Status	AdvancedStatusDis.	Display: External/Internal
Styles/ReturnPath	TrendFunctions2 .OutputValue	I/O box	
Styles/StandardTrend	TrendFunctions2 .FontSize	I/O box	
Styles/Format_InputValue	ProcessValue_AdvancedAnalog Display.Format SetpointValue_AdvancedAnalog Display.Format	AdvancedAnalogDis. AdvancedAnalogDis.	Formats the numbers for process value and setpoint
Styles/Format_OutputValue	OutputValue_AdvancedAnalog Display.Format	AdvancedAnalogDis.	Formats the numbers for the manipulated variable
Styles/Format_xx	Format_xx.OutputValue	I/O box	Further format

13.4.3 Block icon: DIG_MON

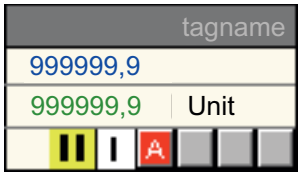
Properties

For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 90/Height = 40		
Links/Status	StatusDisplay.ActualStatus	Status display	.Q
Links/CollectValue	GroupDisplay3.CollectValue	Group display	.EventState

13.4.4 Block icon: DOSE

Properties



For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 110/Height = 63		
General/UnitPV	UnitPV.Text	Stat.Text	Display: PV unit
Links/CollectValue	GroupDisplay.CollectValue	Group display	.EventState
Links/ProcessValue	ProcessValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Process value
Links/SetpointValue	SetpointValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Setpoint
Links/SetpointExternal	External_AdvancedStatus Display.Status SetpointExternValue_ AdvancedAnalogDisplay.Display	AdvancedStatusDis. AdvancedAnalogDis.	.QSPEXTON See below for description
Links/SetpointExternalValue	SetpointExternValue_ AdvancedAnalogDisplay.Value	AdvancedAnalogDis.	Displayed with .QSPEXTON using setpoint
Styles/ReturnPath	TrendFunctions2 .OutputValue	I/O box	
Styles/StandardTrend	TrendFunctions2 .FontSize	I/O box	

Properties	Element and property in user object	Object	Description
Styles/Format_InputValue	ProcessValue_AdvancedAnalog Display.Format SetpointValue_AdvancedAnalog Display.Format	AdvancedAnalogDis. AdvancedAnalogDis.	Formats the numbers for process value and setpoint
Styles/Format_OutputValue	OutputValue_AdvancedAnalog Display.Format	AdvancedAnalogDis.	Formats the numbers for the manipulated variable
Styles/Format_xx	Format_xx.OutputValue	I/O box	Further format

The DOSE block does not contain a parameter which represents the effective setpoint. The program outputs the setpoint display for this reason, depending on the QSPEXTON status.

QSPEXTON = 0 --> "SetpointValue_AdvancedAnalogDisplay" is displayed.

QSPEXTON = 1 --> "SetpointExternValue_AdvancedAnalogDisplay" is displayed.

13.4.5 Block icon: ELAP_CNT

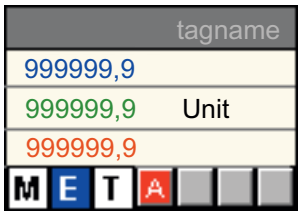
Properties

For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 97/Height = 45		
General/Unit	Unit.Text	Stat.Text	
Links/CollectValue	GroupDisplay.CollectValue	Group display	.EventState
Links/Output_Value	ProcessValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	.HOURS Display: Max. 7 digits
Styles/Format_InputValue	ProcessValue_AdvancedAnalog Display.Format	AdvancedAnalogDis.	Formats the numbers for process value and setpoint
Styles/Format_OutputValue	Format_OutputValue .OutputValue	I/O box	Formats the numbers for the manipulated variable
Styles/Format_xx	Format_xx.OutputValue	I/O box	Further format

13.4.6 Block icon: FMCS_PID

Properties

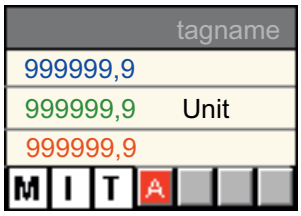


For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 110/Height = 77		
General/UnitPV	Unit.Text/.PV#unit	Stat.Text	
General/Unit_MAN_OP	Unit.Text/.LMN#unit	Stat.Text	
Links/CollectValue	GroupDisplay.CollectValue/ .EventState	Group display	
Links/SetpointValue	SetpointValue_AdvancedAnalog Display.Value/.SP	AdvancedAnalogDis.	Display: Setpoint
Links/ProcessValue	ProcessValue_AdvancedAnalog Display.Value/.PV	AdvancedAnalogDis.	Display: Process value
Links/OutputValue	OutputValue_AdvancedAnalog Display.Value/.LMN	AdvancedAnalogDis.	Display: Manipulated variable
Links/Tracking	Tracking_AdvancedStatus Display.Status	AdvancedStatusDis.	Display: Tracking LMN
Links/Mode_MAN_AUT	Manual_AdvancedStatus Display.Status	AdvancedStatusDis.	Display: Manual/Auto
Links/Mode_INT_EXT	External_AdvancedStatus Display.Status	AdvancedStatusDis.	Display: External/Internal
Styles/ReturnPath	TrendFunctions2 .OutputValue	I/O box	
Styles/StandardTrend	TrendFunctions2 .FontSize	I/O box	
Styles/Format_InputValue	ProcessValue_AdvancedAnalog Display.Format SetpointValue_AdvancedAnalog Display.Format	AdvancedAnalogDis. AdvancedAnalogDis.	Formats the numbers for process value and setpoint
Styles/Format_OutputValue	OutputValue_AdvancedAnalog Display.Format	AdvancedAnalogDis.	Formats the numbers for the manipulated variable
Styles/Format_xx	Format_xx.OutputValue	I/O box	Further format

13.4.7 Block icon: FMT_PID

Properties



You will find more information on the properties in the section: "FMCS_PID block icon (Page 585)"

13.4.8 Block icon: INTERLOK

Properties

For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 108/Height = 20		
Links/Link	Lock.ActualStatus	StatDis.	Lock icon

13.4.9 Block icon: MEAS_MON

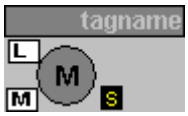
Properties

For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 97/Height = 45		
General/Unit	Unit.Text	Stat.Text	
Links/CollectValue	GroupDisplay.CollectValue	Group display	
Links/OutputValue	ProcessValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Process value
Styles/ReturnPath	TrendFunctions2.OutputValue	I/O box	
Styles/StandardTrend	TrendFunctions2 .FontSize	I/O box	
Styles/Format_InputValue	ProcessValue_AdvancedAnalog Display.Format	AdvancedAnalogDis.	Formats the numbers for process value and setpoint
Styles/Format_OutputValue	Format_OutputValue .OutputValue	I/O box	Formats the numbers for the manipulated variable
Styles/Format_xx	Format_xx.OutputValue	I/O box	Further format

13.4.10 Block icon: MOT_REV

Properties



For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 90/Height = 53		
Links/CollectValue	GroupDisplay_WithASD1.CollectValue	AdvancedStatusDis.	
Links/QMAN_AUT	Mode.Status1	AdvancedStatusDis.	Display: Auto/Manual
Links/LOCK	Interlock.Status1	AdvancedStatusDis.	Display: Lock
Links/QRUN	Motor_Status1.Status1	AdvancedStatusDis.	Display: Motor
Links/QSTOP	Motor_Status1.Status2	AdvancedStatusDis.	Display: Motor
Links/QDIR	Motor_Status1.Status3	AdvancedStatusDis.	Display: Motor

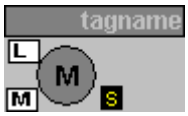
A left-click calls the MOT_REV faceplate. A right-click calls the associated INTERLOK faceplate.

The name of the INTERLOK block is stored as a script transfer parameter.

The default block name is "ILOCK". The INTERLOK block and MOT_REV must be placed into the same CFC chart.

13.4.11 Block icon: MOT_SPED

Properties



For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 90/Height = 53		
Links/CollectValue	GroupDisplay_WithASD1.CollectValue	AdvancedStatusDis.	
Links/QMAN_AUT	Mode.Status1	AdvancedStatusDis.	Display: Auto/Manual
Links/LOCK	Interlock.Status1	AdvancedStatusDis.	Display: Lock
Links/QRUN	Motor_Status.Status1	AdvancedStatusDis.	Display: Motor
Links/QSTOP	Motor_Status.Status2	AdvancedStatusDis.	Display: Motor
Links/QSPEED	Motor_Status.Status3	AdvancedStatusDis.	Display: Motor
Links/QSTOPING	Motor_Status.Status4	AdvancedStatusDis.	Display: Motor

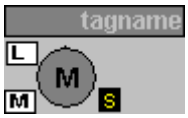
A left-click calls the MOT_SPED faceplate. A right-click calls the associated INTERLOK faceplate.

The name of the INTERLOK block is stored as a script transfer parameter.

The default block name is "ILOCK". The INTERLOK block and MOT_SPED must be placed into the same CFC chart.

13.4.12 Block icon: MOTOR

Properties



For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 90/Height = 54		
Links/CollectValue	GroupDisplay_WithASD.CollectValue	AdvancedStatusDis.	
Links/QMAN_AUT	Mode.Status1	AdvancedStatusDis.	Display: Auto/Manual
Links/LOCK	Interlock.Status1	AdvancedStatusDis.	Display: Lock
Links/QRUN	Motor_Status.Status1	AdvancedStatusDis.	Display: Motor
Links/QSTOP	Motor_Status.Status2	AdvancedStatusDis.	Display: Motor

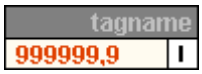
A left-click calls the MOTOR faceplate. A right-click calls the associated INTERLOK faceplate.

The name of the INTERLOK block is stored as a script transfer parameter.

The default block name is "ILOCK". The INTERLOK block and MOTOR must be placed into the same CFC chart.

13.4.13 Block icon: RATIO_P

Properties



For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 97/Height = 32		
General/Unit	Unit.Text	Stat.Text	Display: Unit
Links/OutputValue	ProcessValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Process value
Links/Mode_INT_EXT	External_AdvancedStatus Display.Status	AdvancedStatusDis.	Display: External/Internal
Styles/Format_InputValue	ProcessValue_AdvancedAnalog Display.Format	AdvancedAnalogDis.	Formats the numbers for process value and setpoint
Styles/Format_OutputValue	Format_OutputValue.OutputValue	I/O box	Formats the numbers for the manipulated variable
Styles/Format_xx	Format_xx.OutputValue	I/O box	Further format

13.4.14 Block icon: SWIT_CNT

Properties

For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 97/Height = 45		
General/Unit	Unit.Text	Stat.Text	.V#UNIT
Links/CollectValue	GroupDisplay.CollectValue	Group display	.EventState
Links/OutputValue	ProcessValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Process value
Styles/Format_InputValue	ProcessValue_AdvancedAnalog Display.Format	AdvancedAnalogDis.	Formats the numbers for process value and setpoint
Styles/Format_OutputValue	Format_OutputValue.OutputValue	I/O box	Formats the numbers for the manipulated variable
Styles/Format_xx	Format_xx.OutputValue	I/O box	Further format

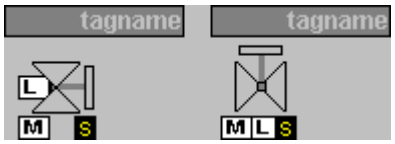
13.4.15 Block icon: VAL_MOT

Properties

Properties and display as Block Icon: VALVE (Page 593)

13.4.16 Block icon: VALVE

Properties



For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 90/Height = 67		
Links/CollectValue	GroupDisplay_WithASD.CollectValue	AdvancedStatusDis.	
Links/QMAN_AUT	Mode.Status1	AdvancedStatusDis.	Display: Auto/Manual
Links/V_LOCK	Interlock.Status1	AdvancedStatusDis.	Display: Lock
Links/QOPENED	Valve_Status.Status1	AdvancedStatusDis.	Display: Valve
Links/QCLOSED	Valve_Status.Status2	AdvancedStatusDis.	Display: Valve
Links/QOPENING	Valve_Status.Status3	AdvancedStatusDis.	Display: Valve
Links/QCLOSING	Valve_Status.Status4	AdvancedStatusDis.	Display: Valve

A left-click calls the VALVE faceplate. A right-click calls the INTERLOK faceplate.

The name of the INTERLOK block is stored as a script transfer parameter.

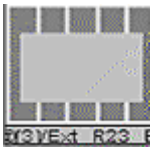
The default block name is "ILOCK". The INTERLOK block and the VALVE block must be placed in the same CFC chart.

13.5 Block Icons: Asset Management

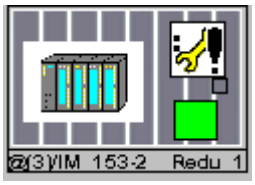
13.5.1 Block Icons: Asset Management

Block Icons Supported by the System

The block icons for asset management supported by the system are contained in the "@@maintenancetypicals.pdl" picture. This file is located under ..\SIEMENS\WinCC\options\pdl\faceplatedesigner_v6" and is copied from the installation directory to the project directory under ..\wincproj\<os-name>\GraCS when a PCS 7 project is generated or when the OS project editor is run.

Icons in the Block Icon	Meaning	Local operation
	**Diagnostic block icon** with the self-diagnosis icon (top right) and the group display with the status of the lower-level hierarchy (bottom right)	

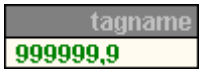
Icons in the Block Icon	Meaning	Local operation
	**"Self-diagnosis (maintenance state)" icon of the component** Depending on the maintenance status, the icons shown in the right-hand column can be displayed within the component. This only applies to non-redundant components. Regardless of which icon is displayed, the icon is used to open the faceplate of the component. The table shown under "Status display for redundant components [Asset] (Page 618)" is applicable for redundant components.	 Maintenance alarm  Maintenance request  Maintenance required  Component good  Local operation  At least one process value is being simulated  Out of service  Component passivated  Unchecked/unknown  Maintenance in progress  Maintenance job requested, alarm priority  Maintenance job requested, demand priority  Maintenance job requested, request priority
	**Group display for same-level components** The small square (group display) at the bottom right indicates the following: An alarm has not yet been acknowledged. The alarm can also have already completed. Alarms remain flashing until they are acknowledged. If the alarm is still pending, the icon stops flashing. If the alarm is no longer pending, the icon disappears.	**Possible icons in the group display**  Maintenance alarm  Maintenance request  Maintenance required Good

Icons in the Block Icon	Meaning	Local operation
	**Group display for lower-level hierarchy** A maintenance alarm, for example, is pending within the nested diagnostic pictures. Only the highest-priority class is displayed. Regardless of which icon is shown in the group display, the left-hand icon is used to open the lower-level hierarchy.	**Possible icons in the group display**  Maintenance alarm Maintenance request Maintenance required Good
	**DeviceIcon** If HW Config contains a bit map, a bit map displaying the device icon is shown in the "DeviceIcon" property.	 

13.6 Block icons: Operator-control blocks

13.6.1 Block icon: OP_A

Properties



For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 97/Height = 32		
General/Unit	Unit.Text	Stat.Text	
Links/OutputValue	ProcessValue_AdvancedAnalog Display.Value	AdvancedAnalogDis.	Display: Process value
Styles/Format_InputValue	ProcessValue_AdvancedAnalog Display.Format	AdvancedAnalogDis.	Formats the numbers for process value and setpoint
Styles/Format_OutputValue	Format_OutputValue .OutputValue	I/O box	Formats the numbers for the manipulated variable
Styles/Format_xx	Format_xx.OutputValue	I/O box	Further format

13.6.2 Block icon: OP_A_LIM

Properties

Properties and display as Block Icon: OP_A (Page 597).

13.6.3 Block icon: OP_A_RJC

Properties

Properties and display as Block Icon: OP_A (Page 597).

13.6.4 Block icon: OP_D

Properties

For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 90/Height = 45		
Links/Status	StatusDisplay.ActualStatus	Status display	.Q0

13.6.5 Block icon: OP_D3

Properties

For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 90/Height = 45		
Links/Output1	StatusDisplay1.Display	Status display	.Q1
Links/Output2	StatusDisplay2.Display	Status display	.Q2
Links/Output3	StatusDisplay3.Display	Status display	.Q3

13.6.6 Block icon: OP_TRIG

Properties

For more information, refer to the section: "General properties of block icons (Page 577)"

Properties	Element and property in user object	Object	Description
Geometry	Width = 90/Height = 40		
Links/Status	StatusDisplay.ActualStatus	Status display	.SIGNAL

Appendix

14

14.1 MODE settings for SM modules

Measuring range coding of the analog input modules

Depending on the measuring-range coding of the analog input modules, the parameter MODE_xx (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 MODE_xx parameter must be calculated according to the following formula and the result written to the MODE input as an INTEGER value:
MODE = 256 * code A + code B

Please note: The table displays codes **A** and **B** in binary format, and the result in hexadecimal format in the **MODE** parameter.

Measurement type	Code (A)	Measuring range	Code(B)	MODE (256*A+B)
Deactivated				16#0000
Voltage	2#0001	± 25 mV	2#1010	16#010A
		± 50 mV	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 10 V	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	2#1100	16#070C
2-wire measuring transducer	2#0011	4 to 20 mA	2#0011	16#0303
		± 20 mA	2#0100	16#0304

Measurement type	Code (A)	Measuring range	Code(B)	MODE (256*A+B)
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
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
Thermocouple, 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 climate 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 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

Measurement type	Code (A)	Measuring range	Code(B)	MODE (256*A+B)
		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
		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

Measurement type	Code (A)	Measuring range	Code(B)	MODE (256*A+B)
Thermocouple, 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 climate 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 standard range	2#0110	16#0906
		Ni 120 standard range	2#1100	16#090C
		Ni 120 climate range	2#1101	16#090D
		Cu10 climate range	2#1110	16#090E
		Cu10 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
		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

Measurement type	Code (A)	Measuring range	Code(B)	MODE (256*A+B)
		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
Thermocouple, 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 climate 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 standard range	2#0110	16#0F06
		Ni 120 standard range	2#1100	16#0F0C
		Ni 120 climate range	2#1101	16#0F0D
		Cu10 climate range	2#1110	16#0F0E
		Cu10 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

Measurement type	Code (A)	Measuring range	Code(B)	MODE (256*A+B)
Thermocouple, linear, reference temperature 0 ºC	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, linear, internal compensation	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

Measurement type	Code (A)	Measuring range	Code(B)	MODE (256*A+B)
Thermocouple, linear, external compensation	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

Effect of the temperature coefficients on the measuring range

- Setting TC = 3850 at GOST Standard Pt10, Pt50, Pt100, Pt500 sets bit 7 in the measuring range byte (0x40)
- Setting TC = 428 at GOST Standard Cu10, Cu50, Cu100 sets bit 8 in the measuring range byte (0x80)

Measuring range coding of the analog output modules

Depending on the measuring-range coding of the analog output modules, the parameter MODE_xx (measuring-range coding) corresponding to the channel must be specified according to the table.

Measurement type	Measuring range	MODE
Voltage	± 5 V	16#0106
	1 to 5 V	16#0107
	0 to 10 V	16#0108
	± 10 V	16#0109
Current	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

With digital input modules and digital output modules, there is no measurement type and no measuring range:
MODE = 16#FFFF (with DI)
MODE = 16#FFFE (with DO)

14.2 OMODE settings for SM modules

OMODE structure

The table below shows the structure and meaning of the outputs OMODE_xx of data type DWORD:

Byte 3:	16#80: Value status "valid value"	
	16#00: Value status "invalid value"	(Channel error)
	16#40: Value status "invalid value"	(Higher-level error)
Byte 2:	16#01: Restart (OB 100) has been carried out	
	16#02: Measuring-range overshoot	(Channel-error diagnostics)
	16#04: Measuring range low limit exceeded	(Channel-error diagnostics)
Byte 1, 0 (low word):	MODE (see above)	

Example:

16#80010203 = value status "valid value", restart has been carried out, current 4 mA to 20 mA

14.3 Technical data, "standard-library blocks"

Overview

The table below contains technical data relating to the blocks. The table columns have the following meanings:

- **Block type name**

  The symbolic identifier in the symbol table of the library for the relevant FB or FC. Must be unique within the project.

- **FB/FC no.**

  Block number

- **Typical execution time**

  Time taken by the CPU to process the corresponding block program under normal circumstances (for example, for a driver, this is the execution time in the cyclic interrupt OB, without message generation in the event of a channel error).

  The table below shows the execution time of blocks in an S7 417-4 CPU. The block execution time on other CPUs depends on the CPU performance.

- **Block length**

  Memory requirements of the program code, once for each block type.

- **Length of instance data**

  Memory requirements of an instance DB.

- **Temporary memory**

  The local-data memory required in a priority class when the block is called. This limit is CPU specific. When it is exceeded, check the CPU configuration and, if necessary, distribute it amongst the OBs to meet the actual requirements.

- **Multiple instance block**

  The specified blocks are used by the block concerned and must exist in the user program. They can be found in the same library.

Block (Type name)	FB/FC no.	Typical runtime CPU 417-4 (µs)	Block length in load /work memory (bytes)	Instance-data length in load/ work memory (bytes)	Temporary memory (bytes)	Multiple-instance block
ADD4_P	FC256	6	194/122	-/-	2	
ADD8_P	FC257	8	298/202	-/-	2	
AL_DELAY	FC 290					
ASSETMON	FB 86	84	6474/5486	1102/546	90	SFB 31, 2 x SFB 35
AVER_P	FB 34	16	508/368	156/54	48	
CH_AI	FC 275	36	6254/5414	-/-	28	
CH_AO	FC 276	30	1782/1510	-/-	34	
CH_CNT	FB 127	33	6466/5928	452/124	70	
CH_CNT1	FB 59	16	1342/1152	214/92	36	
CH_DI	FC 277	14	502/392	-/-	4	
CH_DO	FC 278	11	358/274	-/-	2	
CH_MS	FB 60	22	1680/1366	310/74	20	
CH_U_AI	FC 283	46	8826/7688	-/-	28	
CH_U_AO	FC 284	31	2674/2302	-/-	34	
CH_U_DI	FC 285	20	3064/2666	-/-	6	
CH_U_DO	FC 286	13	1268/1080	-/-	2	
COUNT_P	FB 36	16	484/340	166/54	44	
CPM	FB 140					
CPU_RT	FB 128	67	31434/27370	2800/1784	86	
CTRL_PID	FB 61	178	7922/6502	1322/604	164	2 x FB 46 + SFB 35
CTRL_S	FB 76	189	10386/8672	1496/644	198	2 x FB 46 + SFB 35
DEADT_P	FB 37	20	908/704	252/126	48	
DIF_P	FB 38	21	710/518	208/72	56	
DIG_MON	FB 62	104	2078/1692	802/482	56	SFB 35 + SFB 36
DOSE	FB 63	127	5250/4312	1178/592	76	FB 46 + SFB 35 + SFB 36
ELAP_CNT	FB64	101	1434/1086	610/334	54	SFB 35
FF_A_AI	FB 410	38	6882/5618	1470/516	50	
FF_A_AO	FB 411	56	1174/814	492/154	14	
FF_A_DI	FB 412	32	1976/1438	740/252	20	
FF_A_DO	FB 413	49	926/634	364/90	8	
FMCS_PID	FB 114	143	9712/8112	1870/846	116	FB 46 + SFB 35

Block (Type name)	FB/FC no.	Typical runtime CPU 417-4 (µs)	Block length in load /work memory (bytes)	Instance-data length in load/ work memory (bytes)	Temporary memory (bytes)	Multiple-instance block
FMT_PID	FB 77	142	9422/8002	1856/894	82	FB 46 + SFB 35
GAIN_SHD	FB 141					
INT_P	FB 40	24	1142/892	228/84	60	
INTERLOK	FB 75	27	1492/1154	300/78	46	
LIMITS_P	FB 41	6	308/216	124/58	6	
MEANTM_P	FB 42	53	1586/1306	264/154	20	
MEAS_MON	FB 65	108	1910/1486	690/376	56	SFB 35
MESSAGE	FB 43	98	932/684	506/278	44	SFB 35
MODB_341	FB 80	594	4388/3666	1012/490	54	2 x SFB 35
MOT_REV	FB 67	125	4466/3770	828/382	70	SFB 35
MOT_SPED	FB 68	122	4304/3618	824/382	66	SFB 35
MOTOR	FB 66	114	2570/2108	714/364	54	SFB 35
MPC	FB 142					
MS_MUX	FC 288	9	382/288	-	6	SFB 54 + SFB 35
MSG_NACK	FB 78	99	998/732	520/274	44	SFB 31
MSG_TS	FB 131					
MUL4_P	FC262	5	214/122	-/-	2	
MUL8_P	FC263	9	334/202	-/-	2	
NOISE_GN	FB 143					
OB1_TIME	FB 69	57	2140/1822	216/70	84	
OP_A	FB 45	4	232/156	114/56	2	
OP_A_LIM	FB 46	8	486/358	160/68	6	
OP_A_RJC	FB 47	8	518/388	160/68	6	
OP_D	FB 48	6	376/286	112/44	2	
OP_D3	FB 49	12	1354/1136	150/46	8	
OP_TRIG	FB 50	5	244/166	104/44	2	
PA_AI	FB 101	21	3652/3162	290/96	6	
PA_AO	FB 103	45	4054/3524	406/140	10	
PA_DI	FB 104	19	3438/2992	224/62	6	
PA_DO	FB 105	28	2960/2572	314/82	12	
PA_TOT	FB 102	21	3774/3264	306/100	12	
POLYG_P	FC271	6	1446/1176	-/-	24	
PT1_P	FB 51	7	446/330	136/66	2	
R_TO_DW	FC282	6	344/262	-/-	10	
RAMP_P	FB 52	19	708/524	188/64	54	
RATIO_P	FB 70	19	742/526	316/134	12	FB 46 + SFB 35
RCV_341	FB 121	9	1940/1070	1694/874	12	SFB 35
READ355P	FB 72	18	984/784	258/94	94	

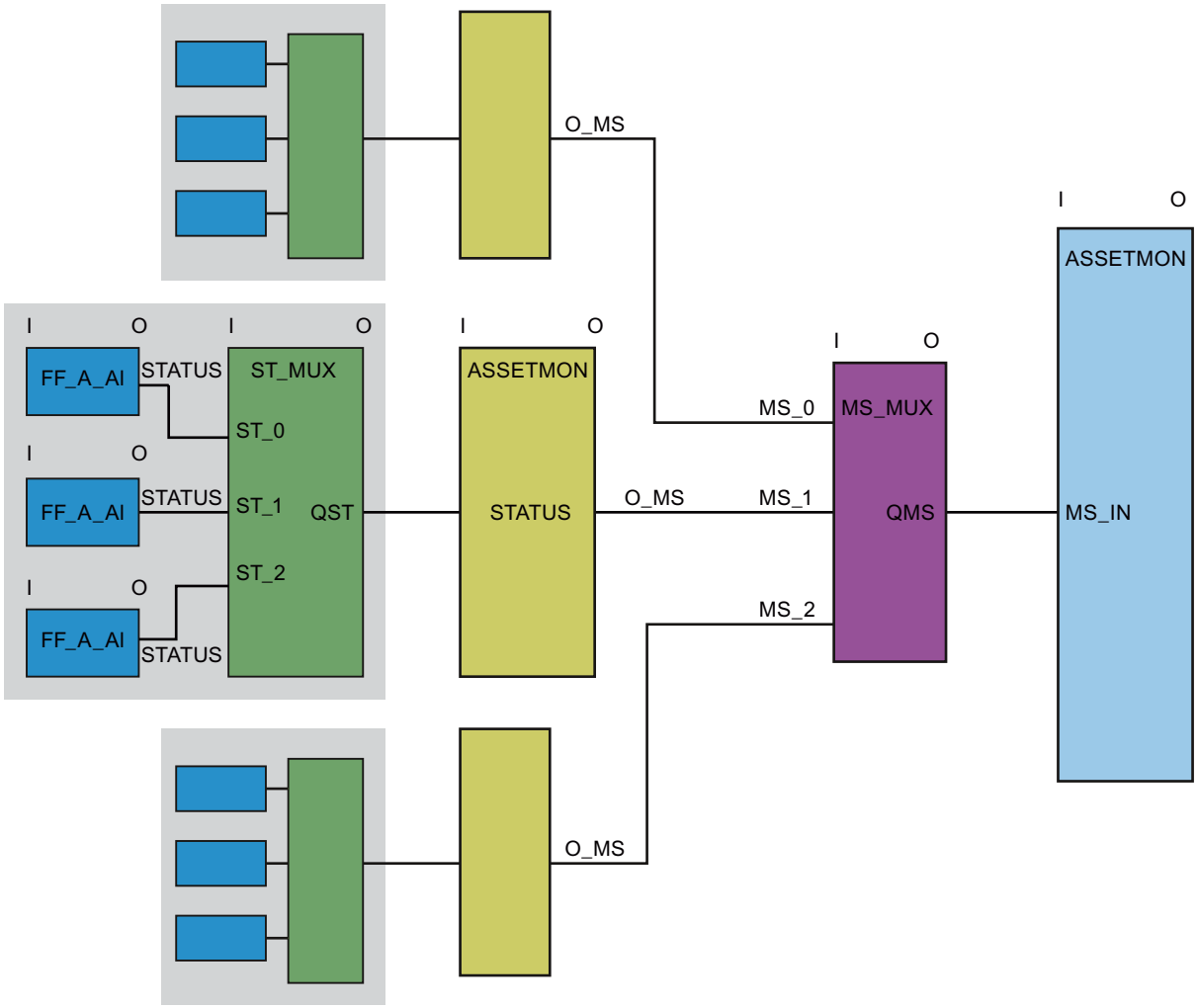
Block (Type name)	FB/FC no.	Typical runtime CPU 417-4 (µs)	Block length in load /work memory (bytes)	Instance-data length in load/ work memory (bytes)	Temporary memory (bytes)	Multiple-instance block
REC_BO	FB 208	69	3246/2356	992/128	2	SFB 13
REC_R	FB 210	69	1838/1332	956/476	2	SFB 13
SEND_BO	FB 207	163	2298/1668	718/110	2	SFB 12
SEND_R	FB 209	195	4486/3886	908/478	2	SFB 12
SIG_SMTH	FB 144					
SND_341	FB 120	86	1608/1226	616/288	12	SFB 35
SPLITR_P	FC272	17	832/644	-/-	10	
ST_MUX	FC 287	114	1612/1408	-	20	
STATEREP	FB 87	15	1900/1574	182/58	4	
SWIT_CNT	FB 71	102	1332/988	606/316	92	SFB 35
VAL_MOT	FB 74	127	4498/3782	828/382	70	SFB 35
VALVE	FB 73	120	3280/2740	734/366	54	SFB 35

14.4 Customizing MODE_LW for FF devices

Blocks	I/O (parameters) (cyclic data) permissible combination and sequence	Input (I)/Output (O) (PLS View)	MODE 16#xxyy, O=xx I=yy
Analog Input (FF_A_AI)	OUT	I	16#0001
Discrete Input (FF_A_DI)	OUT_D	I	16#0002
Analog Output (FF_A_AO)	SP	O	16#0100
Analog Output (FF_A_AO)	SP READBACK POS_D	O I I	16#0103
Analog Output (FF_A_AO)	SP CHECK_BACK	O I	16#0104
Analog Output (FF_A_AO)	SP READBACK POS_D CHECK_BACK	O I I I	16#0105
Analog Output (FF_A_AO)	RCAS_IN RCAS_OUT	O I	16#0206
Analog Output (FF_A_AO)	RCAS_IN RCAS_OUT CHECK_BACK	O I I	16#0207
Analog Output (FF_A_AO)	SP RCAS_IN READBACK RCAS_OUT POS_D CHECK_BACK	O O I I I I	16#0308
Discrete Output (FF_A_DO)	SP_D	I	16#0400
Discrete Output (FF_A_DO)	SP_D READBACK_D	O I	16#0409
Discrete Output (FF_A_DO)	SP_D CHECK_BACK_D	O I	16#040A
Discrete Output (FF_A_DO)	SP_D READBACK_D CHECK_BACK_D	O I I	16#040B
Discrete Output (FF_A_DO)	RCAS_IN_D RCAS_OUT_D	O I	16#050C
Discrete Output (FF_A_DO)	RCAS_IN_D RCAS_OUT_D CHECK_BACK_D	O I I	16#0D05
Discrete Output (FF_A_DO)	SP_D RCAS_IN_D READBACK_D RCAS_OUT _D CHECK_BACK_D	O O I I I	16#0E06

14.5 Integrating FF devices in Asset

FF devices can be integrated in Asset as follows:



Key

	Signal-processing driver blocks of the FF field device [FF_A_xx], xx = AI, DI, AO, DO
	Summarized for one field device [ST_MUX]
	Representative for one field device [ASSETMON]
	Summarized for one functional unit [MS_MUX]
	Representative for one functional unit [ASSETMON]

14.6 Archiving process values

System attribute "s7_archive"

The system attribute "s7_archive" informs WinCC which process values should be archived during runtime. A distinction is made between long-term archiving (s7_archive=longterm), short-term archiving (s7_archive=shortterm) and no archiving (s7_archive=false).

When controller monitoring by the ConPerMon block is activated, the archive data is evaluated by the ConPerMon faceplate.

With the following controllers, short-term archiving is the default parameter setting:

CTRL_PID	CTRL_S	FMCS_PID	FMT_PID
LMN	LMNR_IN	LMN	LMN
PV_IN	PV_IN	PV	PV
QLMN_HLM	QLMNR_HS	QLMN_HLM	QLMN_HLM
QLMN_LLM	QLMNR_LS	QLMN_LLM	QLMN_LLM
QMAN_AUT	QMAN_AUT	QMAN_AUT	QMAN_AUT
SP	SP	SP	SP

With the FMCS_PID and FMT_PID blocks, the default archiving setting is for the continuous controller. When they are used as a step controller, the parameters LMN_A, QLMNR_HS and QLMNR_LS must be set for short-term archiving.

To save resources in WinCC, unnecessary archives can be disabled with "s7_archive=false".

14.7 Text library ASSETMON

The following table lists the text-library message texts and their numbers for the block ASSETMON (FB 86):

Text No.	Message Text
1	Further status available
2	No further status available

14.8 Quality code and status displays

14.8.1 The Quality Code Display

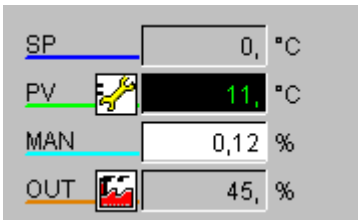
Properties of the quality code display

The quality code display is an expanded status display with seven alternatives.

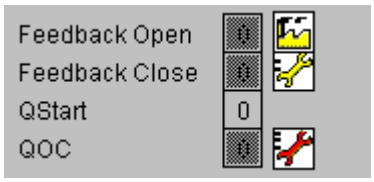
Priority	Quality code	Plain text	Symbol
1	0x44 0x48 0x60	Simulation	
2	0x00 0x14 0x18	Bad, device related	
3	0x28 *) 0x2B	Bad, process related	
4	0x68	Uncertain, device related	
5	0x54 0x55 0x56 0x78	Uncertain, process related	
6	0xA4	Maintenance required	
7	0x80	Good	
	*) Code 0x08 is generated in the OS if communication with the AS is down.		

The plain text appears as short info when you position the mouse pointer over the quality code display.

The quality code is displayed before the display of the **analog values**.



With **binary values**, the quality code display appears in the maintenance view after the binary value displays in the faceplates for valves and motors.



Additional information

The exhaustive Excel table "QC_MS" with the quality code of the PA devices and the application in Asset Management can be found in the LIB readme file.

14.8.2 Maintenance Status of MS

Layout of the maintenance status

The layout of the maintenance status MS (DWORD data type) is as follows:

Bit 0 to 7	Display of the MS
Bit 8 to 15	Display of the MS of the redundant partner
Bit 16	1 = Redundant partner available
Bit 17	0 = primary partner is master, 1 = redundant partner is master
Bit 18	PDM-MS worse than device status
Bit 19 to 21	Reserve
Bit 22	PDM has detected status change
Bit 23	Block takes part in the cyclical updating of PDM
Bit 24 to 27	OS operation
Bit 28 to 31	PDM-MS

The MS is copied 1:1 to the output O_MS.

Displayable Statuses

The maintenance status (MS) can display the following statuses, that are entered in Bit 0 to 7 or Bit 8 to 15 (for redundant partner):

Bit number								State	Symbol	Priority
7	6	5	4	3	2	1	0			0 = highest priority
0	0	0	0	0	0	0	0	Good		9
0	0	0	0	0	0	0	1	Passivated		7
0	0	0	0	0	0	1	0	Out of service		6
0	0	0	0	0	0	1	1	At least one PV simulated		5
0	0	0	0	0	1	0	0	Local operation/function test		4
0	0	0	0	0	1	0	1	Maintenance required		3
0	0	0	0	0	1	1	0	Maintenance request		2
0	0	0	0	0	1	1	1	Maintenance alarm		1
0	0	0	0	1	0	0	0	Untested/unknown		0
0	0	0	0	1	0	0	1	Configuration changed		8

Note

If the maintenance status is "untested/unknown", all other dynamic displays in the faceplates for Asset Management are not relevant to this instance.

Maintenance Status of the Messages

The maintenance status is updated in the driver blocks by a message. The driver blocks generate a message with the following message classes:

Message class	EventState bit	Symbol
AS control system message (S) = fault	25	S
AS control system message (F) = error	24	F
Preventive maintenance (M) = Maintenance	23	M
Status AS (SA)	18	

Redundancy

In case of redundancy, several combinations of the displays are possible. See:

Status display for redundant components [Asset] (Page 618)

14.8.3 Status Display for Redundant Components [Asset]

Status display icons

Redundant component A and redundant component B form the status display (maintenance state) for redundant components. In the following table, the icons of the status display are listed that result from this rule (the bit numbers not listed in the table are always = 0).

Note

The status MS = 9, configuration changed, is for the redundant components and is therefore not listed here.

PV = process value

Bit number								State		
11	10	9	8	3	2	1	0	Redundant component A	Redundant component B	Status display icon
0	0	0	0	0	0	0	0	Good	Good	Good 
0	0	0	1	0	0	0	0	Good	Passivated	Good 
0	0	1	0	0	0	0	0	Good	Out of service	Maintenance request 
0	0	1	1	0	0	0	0	Good	At least 1 PV simulated	Good 
0	1	0	0	0	0	0	0	Good	Local operation/ function test	Good 
0	1	0	1	0	0	0	0	Good	Maintenance required	Maintenance required 
0	1	1	0	0	0	0	0	Good	Maintenance request	Maintenance request 
0	1	1	1	0	0	0	0	Good	Maintenance alarm	Maintenance request 
1	0	0	0	0	0	0	0	Good	Untested/ unknown	Good 
0	0	0	0	0	0	0	1	Passivated	Good	Good 
0	0	0	1	0	0	0	1	Passivated	Passivated	Passivated 
0	0	1	0	0	0	0	1	Passivated	Out of service	Out of service 
0	0	1	1	0	0	0	1	Passivated	At least 1 PV simulated	At least 1 PV simulated 
0	1	0	0	0	0	0	1	Passivated	Local operation/ function test	Local operation/ function test 

Bit number								State			
0	1	0	1	0	0	0	1	Passivated	Maintenance required	Maintenance required	
0	1	1	0	0	0	0	1	Passivated	Maintenance request	Maintenance request	
0	1	1	1	0	0	0	1	Passivated	Maintenance alarm	Maintenance alarm	
1	0	0	0	0	0	0	1	Passivated	Untested/ unknown	Passivated	
0	0	0	0	0	0	1	0	Out of service	Good	Maintenance request	
0	0	0	1	0	0	1	0	Out of service	Passivated	Out of service	
0	0	1	0	0	0	1	0	Out of service	Out of service	Out of service	
0	0	1	1	0	0	1	0	Out of service	At least 1 PV simulated	At least 1 PV simulated	
0	1	0	0	0	0	1	0	Out of service	Local operation/ function test	Local operation/ function test	
0	1	0	1	0	0	1	0	Out of service	Maintenance required	Maintenance required	
0	1	1	0	0	0	1	0	Out of service	Maintenance request	Maintenance request	
0	1	1	1	0	0	1	0	Out of service	Maintenance alarm	Maintenance alarm	
1	0	0	0	0	0	1	0	Out of service	Untested/ unknown	Out of service	
0	0	0	0	0	0	1	1	At least 1 PV simulated	Good	Good	
0	0	0	1	0	0	1	1	At least 1 PV simulated	Passivated	At least 1 PV simulated	
0	0	1	0	0	0	1	1	At least 1 PV simulated	Out of service	At least 1 PV simulated	
0	0	1	1	0	0	1	1	At least 1 PV simulated	At least 1 PV simulated	At least 1 PV simulated	
0	1	0	0	0	0	1	1	At least 1 PV simulated	Local operation/ function test	Local operation/ function test	
0	1	0	1	0	0	1	1	At least 1 PV simulated	Maintenance required	Maintenance required	
0	1	1	0	0	0	1	1	At least 1 PV simulated	Maintenance request	Maintenance request	
0	1	1	1	0	0	1	1	At least 1 PV simulated	Maintenance alarm	Maintenance alarm	
1	0	0	0	0	0	1	1	At least 1 PV simulated	Untested/ unknown	At least 1 PV simulated	

Bit number								State			
0	0	0	0	0	1	0	0	Local operation/ function test	Good	Good	
0	0	0	1	0	1	0	0	Local operation/ function test	Passivated	Local operation/ function test	
0	0	1	0	0	1	0	0	Local operation/ function test	Out of service	Local operation/ function test	
0	0	1	1	0	1	0	0	Local operation/ function test	At least 1 PV simulated	Local operation/ function test	
0	1	0	0	0	1	0	0	Local operation/ function test	Local operation/ function test	Local operation/ function test	
0	1	0	1	0	1	0	0	Local operation/ function test	Maintenance required	Maintenance required	
0	1	1	0	0	1	0	0	Local operation/ function test	Maintenance request	Maintenance request	
0	1	1	1	0	1	0	0	Local operation/ function test	Maintenance alarm	Maintenance alarm	
1	0	0	0	0	1	0	0	Local operation/ function test	Untested/ unknown	Local operation/ function test	
0	0	0	0	0	1	0	1	Maintenance required	Good	Maintenance required	
0	0	0	1	0	1	0	1	Maintenance required	Passivated	Maintenance required	
0	0	1	0	0	1	0	1	Maintenance required	Out of service	Maintenance required	
0	0	1	1	0	1	0	1	Maintenance required	At least 1 PV simulated	Maintenance required	
0	1	0	0	0	1	0	1	Maintenance required	Local operation/ function test	Maintenance required	
0	1	0	1	0	1	0	1	Maintenance required	Maintenance required	Maintenance required	
0	1	1	0	0	1	0	1	Maintenance required	Maintenance request	Maintenance request	
0	1	1	1	0	1	0	1	Maintenance required	Maintenance alarm	Maintenance alarm	
1	0	0	0	0	1	0	1	Maintenance required	Untested/ unknown	Maintenance required	
0	0	0	0	0	1	1	0	Maintenance request	Good	Maintenance request	
0	0	0	1	0	1	1	0	Maintenance request	Passivated	Maintenance request	
0	0	1	0	0	1	1	0	Maintenance request	Out of service	Maintenance request	

Bit number								State			
0	0	1	1	0	1	1	0	Maintenance request	At least 1 PV simulated	Maintenance request	
0	1	0	0	0	1	1	0	Maintenance request	Local operation/ function test	Maintenance request	
0	1	0	1	0	1	1	0	Maintenance request	Maintenance required	Maintenance request	
0	1	1	0	0	1	1	0	Maintenance request	Maintenance request	Maintenance request	
0	1	1	1	0	1	1	0	Maintenance request	Maintenance request	Maintenance alarm	
1	0	0	0	0	1	1	0	Maintenance request	Untested/ unknown	Maintenance request	
0	0	0	0	0	1	1	1	Maintenance alarm	Good	Maintenance request	
0	0	0	1	0	1	1	1	Maintenance alarm	Passivated	Maintenance alarm	
0	0	1	0	0	1	1	1	Maintenance alarm	Out of service	Maintenance alarm	
0	0	1	1	0	1	1	1	Maintenance alarm	At least 1 PV simulated	Maintenance alarm	
0	1	0	0	0	1	1	1	Maintenance alarm	Local operation/ function test	Maintenance alarm	
0	1	0	1	0	1	1	1	Maintenance alarm	Maintenance required	Maintenance alarm	
0	1	1	0	0	1	1	1	Maintenance alarm	Maintenance request	Maintenance alarm	
0	1	1	1	0	1	1	1	Maintenance alarm	Maintenance alarm	Maintenance alarm	
1	0	0	0	0	1	1	1	Maintenance alarm	Untested/ unknown	Maintenance alarm	
1	0	0	0	1	0	0	0	Untested/ unknown	Untested/ unknown	Untested/ unknown	
0	0	0	0	1	0	0	0	Untested/ unknown	Good	Good	
0	0	0	1	1	0	0	0	Untested/ unknown	Passivated	Passivated	
0	0	1	0	1	0	0	0	Untested/ unknown	Out of service	Out of service	
0	0	1	1	1	0	0	0	Untested/ unknown	At least 1 PV simulated	At least 1 PV simulated	
0	1	0	0	1	0	0	0	Untested/ unknown	Local operation/ function test	Local operation/ function test	
0	1	0	1	1	0	0	0	Untested/ unknown	Maintenance required	Maintenance required	

Bit number								State			
0	1	1	0	1	0	0	0	Untested/ unknown	Maintenance request	Maintenance request	
0	1	1	1	1	0	0	0	Untested/ unknown	Maintenance alarm	Maintenance alarm	

PCS 7 advanced process control templates 15

15.1 Process tag types (insertible templates)

15.1.1 Explanations of the process tag types

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 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 to the "processtagtypes" 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.

15.1.2 PID controller with safety logic and control loop monitoring

Description of the template "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 driver for the process value PV_IN, the position feedback LMNR_IN (if available), and an analog output driver for the manipulated variable LMN.
- A simple process simulation of the first order, controlled by the manipulated variable LMN which provides simulation values to analog input PV_IN. 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 quality code of PV_IN.
- An additional function block for control loop monitoring that should be installed in all PID control loop.

Notes on "measured value failure"

If there is no valid measured value for PV_IN, the controller must not continue operation in auto 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. 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 safety value, for example Valve closed or Heater off.
2. Holding the last valid manipulated value LMN 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 operator can take over responsibility for process control.

A combination of reactions 2 and 3 is implemented in the template. The controller changes to manual tracking mode with the last valid manipulated value as start value. The change is made at the AUT_L input of the PID block and the LIOP_MAN_SEL parameter is temporarily set to 1. The "measured value failure" function is implemented as a CFC chart in the chart called PV_BadSafetyLogic.

Caution: It is not advisable to freeze the last valid manipulated variable 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 position controller 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 LMN_TRK, and an OR element is set before input LMN_SEL.
- The tracking mode is activated when the measured value is lost. This variant is not implemented in the template.

You will find information relating to control-loop monitoring in the online help of the CPM block.

15.1.3 Step controller with direct access to the actuator and without position feedback

Description of the template "Step controller without position feedback"

The output of the CTRL_S block is directly connected to the process via two digital output drivers. This is the simplest form of a step control. The operator can control the actuator (jog mode open/close) in the faceplate of the controller. This jog mode is preferable when operating simple actuators without position feedback since it allows bumpless changeover to auto mode. This changeover 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 CTRL_S block without position feedback.

15.1.4 Split-range control

Description of the template "Split-Range control"

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 error signal (or more precisely of the manipulated variable). In other words, it can work with a bidirectional LMN output (for example: -100% < LMN < 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 process 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 LMN_LLM = -50% on the controller.

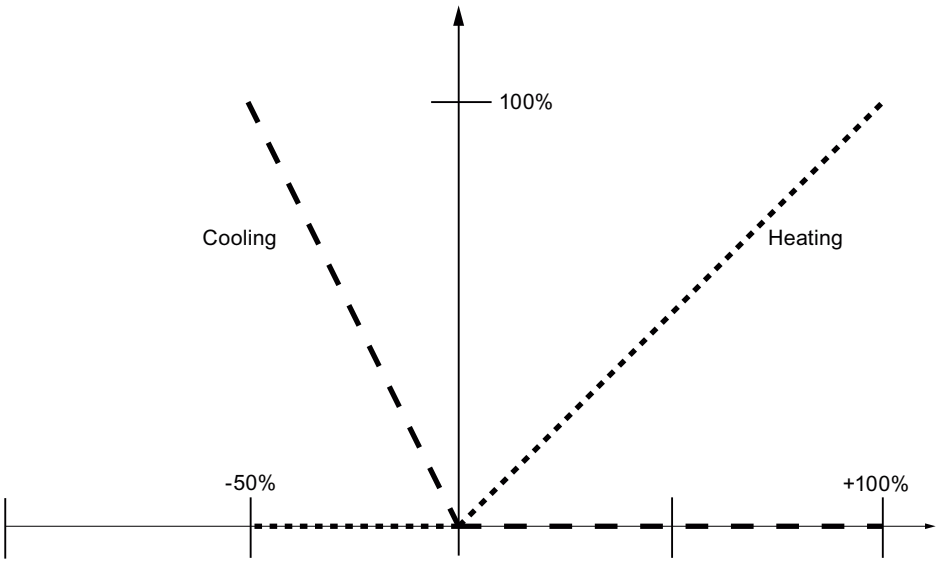


Figure 15-1 Split-range function of the example

The actual split-range function is supplied as separate SPLITR_P 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.

15.1.5 Ratio control

Description of the template "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_P block. The flow setpoint of an additive is obtained from one of these values:

1. From the current PV_IN 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 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.

The ratio control can be expanded by adding further components, in other words, setpoints 3 to n can be derived from SP 1 (or PV_IN 1) with the help of additional Ratio blocks.

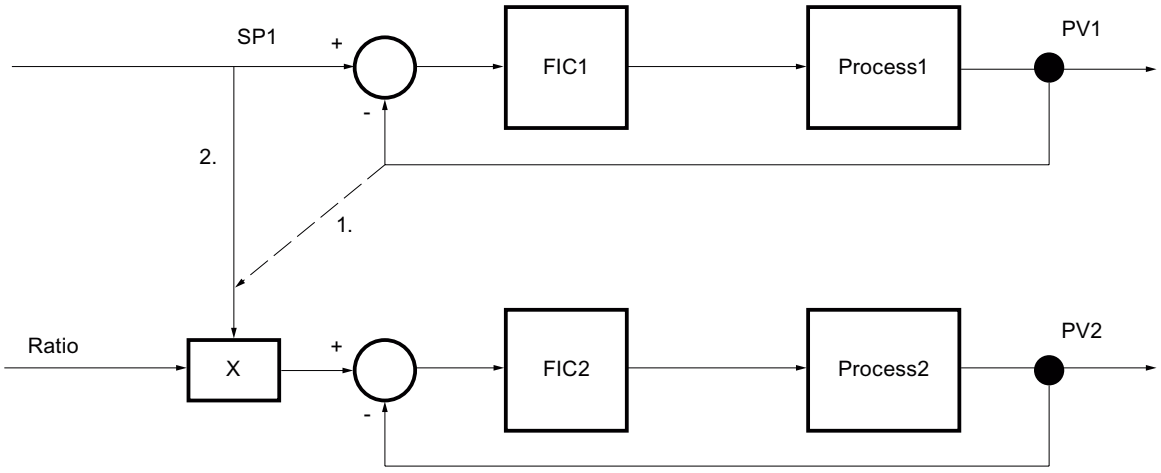


Figure 15-2 Ratio control, process value-oriented (1.) and setpoint-oriented (2.)

In both cases 1 and 2, the current value of the ratio is neither measured nor controlled directly in the closed loop (no feedback-control) but rather uses feedforward-control.

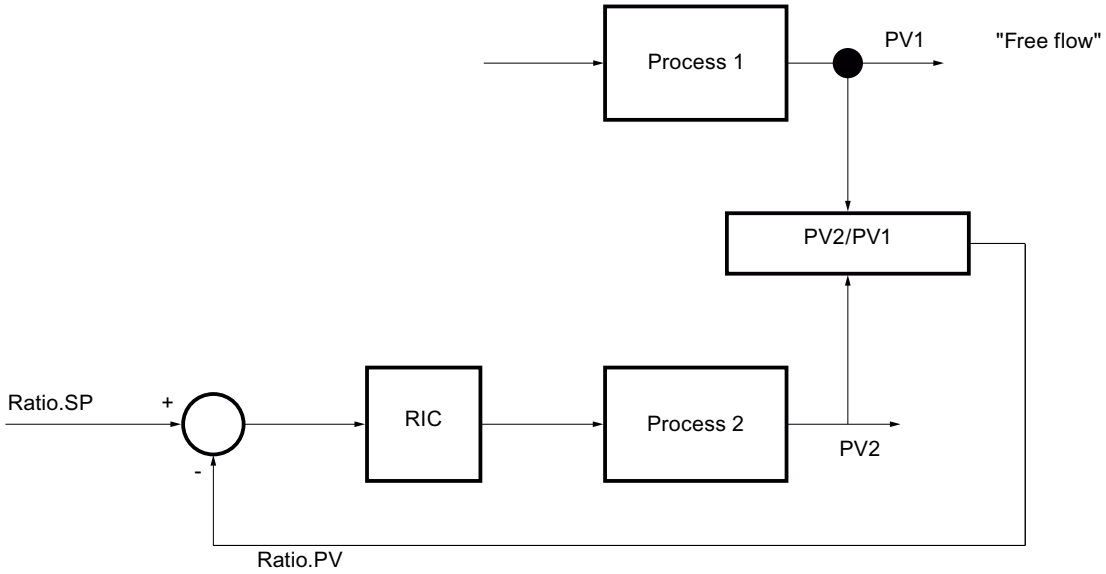


Figure 15-3 Feedback control of the current ratio

If, however, only one of the two flows can be manipulated, a real feedback of the ratio is sometimes used. The current ratio between the dynamic flow PV_IN 1 and the manipulated flow PV_IN 2 is defined by means of division. The controller that adjusts the second flow rate receives the desired ratio as its setpoint. This third alternative is not included in the template, however, it can be easily implemented with the help of a standard PID controller. Steps have to be taken, however, to avoid division by zero when calculating the ratio.

Note: Alternative 1 can also be used to solve this class of application since it avoids division right at the outset.

Examples of applications

- Ratio control of the flow of two reactants fed into to a reactor
- Ratio control of two liquid flows to be mixed, in other words, control of the composition of a mixture
- Control of the ratio between reactor discharge and the reactor recycling stream
- Control of the ratio of burner gas and fresh air for optimized burning performance
- Control of the ratio of liquid and steam feed to an absorber plant

15.1.6 Cascade control

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 connected to the external setpoint of the secondary controller so that both control circuits 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 operates in any mode (for example, manual mode or automatic with local setpoint) instead of "cascade" mode, 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 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 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 higher priority than the manual mode on the primary controller.

15.2 Example projects

15.2.1 Process simulation including noise generator

Description of the template "Process simulation incl. 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 (APC_ExaSP). 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). The block provides 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 usually 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. By contrast, the gain of an overpressure inlet valve is usually positive. PV0 > 0 is the pressure when the valve is fully 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, with MotorHL = 100%, MotorLL = 0%, TI = MotorTime. The integrator input is derived from the binary inputs according to the following formula:

$$\overset{\text{I}}{Integ.Input} = \begin{Bmatrix} 100 & if\ Up = True \\ -100 & if\ Down = True \\ 0 & otherwise \end{Bmatrix}$$

Noise generator function block

The noise generator is supplied as a compiled block NoiseGen and with a minimum online help. The task of the block is to generate a pseudo-stochastic noise signal with specified standard deviation and offset. The algorithm is derived from http://www.code4gold.net/tut/zufall.pdf.

I/O (parameters)	Meaning	Data type	Default	Type
Enable	Noise generation enabled	BOOL	1	I
Offset	Constant offset	REAL	0	I
Noise	Output noise sample	REAL	0	O
Restart	Restart (Initialization)	BOOL	1	I
StdDev	Standard deviation of the noise signal (Variance = StdDev^2)	REAL	1	I

All I/Os are visible and interconnectable in the CFC, and are not intended for operating and monitoring functions in a faceplate.

15.2.2 Cascade control of a temperature by using the heat flow

Description of the template "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 630) 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 (APC_ExaSP) for cascade control

ProcSimC	Gain	TimeLag1	TimeLag2	PV0	NoiseVariance
Flow control loop	8	1	1	0	0.05
Temperature control loop	0.3	8	1	20	0.01

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 deviation but with strong actuator intervention.

PID	Gain	TI	TD
TIC-501	10	8.44	2.5
FIC501	0.1	1.4	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
TIC-501	10.5	6.8	0
FIC501	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 deviation 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 INT_HNEG and INT_HPOS. 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 LMN_OFF that approximates the typical LMN 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

Further examples of applications

- 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.

Use of the secondary controller

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.

15.2.3 Control loop monitoring with simulation of colored noise

Description of the template "Control loop monitoring with simulation of colored noise"

The interconnection of the CPM block with a PID controller can be found in the process tag type PIDCTRL_ConPerMon (see PID controller with safety logic and control loop monitoring (Page 624)). The example project (APC_ExaSP) supports you and helps to familiarize you with the concept and the potential of 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 CPM 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 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.
- Ask the controller for a setpoint step change:
  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 CPM 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 will find detailed information on the CPM block and notes on interpreting its displays in the online help on the block.

15.2.4 Dynamic feedforward to compensate for a measurable variable

Description of the template "Dynamic feedforward to compensate a measurable disturbance variable"

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 unmeasurable disturbances)".

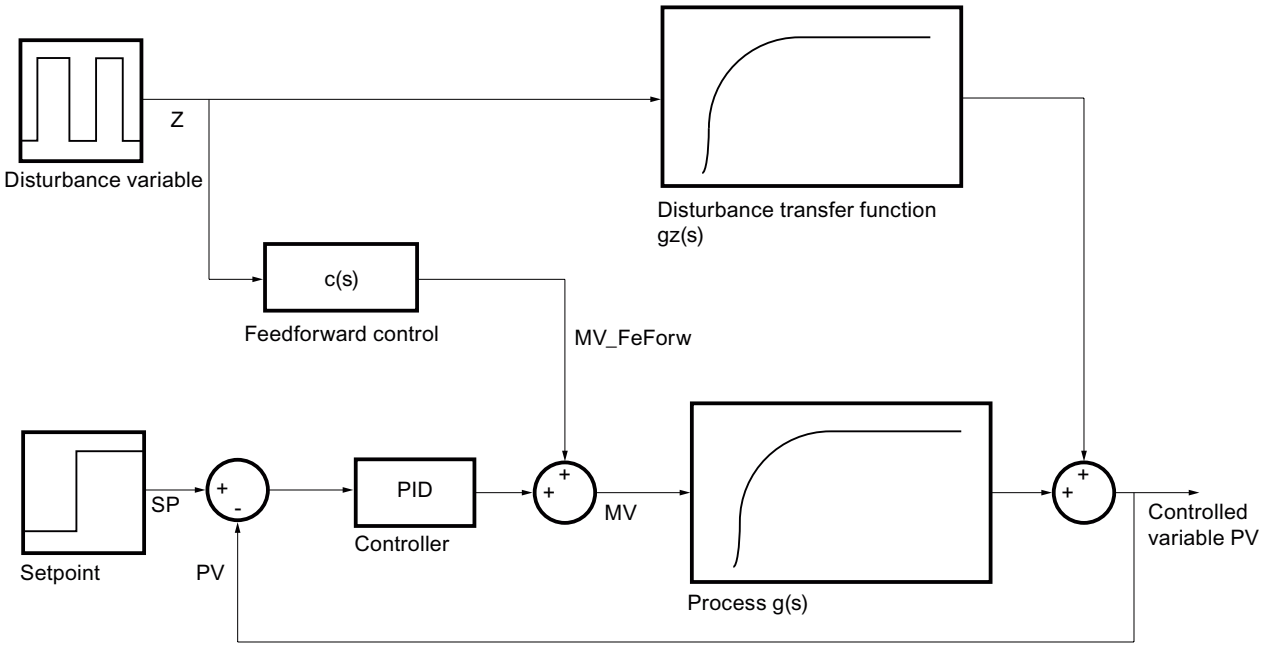


Figure 15-4 Feedforward control

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):

$$g_z(s)z + c(s)g(s)z = \left(g_z(s) + c(s)g(s)\right)z = 0$$

To meet this equation, the compensation block must approximate the equation

$$c(s) = -\frac{g_z(s)}{g(s)}$$

as well as possible. This means that 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 $\theta < \theta_z$ applies, the resulting compensation element must represent the transfer function

$$c(s) = -\frac{k_{SZ}}{k_S} \quad \frac{1 + t_1 s}{1 + t_{1Z} s} \quad e^{-s\,(\theta_Z - \theta)}$$

.

In general, the following dynamic transfer function is required for additive feedforward control:

$$FFwd(s) = -k_c \frac{t_{cd} s + 1}{t_{cl} + 1} \cdot e^{-\theta_c s} \cdot z(s)$$

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 as the series connection of a DT1 (DIF_P) and a PT1 (PT1_P) block.

The input parameters k_c , t_{cd} , t_{cl} must be set by the user. For a static feedforward control, both time constants are set to zero.

The following parameter sets are used in the template:

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 ").

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.

15.2.5 Operating point-oriented adaptation of parameters (gain scheduling) for non-linear processes

Description of the template "Operating point-oriented adaptation of parameters for non-linear processes"

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.

Gain-Scheduling

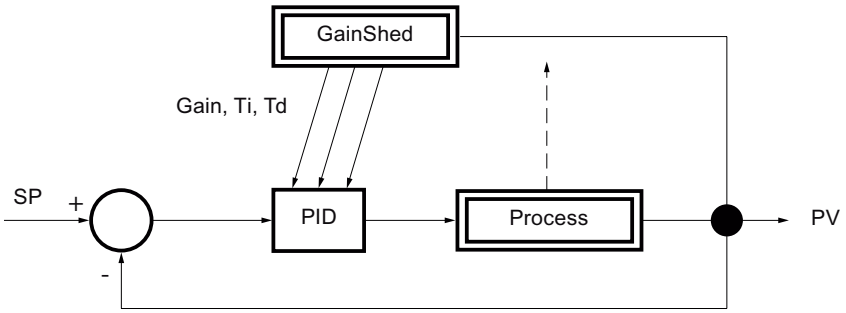


Figure 15-5 Gain scheduling for non-linear processes

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. With non-linearities that are discontinuous or cannot be differentiated or with nonmonotonic non-linearities, great caution is needed.

Using the parameters

In the example project (APC_ExaSP), 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").

15.2.6 Override control

Description of the template "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".

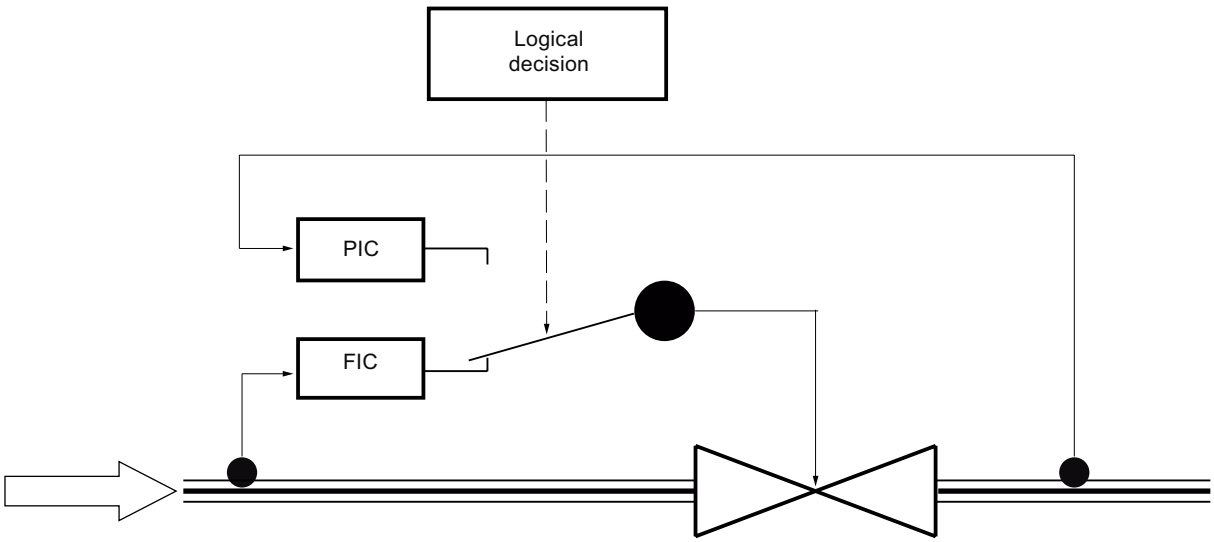


Figure 15-6 Override control with main controller FIC and limiting controller PIC

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 or 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).

Using the parameters

The following parameters are used in the simulation example:

Primary process (flow control): $g(s) = \frac{3}{(2s+1)^2}$, in other words, 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): $g_p(s) = \frac{-0.8}{(7s+1)(1s+1)}$, in other words, 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.

Further examples of applications

- **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.

15.2.7 PID controller with Smith predictor for dead times

Description of the template "PID controller with Smith predictor for dead times"

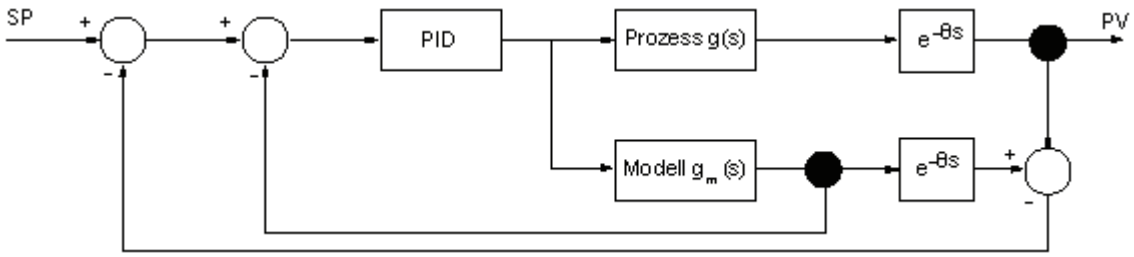


Figure 15-7 Smith predictor in IMC representation

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 (Internal Model Control) of model-based control. To achieve this, the transfer function $g_s(s) = g(s)\ e^{-s\theta}$ of the controlled system is split up into a part without dead time g(s) and a purely dead time part $e^{-s\theta}$ 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 dead time 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, first the dead time must be constant and secondly it must be known.

Note: To control processes with large dead times, a model predictive controller (see Model-based predictive control (Page 646)) is also suitable in a monovariable 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.

In the template, 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").

15.2.8 Filtering of noisy measured values in a control loop

Description of the template "Filtering of noisy measured values in a control loop"

The example illustrates the use of the new SIG_SMTH 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 = 7s, sample time = 0.1s

Butterworth filter: TimeConstant = 3s. At 0.3s, hardly and smoothing effect can be recognized, at 15s 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 will find detailed information on the SIG_SMTH block in the online help on the block.

15.2.9 Model-based predictive control

Description of the template "Model predictive control"

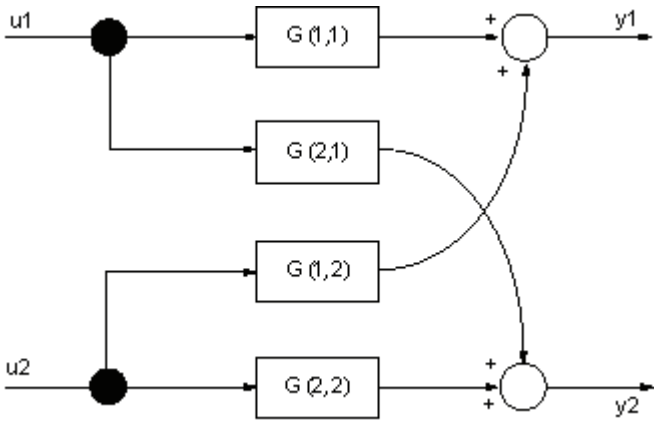


Figure 15-8 MIMO 2x2 process with a p-canonical structure

The template shows the application of the MPC block for simulating a 2x2 multivariable 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 multivariable control systems helps familiarize newcomers with the concept and application of model-based multivariable controllers. The template also shows how users can expand the MPC block by adding extra functions: External alarm reporting using the MEAS_MON block, and control quality monitoring using the CPM block. The safety logic for measured-value failure is not included in the example project (APC_ExaSP), but in the process-tag type.

You will find detailed information on the MPC block in the online help on the block.

Note on using the CPM block in a system with mutivariables

The mathematical concept of the CPM block is designed for single variable applications. If any increase of variance is detected in the channel of a multivariable control, the CPM 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 CPM block for each control channel of a multivariable system to monitor whether the control quality 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 CPM block:

- If one or several of the other channels of the multivariable 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=true, 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 multivariable system exhibit strong variations as indicated by output bit CPI_WrnAct = true, 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.

Note

The templates in the PCS 7 Library (advanced process control templates) include an example of how to use the MPC block with CPM block.

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 or temperature etc.
- Vaporizers, for example, drum steam generators
- Mills, for example, cement mills

See also

PID controller with safety logic and control loop monitoring (Page 624)

15.2.10 Sample project APC_ExaSP

15.2.10.1 Introduction to PCS 7 advanced process control templates

Introduction

In contrast to the templates (insertible process tag types), the simulation templates (sample project "APC_ExaSP") are 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 templates provide realistic process simulation. Working with these templates 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 example projects (APC_ExaSP) contain a third-order simulation model with gain, equivalence value, and measurement noise but no analog driver blocks. The process model is supplied as the "ProcSimC" CFC chart and incorporated in the templates using the chart-in-chart technique.

The following templates are included in the PCS 7 sample project "APC_ExaSP":

- Cascade control of a temperature based on flow of the heating medium (CaskadeSim).
- Control loop monitoring of a process with pink noise interference (ConPerMonSim).
- Dynamic feedforward control to compensate a measurable disturbance variable (DisturCompSim).
- Operating point oriented adaptation of parameters (gain scheduling) for non-linear processes (GainSchedSim).
- Override control in which the primary and limiting controller access a common actuator (OverrideSim)
- PID with Smith predictor for controlling processes with long dead times (SmithPredictorSim).
- Filtering of noisy measured values in a control loop (SigSmoothSim).
- Model predictive control (ModPreConSim).

A template for Fuzzy Control using the "FuzzyControl++" PCS 7 optional package can be downloaded from the *Siemens I&S* Web pages and is, therefore, not included in the demo project.

A separate hierarchy folder ("Unit") is provided for each simulation template in the PCS 7 sample project "APC_ExaSP". 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.

15.2.10.2 Tag types

The PCS 7 library in the Templates folder contains the following process tag types:

You will find control engineering information on the templates (insertable process tag types) of the PCS 7 library in a separate section in this help:

- PID controller with safety logic and control loop monitoring (PIDCTRL_ConPerMon)
- Step controller with direct access to the actuator without position feedback (Step_CTRL_Direct)
- Split-range controller (SPLITRING)
- Ratio controller (RATIO)
- Cascade controller (Cascade CTRL)
- Continuous PID controller (PIDCTRL)
- PID control with feedforward control (PIDCTRL_DistComp)
- PID control with operating point oriented adaptation of parameters (gain scheduling) (PIDCTRL_GainSched)
- Override control (PIDCTRL_Override)
- PID control with Smith predictor (PIDCTRL_SmithPredictor)
- Model predictive control (MPC_CTRL)

The "Templates" folder contains prototype process tag types for several of the control loop structures shown in the simulation examples. 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.

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

V