SIEMENS

Example projects for SIMATIC PID Professional

Application Examples

01/2012
A5E03806704-01

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

Proper use of Siemens products

Note the following:

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

Trademarks

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

Disclaimer of Liability

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

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E03806704-01
℗ 01/2012 Technical data subject to change

Table of contents

Standard PID Control 1

1.1 Example01: Step controller

Application

Example01 comprises a standard step controller (PID_ES) in combination with a simulated process, consisting of an actuator with integrating behavior and a downstream third-order delay element (PT3).

With the help of Example01, you can easily generate a step controller and parameterize and test all properties of this controller offline in a typical process.

The example makes it easy to understand the operating principle and configuration of controllers with discontinuous output, such as they are very often used to control processes with motorized actuators. It can therefore also be used for familiarizing and training purposes.

Functions of Example01

The main components of Example01 are the PID_ES and PROC_S instructions. PID_ES represents the utilized standard controller, and PROC_S simulates a process with the "Valve" and PT3 function elements. In addition to the controlled variable, information about the position of the actuator and any triggered endstop signals are transferred to the controller.

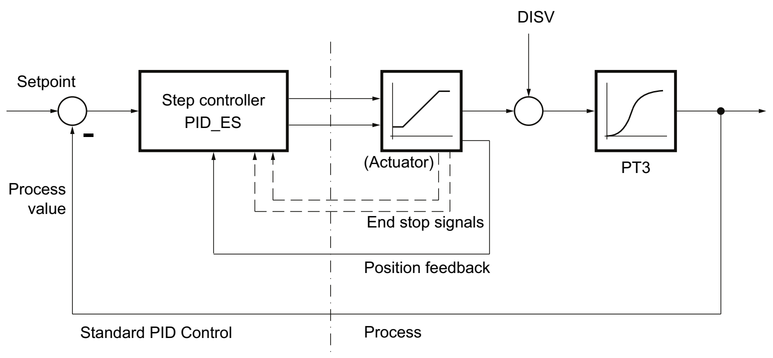


Figure 1-1 Example01, control loop

The PROC_S function block simulates a series connection that consists of the integrating actuator and three first-order delay elements. The DISV disturbance variable is always added to the output signal of the actuator, so that process disturbances can be manually injected at this point. The GAIN factor allows determination of the static system gain.

The parameter for the **MTR_TM** motor transition time defines the time that the actuator requires for one endstop-to-endstop cycle.

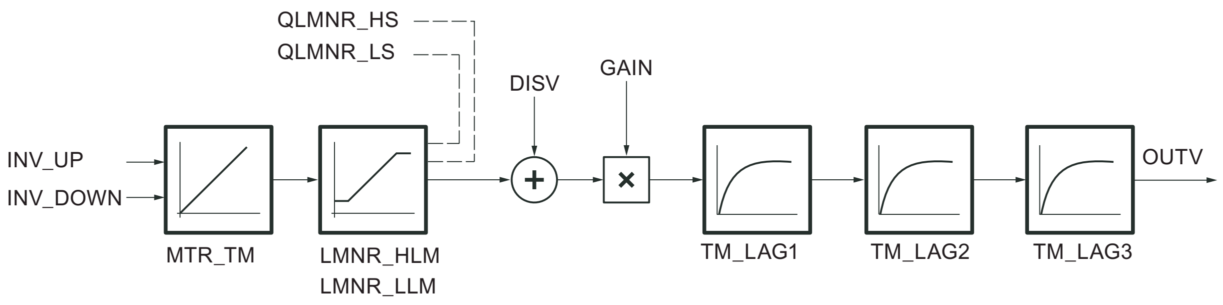


Figure 1-2 Structure and parameters of PROC_S process block

Block structure

Example01 comprises the EX01 function, which includes the blocks for the controller and the simulated process, as well as the call blocks for warm restart (OB 100) and a cyclic interrupt level (OB 35 with 100 ms cycle clock).

Table 1- 1 Blocks of Example01

Block	Name (in the toolbar)	Description
OB 100	RESTART	Warm restart OB
OB 35	CYC_INT1	Time-controlled OB: 100 ms
FC 100	EX01	Example 1
FB 2	PID_ES	Step controller
FB 100	PROC_S	Process for step controller
DB 100	PROCESS	Instance DB for PROC_S
DB 101	CONTROL	Instance DB for PID_ES

The DB 100 instance data block for the process and DB 101 instance data block for the controller are assigned to the two function blocks.

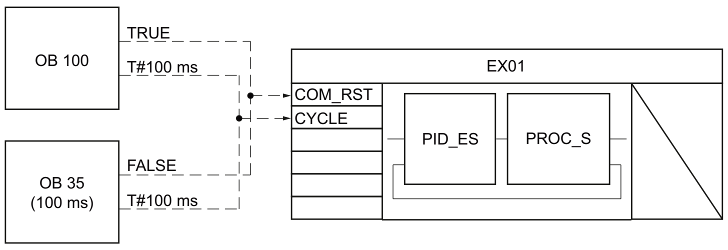


Figure 1-3 Blocks of Example01: Interconnection and call

Parameters of the process model

The parameters of the PID_ES controller block and their meaning are described in the online help. The parameters of the PROC_S process block are listed in the following table.

Table 1- 2 Parameters of the PROC_S process block (DB100: FB100)

Parameter	Type	Value range	Description
INV_UP	BOOL		Input signal up (more)
INV_DOWN	BOOL		Input signal down (less)
COM_RST	BOOL		Warm restart
CYCLE	TIME	≥ 1 ms	Sampling time
DISV	REAL		Disturbance variable
GAIN	REAL		Process gain
MTR_TM	TIME		Motor transition time
LMNR_HLM	REAL	LMNR_LLM ... 100.0 [%]	Upper limit of position feedback
LMNR_LLM	REAL	-100.0...LMNR_HLM [%]	Lower limit of position feedback
TM_LAG1	TIME	≥ CYCLE/2	Delay time 1
TM_LAG2	TIME	≥ CYCLE/2	Delay time 2
TM_LAG3	TIME	≥ CYCLE/2	Delay time 3
OUTV	REAL		Output variable
LMNR	REAL		Position feedback
QLMNR_HS	BOOL		Actuator at high endstop
QLMNR_LS	BOOL		Actuator at low endstop

After a **warm restart**, the OUTV output variable as well as all internal memory variables are set to zero.

Interconnection and call of Example01

The following figure shows how the step controller uses the EX01 function internally along with the process model to interconnect to a control loop.

Obviously, if the LMNR - LMNR_IN connection is opened, this also produces a step control without position feedback.

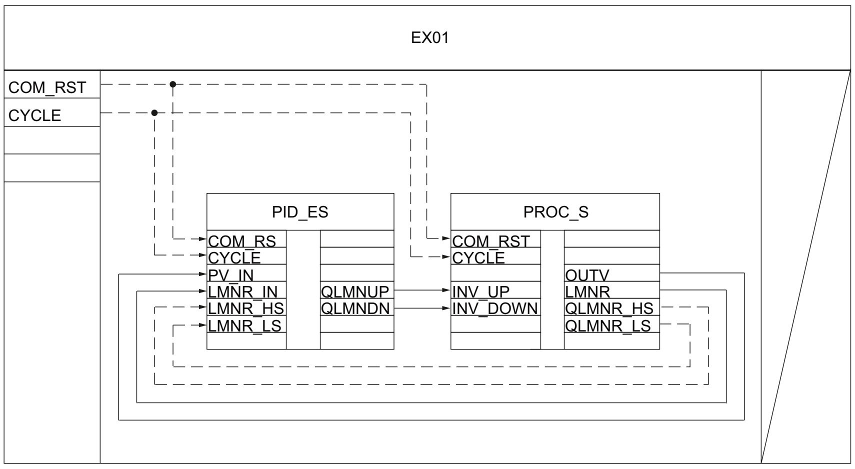
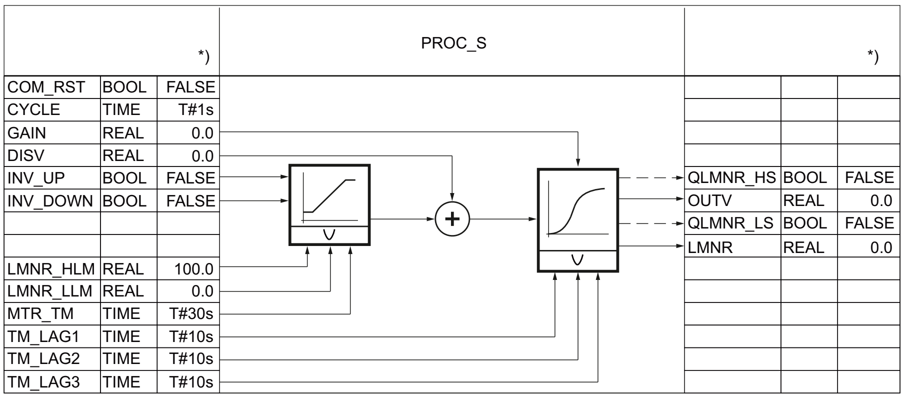


Figure 1-4 Interconnection and call of EX01 function

Parameters of the model process for step controller

The following figure shows the function scheme and the parameters of the process.

In case of a **warm restart** or **hot restart**, the control responds as described in the online help.



		*)
COM_RST	BOOL	FALSE
CYCLE	TIME	T#1s
GAIN	REAL	0.0
DISV	REAL	0.0
INV_UP	BOOL	FALSE
INV_DOWN	BOOL	FALSE
LMNR_HLM	REAL	100.0
LMNR_LLM	REAL	0.0
MTR_TM	TIME	T#30s
TM_LAG1	TIME	T#10s
TM_LAG2	TIME	T#10s
TM_LAG3	TIME	T#10s

		*)
QLMNR_HS	BOOL	FALSE
OUTV	REAL	0.0
QLMNR_LS	BOOL	FALSE
LMNR	REAL	0.0

Figure 1-5 Function scheme and parameters of process model PROC_S

*) Default for creation of new instance DB

Parameters and step response

The reaction of a control loop with a simulated third-order PT process is shown based on a specific parameter assignment of the step controller with PI-action and activated dead band. The selected process parameters with 10 s delay time in each case approximate the behavior of a rapid temperature process or a fill level control.

Setting one of the delay times TM_LAGx = 0 s reduces the order of the process by one degree.

The figure below shows the transient and settling response of the closed control loop after a setpoint value change of 60 percent. The table contains the currently set values of the relevant parameters for the controller and process.

Parameter	Type	Parameter assignment	Description
Controller:			
CYCLE	TIME	100 ms	Sampling time
GAIN	REAL	0.31	Proportional gain
TI	TIME	19.190 s	Integral action time
MTR_TM	TIME	20 s	Motor transition time
PULSE_TM	TIME	100 ms	Minimum pulse time
BREAK_TM	TIME	100 ms	Minimum break time
DEADB_ON	BOOL	TRUE	Dead band ON
DEADB_W	REAL	0.5	Dead band width
Process			
GAIN	REAL	1.5	Process gain
MTR_TM	TIME	20 s	Motor transition time
TM_LAG1	TIME	10 s	Delay time 1
TM_LAG2	TIME	10 s	Delay time 2
TM_LAG3	TIME	10 s	Delay time 3

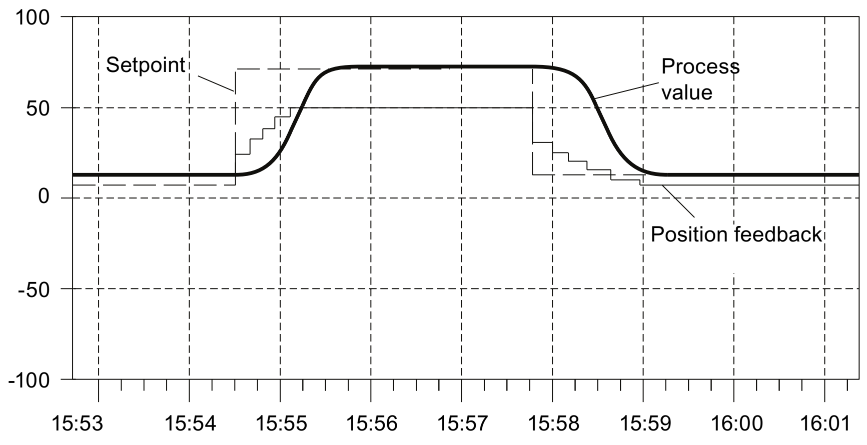


Figure 1-6 Control loop with step controller after setpoint step change

Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX01 function.
Use the TIA Portal to download the example to the CPU.
Open the commissioning interface of the CONTROL technology object in the "Technology objects" folder, and perform the commissioning as described in the online help.

1.2 Example02: Continuous controller

Application

Example02 comprises a continuous controller (PID_CP) in combination with a simulated process, consisting of a third-order delay element (PT3).

With the help of Example02, you can easily generate a continuous PID controller and parameterize and test all properties of this controller offline in a typical process.

The example enables you to easily understand the operating principle and configuration of controllers with analog output signal, in such a way as they are often used to control processes with proportional-action actuators. It can therefore also be used for familiarizing and training purposes.

Functions of Example02

The main components of Example02 are the PID_CP (FB 1) and PROC_C (FB 100) function blocks. PID_CP represents the controller utilized, and PROC_C simulates a third-order self-regulating process.
The example also includes a preconfigured time-dependent setpoint characteristic curve for the time scheduler. The data required for this are stored in the DB_RMPSK global data block.

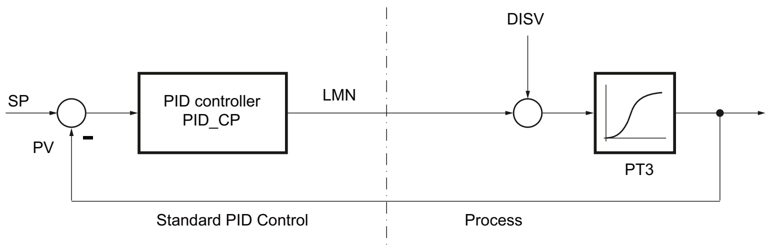


Figure 1-7 Control loop of Example02

The PROC_C function block simulates a series connection that consists of three first-order delay elements. The **DISV** disturbance variable is always added to the output signal of the actuator, so that process disturbances can be manually injected at this point. The **GAIN** factor allows determination of the static system gain.

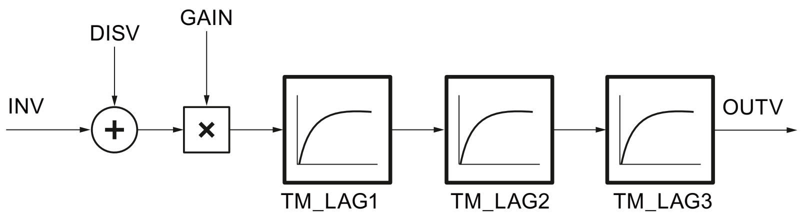


Figure 1-8 Structure and parameters of PROC_C process block

Block structure

Example02 comprises the EX02 function, which includes the blocks for the controller and the simulated process, as well as the call blocks for warm restart (OB 100) and a cyclic interrupt level (OB 35 with 100 ms cycle clock).

Table 1- 3 Blocks of Example02

OB 100	RESTART	Warm restart OB
OB 35	CYC_INT1	Time-controlled OB: 100 ms
FC 100	EX02	Example 2
FB 1	PID_CP	Continuous PID controller
FB 100	PROC_C	Process for continuous controller
DB 100	PROCESS	Instance DB for PROC_C
DB 101	CONTROL	Instance DB for PID_CP
DB 2	DB_RMPSK	Global DB for call data of the time scheduler

The DB 100 instance data block for the process and the DB 101 instance data block for the controller are assigned to the two function blocks.

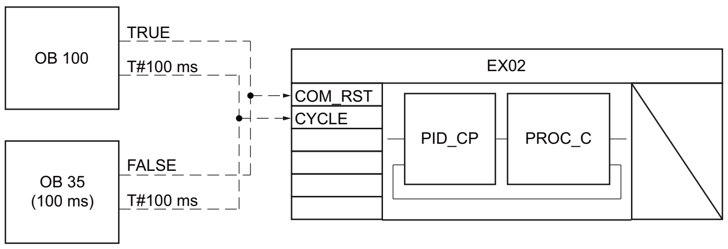


Figure 1-9 Blocks of Example02: Interconnection and call

Parameters of the process model

The parameters of the PID_CP controller block and their meaning are described in the online help. The parameters of the PROC_C process block are listed in the following table.

Table 1- 4 Parameters of the PROC_C process block (DB 100: FB 100)

Parameter	Type	Value range	Description
INV	REAL		Input variable
COM_RST	BOOL		Warm restart
CYCLE	TIME	≥ 1 ms	Sampling time
DISV	REAL		Disturbance variable
GAIN	REAL		Process gain factor
TM_LAG1	TIME	≥ CYCLE/2	Delay time 1
TM_LAG2	TIME	≥ CYCLE/2	Delay time 2
TM_LAG3	TIME	≥ CYCLE/2	Delay time 3
OUTV	REAL		Output variable

Interconnection and call of Example02

The following figure shows how the continuous controller uses the EX02 function internally along with the process model to interconnect to a control loop.

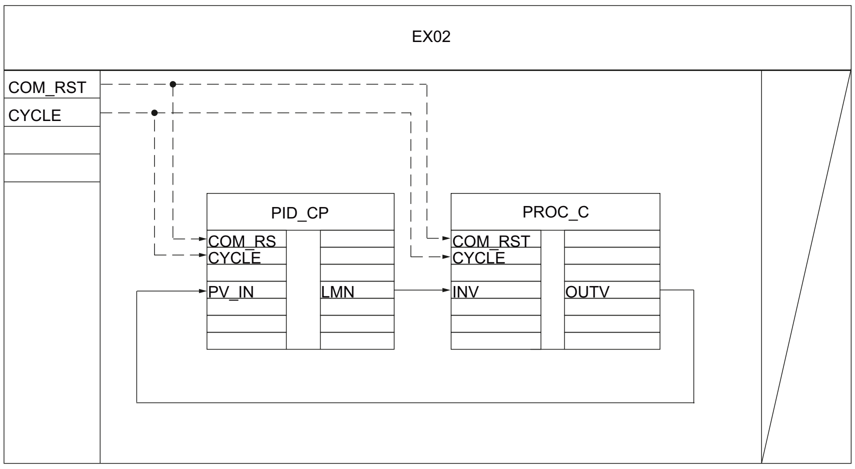


Figure 1-10 Interconnection and call of EX02 function

Parameters of the model process for continuous controllers

The following figure shows the function scheme and the parameters of the process.

In case of a **warm restart** or **hot restart**, the control responds as described in the online help.

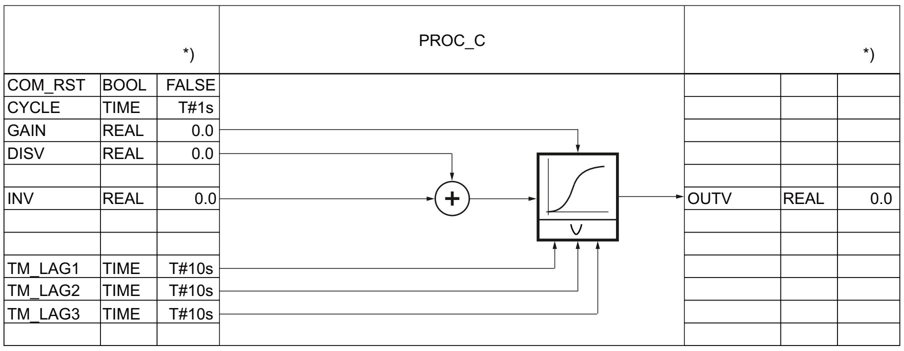


Figure 1-11 Function scheme and parameters of process model PROC_C

*) Default for creation of new instance DB

Parameters and step response

The reaction of a control loop with a simulated third-order PT process is shown based on a specific parameter assignment of a continuous controller with PID action. The selected process parameters with 10 s delay time in each case approximate the behavior of a pressure control or a fill level control.

Setting one of the delay times TM_LAG x = 0 s reduces the order of the process by one degree.

The following figure shows the transient and settling response of the closed control loop after a series of setpoint changes of 20 percent of the measuring range in each case. The table contains the currently set values of the relevant parameters for the controller and process.

Parameter	Type	Parameter assignment	Description
Controller:			
CYCLE	TIME	100 ms	Sampling time
GAIN	REAL	1.535	Proportional gain
TI	TIME	22.720 s	Integral action time
TD	TIME	5.974 s	Derivative action time
TM_LAG	TIME	1.195 s	Delay time of derivative component
Process:			
GAIN	REAL	1.5	Process gain
TM_LAG1	TIME	10 s	Delay time 1
TM_LAG2	TIME	10 s	Delay time 2
TM_LAG3	TIME	10 s	Delay time 3

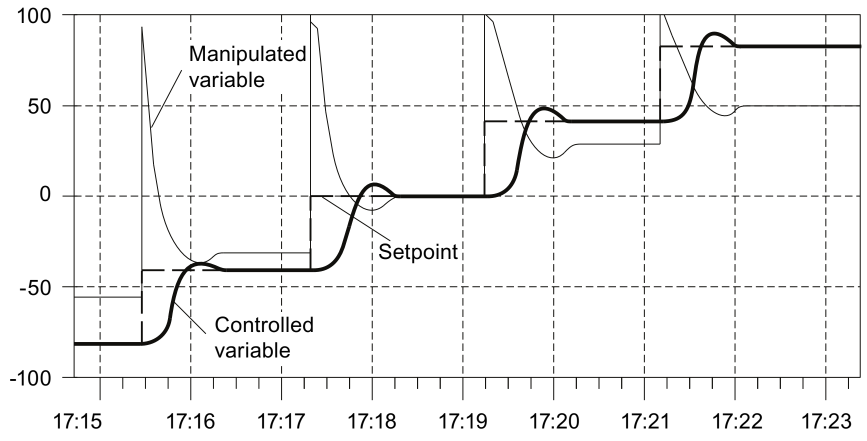


Figure 1-12 Control with continuous controller and setpoint step changes across the entire measuring range

Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX02 function.
Use the TIA Portal to download the example to the CPU.
Open the commissioning interface of the CONTROL technology object in the "Technology objects" folder, and perform the commissioning as described in the online help. If you want to make use of the preconfigured time-dependent setpoint characteristic curve for the example, activate the time scheduler by setting CONTROL.RMPSK_ON=TRUE.

1.3 Example03: Cascade control

Application

Example03 comprises all of the blocks that are required to configure a cascade control with one final controlled variable and one secondary controlled variable.

Example03 allows you to easily generate a cascade control with a master control loop and a follow-up control loop. The structure can be easily expanded to include more than one follow-up control loop.

Functions of Example03

Example03 includes the loop scheduler (LP_SCHED) with the associated global data block (DB–LOOP), the FB 1 function block for the continuous standard controller (master controller), and the FB 2 function block for the step controller (slave controller), together with the two instance DBs for the configuration data of the controllers. The PROC_S and PROC_C function blocks generate the simulated process. They are described in Example01 and Example02.

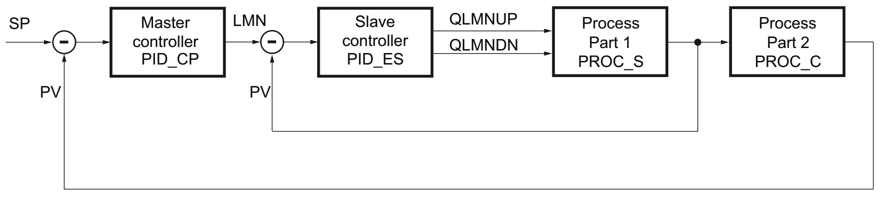


Figure 1-13 Two-loop cascade control (Example03)

By means of the loop scheduler, the controllers are each called cyclically from the cyclic interrupt level with 100 ms cycle clock.

The controller with continuous output (PID_CP) thereby acts as a master controller on the setpoint of the slave controller in such a way that the final controlled variable at the output of process section 2 is kept at setpoint SP. The step controller controls disturbances affecting process section 1 in the secondary control loop (PID_ES) without influencing the PV final controlled variable.

Block structure

Example03 comprises the EX03 function, which includes the blocks for the loop scheduler and the two controllers, as well as the call blocks for warm restart (OB 100) and a cyclic interrupt level (OB 35 with 100 ms cycle clock).

Table 1- 5 Blocks of Example03

Block	Name (in the toolbar)	Description
OB 100	RESTART	Warm restart OB
OB 35	CYC_INT1	Time-controlled OB: 100 ms
FC 100	EX03	Example 3
FC 1	LP_SCHED	Loop scheduler
FB 1	PID_CP	Continuous PID controller
FB 2	PID_ES	Step controller
FB 102	PROC_S	Process section 1
FB 103	PROC_C	Process section 2
DB 1	DB_LOOP	Global DB for call data for LP_SCHED
DB 100	CONTROL_C	Instance DB for PID_CP
DB 101	CONTROL_S	Instance DB for PID_ES
DB 102	PROCESS_S	Instance DB for PROC_S
DB 103	PROCESS_C	Instance DB for PROC_C

The DB 100 and DB 101 instance blocks are assigned to the PID_CP and PID_ES function blocks, respectively.

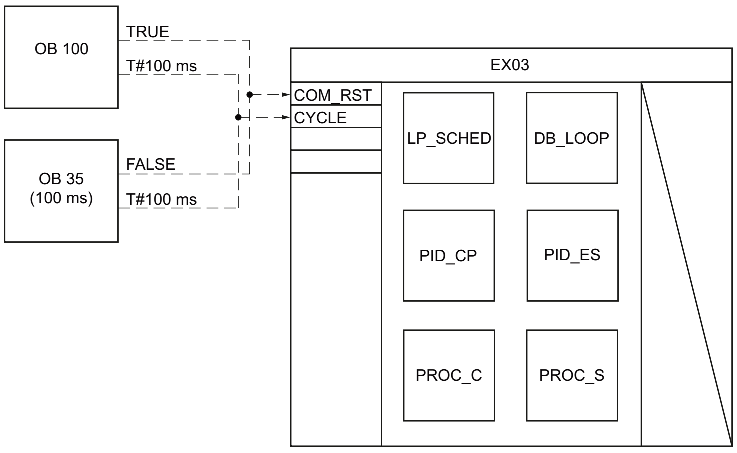


Figure 1-14 Blocks of Example03: Interconnection and call

Parameter assignment of Example03

The following figure reveals how the controllers use the EX03 function internally to interconnect to the loop scheduler and with one another.

In case of a **warm restart** or **hot restart**, the control responds as described in the online help.

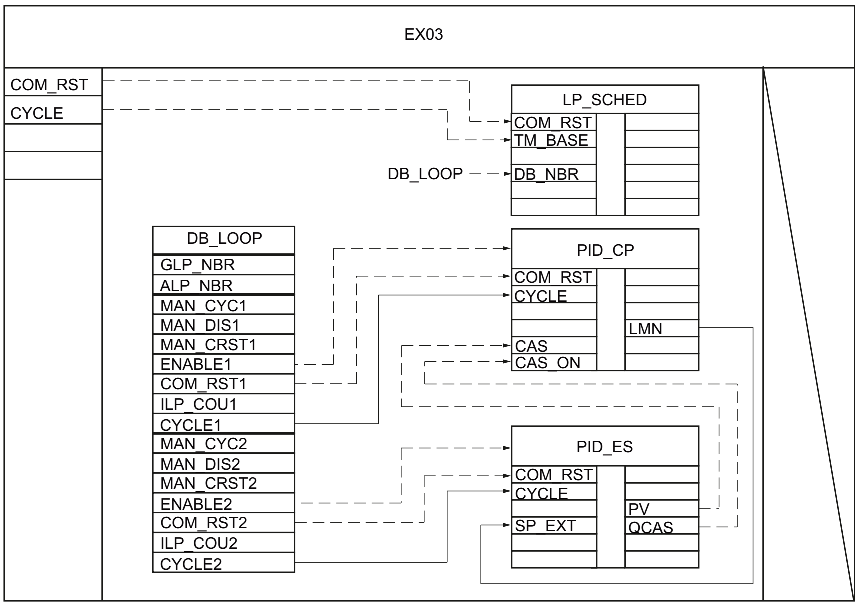


Figure 1-15 Circuit diagram and parameters of EX03 function (representation without process simulation blocks)

Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX03 function.
Use the TIA Portal to download the example to the CPU.
Open the commissioning interface of the CONTROL_C technology object in the "Technology objects" folder, and perform the commissioning as described in the online help.
The commissioning interface of the CONTROL_S technology object (slave controller) can be used to operate the follow-up control loop independently from the master controller. However, this is not necessary to implement the example.

1.4 Example04: Pulse controller

Application

Example04 comprises a pulse controller (PID_CP) with positive and negative output in combination with a simulated process, consisting of a third-order delay element (PT3).

With the help of Example04, you can easily generate a pulse controller and parameterize and test all properties of this controller in a typical process.

The example enables you to easily understand the operating principle and configuration of controllers with binary pulse outputs, as they are often used to control processes with proportional-action actuators. These types of controllers are used, for example, for temperature processes involving electric heating. It can therefore also be used for familiarizing and training purposes.

Functions of Example04

The main components of Example04 are the PID_CP (FB 1) and PROC_CP (FB 100) function blocks. PID_CP represents the pulse controller utilized, and PROC_CP simulates a third-order self-regulating process.

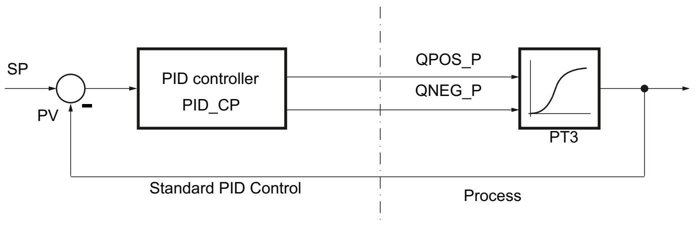


Figure 1-16 Control loop of Example04

The PROC_CP function block simulates a series connection that consists of three first-order delay elements. In addition to the POS_P and NEG_P pulse inputs, the **DISV** disturbance variable also acts as an input signal in the process, so that process disturbances can be manually injected at this point. The **GAIN** factor allows determination of the static system gain.

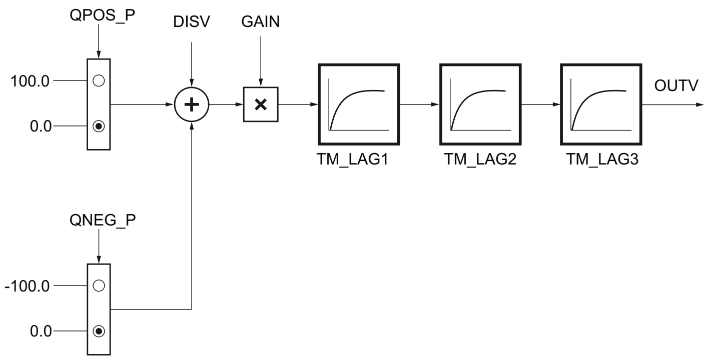


Figure 1-17 Structure and parameters of PROC_CP process block

Block structure

Example04 comprises the EX04 function, which includes the blocks for the controller and the simulated process, as well as the call blocks for warm restart (OB 100) and a cyclic interrupt level (OB 35 with 100 ms cycle clock).

Table 1- 6 Blocks of Example04

Block	Name (in the toolbar)	Description
OB 100	RESTART	Warm restart OB
OB 35	CYC_INT1	Time-controlled OB: 100 ms
FC 100	EX04	Example 4
FB 1	PID_CP	Continuous controller with pulse generator
FB 100	PROC_CP	Process for continuous controller with pulse inputs
DB 100	PROCESS	Instance DB for PROC_C
DB 101	CONTROL	Instance DB for PID_CP

The PROCESS DB 100 instance data block for the process and the CONTROL DB 101 instance data block for the controller are assigned to the two function blocks.

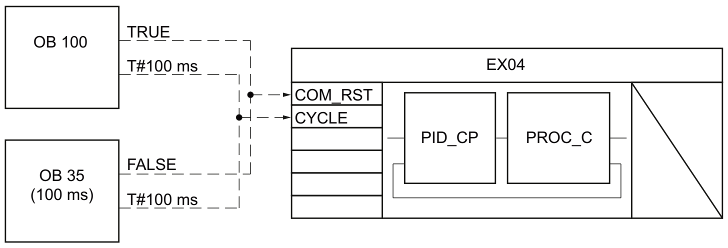


Figure 1-18 Blocks of Example04: Interconnection and call

Parameters of the process model

The parameters of the PID_CP controller block and their meaning are described in the online help. The parameters of the PROC_CP process block are listed in the following table.

Table 1- 7 Parameters of the "PROC_CP" process block (DB 100: FB 100)

Parameter	Type	Value range	Description
DISV	REAL		Disturbance variable
GAIN	REAL		Process gain factor
TM_LAG1	TIME	≥ CYCLE/2	Delay time 1
TM_LAG2	TIME	≥ CYCLE/2	Delay time 2
TM_LAG3	TIME	≥ CYCLE/2	Delay time 3
POS_P	BOOL		Positive pulse
NEG_P	BOOL		Negative pulse
COM_RST	BOOL		Warm restart
CYCLE	TIME	≥ 1 ms	Sampling time
OUTV	REAL		Output variable

Interconnection and call of Example04

The following figure reveals how the continuous controller uses the EX04 function internally along with the process model to interconnect to a control loop.

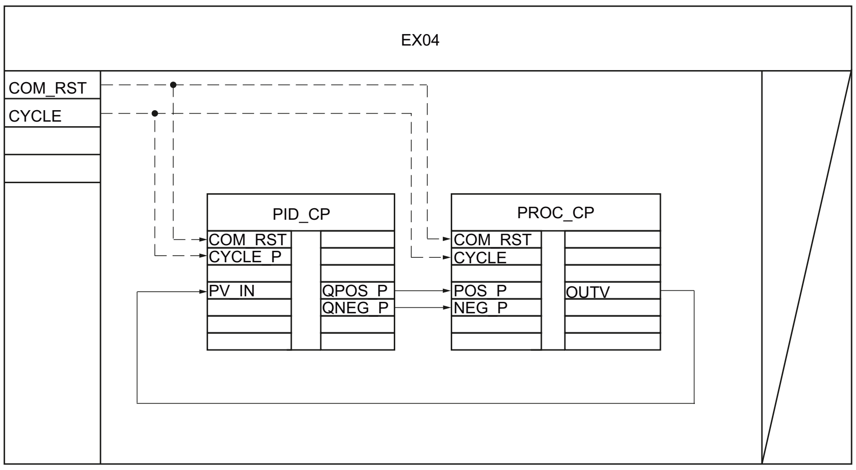


Figure 1-19 Interconnection and call of EX04 function

Parameters of the model process for continuous controllers

The following figure shows the function scheme and the parameters of the process.

In case of a **warm restart** or **hot restart**, the control responds as described in the online help.

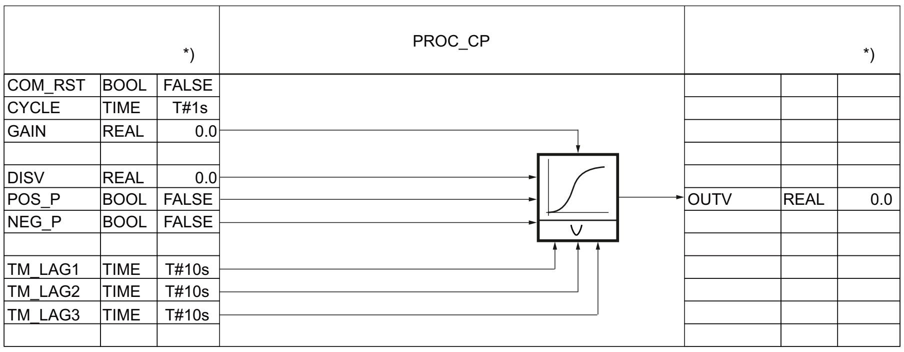


Figure 1-20 Function scheme and parameters of PROC_CP process model

*) Default for creation of new instance DB

Parameters and step response

The reaction of a control loop with a simulated third-order PT process is shown based on a specific parameter assignment of a continuous controller with PID action. The selected process parameters with 10 s delay time in each case produce a more rapid process than is achieved during temperature control, in practice. However, the relatively rapid process allows you to test the controller function faster. By changing the lag time constant, you can easily produce a simulated process whose characteristics approximate a real process.

The following figure shows the transient and settling response of the closed control loop after a series of setpoint changes of 20 percent of the measuring range in each case. In this case, the continuous manipulated variable of the controller is mapped and not the pulse outputs. The table contains the currently set values of the relevant parameters for the controller and process.

Parameter	Type	Parameter assignment	Description
Controller:			
CYCLE	TIME	1 s	Sampling time of the controller
CYCLE_P	TIME	100 ms	Sampling time
GAIN	REAL	1.535	Proportional gain
TI	TIME	22.720 s	Integral action time
TD	TIME	5.974 s	Derivative action time
TM_LAG	TIME	1.195 s	Delay time of derivative component

Parameter	Type	Parameter assignment	Description
Process:			
GAIN	REAL	1.5	Process gain
TM_LAG1	TIME	10 s	Delay time 1
TM_LAG2	TIME	10 s	Delay time 2
TM_LAG3	TIME	10 s	Delay time 3

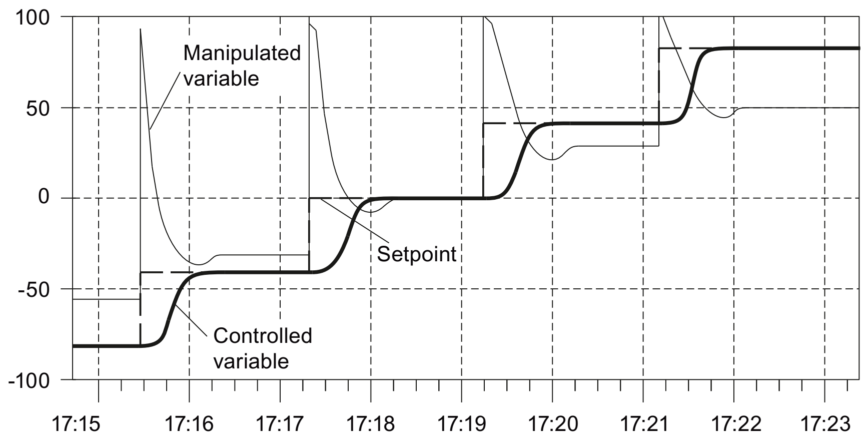


Figure 1-21 Control with continuous controller with pulse outputs and setpoint step changes across the entire measuring range

Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX04 function.
Use the TIA Portal to download the example to the CPU.
Open the commissioning interface of the CONTROL technology object in the "Technology objects" folder, and perform the commissioning as described in the online help.

1.5 Example05: Optimize continuous controller with PID Self-Tuner

Overview

This example contains a simple control loop that consists of a PID controller and a Delay time 2 element as a model controlled system for a temperature process. It is an example of the application of the Self-Tuner with a continuous PID controller.

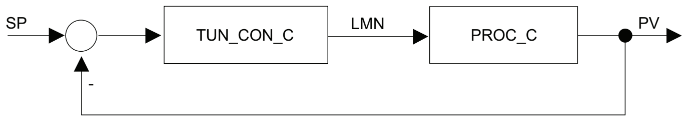


Figure 1-22 Control loop of Example05

Program structure

The IDB_TUN_CON_C block is an instance DB of the TUN_CON_C FB that contains the calls of the Self-Tuner and controller. The TUN_CON_C FB and the PROC_C process simulation FB are called in OB 35.

Process block for simulating a temperature process

The block simulates a process with a third-order delay element. For temperature processes, you choose Delay time 2 response with a large and a small time constant (TM_LAG1 = 15 × TM_LAG2 and TM_LAG3 = 0s).

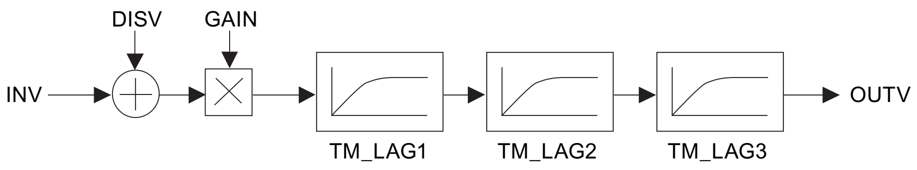


Figure 1-23 Block diagram of the process

Parameters

INV	Input variable (manipulated variable of controller)
DISV	Disturbance variable
GAIN	Process gain
TM_LAG1	Delay time 1 (for temperature processes)
TM_LAG2	Delay time 2 (TM_LAG1 = 15 × TM_LAG2)
TM_LAG3	Delay time 3 (= 0 for temperature processes)
OUTV	Output variable (e.g., temperature)

After addition of the analog input signal and a disturbance variable and multiplication with the process gain, three first-order delay elements are run through.

During initialization, the output variable is set to OUTV = DISV × GAIN.

Operator control and monitoring

Operator control for the example is via the commissioning interface of the Self-Tuner, which is described in the online help for the Self-Tuner. The commissioning interface can be found in the "Technology objects" folder for the "tuner" technology object.

Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the IDB_TUN_CON_C.CYCLE_P sampling time.
Use the TIA Portal to download the example to the CPU.
Open the commissioning interface of the "tuner" technology object in the "Technology objects" folder, and perform the commissioning as described in the online help.

1.6 Example06: Optimize step controller with PID Self-Tuner

Overview

This example contains a simple control loop that consists of a PID controller and a Delay time 2 element with an actuator with integrating behavior as a model controlled system for a temperature process. It is an example of the application of the Self-Tuner with a PID step controller.

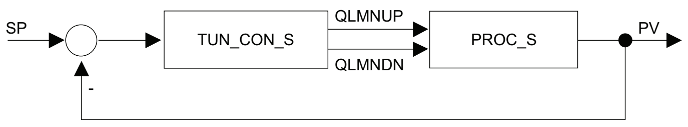


Figure 1-24 Control loop of Example06

Program structure

The IDB_TUN_CON_S block is an instance DB of the TUN_CON_S FB that contains the calls of the Self-Tuner and controller. The TUN_CON_S FB and the PROC_S process simulation FB are called in OB 35.

Process block for simulating a temperature process

The block simulates a process with a third-order delay element. For temperature processes, you choose Delay time 2 response with a large and a small time constant (TM_LAG1 = 15 × TM_LAG2 and TM_LAG3 = 0 s).

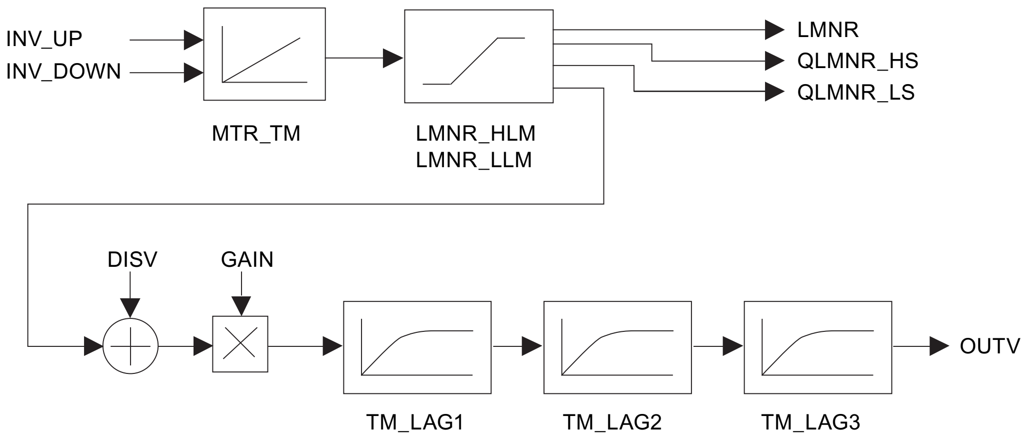


Figure 1-25 Block diagram of the process

Parameters

INV_UP	Manipulated variable signal up
INV_DOWN	Manipulated variable signal down
DISV	Disturbance variable
GAIN	Process gain
MTR_TM	Motor transition time
LMNR_HLM	High limit of actuator
LMNR_LLM	Low limit of actuator
TM_LAG1	Delay time 1
TM_LAG2	Delay time 2 (for temperature processes: TM_LAG1 = 15 × TM_LAG2)
TM_LAG3	Delay time 3 (= 0 for temperature processes)
OUTV	Output variable (e.g., temperature)
LMNR	Position feedback
QLMNR_HS	High limit of endstop signal
QLMNR_LS	Low limit of endstop signal

Depending on the INV_UP and INV_DOWN input signals, the LMNR position feedback is calculated using an integrator. The position feedback is limited to LMNR_HLM and LMNR_LLM . When the limit is reached, the QLMNR_HS and QLMNR_LS endstop signals are set.

After addition of a disturbance variable and subsequent multiplication with the process gain, three first-order delay elements are run through.

During initialization, the OUTV and LMNR output variables are set to zero.

Operator control and monitoring

Operator control for the example is via the commissioning interface of the Self-Tuner, which is described in the online help for the Self-Tuner. The commissioning interface can be found in the "Technology objects" folder for the "tuner" technology object.

Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 20 ms. If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the IDB_TUN_CON_S.CYCLE_P sampling time.
Use the TIA Portal to download the example to the CPU.
Open the commissioning interface of the "tuner" technology object in the "Technology objects" folder, and perform the commissioning as described in the online help.

1.7 Example07: Optimize pulse controller with PID Self-Tuner

Overview

This example contains a simple control loop that consists of a PID controller with pulse generation and a Delay time 2 element as a model process for a temperature process. It is an example of the application of the Self-Tuner with a continuous PID controller with pulse generation.

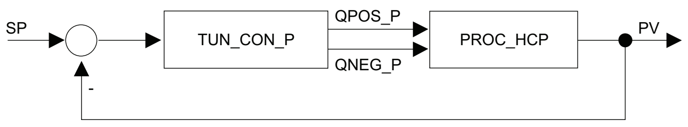


Figure 1-26 Control loop of Example07

Program structure

The IDB_TUN_CON_P block is an instance DB of the TUN_CON_P FB that contains the calls of the Self-Tuner and controller. The TUN_CON_P FB and the PROC_HCP process simulation FB are called in OB 35.

Process block for simulating a temperature heating zone

The block simulates a typical temperature process for heating and cooling as it can occur as a control zone in a extruder, an injection molding machine, a tempering machine or as a separate furnace in reality.

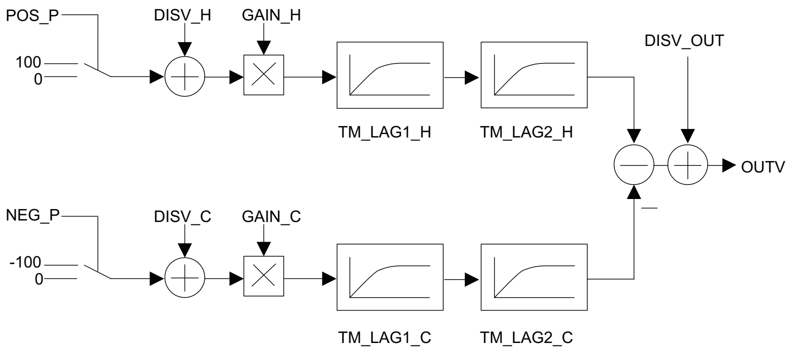


Figure 1-27 Block diagram of the process

Parameters

POS_P	Binary input signal heating
NEG_P	Binary input signal cooling
DISV_H	Disturbance variable heating
DISV_C	Disturbance variable cooling
DISV_OUT	Disturbance variable at output (e.g., ambient temperature)
GAIN_H	Process gain heating
GAIN_C	Process gain cooling
TM_LAG1_H	Heating delay time 1
TM_LAG2_H	Heating Delay time 2
TM_LAG1_C	Cooling delay time 1
TM_LAG2_C	Cooling Delay time 2
OUTV	Output variable (e.g., control zone temperature)

The binary input signals are converted to continuous floating-point values (-100, 0, 100). After addition of a disturbance variable (for example, ambient temperature) and multiplication with the process gain, the two first-order delay elements are run through. This procedure is performed separately in the heating and cooling processes.

During initialization, the output variable is set to
OUTV = DISV_H × GAIN_H - DISV_C × GAIN_C + DISV_OUT.

Operator control and monitoring

Operator control for the example is via the commissioning interface of the Self-Tuner, which is described in the online help for the Self-Tuner. The commissioning interface can be found in the "Technology objects" folder for the "tuner" technology object.

Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 20 ms. If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the IDB_TUN_CON_P.CYCLE_P sampling time.

Use the TIA Portal to download the example to the CPU.

Open the commissioning interface of the "tuner" technology object in the "Technology objects" folder, and perform the commissioning as described in the online help.

Modular PID Control 2

2.1 Example08: Step controller

2.1.1 Overview

Control loop

Example08 comprises a PID step controller (fixed-setpoint controller with switching output for actuators with integrating behavior) and a simulated process.

The following figure shows the complete control loop of Example08.

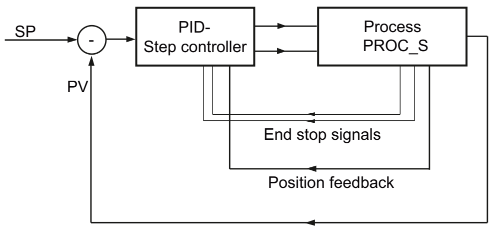


Figure 2-1 Control loop of Example08

Block call and interconnection

The following figure shows the block call and the interconnection of Example08.

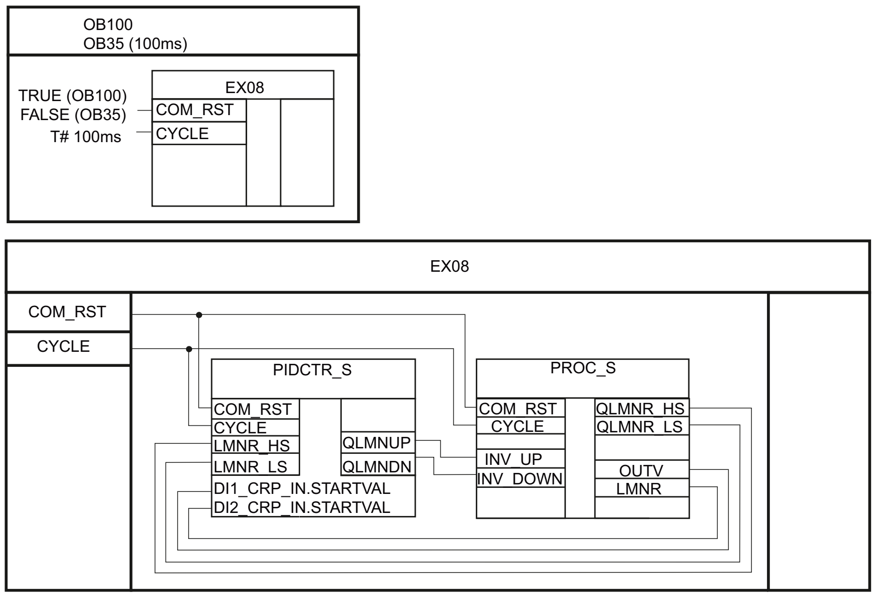


Figure 2-2 Block call and interconnection of Example08

2.1.2 PIDCTR_S: Fixed-setpoint controller with switching output for actuators with integrating behavior

Application

The PIDCTR_S block realizes a PID step controller for actuators with integrating behavior (for example, motor-driven control valves in process engineering applications). The following figure shows the block interconnection of PIDCTR_S.

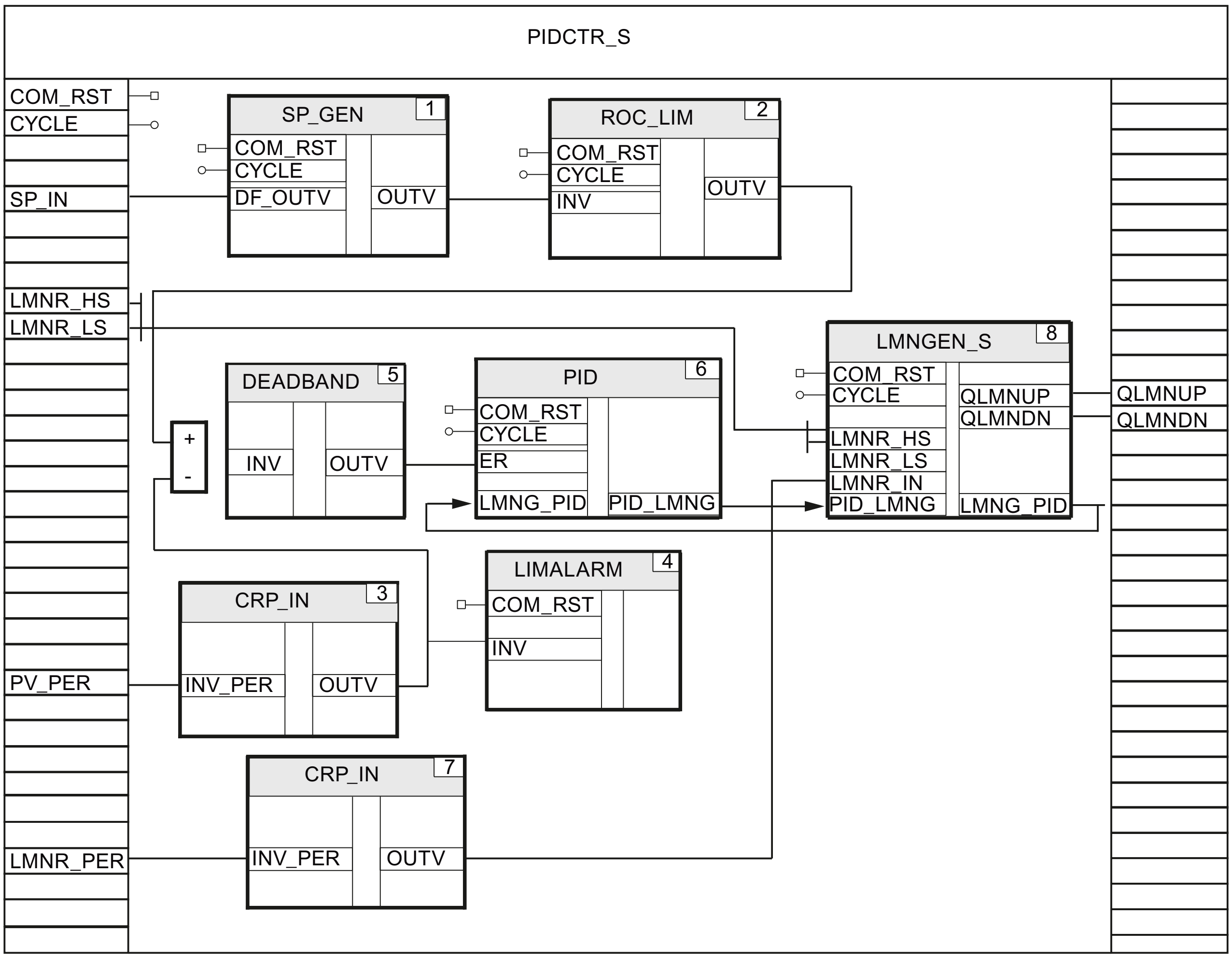


Figure 2-3 Block interconnection of PIDCTR_S

Description of functions

The SP_GEN setpoint generator specifies the setpoint whose slope is limited by the ROC_LIM ramp-function generator. The I/O process value is converted to a floating-point value by means of CRP_IN and monitored for preset limit values by the LIMALARM limit monitor. The control deviation is fed via a DEADBAND dead band element to the PID PID algorithm. The position feedback is read in via a second CRP_IN block. The LMNGEN_S manipulated variable processing block sets the QLMNUP and QLMNDN output signals.

Warm restart

Each individual block is called at a warm restart. Individual blocks with a warm restart routine are called in their warm restart routine.

2.1.3 PROC_S: Process for step controller

Application

The PROC_S block simulates an integrating control valve with a third-order delay process.

The following figure shows the block diagram of PROC_S.

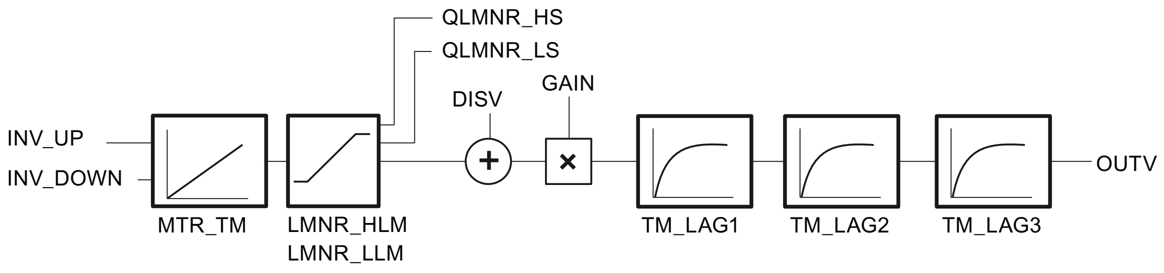


Figure 2-4 Block diagram of PROC_S

Description of functions

The block generates a series connection of an integrating control valve and three first-order delay elements. The DISV disturbance variable is always added to the output of the control valve. The MTR_TM motor transition time is the time that the valve requires to pass from endstop to endstop.

Warm restart

At a warm restart, the OUTV output variable and the internal memory variables are all set to 0.

2.1.4 Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX08 function.
Use the TIA Portal to download the example to the CPU.
You can then carry out operator control of the control system using the watch table contained in the example.
Deactivate manual mode by setting DI_PIDCTR_S.DI_LMNGEN_S.MAN_ON=FALSE.
You can specify a setpoint directly with the DI_PIDCTR_S.SP_IN parameter.
If you want to the use the setpoint generator instead, set DI_PIDCTR_S.DI_SP_GEN.DFOUT_ON=FALSE. You can then use the OUTVUP and OUTVDN input parameters of the setpoint generator to continuously increase or decrease the setpoint.
You can monitor the process value with the DI_PIDCTR_S.DI1_CRP_IN.OUTV parameter and the actuation signals with the DI_PIDCTR_S.QLMNUP and DI_PIDCTR_S.QLMNDN parameters.

2.2 Example09: Continuous controller

2.2.1 Overview

Control loop

Example09 comprises a continuous PID controller and a simulated process.
In this example, the process simulation is expanded to include a dead time element. This is realized with the DEAD_T function block, which uses the DB_DEADT global data block for data buffering.
If you want to deactivate the dead time element, connect EX09 DI_PIDCTR_C.DI_LMNGEN_C.LMN directly to DI_PROC_C.INV.
The following figure shows the complete control loop of Example09.

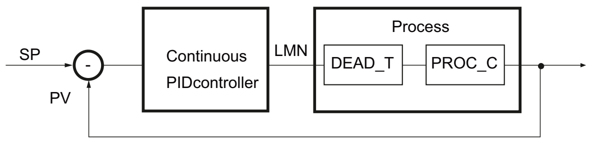


Figure 2-5 Control loop of Example09

Block call and interconnection

The following figure shows the block call and the interconnection of Example09.

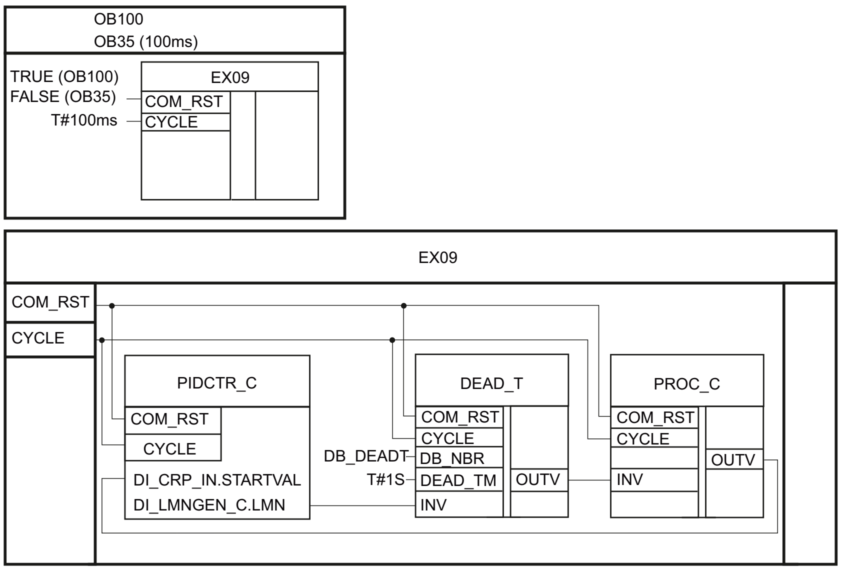


Figure 2-6 Block call and interconnection of Example09

2.2.2 PIDCTR_C: Fixed-setpoint controller with continuous output

Application

The PIDCTR_C block realizes a PID controller for continuous actuators. The following figure shows the block interconnection of PIDCTR_C.

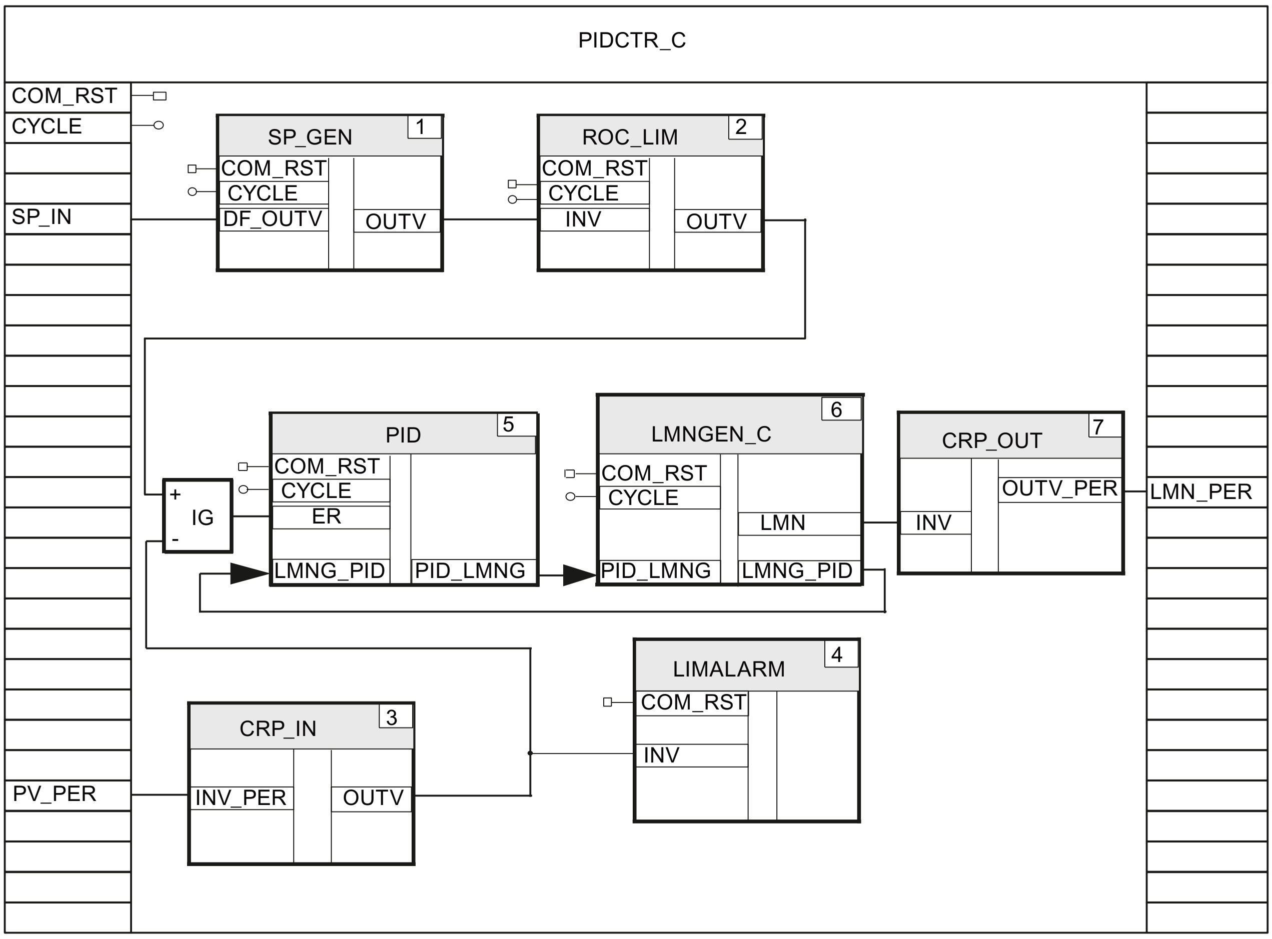


Figure 2-7 Block interconnection of PIDCTR_C

Description of functions

The SP_GEN setpoint generator specifies the setpoint whose slope is limited by the ROC_LIM ramp-function generator. The I/O process value is converted to a floating-point value by means of CRP_IN and monitored for preset limit values by the LIMALARM limit monitor. The control deviation is applied to the PID PID algorithm. The LMNGEN_C manipulated variable processing block generates the LMN analog manipulated variable, which is converted to the I/O format with CRP_OUT.

Warm restart

Each individual block is called at a warm restart. Individual blocks with a warm restart routine are called in their warm restart routine.

2.2.3 PROC_C: Process for continuous controllers

Application

The PROC_C block simulates a third-order delay process.

The following figure shows the block diagram of PROC_C.

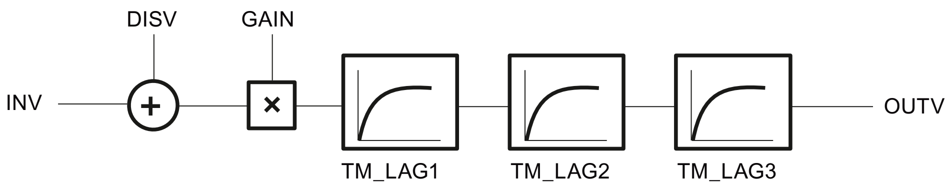


Figure 2-8 Block diagram of PROC_C

Description of functions

The block generates a series connection of 3 first-order delay elements. The DISV disturbance variable is always added to the INV input.

Warm restart

At a warm restart, the OUTV output variable and the internal memory variables are all set to 0.

2.2.4 Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX09 function.
Use the TIA Portal to download the example to the CPU.
You can then carry out operator control of the control system using the watch table contained in the example.
Deactivate manual mode by setting DI_PIDCTR_C.DI_LMNGEN_C.MAN_ON=FALSE.
You can specify a setpoint directly with the DI_PIDCTR_C.SP_IN parameter.
If you want to the use the setpoint generator instead, set DI_PIDCTR_C.DI_SP_GEN.DFOUT_ON=FALSE. You can then use the OUTVUP and OUTVDN input parameters of the setpoint generator to continuously increase and decrease the setpoint, respectively.

You can use the DI_PIDCTR_C.DI_CRP_IN.OUTV parameter to monitor the process value and the DI_PIDCTR_C.DI_LMNGEN_C.LMN parameter to monitor the manipulated variable.

2.3 Example10: Pulse controller

2.3.1 Overview

Control loop

Example10 comprises a pulse controller and a simulated process.

The following figure shows the complete control loop of Example10.

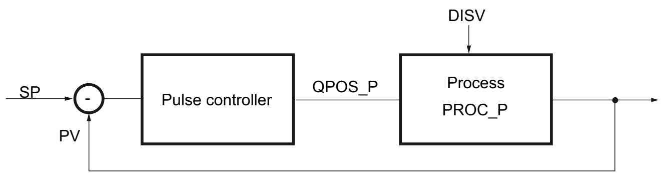


Figure 2-9 Control loop of Example10

Block call and interconnection

The following figure shows the block call and the interconnection of Example10.

Note

You must set the cycle time of OB 35 to 10 ms in the device configuration once you have selected your CPU.

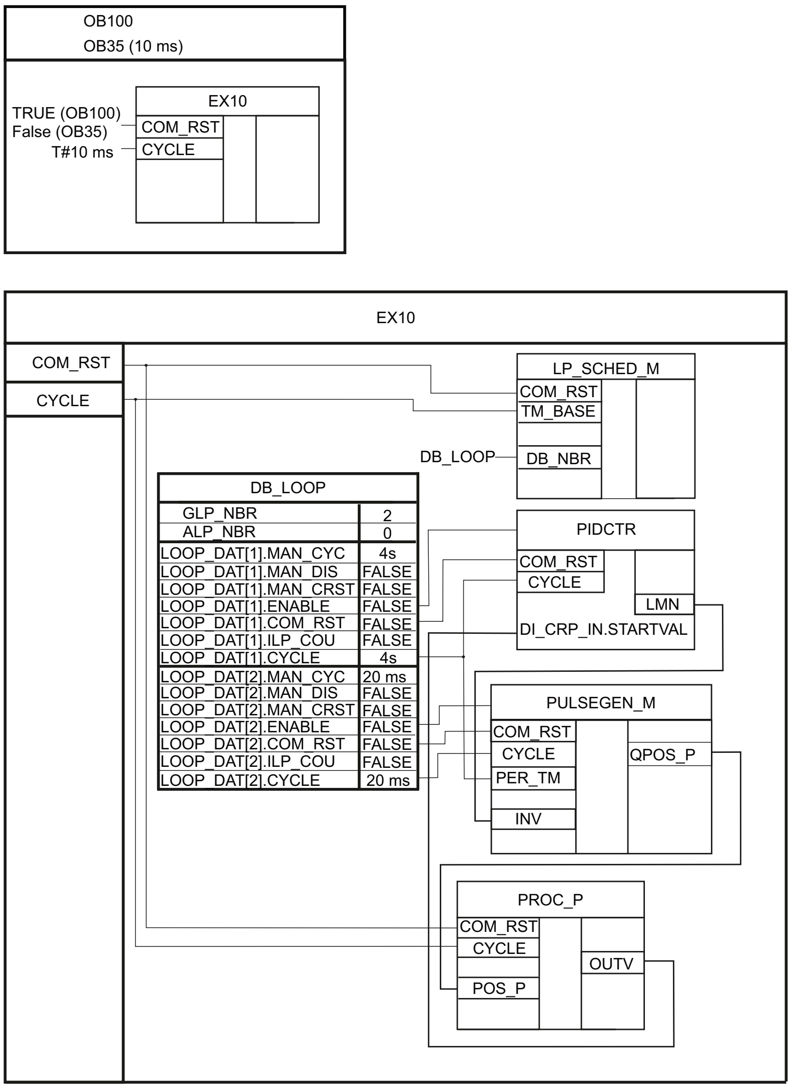


Figure 2-10 Block call and interconnection of Example10

2.3.2 PIDCTR: Master controller for continuous controller with pulse generator

Application

The PIDCTR block realizes a PID controller with continuous output. It is used to calculate the analog manipulated variable within a pulse-pause controller. It is furthermore used as a master controller in ratio and blending controls and in the cascade control. The following figure shows the block interconnection of PIDCTR.

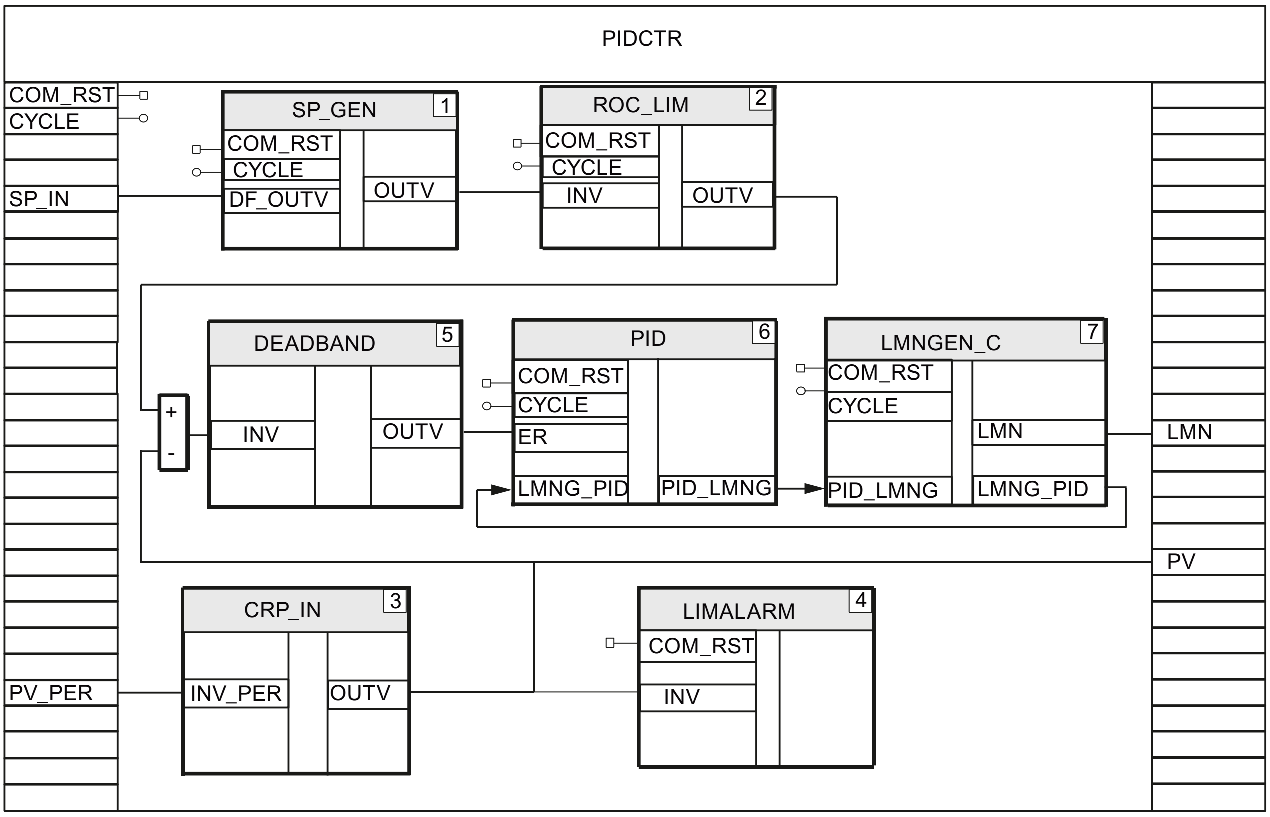


Figure 2-11 Block interconnection of PIDCTR

Description of functions

The SP_GEN setpoint generator specifies the setpoint whose slope is limited by the ROC_LIM ramp-function generator. The I/O process value is converted to a floating-point value by means of CRP_IN and monitored for preset limit values by the LIMALARM limit monitor. The control deviation is applied to the PID PID algorithm. The LMNGEN_C manipulated variable processing block generates the LMN analog manipulated variable.

Warm restart

Each individual block is called at a warm restart. Individual blocks with a warm restart routine are called in their warm restart routine.

2.3.3 PROC_P: Process for continuous controller with pulse generator

Application

The PROC_P block simulates a continuous control valve with digital input and a third-order delay process.

The following figure shows the block diagram of PROC_P.

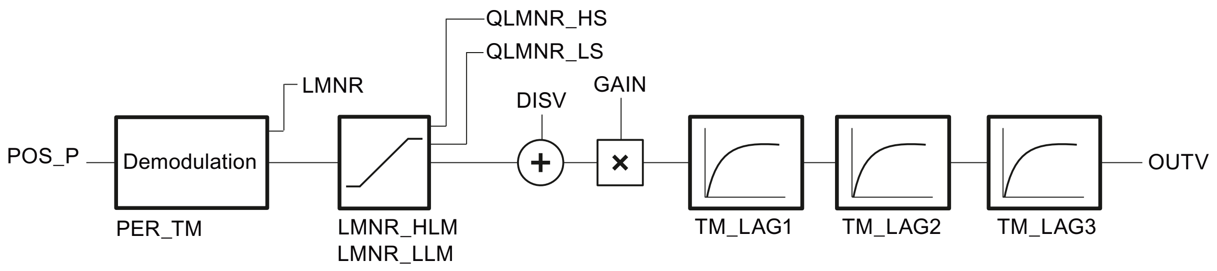


Figure 2-12 Block diagram of PROC_P

Description of functions

The block converts the binary input values of the pulse width modulation into continuous analog values and delays the output signal with three first-order delay elements after the feedforward control.

Warm restart

At a warm restart, the **OUTV** output variable and the internal memory variables are all set to 0.

2.3.4 Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 10 ms. If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX10 function.
Use the TIA Portal to download the example to the CPU.
You can then carry out operator control of the control system using the watch table contained in the example.
Deactivate manual mode by setting DI_PIDCTR.DI_LMNGEN_C.MAN_ON=FALSE.
You can specify a setpoint directly with the DI_PIDCTR.SP_IN parameter.
If you want to the use the setpoint generator instead, set DI_PIDCTR.DI_SP_GEN.DFOUT_ON=FALSE. You can then use the OUTVUP and OUTVDN input parameters of the setpoint generator to continuously increase or decrease the setpoint.
You can monitor the process value with the DI_PIDCTR.PV parameter and the manipulated variable with the DI_PIDCTR.LMN parameter.

2.4 Example11: Single-loop ratio controller

2.4.1 Overview

Control loop

Example11 comprises a single-loop ratio controller and a simulated process.
PROC_C is used as the process simulation block. This is described in Example09.
In this example, the RMP_SOAK time scheduler is used to simulate the time-dependent PV1 process value, which cannot be influenced by the controller. The time schedule for the PV1 process value is stored in the DB_RMPSK global data block. The output value of this time scheduler is then used in conjunction with the PV2 process value of the PROC_C block to generate the PV process value.
The following figure shows the application of Example11 in a complete control loop.

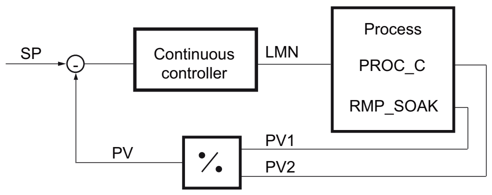


Figure 2-13 Control loop of Example11

Block call and interconnection

The following figure shows the block call of Example11.

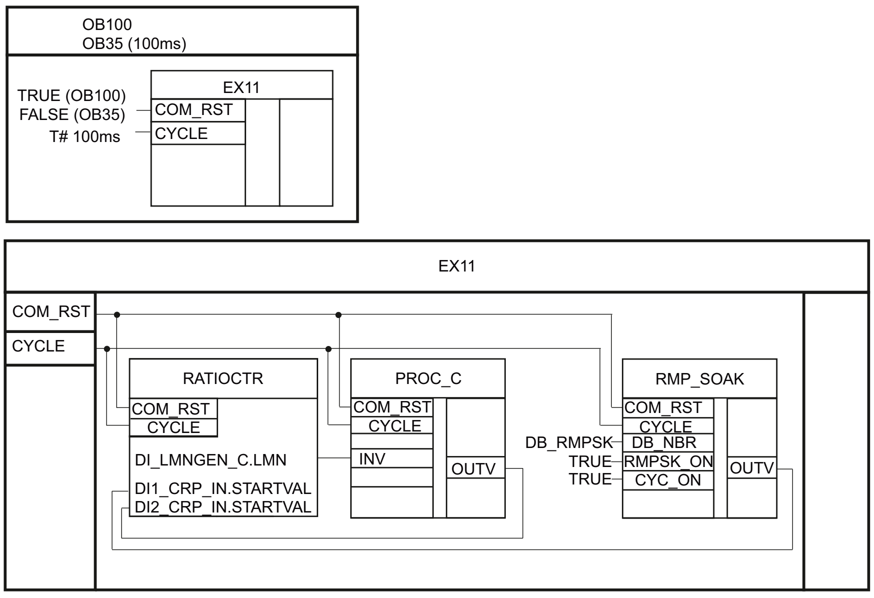


Figure 2-14 Block call and interconnection of Example11

2.4.2 RATIOCTR: Single-loop ratio controller

Application

The RATIOCTR block realizes a single-loop ratio controller for continuous actuators. The following figure shows the block interconnection of RATIOCTR.

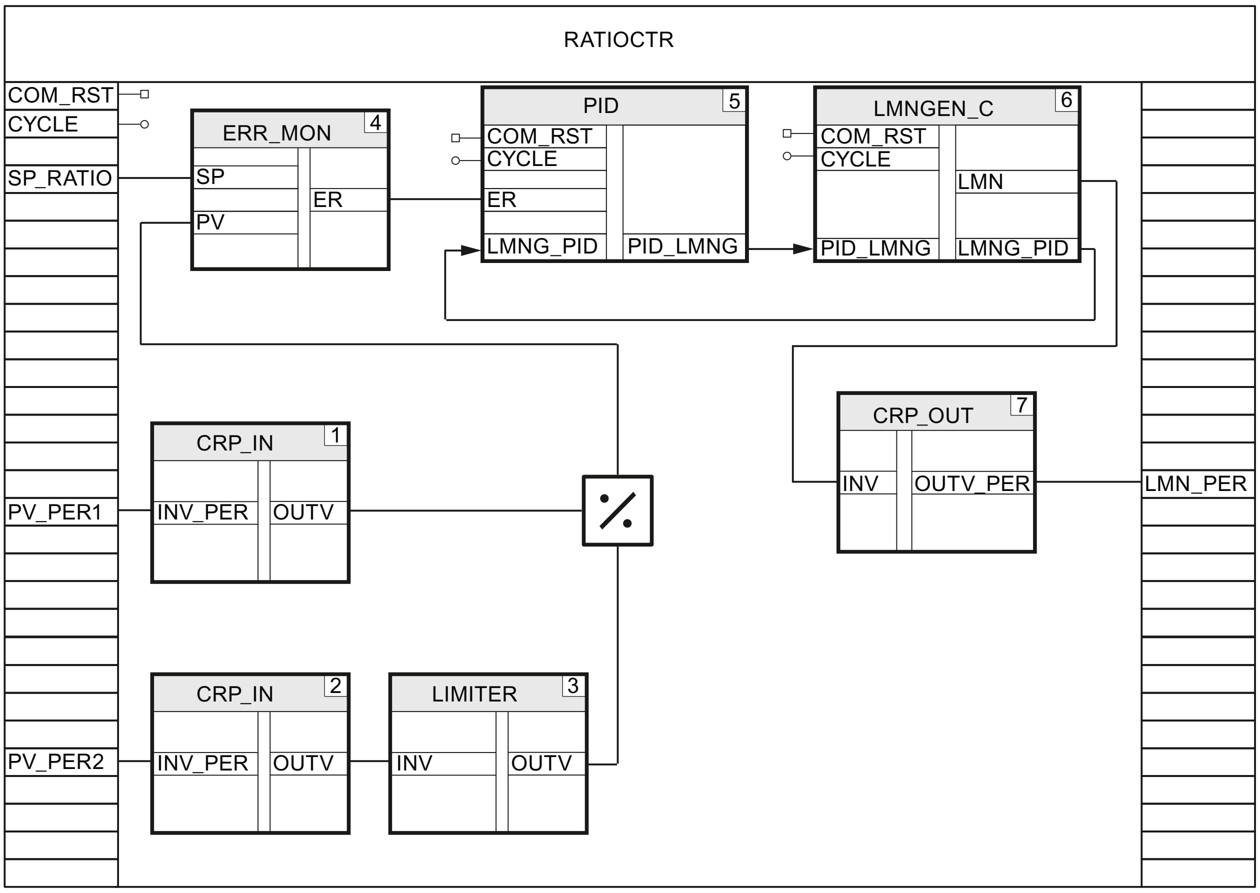


Figure 2-15 Block interconnection of RATIOCTR

Description of functions

The ratio setpoint is defined with the SP_RATIO input parameter. The PV_PER1 and PV_PER2 I/O process values are converted to floating-point values by means of CRP_IN and the ratio is generated. The floating-point value of PV_PER2 is limited by LIMITER so that division by zero is not possible. The control deviation is applied to the PID PID algorithm. The LMNGEN_C manipulated variable processing block generates the LMN analog manipulated variable that is converted to the I/O format with CRP_OUT .

Warm restart

Each individual block is called at a warm restart. Individual blocks with a warm restart routine are called in their warm restart routine.

2.4.3 Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX11 function.
Use the TIA Portal to download the example to the CPU.
You can then carry out operator control of the control system using the watch table contained in the example.
Deactivate manual mode by setting DI_RATIOCTR.DI_LMNGEN_C.MAN_ON = FALSE.
You can specify a ratio setpoint directly with the DI_RATIOCTR.SP_RATIO parameter. It must be between 0.75 and 3.0.
You can use the DI_RATIOCTR.DI_ERR_MON.PV parameter to monitor the ratio of the process value and the DI_RATIOCTR.DI_LMNGEN_C.LMN parameter to monitor the manipulated variable.

2.5 Example12: Blending controller

2.5.1 Overview

Control loop

Example12 demonstrates a blending controller. It consists of a master controller, three slave controllers, and the associated simulated processes.
PROC_C is used as the process simulation block. This is described in Example09.
The PIDCTR block from Example10 is used as the master controller. Its mode of operation is described there.
The following figure shows the application of Example12 in a complete control loop.

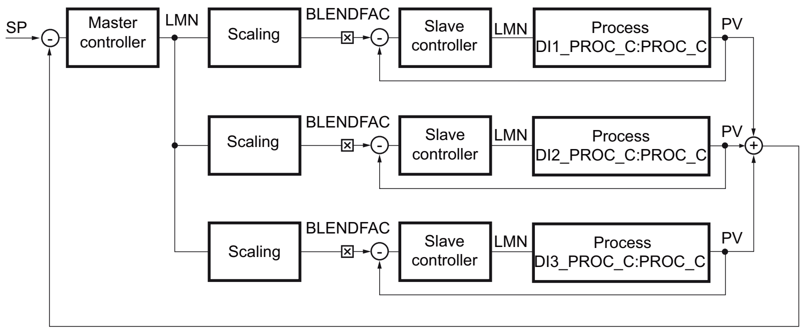


Figure 2-16 Control loop of Example12

Block call and interconnection

The following figure shows the block call and the interconnection of Example12.

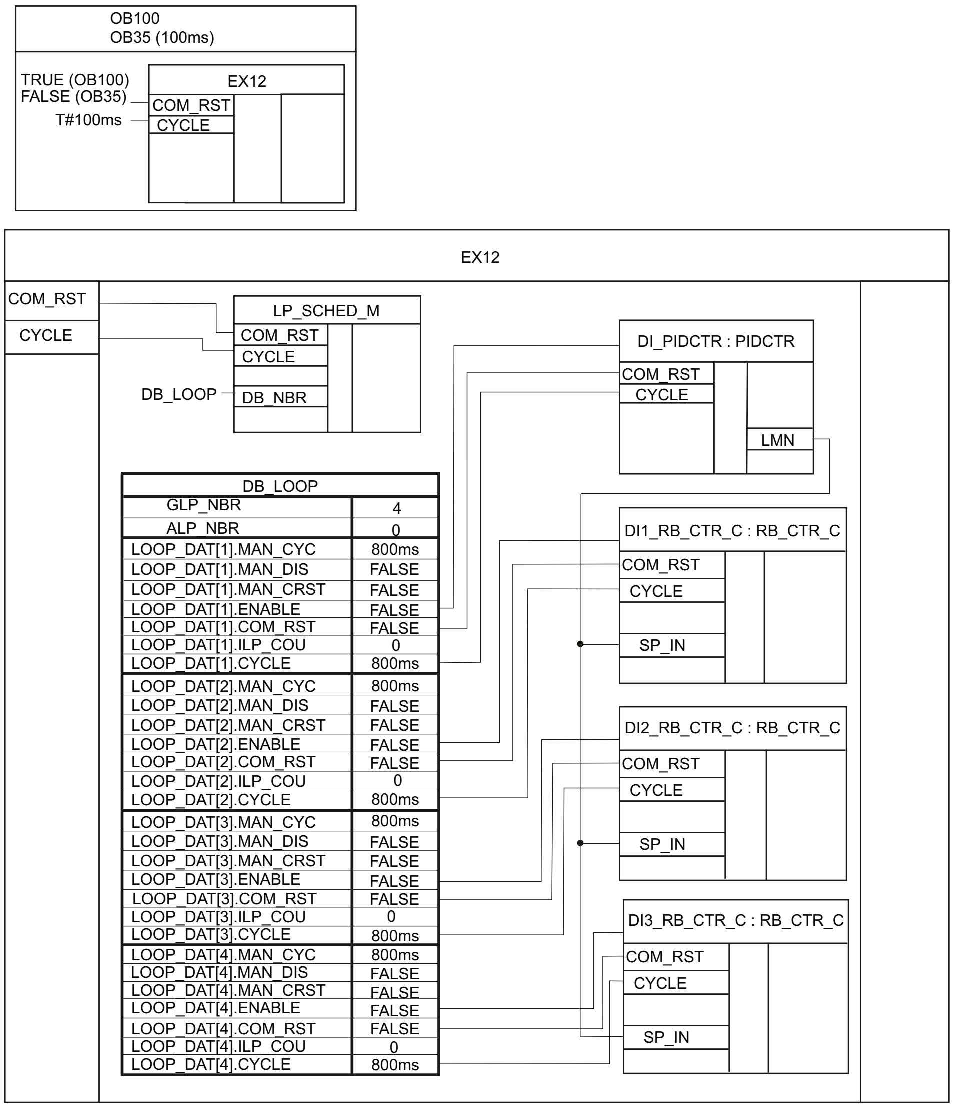


Figure 2-17 Block call and interconnection of Example12 (representation without process simulation blocks)

2.5.2 RB_CTR_C: Continuous slave controller for multi-loop ratio or blending controls

Slave controller

The RB_CTR_C block is used as a slave controller. This block is a continuous PID controller that can be used as a slave controller in a multi-loop ratio or blending control. The following figure shows the block interconnection of RB_CTR_C.

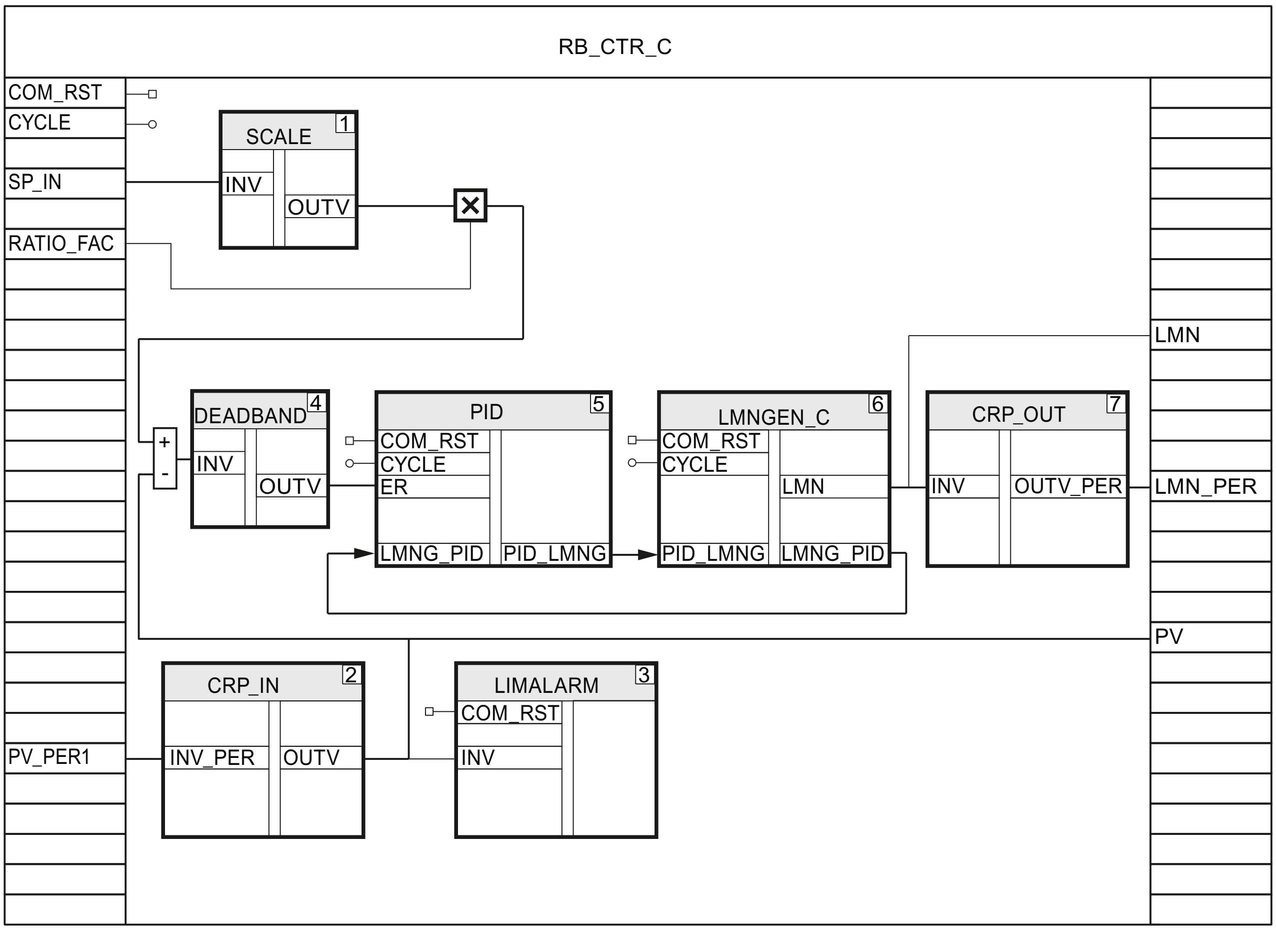


Figure 2-18 Block interconnection of RB_CTR_C

Function description of the slave controller

The mode of operation of the RB_CTRL_C slave controller is similar to that of the PIDCTR_C continuous PID controller from Example09. The input of the slave controller is adapted to the output of the master controller by means of scaling and is multiplied by a preset ratio or blending factor.

Warm restart

Each individual block is called at a warm restart. Individual blocks with a warm restart routine are called in their warm restart routine.

2.5.3 Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX12 function.
Use the TIA Portal to download the example to the CPU.
You can then carry out operator control of the control system using the watch table contained in the example.
Deactivate manual mode by setting DI_PIDCTR.DI_LMNGEN_C.MAN_ON = FALSE.
You can specify a setpoint directly with the DI_PIDCTR.SP_IN parameter.
If you want to the use the setpoint generator instead, set DI_PIDCTR.DI_SP_GEN.DFOUT_ON = FALSE. You can then use the OUTVUP and OUTVDN input parameters of the setpoint generator to continuously increase or decrease the setpoint.
You can monitor the process value with the DI_PIDCTR.PV parameter and the manipulated variable of the master controller with the DI_PIDCTR.LMN parameter.
The blending factors for the slave controllers are stored in the DIx_RB_CTR_C.RATIO_FAC parameters. The sum of all blending factors must equal "1".
You can use the DIx_RB_CTR_C.PV parameters to monitor the process values and the DIx_RB_CTR_C.LMN parameters to monitor the manipulated variables of the slave controllers.

2.6 Example13: Controller with feedforward control

2.6.1 Overview

Control loop

Example13 comprises a controller with feedforward control and a simulated process. PROC_C is used as the process simulation block. This is described in Example09.

The following figure shows the application of Example13 in a complete control loop.

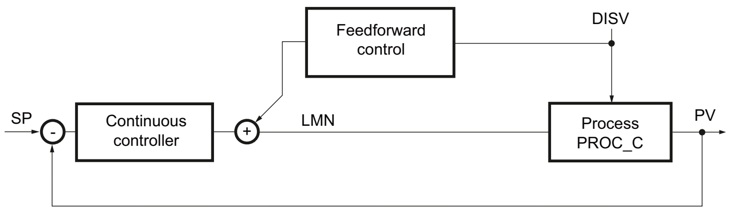


Figure 2-19 Control loop of Example13

Block call and interconnection

The following figure shows the block call of Example13.

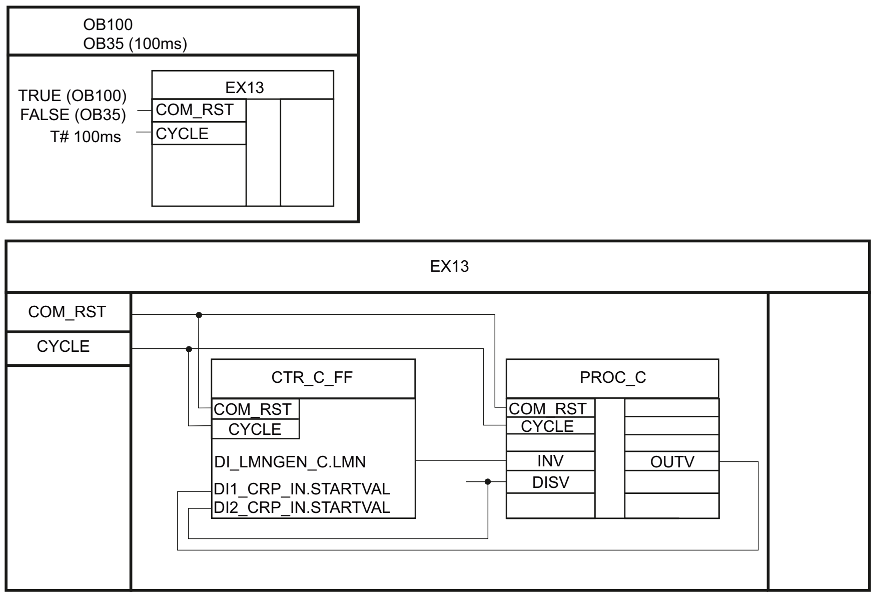


Figure 2-20 Block call and interconnection of Example13

2.6.2 CRT_C_FF: Controller with feedforward control

Application

The CRT_C_FF block is a PID controller with feedforward control for continuous actuators. The following figure shows the block interconnection of CTR_C_FF.

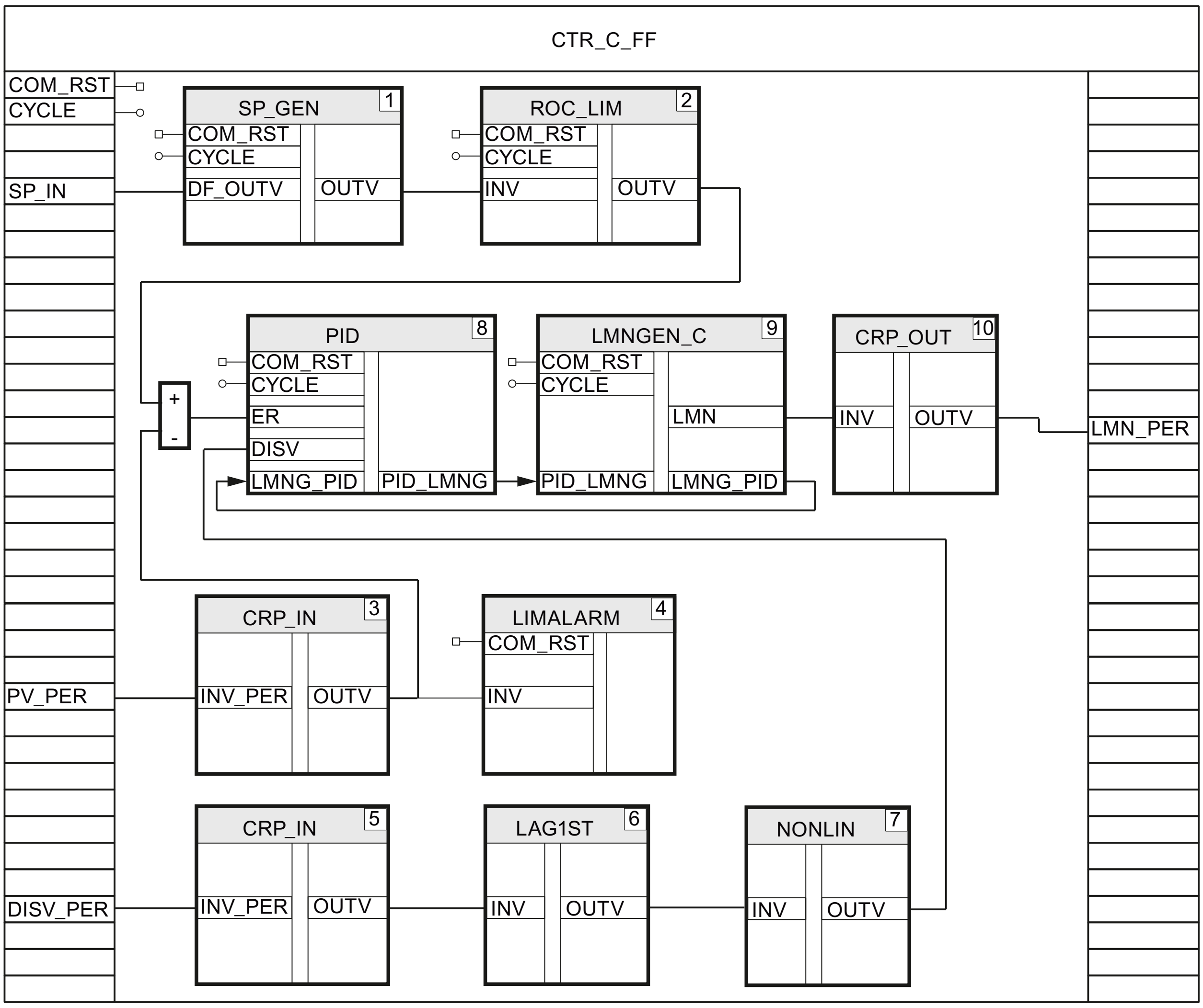


Figure 2-21 Block interconnection of CTR_C_FF

Description of functions

The SP_GEN setpoint generator specifies the setpoint whose slope is limited by the ROC_LIM ramp-function generator. The I/O process value is converted to a floating-point value by means of CRP_IN and monitored for preset limit values by the LIMALARM limit monitor. The control deviation is applied to the PID PID algorithm. The I/O disturbance variable is converted to a floating-point value by means of CRP_IN , smoothened with LAG1ST, and linearized with NONLIN via a characteristic curve. The LMNGEN_C manipulated variable processing block generates the LMN analog manipulated variable that is converted to the I/O format with CRP_OUT .

Warm restart

Each individual block is called at a warm restart. Individual blocks with a warm restart routine are called in their warm restart routine.

2.6.3 Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX13 function.
Use the TIA Portal to download the example to the CPU.
You can then carry out operator control of the control system using the watch table contained in the example.
Deactivate manual mode by setting DI_CTR_C_FF.DI_LMNGEN_C.MAN_ON = FALSE.
You can specify a setpoint directly with the DI_CTR_C_FF.SP_IN parameter.
If you want to the use the setpoint generator instead, set DI_CTR_C_FF.DI_SP_GEN.DFOUT_ON = FALSE. You can then use the OUTVUP and OUTVDN input parameters of the setpoint generator to continuously increase or decrease the setpoint.
You can monitor the process value with the DI_CTR_C_FF.DI1_CRP_IN.OUTV parameter and the manipulated variable with the DI_CTR_C_FF.DI_LMNGEN_C.LMN parameter.
You can inject a disturbance with the DI_PROC_C.DISV input parameter of the simulated process.
You can also use the DI_CTR_C_FF.DI_PID.DISV_SEL input bit of the controller to deactivate the feedforward control by resetting the bit.

2.7 Example14: Range selection controller

2.7.1 Overview

Control loop

Example14comprises a range selection controller and a simulated process.
The following figure shows the application of Example14 in a complete control loop.

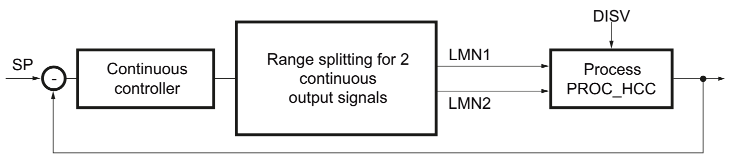


Figure 2-22 Control loop of Example14

Block call and interconnection

The following figure shows the block call of Example14.

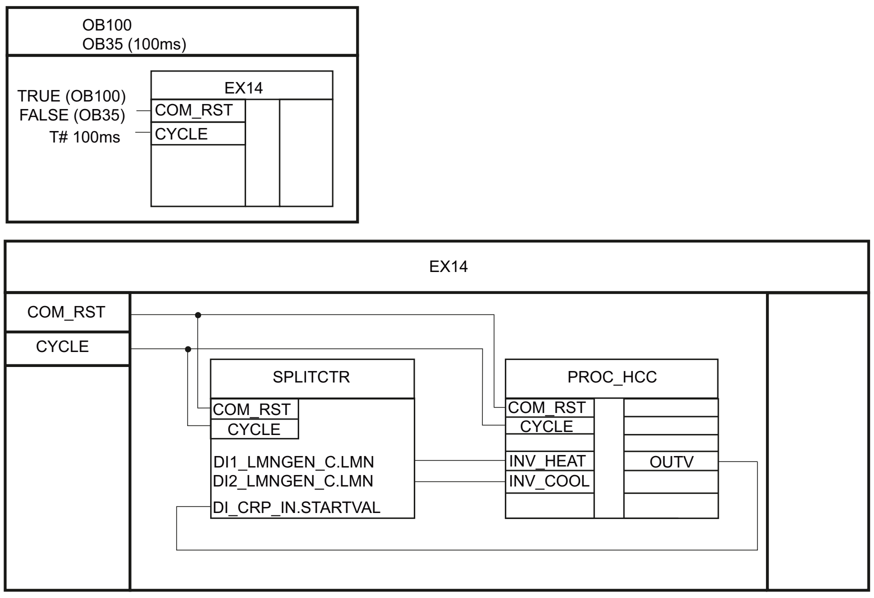


Figure 2-23 Block call and interconnection of Example14

2.7.2 SPLITCTR: Range selection controller

Application

The SPLITCTR block is a PID controller with range selection for 2 continuous actuators. The following figure shows the block interconnection of SPLITCTR.

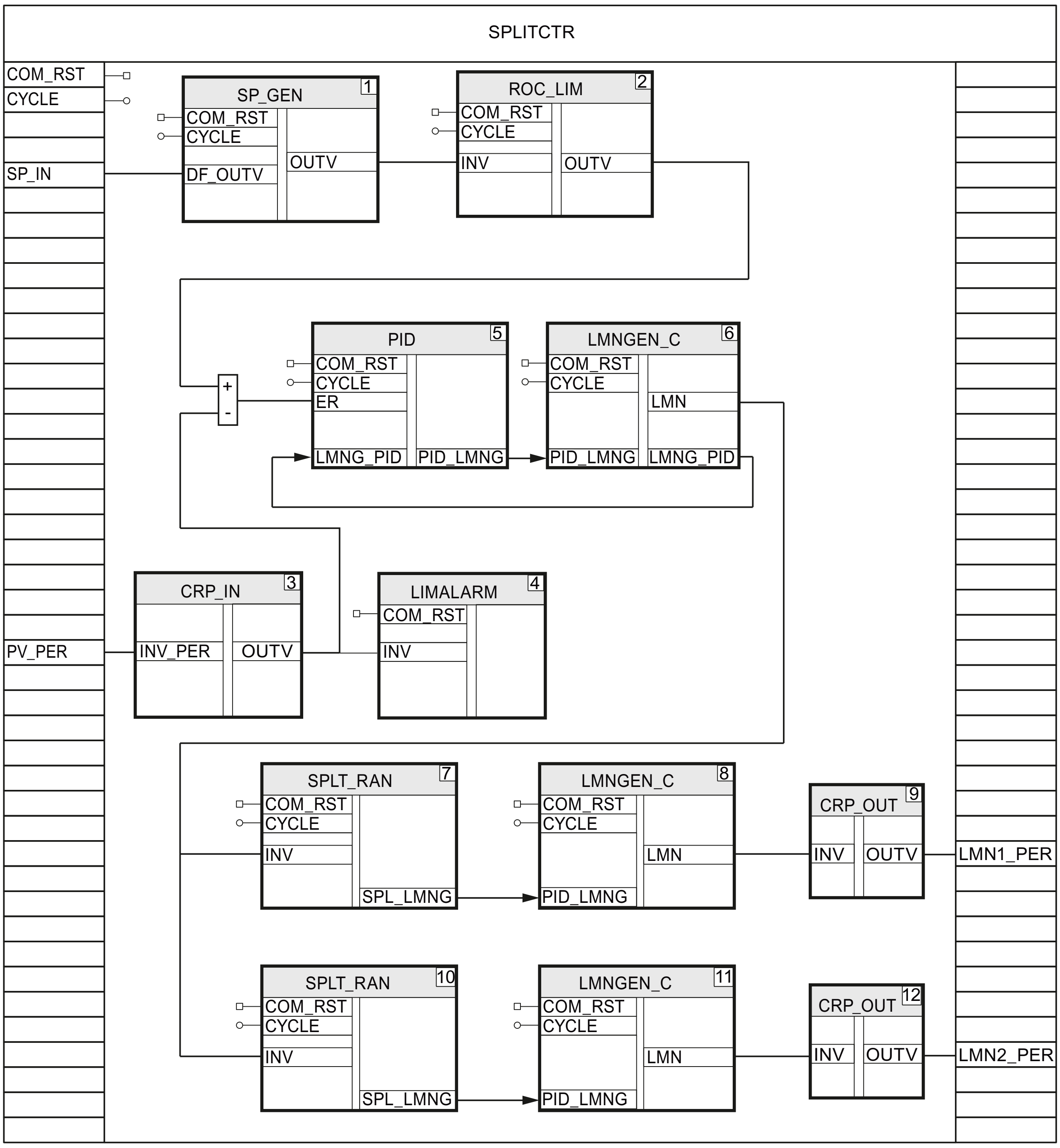


Figure 2-24 Block interconnection of SPLITCTR

Description of functions

The SP_GEN setpoint generator specifies the setpoint whose slope is limited by the ROC_LIM ramp-function generator. The I/O process value is converted to a floating-point value by means of CRP_IN and monitored for preset limit values by the LIMALARM limit monitor. The control deviation is applied to the PID PID algorithm. The LMNGEN_C manipulated variable processing block generates the LMN analog manipulated variable. The manipulated variable range is split into 2 ranges by means of 2 SPLT_RAN blocks. An analog manipulated variable is determined for each range with LMNGEN_C and converted to the I/O format with CRP_OUT.

Warm restart

Each individual block is called at a warm restart. Individual blocks with a warm restart routine are called in their warm restart routine.

2.7.3 PROC_HCC: Process with heating and cooling branches

Application

The PROC_HCC block simulates a third-order delay process. It consists of a heating branch and a cooling branch with one continuous input signal each. For this reason, the block is suitable for simulating temperature processes that have an actuator for heating and an actuator for cooling.
The following figure shows the block diagram of PROC_HCC.

Description of functions

The block generates a series connection of 3 first-order delay elements. The process structure is doubled for realizing the additional cooling branch so that the block has two continuous inputs. The **DISV_H** and **DISV_C** disturbance variables are always added to the INV_HEAT and INV_COOL inputs, respectively.

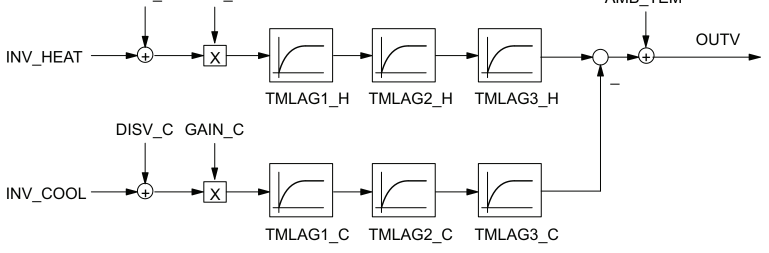


Figure 2-25 Block diagram of PROC_HCC

Warm restart

On warm restart, the output variable is set to
OUTV = (INV_HEAT + DISV_H) * GAIN_H – (INV_COOL + DISV_C) * GAIN_C + AMB_TEM.

2.7.4 Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX14 function.
Use the TIA Portal to download the example to the CPU.
You can then carry out operator control of the control system using the watch table contained in the example.
Deactivate manual mode by setting DI_SPLITCTR.DI_LMNGEN_C.MAN_ON = FALSE.
You can specify a setpoint directly with the DI_SPLITCTR.SP_IN parameter.
If you want to the use the setpoint generator instead, set DI_SPLITCTR.DI_SP_GEN.DFOUT_ON = FALSE. You can then use the OUTVUP and OUTVDN input parameters of the setpoint generator to continuously increase or decrease the setpoint.
You can monitor the process value with the DI_SPLITCTR.DI_CRP_IN.OUTV parameter and the manipulated variable with the DI_SPLITCTR.DI_LMNGEN_C.LMN parameter.
You can use the DI_SPLITCTR.DI1_LMNGEN_C.LMN and DI_SPLITCTR.DI2_LMNGEN_C.LMN parameters to monitor the manipulated variables that result from the range splitting for the continuous actuators.

2.8 Example15: Override controller

2.8.1 Overview

Control loop

Example15 comprises an override controller and a simulated process.
PROC_S and PROC_C are used as the process simulation blocks. They are described in Example08 and Example09.
In the example, the override controller is parameterized as a limit controller. For this purpose, the controller with the smaller manipulated variable is used to generate the actuation signals.
The DI_OVR_CTR.DI1_PID controller is active in normal operation and controls the PV1 process value from PROC_C to the SP1 setpoint.
The DI_OVR_CTR.DI2_PID controller only intervenes if the PV2 process value from PROC_S approaches the SP2 limit value (default 50.0) too closely.
The following figure shows the application of Example15 in a complete control loop.

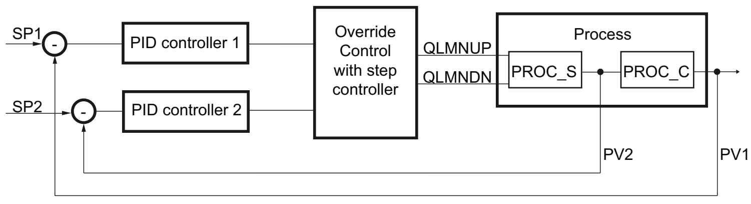


Figure 2-26 Control loop of Example15

Block call and interconnection

The following figure shows the block call and the interconnection of Example15.

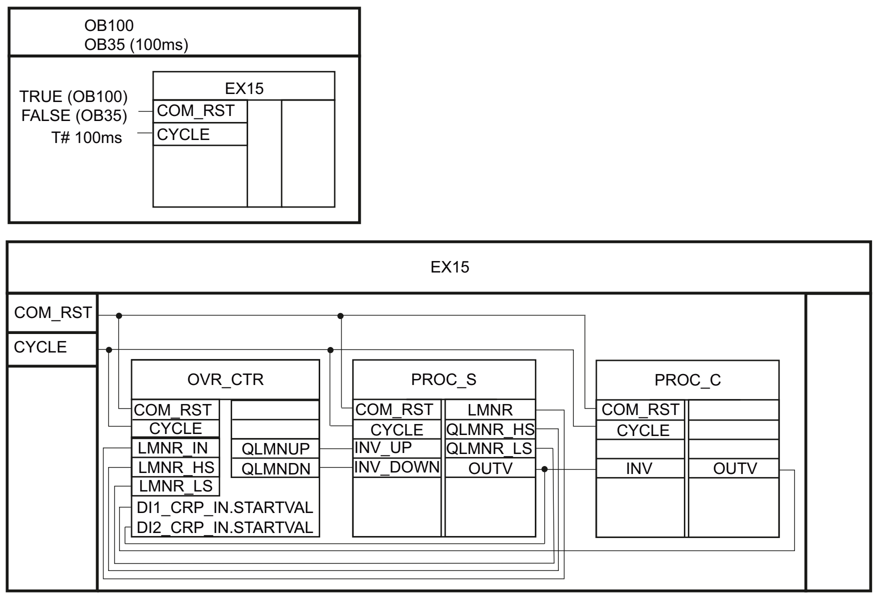


Figure 2-27 Block call and interconnection of Example15

2.8.2 OVR_CTR: Override controller

Application

The OVR_CTR block is an override controller. Two PID controllers are connected to a step controller output. The following figure shows the block interconnection of OVR_CTR.

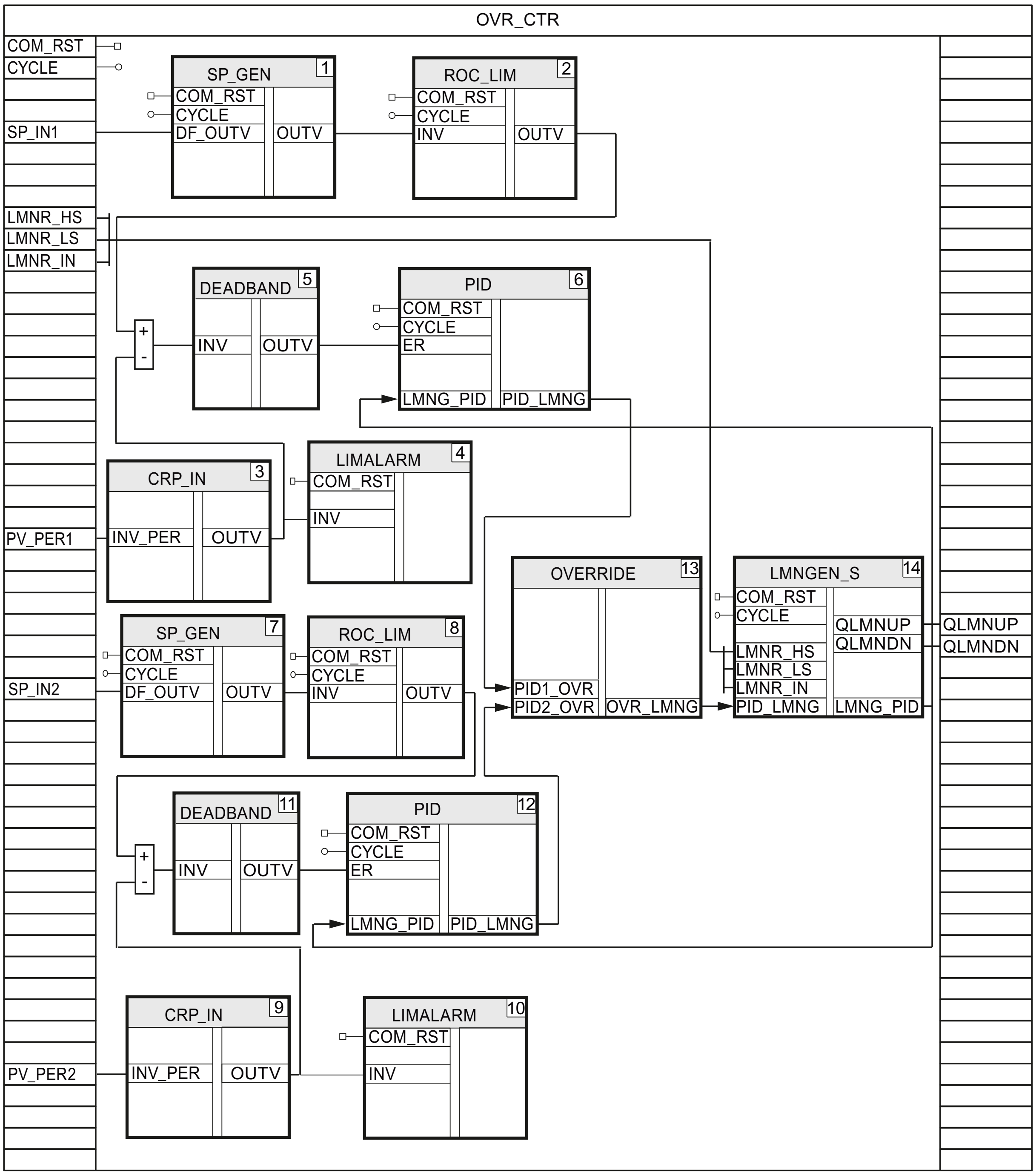


Figure 2-28 Block interconnection of OVR_CTR

Description of functions

The SP_GEN setpoint generators specify the setpoints whose slopes are limited by the ROC_LIM ramp-function generators. The I/O process values are converted to floating-point values by means of the CRP_IN instruction and monitored for preset limit values by the LIMALARM limit monitors. The control deviations are applied to the PID blocks. The manipulated variables of the two PID blocks are suppled to an OVERRIDE block. Here, either the maximum or the minimum of the two manipulated variables is determined and forwarded to the LMNGEN_S manipulated variable processing block. LMNGEN_S can only function in "Step controller with position feedback" mode in the override controller.

Warm restart

Each individual block is called at a warm restart. Individual blocks with a warm restart routine are called in their warm restart routine.

2.8.3 Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the CYCLE sampling time of the EX15 function.

Use the TIA Portal to download the example to the CPU.

You can then carry out operator control of the control system using the watch table contained in the example.

Deactivate manual mode by setting DI_OVR_CTR.DI_LMNGEN_S.MAN_ON = FALSE.

First specify a low SP1 setpoint (e.g., 28.0) for the DI_OVR_CTR.SP_IN1 parameter, and wait until the control has responded.

Then specify a high setpoint (e.g., 92.0) for the DI_OVR_CTR.SP_IN1 parameter.

From the DI_OVR_CTR.DI_OVERRIDE.QPID1 and DI_OVR_CTR.DI_OVERRIDE.QPID2 parameters, you can recognize that the second controller takes over the actuation signal generation for a limited time when this setpoint step change occurs. This prevents process value PV2 DI_OVR_CTR.DI2_CRP_IN.OUTV from exceeding its limit value SP2 DI_OVR_CTR.SP_IN2 (default 50.0).

You can also use the DI_OVR_CTR.DI_OVERRIDE.PID1_ON and DI_OVR_CTR.DI_OVERRIDE.PID2_ON input bits to manually specify which controller is to be used to generate the actuation signal.

You can monitor the PV1 process value with the DI_OVR_CTR.DI1_CRP_IN.OUTV parameter and the actuation signals with the DI_OVR_CTR.QLMNUP and DI_OVR_CTR.QLMNDN parameters.

2.9 Example16: Optimize continuous controller with PID Self-Tuner

Overview

This example contains a simple control loop that consists of a PID controller and a Delay time 2 element as a model controlled system for a temperature process. It is an example of the application of the Self-Tuner with a continuous PID controller.

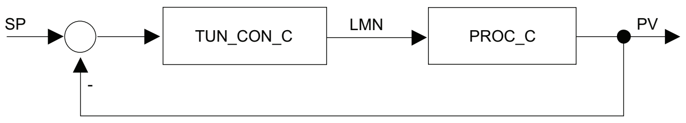


Figure 2-29 Control loop of Example16

Program structure

The IDB_TUN_CON_C block is an instance DB of the TUN_CON_C FB that contains the calls of the Self-Tuner and controller. The TUN_CON_C FB and the PROC_C process simulation FB are called in OB 35.

Process block for simulating a temperature process

The block simulates a process with a third-order delay element. For temperature processes, you choose Delay time 2 response with a large and a small time constant (TM_LAG1 = 15 × TM_LAG2 and TM_LAG3 = 0s).

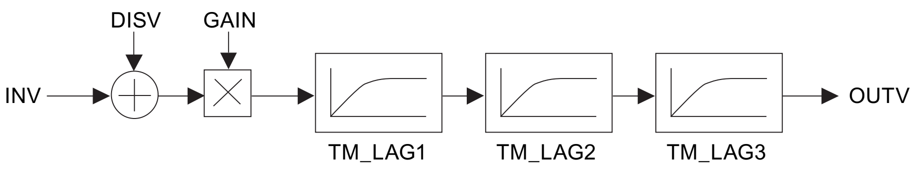


Figure 2-30 Block diagram of the process

Parameters

INV	Input variable (manipulated variable of controller)
DISV	Disturbance variable
GAIN	Process gain
TM_LAG1	Delay time 1 (for temperature processes)
TM_LAG2	Delay time 2 (TM_LAG1 = 15 × TM_LAG2)
TM_LAG3	Delay time 3 (= 0 for temperature processes)
OUTV	Output variable (e.g., temperature)

After addition of the analog input signal and a disturbance variable and multiplication with the process gain, three first-order delay elements are run through.

During initialization, the output variable is set to OUTV = DISV × GAIN.

Operator control and monitoring

Operator control for the example is via the commissioning interface of the Self-Tuner, which is described in the online help for the Self-Tuner. The commissioning interface can be found in the "Technology objects" folder for the "tuner" technology object.

Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 100 ms (default setting). If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the IDB_TUN_CON_C.CYCLE_P sampling time.
Use the TIA Portal to download the example to the CPU.
Open the commissioning interface of the "tuner" technology object in the "Technology objects" folder, and perform the commissioning as described in the online help.

2.10 Example17: Optimize step controller with PID Self-Tuner

Overview

This example contains a simple control loop that consists of a PID controller and a Delay time 2 element with an actuator with integrating behavior as a model controlled system for a temperature process. It is an example of the application of the Self-Tuner with a PID step controller.

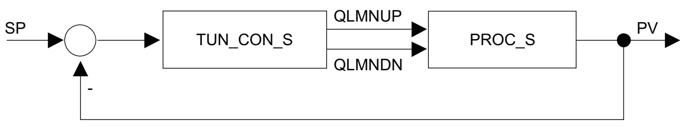


Figure 2-31 Control loop of Example17

Program structure

The IDB_TUN_CON_S block is an instance DB of the TUN_CON_S FB that contains the calls of the Self-Tuner and controller. The TUN_CON_S FB and the PROC_S process simulation FB are called in OB 35.

Process block for simulating a temperature process

The block simulates a process with a third-order delay element. For temperature processes, you choose Delay time 2 response with a large and a small time constant (TM_LAG1 = 15 × TM_LAG2 and TM_LAG3 = 0 s).

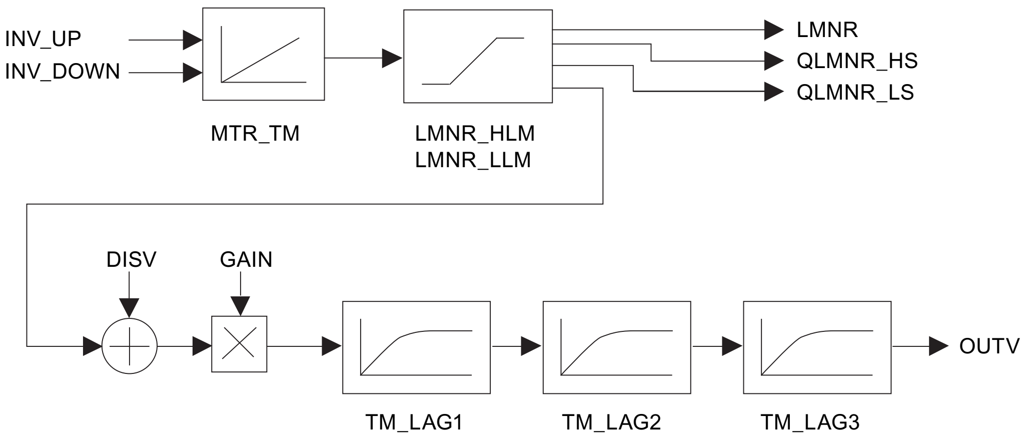


Figure 2-32 Block diagram of the process

Parameters

INV_UP	Manipulated variable signal up
INV_DOWN	Manipulated variable signal down
DISV	Disturbance variable
GAIN	Process gain
MTR_TM	Motor transition time
LMNR_HLM	High limit of actuator
LMNR_LLM	Low limit of actuator
TM_LAG1	Delay time 1
TM_LAG2	Delay time 2 (for temperature processes: TM_LAG1 = 15 × TM_LAG2)
TM_LAG3	Delay time 3 (= 0 for temperature processes)
OUTV	Output variable (e.g., temperature)
LMNR	Position feedback
QLMNR_HS	High limit of endstop signal
QLMNR_LS	Low limit of endstop signal

Depending on the INV_UP and INV_DOWN input signals, the LMNR position feedback is calculated using an integrator. The position feedback is limited to LMNR_HLM and LMNR_LLM. When the limit is reached, the QLMNR_HS and QLMNR_LS endstop signals are set.

After addition of a disturbance variable and subsequent multiplication with the process gain, three first-order delay elements are run through.

During initialization, the OUTV and LMNR output variables are set to zero.

Operator control and monitoring

Operator control for the example is via the commissioning interface of the Self-Tuner, which is described in the online help for the Self-Tuner. The commissioning interface can be found in the "Technology objects" folder for the "tuner" technology object.

Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 20 ms. If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the IDB_TUN_CON_S.CYCLE_P sampling time.

Use the TIA Portal to download the example to the CPU.

Open the commissioning interface of the "tuner" technology object in the "Technology objects" folder, and perform the commissioning as described in the online help.

2.11 Example18: Optimize pulse controller with PID Self-Tuner

Overview

This example contains a simple control loop that consists of a PID controller with pulse generation and a Delay time 2 element as a model process for a temperature process. It is an example of the application of the Self-Tuner with a continuous PID controller with pulse generation.

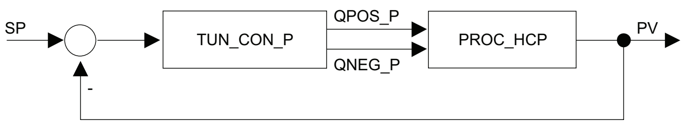


Figure 2-33 Control loop of Example18

Program structure

The IDB_TUN_CON_P block is an instance DB of the TUN_CON_P FB that contains the calls of the Self-Tuner and the controller. The TUN_CON_P FB and the PROC_HCP process simulation FB are called in OB 35.

Process block for simulating a temperature heating zone

The block simulates a typical temperature process for heating and cooling as it can occur as a control zone in a extruder, an injection molding machine, a tempering machine or as a separate furnace in reality.

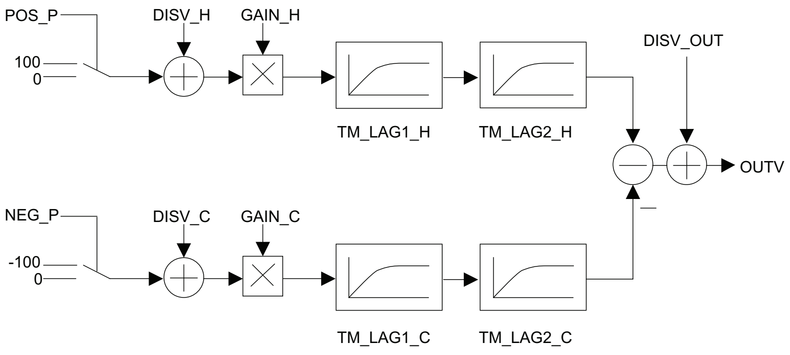


Figure 2-34 Block diagram of the process

Parameters

POS_P	Binary input signal heating
NEG_P	Binary input signal cooling
DISV_H	Disturbance variable heating
DISV_C	Disturbance variable cooling
DISV_OUT	Disturbance variable at output (e.g., ambient temperature)
GAIN_H	Process gain heating
GAIN_C	Process gain cooling
TM_LAG1_H	Heating delay time 1
TM_LAG2_H	Heating Delay time 2
TM_LAG1_C	Cooling delay time 1
TM_LAG2_C	Cooling Delay time 2
OUTV	Output variable (e.g., control zone temperature)

The binary input signals are converted to continuous floating-point values (-100, 0, 100). After addition of a disturbance variable (for example, ambient temperature) and multiplication with the process gain, the two first-order delay elements are run through. This procedure is performed separately in the heating and cooling processes.

During initialization, the output variable is set to
OUTV = DISV_H × GAIN_H - DISV_C × GAIN_C + DISV_OUT.

Operator control and monitoring

Operator control for the example is via the commissioning interface of the Self-Tuner, which is described in the online help for the Self-Tuner. The commissioning interface can be found in the "Technology objects" folder for the "tuner" technology object.

Putting the example into operation

Use the "device configuration" to adapt the configuration of the hardware structure from the example to your real plant. Choose the appropriate CPU (right-click "Unspecified CPU" -> "Change device"), and set the cycle time of OB 35 to 20 ms. If a time-out occurs in the cyclic interrupt level, you must increase the cycle time. In this case the simulation is then executed more slowly. If you are controlling a real process, the cycle time of OB 35 must agree with the IDB_TUN_CON_P.CYCLE_P sampling time.
Use the TIA Portal to download the example to the CPU.
Open the commissioning interface of the "tuner" technology object in the "Technology objects" folder, and perform the commissioning as described in the online help.