WinAC Target for MATLAB/Simulink:

Integrating and calling Simulink models using STEP 7 and WinAC ODK using PID control as an example

WinAC RTX, STEP 7, WinAC ODK

Application • July 2013

Applications & Tools

Answers for industry.

SIEMENS

Siemens Industry Online Support

This article is taken from the Siemens Industry Online Support. The following link takes you directly to the download page of this document:

http://support.automation.siemens.com/WW/view/en/56969417

Caution:
The functions and solutions described in this article confine themselves to the realization of the automation task predominantly. Please take into account furthermore that corresponding protective measures have to be taken up in the context of Industrial Security when connecting your equipment to other parts of the plant, the enterprise network or the Internet. Further information can be found under the Entry ID 50203404.

http://support.automation.siemens.com/WW/view/en/50203404

SIEMENS

SIMATIC
WinAC Target for MATLAB/Simulink

Integrating and calling Simulink models using STEP 7 and WinAC ODK using PID control as an example

Warranty and liability

Note

The Application Examples are not binding and do not claim to be complete regarding the circuits shown, equipping and any eventuality. The Application Examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible for ensuring that the described products are used correctly. These application examples do not relieve you of the responsibility to use safe practices in application, installation, operation and maintenance. When using these Application Examples, you recognize that we cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these Application Examples at any time without prior notice.
If there are any deviations between the recommendations provided in these application examples and other Siemens publications – e.g. Catalogs – the contents of the other documents have priority.

We do not accept any liability for the information contained in this document.

Any claims against us – based on whatever legal reason – resulting from the use of the examples, information, programs, engineering and performance data etc., described in this Application Example shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act (“Produkthaftungsgesetz”), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract (“wesentliche Vertragspflichten”). The damages for a breach of a substantial contractual obligation are, however, limited to the foreseeable damage, typical for the type of contract, except in the event of intent or gross negligence or injury to life, body or health. The above provisions do not imply a change of the burden of proof to your detriment.

Preface

Objective of the Application

MATLAB/Simulink from MathWorks is software for modeling and simulating systems. Simulink provides a graphical editor that allows the user to create realistic processes and develop solutions, for example in control engineering. The Embedded Coder add-on is capable of compiling Simulink models directly into C/C++ code. WinAC ODK makes it possible to run C/C++ code in the Windows or real-time environment of WinAC RTX.

WinAC Target supports the user, allowing easy integration of Simulink models in STEP 7.

Main contents of this application note

This application discusses the following main points:

- Creating a simple block (PID controller) using Simulink
- Compiling into a DLL/RTDLL file and SCL source using WinAC Target
- Integrating into a STEP 7 V5.X and V1x project
- Monitoring a model online using Simulink and changing parameters during operation
- Changing model parameters with the STEP 7 program

Validity

This application does not include an in-depth description of:

- Control engineering basics
- MATLAB/Simulink and Embedded Coder
- WinAC RTX and WinAC ODK
- IntervalZero
- C/C++ programming
- STEP 7 programming

Basic knowledge of these topics is required.

Table of Contents

1 Automation task

Requirement

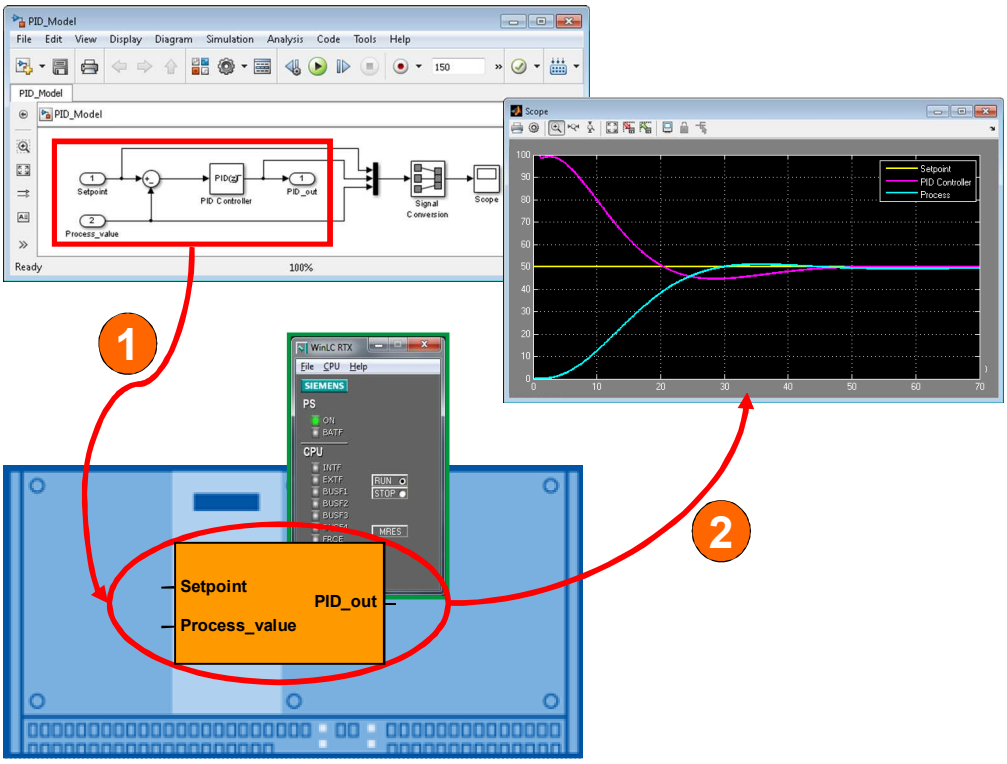
In automation and control engineering, MATLAB/Simulink is frequently used to simulate processes and create PID controllers or more complex algorithms. The requirement is to run algorithms, functions or systems created in a few steps on a programmable logic controller (PLC).

MathWorks provides the option to compile the Simulink models into C/C++ code using Embedded Coder. This code, in turn, can be integrated into a WinAC ODK project and called and run from the STEP 7 program via the DLL/RTDLL.

General task definition

1. Models created in Simulink are to be ported to WinAC RTX and run.
2. External mode allows you to monitor the model using Simulink and change parameters during operation.

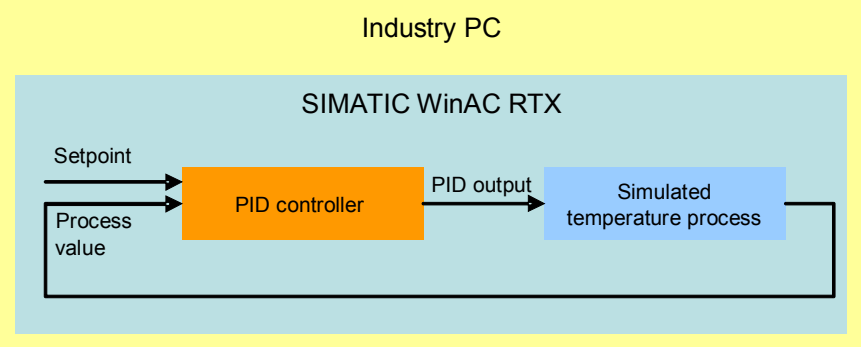
Figure 1-1: Simulink models on WinAC RTX



Description of the automation task

The sample application shows you how to port models created with Simulink to WinAC RTX. The application is a simple temperature control application with a simulated plant (temperature process).

Figure 1-2: Automation task: PID closed-loop control



The PID controller is created using Simulink and ported to WinAC RTX. The temperature process is simulated by a function block.

Simulink allows you to monitor the interfaces of the model and change parameters.

Note The application example is deliberately kept simple to focus on the description of the use of WinAC Target and External mode.

Learning contents of this application

After studying this application, you will have learned the following:

- Examples of simulating processes and creating a controller in Simulink
- Principle of operation of WinAC Target
- Encoding of a Simulink model for porting to WinAC RTX
- Principle of operation of External mode with MATALB/Simulink

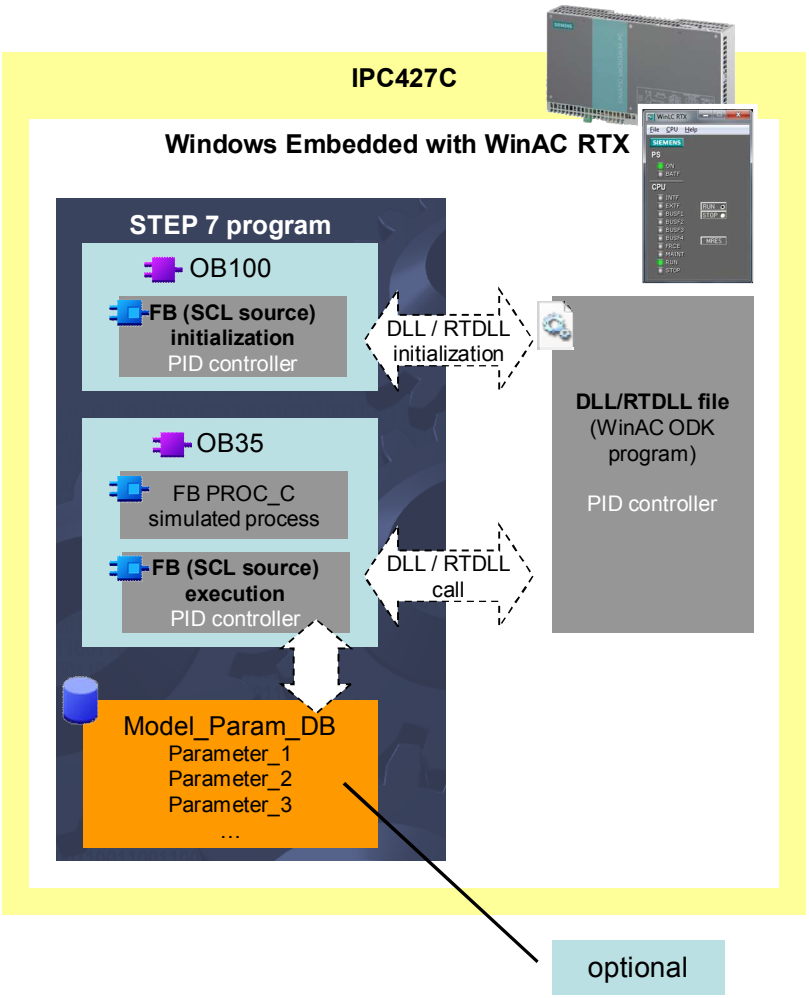
2 Automation solution

2.1 Overview of the overall solution

This application serves as an example of temperature control. The process is simulated using the “PROC_C” function block in the user program. The PID controller created in Simulink is called via the ODK interface in the form of a DLL/RTDLL. The process and the PID controller are called in the cyclic OB35. The DLL/RTDLL file is initialized in OB100.

In this application, the runtime system is a SIMATIC IPC427C.

Figure 2-1: Overview of the overall solution



The “Model_Param_DB” data block allows you to optionally access the internal parameters of the Simulink model. This enables you to manipulate these parameters during operation with the STEP 7 user program.

2.1.1 WinAC Target

Porting Simulink models to WinAC RTX

Figure 2-2 and Table 2-1 show the simple process for creating a model in Simulink in order to run it in WinAC RTX using WinAC Target. The gray area shows the engineering components.

Figure 2-2: Porting Simulink models to WinAC RTX

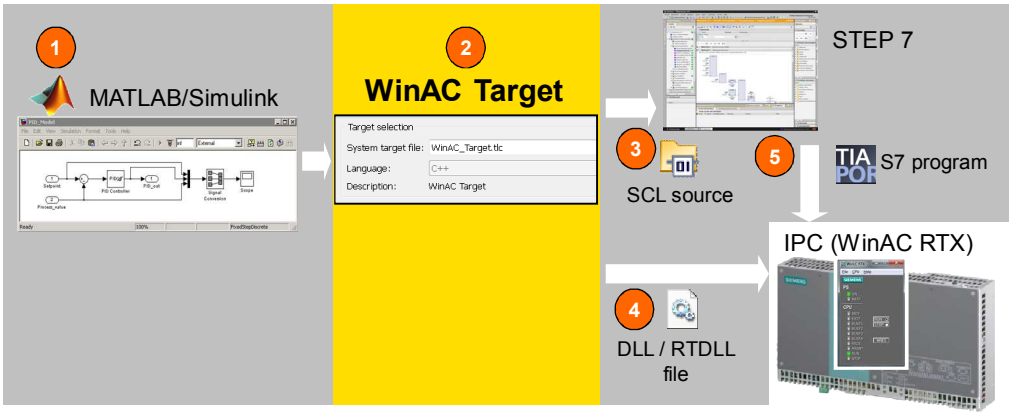
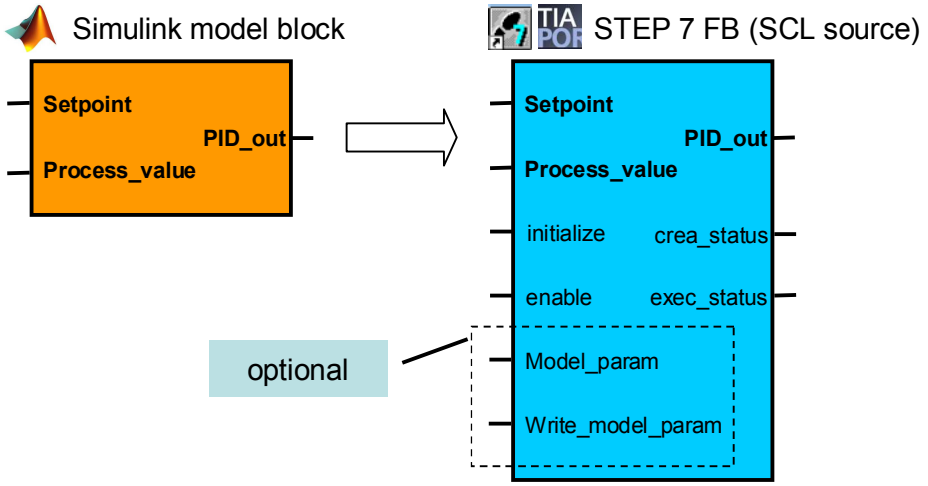


Table 2-1: Instructions

Step	Action	Explanation
1	Create a model in Simulink	In this case: A PID controller
2	Compile the model using WinAC Target	An SCL source and a DLL/RTDLL file will be generated.
3	Integrate the SCL source into the S7 program	The source is manually integrated into STEP 7 V5.5 and STEP 7 V11 using “External source files” in the Project tree.
4	Copy the DLL/RTDLL file to the PC-based controller	The DLL/RTDLL file must be manually copied to the PC system. The path is defined in the SCL source (default path: C:\).
5	Download the S7 program to WinAC RTX	-

When the SCL source is generated, the interfaces of the Simulink model are applied. If a function block is created from the SCL source in the S7 program, it has the same interfaces as the Simulink model. The block is provided with additional parameters (initialize, enable, crea_status, exec_status) for WinAC ODK communication. The “Model_param” and “Write_model_param” inputs will only be additionally created when the “Parameter access with STEP 7” setting is checked in WinAC Target.

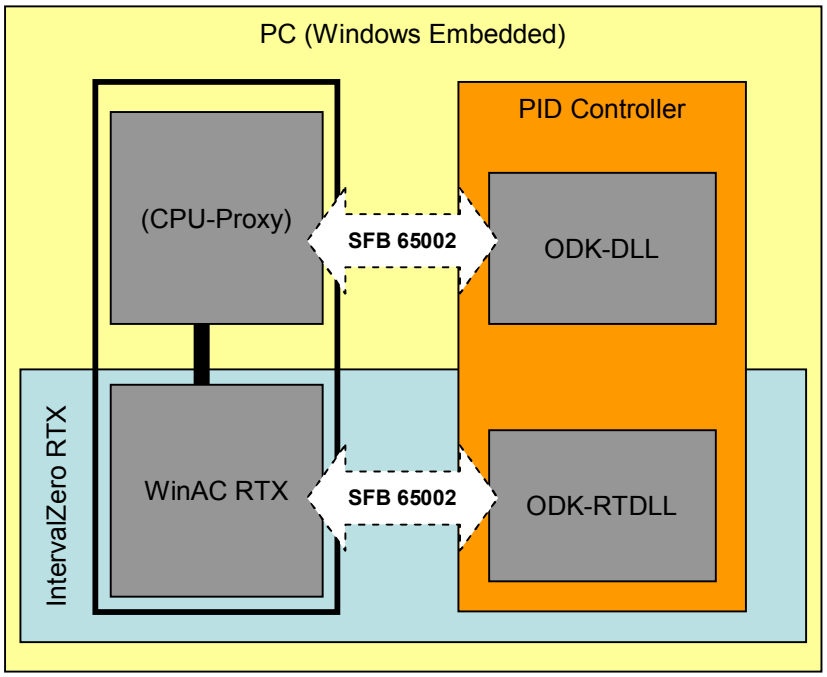
Figure 2-3: Simulink model block and STEP 7 function block



WinAC ODK interface with synchronous call

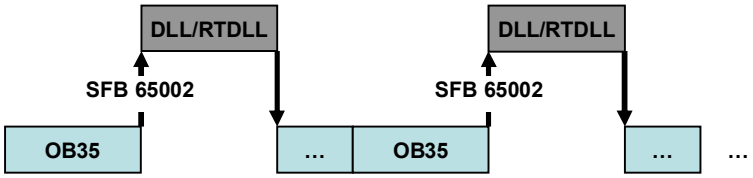
The WinAC ODK interface is used to call the PID controller as a DLL/RTDLL file.

Figure 2-4: Schematic diagram of the WinAC ODK functionality



The ODK CCX (Custom Code Extension) interface is used to directly call the DLL/RTDLL from the STEP 7 program. The call is synchronous using SFB65002 (EXEC_COM).

Figure 2-5: Synchronous call using SFB65002



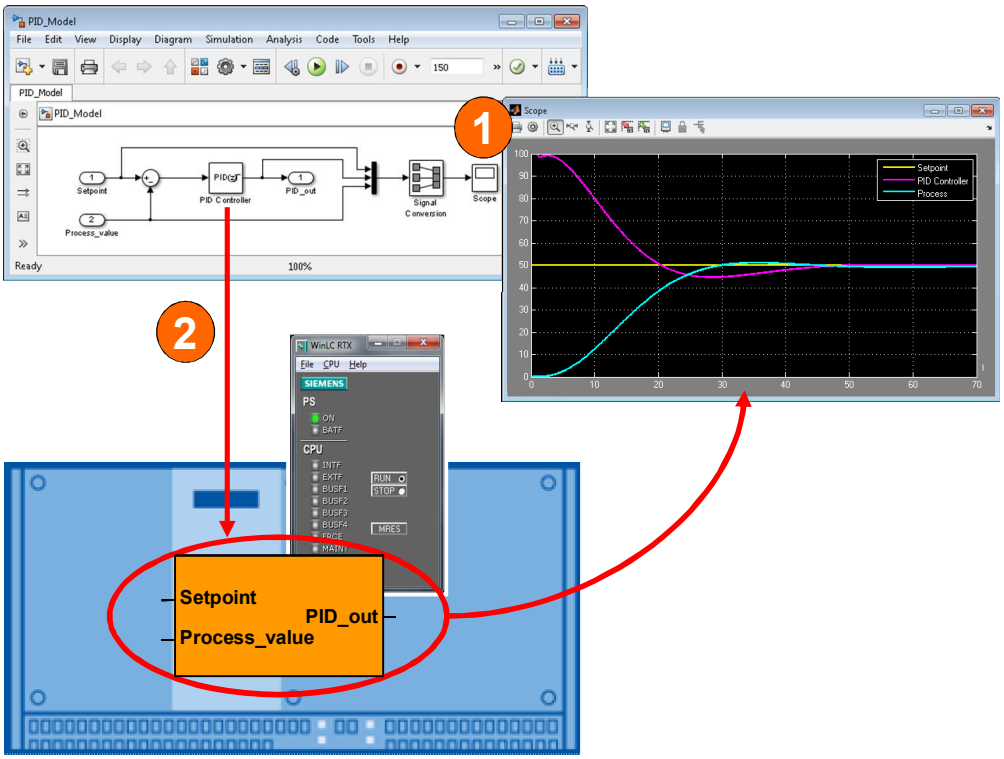
For more information on WinAC ODK, please refer to the following manual.
http://support.automation.siemens.com/WW/view/en/19256518

2.1.2 External mode

Simulink External mode provides you with the option to monitor the ported model during operation on the runtime system (here: WinAC RTX on IPC427C) and change block parameters online.

In this case, the External mode interface must be enabled and configured in the code generation settings before porting the model to WinAC RTX.

Figure 2-6: Using External mode



(1) Monitoring

The "**Scope**" block allows you to graphically display signals in Simulink. In this case, these are the Setpoint, Process_value and PID_out signals.

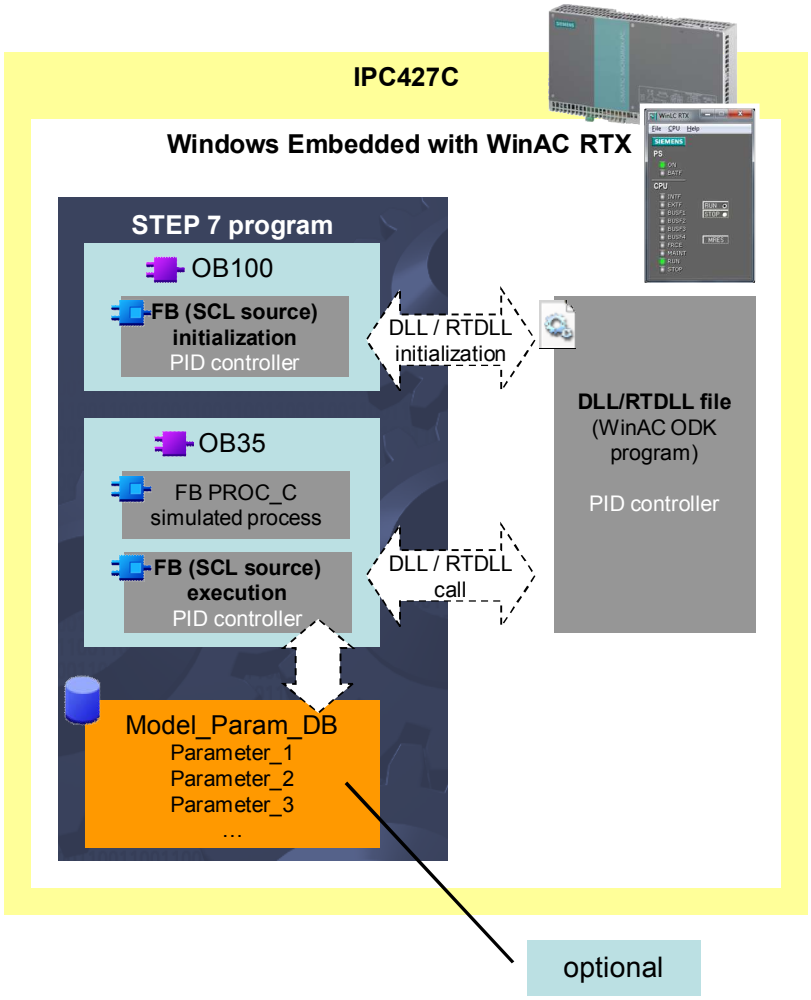
(2) Changing parameters

Some Simulink blocks (here: "PID Controller") have internal parameters such as P, I, D, etc. You can double-click to open the block screen form and change the parameters during operation.

2.1.3 Manipulating model parameters using STEP 7

In some cases, it is necessary to access the internal parameters of the Simulink models during operation. For example, if parameters have to be adjusted to the running process. Optionally, an additional global data block, “Model_Param_DB”, can be created with all parameters of the Simulink model when encoding the SCL source and the DLL/RTDLL file. This data block can be used to manipulate all parameters via the user program in STEP 7.

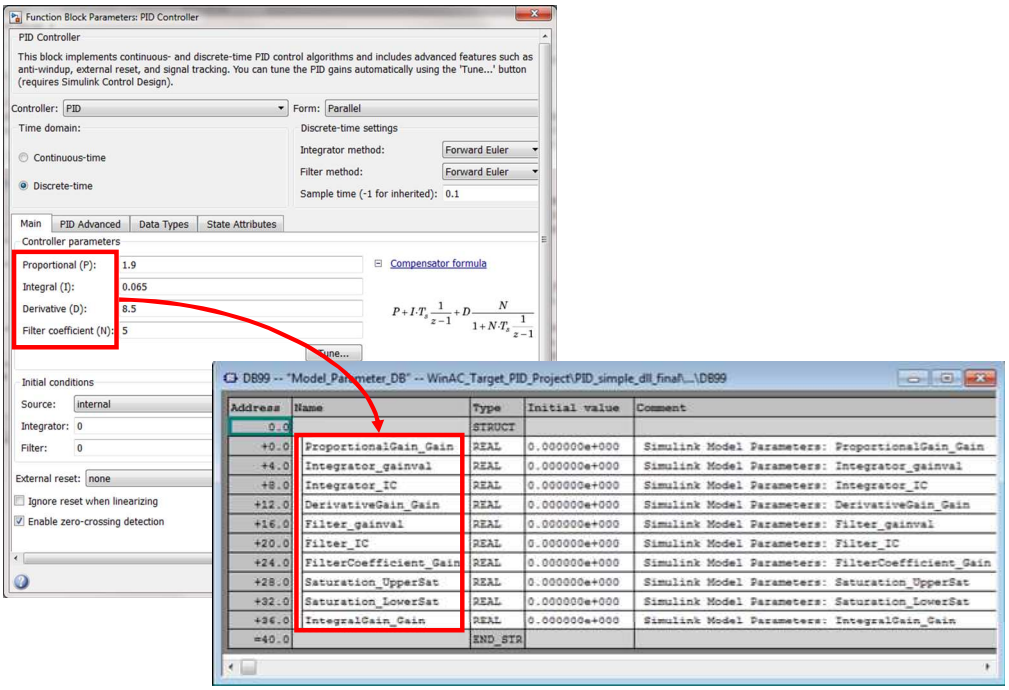
Figure 2-7: Overview of the overall solution



Note If you want to manipulate the model parameters using STEP 7, you have to enable the “Parameter access with STEP 7” parameter before encoding (see Chapter 10 Manipulating model parameters using STEP 7).

The figure below shows the internal parameters of “PID Controller” in Simulink and the relevant “Model_Param_DB” data block.

Figure 2-8: Model parameters in Simulink and in the data block in STEP 7



2.2 Hardware and software components used

The application was created with the following components:

Hardware components

Table 2-2: SIMATIC components

Component	Qty.	Order no.	Note
SIMATIC IPC427C	1	6ES7675-1D…-…1	Different bundles are available for the IPC427C, see http://support.automation.siemens.com/WW/view/en/37954208. The bundle must include WinAC RTX 2010 UPD 3.

Software components

Table 2-3: SIMATIC software components

Component	Qty.	Order no.	Note
STEP 7 V5.5	1	6ES7810-4C.10-..	-
S7-SCL V5.3	1	6ES7811-1CC05-..	-
STEP 7 Prof. V11 SP2 Update 4	1	6ES7822-1A.01-..	V11 projects can also be processed with STEP 7 V12
WinAC ODK V4.2 SP1	1	6ES7806-1CC03-0BA0	-

Table 2-4: Third-party software

Component	Qty.	Order no.	Note
MATLAB V8.0 (R2012b)	1	-	www.mathworks.com
Simulink V8.0	1	-	
Embedded Coder V6.3	1	-	Embedded Coder requires that MATLAB Coder and Simulink Coder be installed.
MATLAB Coder V2.3	1	-	
Simulink Coder V8.2	1	-	
Microsoft Visual Studio 2008/2010 Professional	1	-	www.microsoft.com
IntervalZero SDK V9.1.2	1	-	www.intervalzero.com (optional, for generating RTDLL files)

Sample files and projects

The following list contains all files and projects that are used in this example.

Table 2-5

Component	Note
Setup: **WinAC Target**	Setup for installing WinAC Target for Simulink
Compressed (zipped) folder: **WinAC_Target_Examples_v11**	This zip archive contains the following files: • **01_Simulink_Model** "PID_Process_Model.mdl" Simulink model "PID_Model.mdl" Simulink model • **02_WinAC_Target_Example_Code** Files generated with WinAC Target, including External mode for - STEP 7 V5.x with DLL - STEP 7 V5.x with RTDLL - STEP 7 V1x with DLL - STEP 7 V1x with RTDLL • **03_STEP7_V5x_Project** Sample project for STEP 7 V5.5 • **04_STEP7_V11_Project** Sample project for STEP 7 V11

3 Basics

For all basic topics on the software components used in this application, please use the following links.

MathWorks

- **MATLAB**
  Program to solve mathematical problems and graphically display the results. In MATLAB, programming is done in a proprietary programming language. It offers a great scope of mathematical functions.
- **Simulink**
  Add-on for MATLAB for graphical programming of models and simulations. Simulink offers a comprehensive library with functions to create, for example, control engineering applications.
- **Embedded Coder**
  Add-on for Simulink for compiling models into C/C++ code. Embedded Coder requires that MATLAB Coder and Simulink Coder be installed.

www.mathworks.com

Microsoft Visual Studio 2008/2010

Visual Studio is a development environment with integrated high-level languages such as C, C++, C#, Basic.

www.microsoft.com

IntervalZero

IntervalZero offers a real-time environment for PC systems: IntervalZero RTX. To create RTDLL files using Microsoft Visual Studio, you need the appropriate IntervalZero RTX SDK (software development kit).

www.intervalzero.com

STEP 7

STEP 7 is the development environment for programming and commissioning SIMATIC programmable logic controllers.

http://www.automation.siemens.com/mcms/simatic-controller-software/en/Pages/Default.aspx

SIMATIC WinAC RTX and WinAC ODK

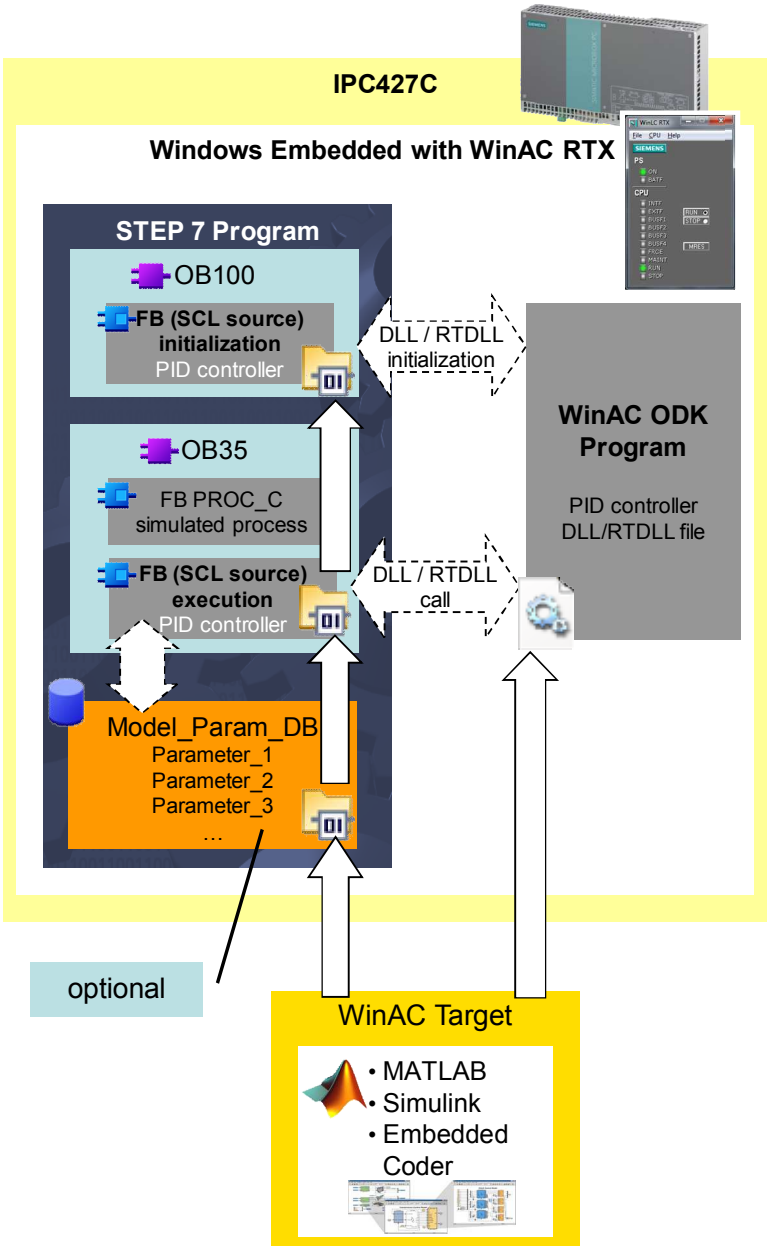
- **WinAC RTX**
  SIMATIC WinAC RTX is the SIMATIC software controller for PC-based automation solutions and allows real-time capable, deterministic control on the PC.
- **WinAC ODK**
  Via three different interfaces, the WinAC option Open Development Kit (ODK) allows flexible use of all PC resources from the control program, providing a high-performance extension of the PLC functionality. All Windows operating system functions and system resources are available to the programmer and therefore also access to external hardware and software components.

http://www.automation.siemens.com/mcms/programmable-logic-controller/en/software-plc/Pages/Default.aspx

4 Function mechanisms of this application

The figure below shows the complete function mechanism of the application: From the creation of the PID controller with Simulink to the execution on the target system (here: WinAC RTX on IPC427C).

Figure 4-1: Complete overview of the application



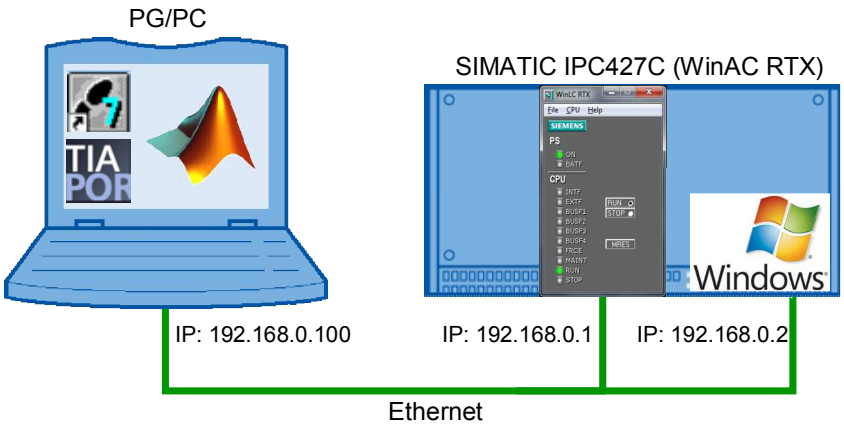
1. An **IPC427C** where **WinAC RTX** is installed is used as the runtime system.
2. The **STEP 7 program**:
    - **OB100** (startup OB for restart (warm restart)): Calls the ported Simulink model once to initialize the DLL/RTDLL file.
    - **OB35** (time interrupt OB with 100ms cycle)): Calls the simulated process (PROC_C) and the ported Simulink model (PID controller).
    - Optional **Model_Param_DB** (global data block to manipulate internal model parameters)
3. **FB PID_Model** (ported Simulink model) reads inputs, communicates with the DLL/RTDLL file and writes outputs.
4. WinAC Target encodes the Simulink model and generates the **SCL source** and the **DLL/RTDLL file**.

5 Installation

5.1 Hardware installation

In general, all PC systems where WinAC RTX is installed can be used for this application. A SIMATIC IPC427C was used in this example.

Figure 5-1: Hardware configuration



The programming device (PG/PC) must be connected to the runtime system (IPC427C) using an Ethernet cable.

The following Ethernet addresses were used for this application:

- PG/PC interface: Ethernet (192.168.0.100)
- IPC427C interface: PROFINET CP1616 (192.168.0.1)
  Interface to WinAC RTX
- IPC427C interface: Industrial Ethernet (192.168.0.2)
  Interface to Windows

Subnet mask 255.255.255.0 was parameterized in both devices.

Note In order to use External mode with Simulink, TCP/IP communication must be established between the PG/PC and the runtime system. For this purpose, the Windows interface must be parameterized in WinAC Target (here: 192.168.0.2).

For more information on the IPC427C PC system, please refer to this manual:
http://support.automation.siemens.com/WW/view/en/37028954

5.2 Software installation

Installing the basic software

Install the following software on the programming device.

Table 5-1: Basic software installation instructions

No.	Action	Remark
1	Install **MATLAB V8.0 (R2012b)** **Simulink V8.0** **MATLAB Coder V2.3** **Simulink Coder V8.3** **Embedded Coder V6.3**	-
2	Install **Microsoft Visual Studio 2008** or **2010 Professional**	-
3	Optionally, install **IntervalZero SDK V9.1.2**	Only required for RTDLL generation
4	Install **STEP V5.5** and **S7-SCL V5.3** Alternatively, install **STEP 7 V11 Professional with Service Pack 2 Update 4** or **STEP 7 V12 Professional**	Manual: http://support.automation.siemens.com/WW/view/en/10805384/133300
5	Install **WinAC ODK V4.2 SP1**	Manual: http://support.automation.siemens.com/WW/view/en/12840073/133300

Install the following software on the runtime system (here: IPC427C).

Table 5-2: Basic software installation instruction

No.	Action	Remark
1	Install **WinAC RTX 2010 Update 3** on a PC system	Manual: http://support.automation.siemens.com/WW/view/en/10805641/133300 Update: http://support.automation.siemens.com/WW/view/en/15227402

Installing WinAC Target

Install the following software on the programming device.

Table 5-3: WinAC Target installation instruction

No.	Action	Remark
1	Install **WinAC Target** (run setup.exe)	Prerequisite: Windows XP Professional SP3 or Windows 7 (32-bit) STEP 7 V5.5 and S7-SCL V5.3 or STEP 7 V11 Professional with Service Pack 2 or STEP 7 V12 Professional WinAC ODK V4.2 SP1

Integrating WinAC Target into Simulink

In order to use WinAC Target in Simulink, the software has to be integrated first. Perform the following steps once.

Table 5-4: WinAC Target into Simulink – installation instructions

No.	Action	Remark
1	Open **"MATLAB"**.	-
2	In the "Current Folder", navigate to the **WinAC Target installation folder**. It contains the **"startup.m"** file. Right-click and select Run to run this file. WinAC Target will be integrated into Simulink.	-

Note This has to be done **only once when using** WinAC Target **for the first time**.

Installing the sample projects

Table 5-5: Sample projects installation instruction

No.	Action	Remark
1	Unzip the "**56969417_WinAC_Target_Examples_v11.zip**" file	-

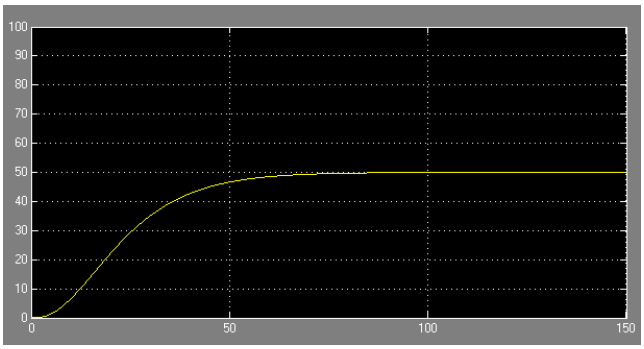
6 MATLAB/Simulink

6.1 Model simulation with MATLAB/Simulink

6.1.1 The process

This application simulates a process that behaves similarly to a temperature control application. The response of the process to a step from 0°C to 50°C is very slow and aperiodic approaches the value 50 (Figure 6-1). The Y-axis represents the temperature in degrees Celsius [°C] and the X-axis represents the time in seconds [s].

Figure 6-1: Step response of the process



As a mathematical model, there is a continuous PT3 plant with the following formula:

$$G(s) = \frac{1}{10s+1} \cdot \frac{1}{10s+1} \cdot \frac{1}{5s+1}$$

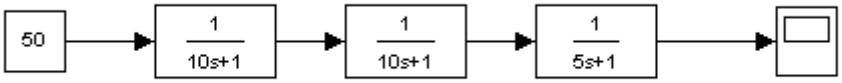
or

Table 6-1: Plant parameters of the continuous process

Plant parameters	Remark
GAIN = 1	Gain
TM_LAG_1 = 10 seconds	Time lags
TM_LAG_2 = 10 seconds	
TM_LAG_3 = 5 seconds	

In Simulink, the process is simulated in the form of three PT1 functions connected in series.

Figure 6-2: Simulink model: Step response of the continuous process



This model is a continuous process. Later the process and the PID controller will be called in the PLC (here: WinAC RTX) in a cyclic organization block with the T = 100ms cycle. This means it is a discrete process simulation and a discrete PID controller. For this reason, the continuous plant must be converted to a discrete plant.

The TUSTIN transformation is used in this case:

$$s = \frac{1}{T} \cdot \ln(z) = \frac{1}{T}\left[(z-1) - \frac{(z-1)^2}{2} + -...\right] \rightarrow \boxed{s \approx \frac{1}{T} \cdot (z-1)}$$

$$T = 100ms = 0.1$$

The continuous process …

$$G(s) = \frac{1}{10s+1} \cdot \frac{1}{10s+1} \cdot \frac{1}{5s+1}$$

after the TUSTIN transformation …

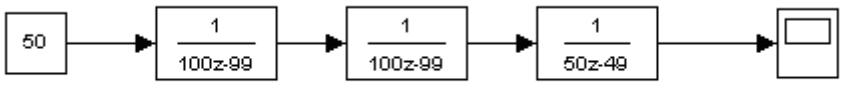
$$G(z) = \frac{1}{10 \cdot \frac{1}{0.1} \cdot (z-1) + 1} \cdot \frac{1}{10 \cdot \frac{1}{0.1} \cdot (z-1) + 1} \cdot \frac{1}{5 \cdot \frac{1}{0.1} \cdot (z-1) + 1}$$

becomes the discrete process:

$$G(z) = \frac{1}{100z - 99} \cdot \frac{1}{100z - 99} \cdot \frac{1}{50z - 49}$$

The model in Simulink then looks as follows:

Figure 6-3: Simulink model: Step response of the discrete process



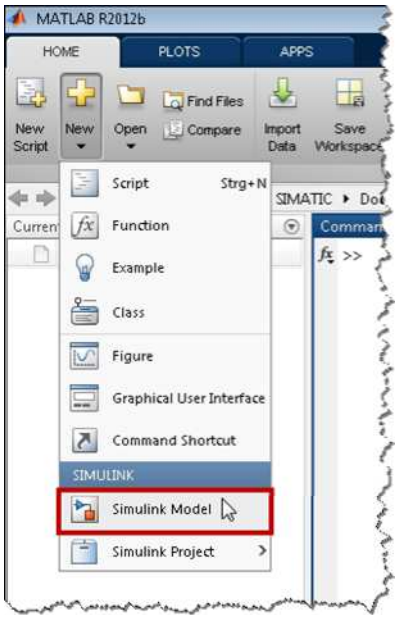
6.1.2 The PID controller

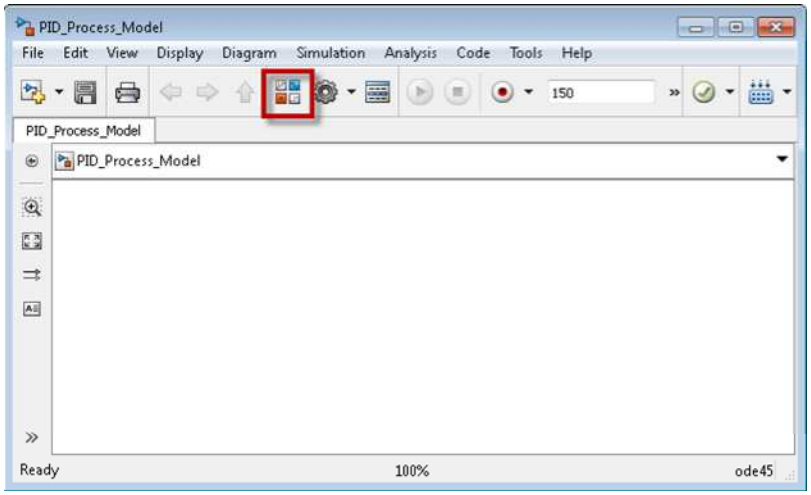
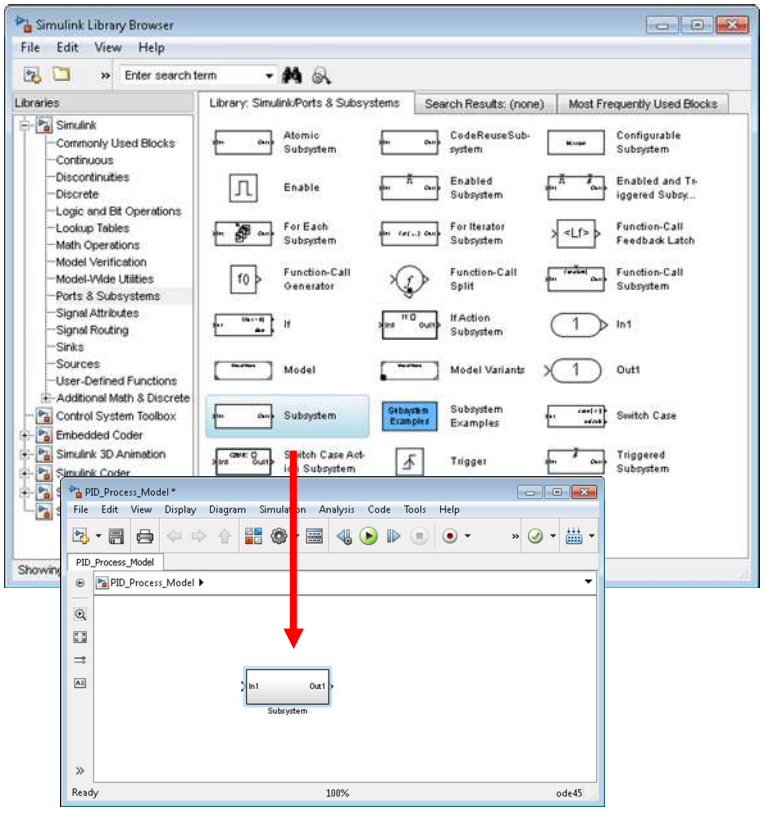
The following section shows you how to create a model in Simulink. In this case, this is a simple PID controller with the process described in Chapter 6.1 Model simulation with MATLAB/Simulink.

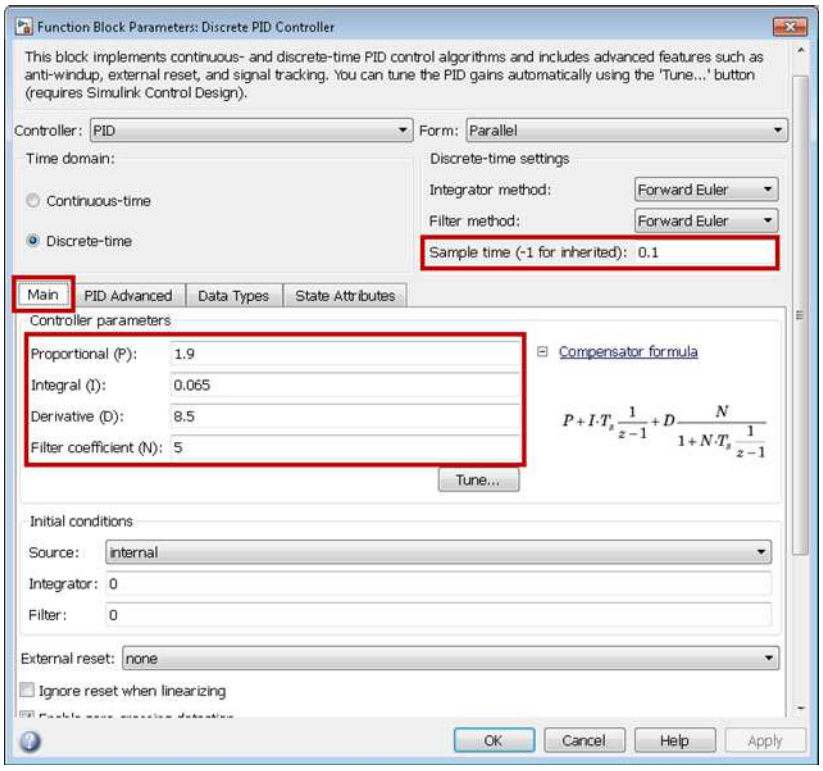
Note Alternatively, you can also open the “PID_Process_Model.mdl” Simulink model.

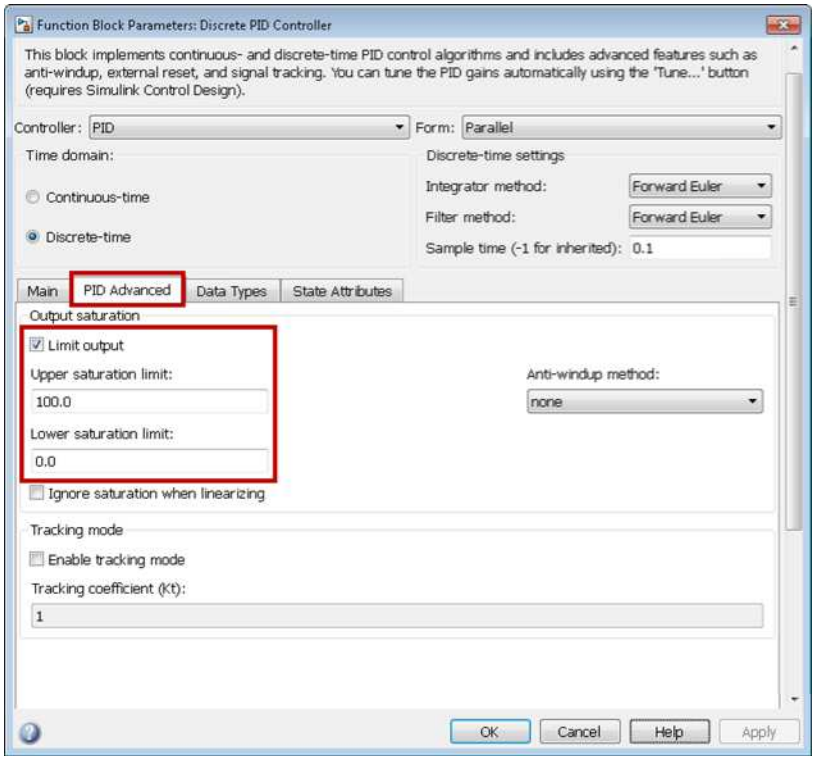
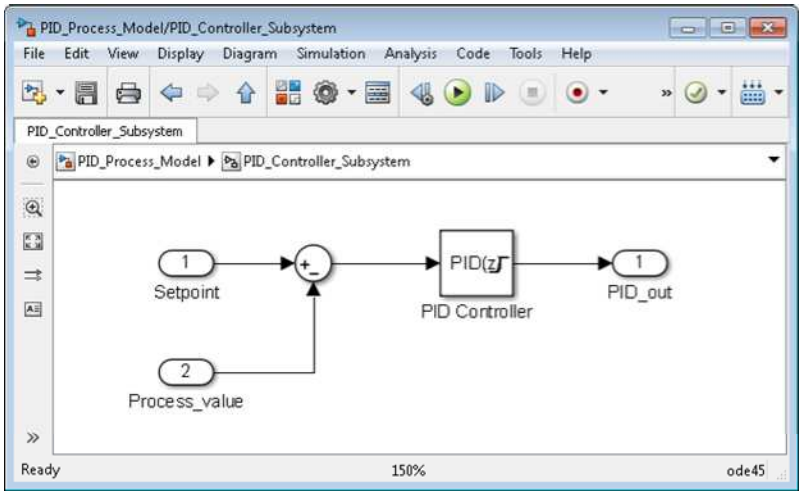
Creating a model with PID controller in Simulink

Table 6-2: Creating a Simulink model

Step	Instruction
1.	Open “**MATLAB**”.
2.	Create a new model using Simulink and save it under the name “**PID_Process_Model**”. 

Step	Instruction
3.	Open the "**Simulink Library Browser**". 
4.	Use drag and drop to insert a "**Subsystem**" from the "**Simulink - Ports & Subsystems**" category into the model. 
5.	Double-click the block to open the subsystem.

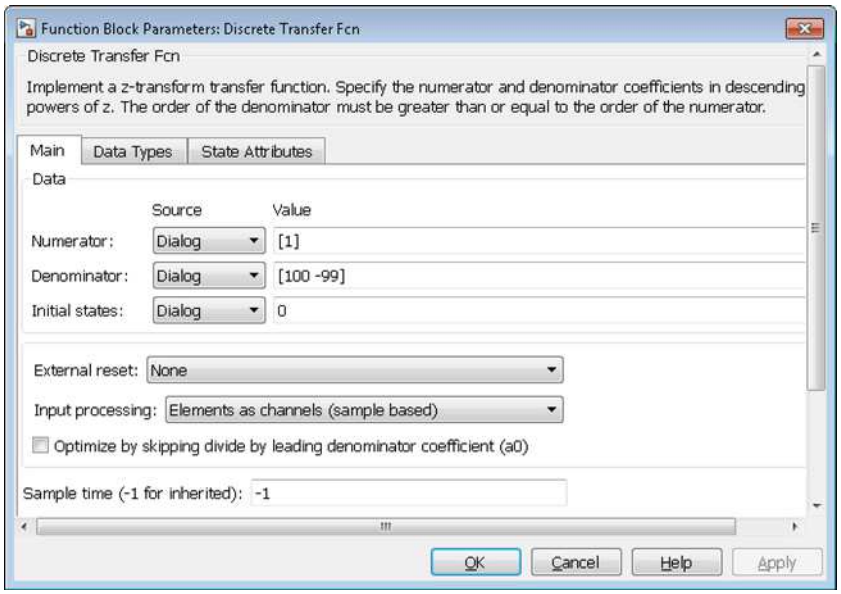
Step	Instruction
6.	Use drag and drop to insert the following blocks from the Simulink library into the subsystem: • More input: "**Simulink – Sources – In1**" • Adding block: "**Simulink – Math Operations – Sum**" • PID controller: "**Simulink – Discrete – Discrete PID Controller**"
7.	Double-click to open the block parameters of "**Sum**" and change the "**List of signs**" parameter to " **|+-** ".
8.	Then close the parameterization screen form.
9.	Double-click to open the block parameters of "**Discrete PID controller**" and apply the following values.  **Note:** • Sample time is set to 0.1 for 100ms. The controller and the process will later be called in WinAC RTX in OB35 (cycle: 100ms). • The PID values were determined empirically for this application. A good response to setpoint changes with little overshoot is achieved with this parameterization. • The sample time in the Simulink model must match the sample time later desired in the PLC application.

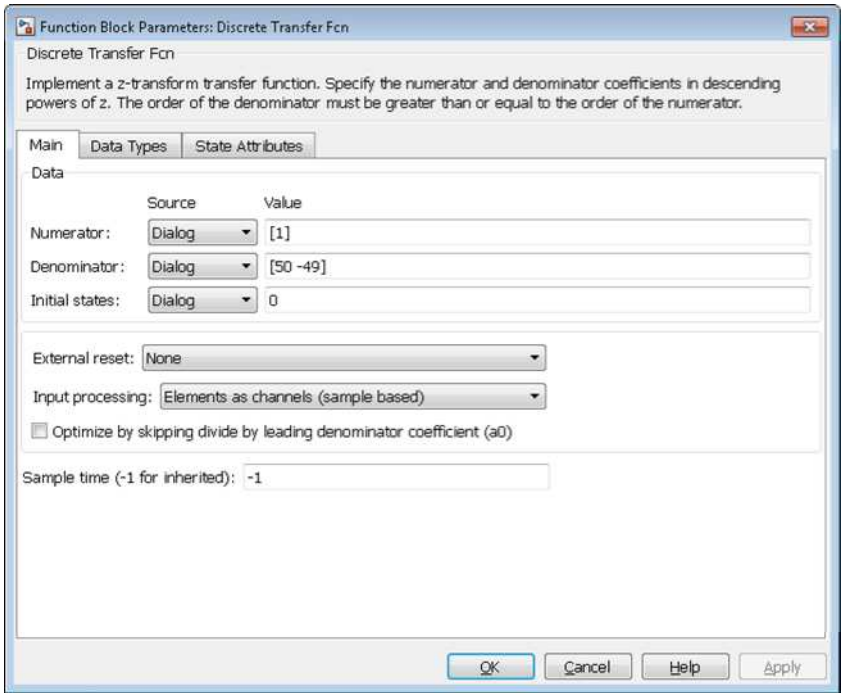
Step	Instruction
10.	In the "**PID Advanced**" tab, set a limit from 0 to 100 for the PID output. 
11.	Close the parameterization screen form.
12.	Wire the blocks and assign the names as shown below. 
13.	Close the subsystem.

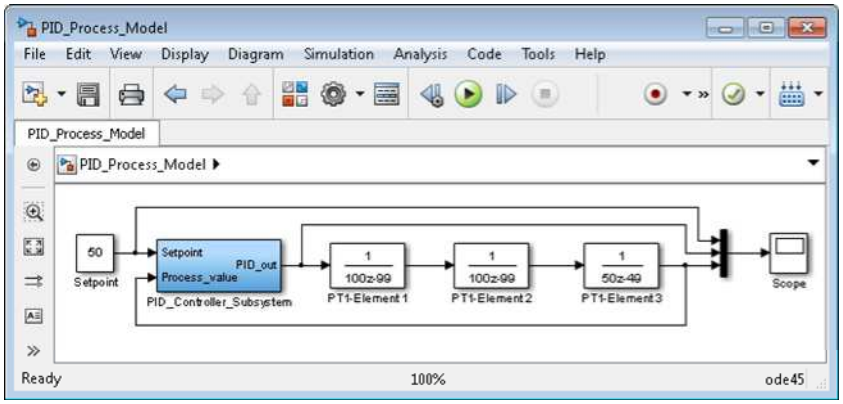
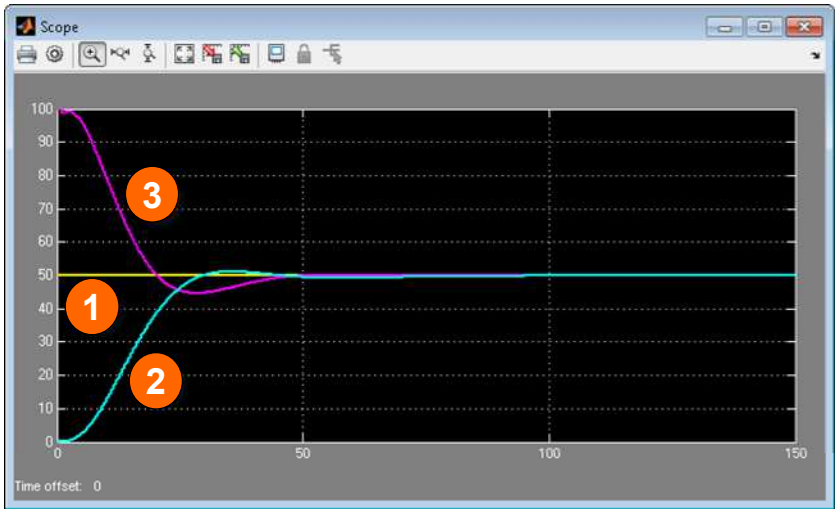
6.1.3 Simulation of the complete control loop

The simulation of the process and the created PID controller allows you to check the response of the complete control loop directly in Simulink. To do so, proceed as follows.

Table 6-3: Simulation of the complete control loop

Step	Instruction
1.	Use drag and drop to insert three blocks of the "**Discrete Transfer Fcn**" type from the Simulink library into the Simulink model: • "**Simulink – Discrete**" **–** "**Discrete Transfer Fcn**"
2.	Successively open the "**Discrete Transfer Fcn**" blocks and parameterize them as follows.
3.	Parameterization for blocks 1 and 2: 

Step	Instruction
4.	Parameterization for block 3:  Close the parameterization screen forms.
5.	In addition, insert the following blocks: • "**Simulink – Sources – Constant**" for the setpoint • "**Simulink – Sinks – Scope**" to visualize the control loop • "**Simulink – Signal Routing – Mux**" to merge the signals
6.	Double-click to open the "**Mux**" block and change the value of the "**number of inputs**" parameter to "**3**". Close the parameterization screen form.
7.	Double-click to open the "**Constant**" block and change "**Constant Value**" to 50 and "**Sample time**" to 0.1 for 100ms. Close the parameterization screen form.

Step	Instruction
8.	Wire the complete control loop and assign the names as shown below. 
9.	• Double-click to open the "**Scope**" block. • Select 150 for "**Simulation stop time**" and use the "**Start simulation**" icon in the Simulink model menu bar to start the simulation. • Click the "**Autoscale**" icon in the menu bar of the "**Scope**" block. As a result, the simulation of the complete control loop is displayed.  1. Setpoint 2. Process_value 3. PID_out

Note

Before porting the PID controller to the runtime system, you have to extract the model. This means that the plant simulation is removed from the model as the complete model is compiled when encoding. The next chapter describes the extraction of the controller.

6.2 Extracting the controller for code generation

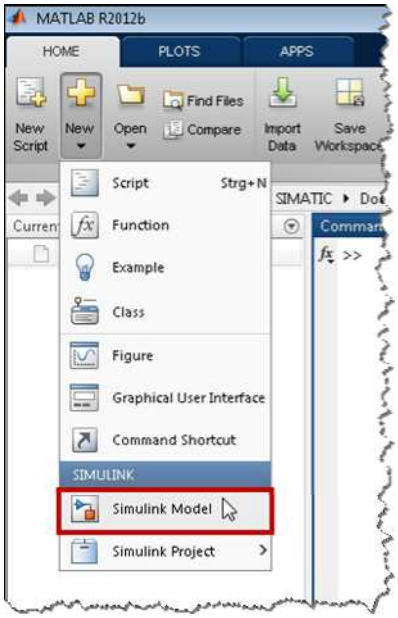
To port the PID controller to WinAC RTX, the model has to be adjusted. This is necessary as only the controller is to run on the runtime system and not the complete control loop with controller and process.

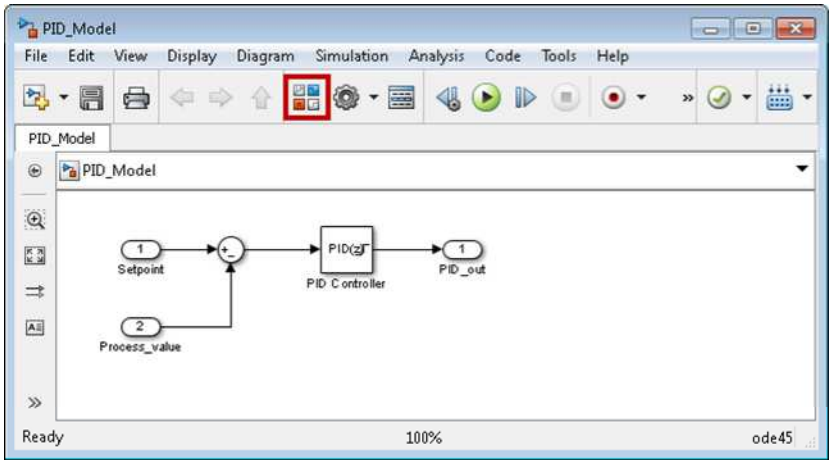
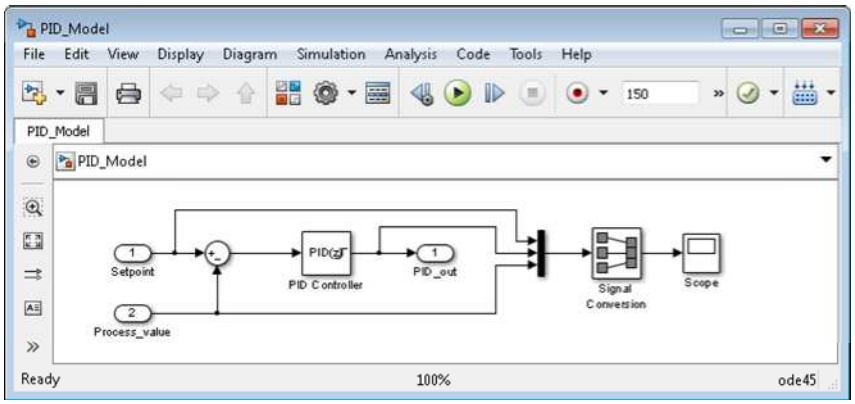
Note Conversion with WinAC Target always compiles the complete Simulink model.

Note Alternatively, you can also open the “PID_Model.mdl” Simulink model.

To do so, follow the steps in the table below:

Table 6-4: Extracting the controller

Step	Instruction
1.	Open the “**PID_Process_Model**” model.
2.	Create a new model and save it under the name “**PID_Model**”. 
3.	Double-click the “**PID_Controller_Subsystem**” block to open the subsystem from the “**PID_Process_Model**”.
4.	Select the entire content of the subsystem and use drag and drop to move it to the newly created “**PID_Model**”.

Step	Instruction
5.	Open the "**Simulink Library Browser**". 
6.	Use drag and drop to insert the following blocks from the Simulink library into the new "**PID_Model**": • "**Simulink – Signal Attributes – Signal Conversion**" • "**Simulink – Sinks – Scope**" • "**Simulink – Signal Routing – Mux**"
7.	Double-click to open the "**Mux**" block and change the value of the "**number of inputs**" parameter to "**3**". Close the parameterization screen form.
8.	Wire the model and assign the names as shown below. 

The model now only includes the PID controller with the links to the variables: "Setpoint", "Process_value" and "PID_out". To graphically display the signals, the model also includes the "Signal Conversion" and "Scope" blocks.

Now the model can be encoded for WinAC RTX and run (see 7.3 Code generation with WinAC Target).

7 WinAC Target

7.1 General

WinAC Target is a Siemens add-on for Simulink that generates the DLL/RTDLL file and the SCL source from Simulink models.

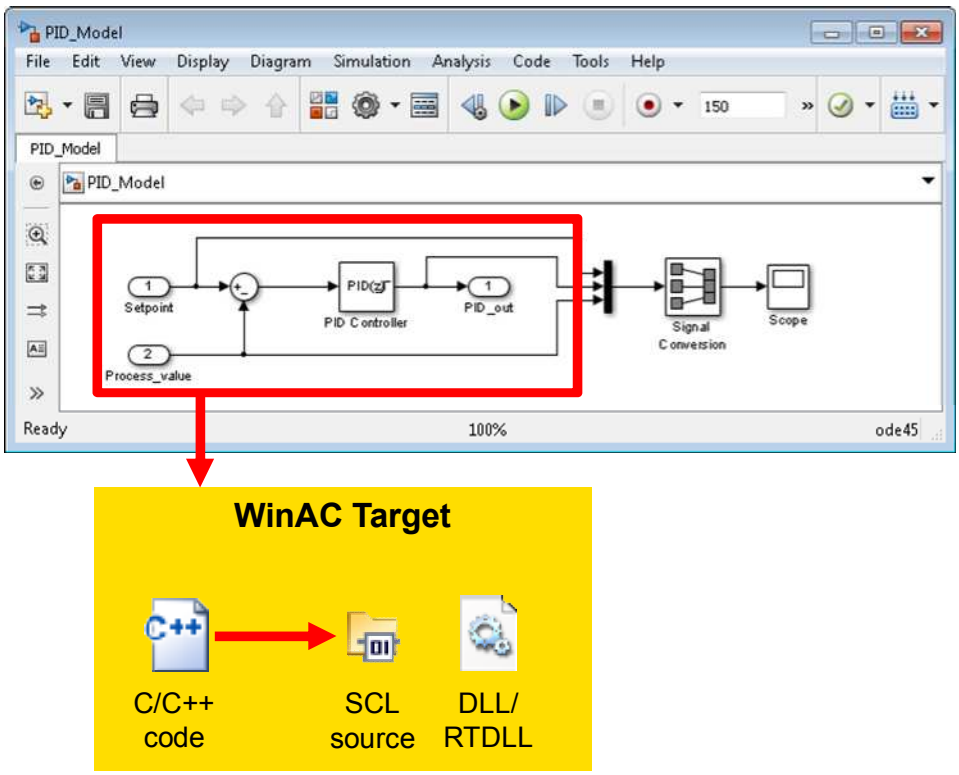
To integrate the Simulink model into a STEP 7 project, WinAC Target automatically generates all necessary blocks and files. An SCL source and a DLL/RTDLL file are created from the generated C/C++ code.

The SCL source must be manually integrated into the STEP 7 project. This is done using the “External Sources” function in the Project tree of the STEP 7 project. Finally, you have to download the STEP 7 program and the DLL or RTDLL to a PC system where WinAC RTX is installed.

Note If you want to use External mode, you have to activate it before encoding (see Chapter 7.2 Activating External mode).

Note If you want to manipulate the model parameters using STEP 7, you have to enable the “Parameter access with STEP 7” parameter before encoding (see Chapter 10 Manipulating model parameters using STEP 7).

Figure 7-1: Code generation – schematic diagram



Note

IntervalZero SDK (software development kit) is required for generating an RTDLL.

http://www.intervalzero.com/

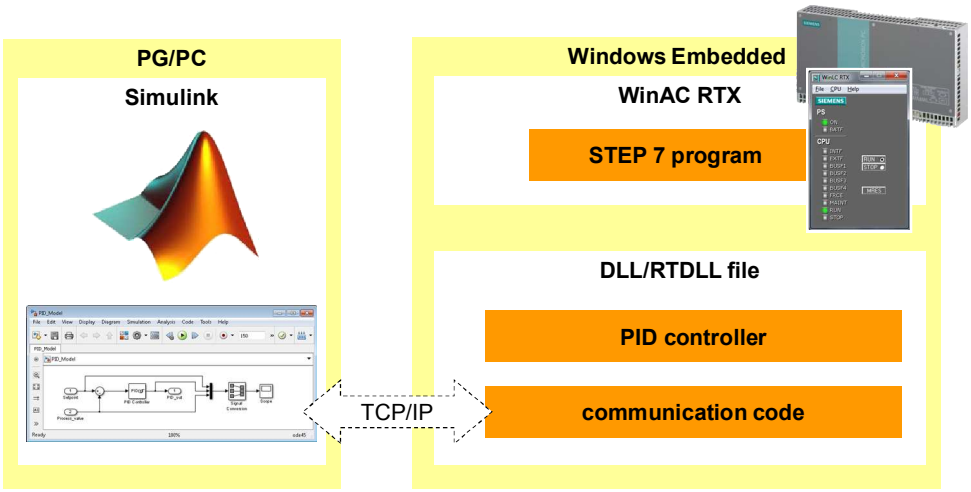
7.2 Activating External mode

7.2.1 General

Simulink External mode provides you with the option to monitor the ported model during operation on the runtime system (here: WinAC RTX on IPC427C) and change internal model parameters online.

To enable Simulink to communicate with the ported model on the runtime system via External mode, External mode must be activated before encoding. In this case, the code is extended by the communication code to allow data exchange between Simulink and the model in the runtime system. Communication is implemented based on TCP/IP.

Figure 7-2: Communication between Simulink and model on WinAC RTX



Note

When External mode is activated for encoding, the size of the generated DLL/RTDLL file increases and it needs more time to run. The reason for this is the additional code for communication.

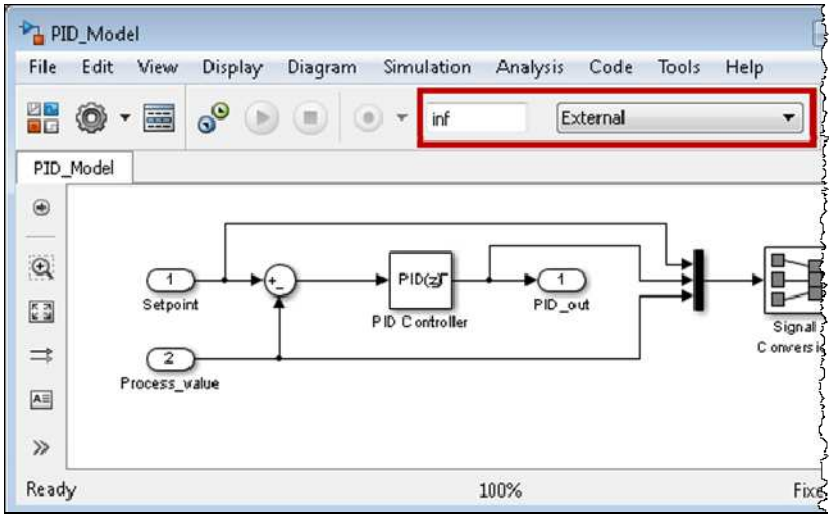
Note

If communication with an RTDLL file is to take place, an additional service must be started on the runtime system (see Chapter 9.3 RTDLL Communication Service for External mode).

7.2.2 Parameterizing External mode

If you want to use External mode, you have to make the following settings.

Table 7-1: Parameterizing External mode

Step	Instruction
1.	Open the following Simulink model: "**PID_Model**".
2.	Set "**Simulation stop time**" to "**inf**" and "**Simulation Mode**" to "**External**". 
3.	In the menu, click "**Code – C/C++ Code – Code Generation Options …**".
4.	A window with a navigation bar appears. "**Code Generation**" is checked by default.
5.	Set the following parameters: "**Code Generation**" navigation item - **System target file: WinAC_Target.tlc** "**Interface**" navigation item - **Data exchange** o **Interface: External Mode** o **Transport layer: tcpip** o **MEX-file arguments: '192.168.0.2'** (IP address of the runtime system, Windows)

7.3 Code generation with WinAC Target

7.3.1 Settings

In this application, the following settings are made for WinAC Target.

Table 7-2: Settings for code generation

Step	Instruction
1.	Open the following Simulink model: "**PID_Model**".
2.	In the menu, click "**Code – C/C++ Code – Code Generation Options …**".
3.	A window with a navigation bar appears. The "**Code Generation**" item is checked by default. Set the following parameters: "**Code Generation**" navigation item - **System target file: WinAC_Target.tlc**
4.	The following settings will be set automatically after selecting WinAC Target: "**Solver**" navigation item - **Type: Fixed-step** - **Solver: discrete (no continuous states)** - **Fixed-step size: auto** "**Interface**" navigation item - **Data exchange** o **Interface: External Mode** o **Transport layer: tcpip** o **MEX-file arguments: '192.168.0.2'** (IP address of the runtime system) "**Data Import/Export**" navigation item - **Save to workspace:** Uncheck the Time, States, Output, Final states variables "**WinAC Options**" navigation item - In "**WinAC ODK project type**", you can select whether you want to generate a Windows DLL (**DLL**) or a real-time DLL (**RTDLL**). A Windows DLL is selected by default. - In "**STEP 7 Version**", you can select the automation software you want to use later. The STEP 7 V5.5 setting is selected by default. - In "**Data cycle time (Extended mode) [ms]**", you can customize the cycle time for External mode. The default setting is "50". - In "**STEP 7 Function Block name**", you can select the name of the function block. The default setting is the name of the Simulink model. - In "**Parameter access with STEP 7**", you can enable parameter access to the model parameters with the STEP 7 user program.

Note If the development environment and WinAC RTX are installed on one computer, **MEX-file arguments** must be modified.
Change the IP address to **'localhost'**.

If the IP address of your target system differs from the one in this application example, change **MEX-file arguments**.

Note When selecting WinAC Target, External mode will be activated by default. If you do not want to use External mode, proceed as follows:

- In the menu, click "**Code – C/C++ Code – Code Generation Options…**"
- "**Interface**" navigation item
- In the "**Data exchange**" menu option, set "**Interface**" to "**None**"

7.3.2 Generating the code

Note Alternatively, you can also use the code from the sample files:

02_WinAC_Target_Example_Code

Files generated using WinAC Target with **External mode** for

- STEP 7 V5.x with DLL
- STEP 7 V5.x with RTDLL
- STEP 7 V1x with DLL
- STEP 7 V1x with RTDLL

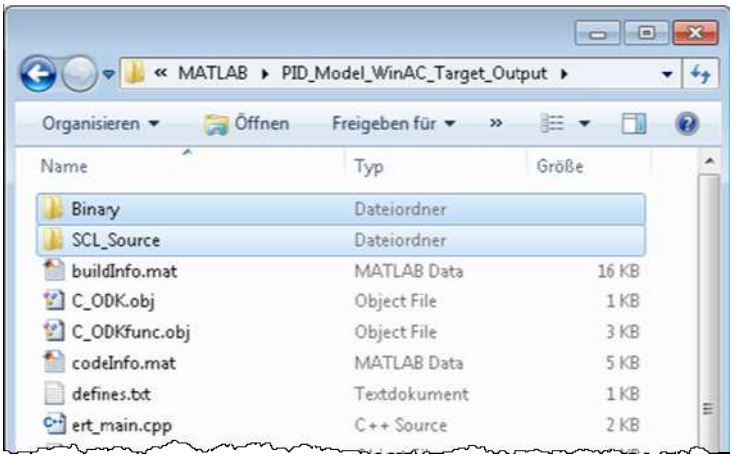
WinAC Target Output

To compile a Simulink model into C++ code and generate a DLL/RTDLL and an SCL file, proceed as follows:

Table 7-3: Settings for code generation

Step	Instruction
1.	In the Simulink model menu, click "**Code – C/C++ Code – Build Model**".
2.	In this case, the "PID Controller" block from the Simulink model (PID_Model.mdl) is compiled into C/C++. Then WinAC Target generates the DLL/RTDLL file and the SCL source from this source code.
3.	The "**...\PID_Model_WinAC_Target_Output**" Simulink model directory contains the generated code. • "**Binary**" folder: DLL/RTDLL file • "**SCL_Source**" folder: SCL source

Figure 7-3: Files generated with WinAC Target



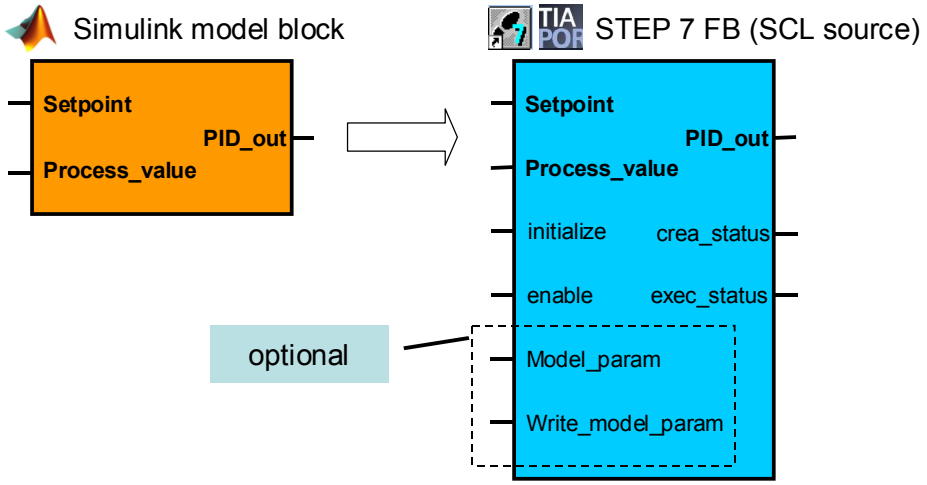
Addition to the SCL source

The interfaces were applied when porting the Simulink code. In STEP 7, the block has the same interfaces as the Simulink model.

The additional interfaces (initialize, enable, crea_status, exec_status) for WinAC ODK communication will be described in the next chapter.

The “Model_param” and “Write_model_param” inputs will only be additionally created when the “Parameter access with STEP 7” setting is checked in WinAC Target.

Figure 7-4: Simulink model block and STEP 7 function block



7.4 Data type conversion

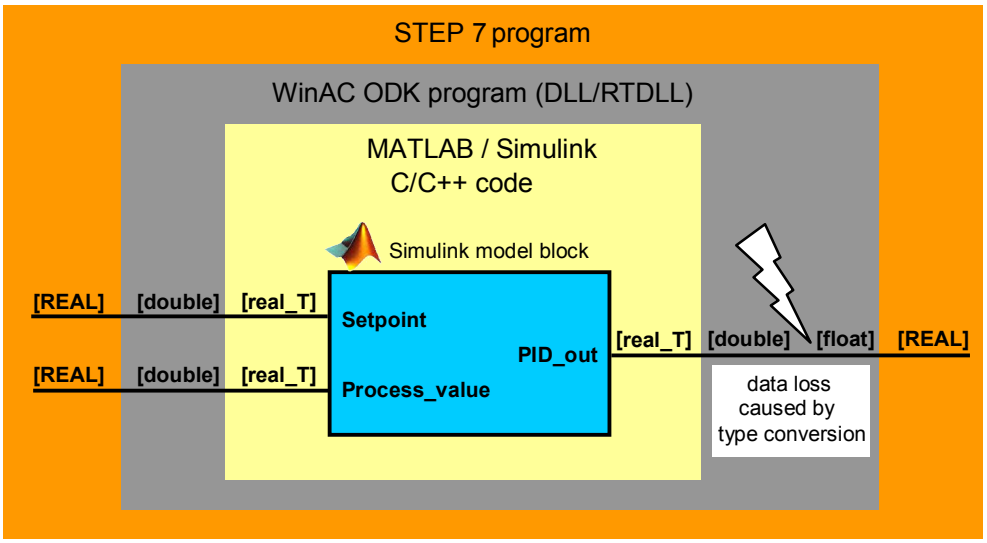
As STEP 7, WinAC ODK and Embedded Coder use different data types, type conversion must take place on the interfaces of the programs. WinAC Target recognizes the data types of the Embedded Coder C/C++ program and adjusts the data in the WinAC ODK program and the STEP 7 program (SCL source). Type conversion takes place when running WinAC Target. The following table shows the relevant data types.

Table 7-4: Data types overview

C/C++ (Embedded Coder)	C/C++ (WinAC ODK)	Bytes	STEP 7
boolean_T	bool	1 bit	BOOL
int8_T, unint8_T, char_T, uchar_T, byte_T	char	1	CHAR
int16_T, uint16_T	short	2	INT
int32_T, uint32_T, int_T, uint_T	long	4	DINT
ulong_T	long	4	DINT
real_T	float	4	→ REAL (4 Byte) Notice: Loss of data
real32_T	float	4	REAL
“type”_T [“length”]	“type” [“length”]	Depending on array size and data type	Array [lo .. hi] of type

If the data types used in the WinAC ODK program are larger than the ones used in STEP 7, loss of data may occur. The figure below shows the interfaces and data types of this application. In this case, data conversion with loss of data takes place from the [double] to the [float] type in the WinAC ODK program. However, this does not affect the application example as the accuracy is sufficiently high despite the loss of data.

Figure 7-5: Example of data type conversion

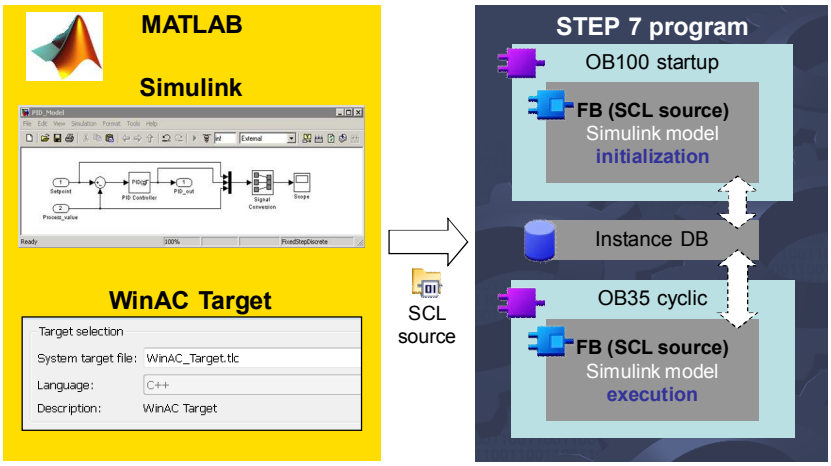


8 Commissioning with STEP 7

This chapter describes the programming and commissioning with STEP 7 V5.5 and STEP 7 V11 (TIA Portal).

8.1 General

Figure 8-1: Simulink model in the STEP 7 program



When porting Simulink models in STEP 7, the following steps are recommended:

Table 8-1: Tips for porting in STEP 7

Instruction	Note
Initialization The ported Simulink model (FB from SCL source) is to be initialized in OB100 (startup).	The initialization operation of the DLL/RTDLL file may take several seconds (depending on the operating system and the complexity of the DLL/RTDLL). If initialization in the time interrupt OB is performed, for example, with the 100 ms cycle, the runtime system will go to STOP due to a timeout.
During operation During operation, it is standard practice to call the Simulink model in a cyclic time interrupt OB. This ensures an equidistant call.	When creating the Simulink model, you should know from the beginning in which cycle the model is called on the runtime system. Accordingly, the model in Simulink and the time interrupt OB are parameterized with the same cycle time.

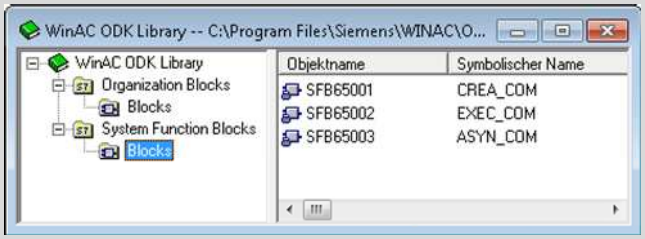
8.2 STEP 7 V5.5

8.2.1 WinAC ODK Library

Note

To integrate models, the WinAC ODK software has to be installed. In addition, the **WinAC ODK Library** must be integrated in the SIMATIC Manager as the SFB65001 (CREA_COM) and SFB65002 (EXEC_COM) blocks are required to run the DLL/RTDLL.

For more information on WinAC ODK, please refer to
http://www.automation.siemens.com/mcms/programmable-logic-controller/en/software-plc/simatic-winac-odk/Pages/Default.aspx



8.2.2 Contents of the sample program

The "**…\03_STEP7_V5x_Project**" folder contains the sample project for STEP7 V5.5 with the following contents:

- **WinAC RTX**
  This is only the hardware configuration of a PC station with WinAC RTX. The device is empty. This configuration serves as a template.

- **PID Simple**
  This example provides a prepared program for integration by WinAC Target. The following blocks have already been created:

  - **OB100 (Complete Restart**) is called when starting the PLC. The PID controller is initialized in this OB.
  - **OB35 (CYC_INT5)** as a cyclic OB with a 100 ms cycle. The simulated plant and the PID controller (PID_Controller) are called in this block.
  - **DB35 (DATA)** as a global data block. It includes all necessary variables:
    - **Setpoint [Real]**,
    - **PID_output [Real]**,
    - **Process_value [Real]**,
    - **crea_status [Word]** (provides the status via CREA_COM),
    - **exec_status[Word]** (provides the status via EXEC_COM),
    - **initialize [Bool]**
    - **enable [Bool]**
    - **Model_param [Any]**
    - **Write_model_param [Bool]**

- **FB100 (PROC_C)** with **instance DB100**
  Simulated PT3 process (parameterized like the process in Simulink)
- **SFB65001 (CREA_COM)** to initialize the DLL/RTDLL file
- **SFB65002 (EXEXC_COM)** to run the DLL/RTDLL file
- **VAT_1**
  Variable table with the variables of DB35 (DATA) and DB1 (Model_Param_DB)

- **PID_Simple_dll_final**
  This program includes ready-to-use programming with integration of the PID controller from Simulink by WinAC Target. This program uses a DLL call.

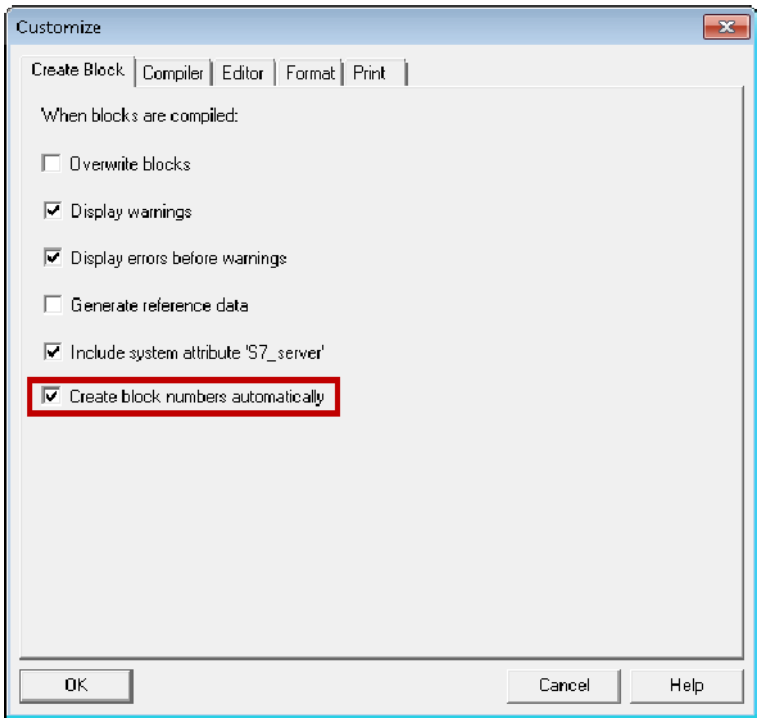
- **PID_Simple_rtdll_final**
  This program includes ready-to-use programming with integration of the PID controller from Simulink by WinAC Target. This program uses an RTDLL call.

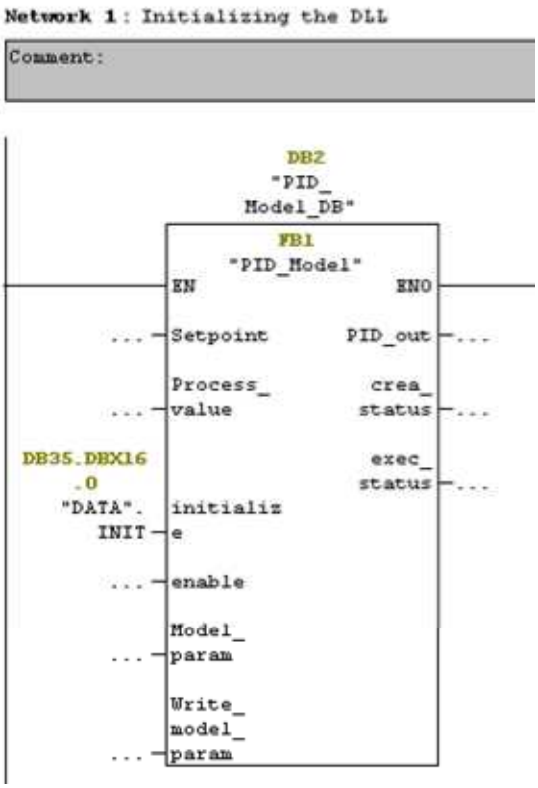
Note It is recommended to initialize the ported Simulink model in an OB100 as the time required for this operation can be significantly longer than the normal call.

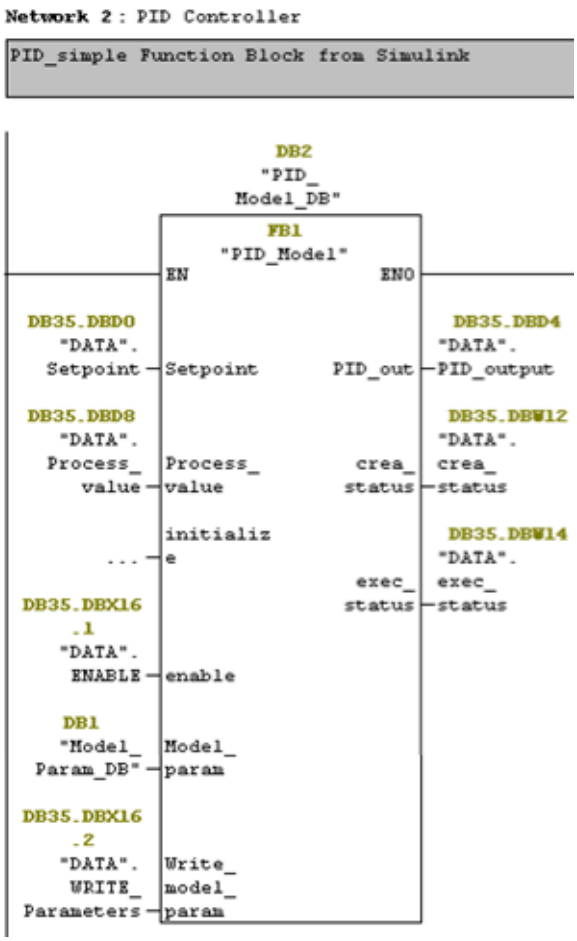
8.2.3 Integrating the Simulink model into STEP 7 V5.5

The following steps complete the "**PID_simple**" program with the "**PID_Model**" block from the Simulink model:

Table 8-2: Integrating the Simulink model into STEP 7 V5.5

Step	Instruction
1.	Open the SIMATIC Manager.
2.	Open the "**WinAC_Target_PID_Project**" sample project in the "**...\03_STEP7_V5x_Project**" directory.
3.	Expand the tree under "**WinAC_Target_PID_Project**".
4.	Expand the tree under "**PID_simple**".
5.	Click the **Source** folder in the "**PID_simple**" S7 program.
6.	Then use "**Insert – External Source**" to add the SCL source generated by WinAC Target.
7.	Open the SCL source and click at the menu **"Options – Customize…"**
8.	Check **"Create block numbers automatically"** and confirm the window with **"OK"** 

Step	Instruction
9.	Generate the function block with "**File – Compile**".
10.	Open OB 100 and insert FB "**PID_Model**" into "**Network 1: Initializing the DLL**". Interconnect the inputs and outputs as shown in the figure. 

Step	Instruction
11.	Open OB 35 and insert FB "**PID_Model**" into "**Network 2: PID Controller**". Interconnect the inputs and outputs as shown in the figure. 
12.	Download the "**PID_simple**" program to WinAC RTX.
13.	Now copy the **DLL (PID_Model.dll)** or **RTDLL (PID_Model.rtdll)** to the **C:** directory of the PC system with WinAC RTX. RTDLL files have to be registered (see notes below).
14.	Set WinLC RTX to RUN mode.

Note

If you want to use a different path to store the DLL/RTDLL file, the path also has to be modified in the SCL source:

Variable: "**DLL_name**"

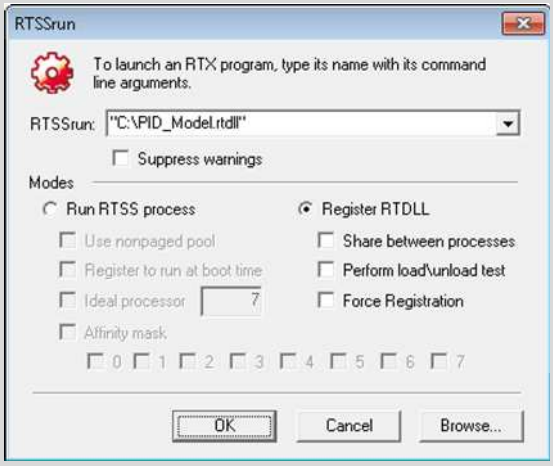
Alternatively, you can also change the variable directly in the generated function block

"**Interface – Static – DLL_name**"

Note

If an RTDLL is to be called in WinAC RTX, it must be registered in IntervalZero Runtime after copying to the runtime system.

In the PC system, select "**Start - Programs - IntervalZero - RTX 2009 - Tools – RtssRun**". Register the RTDLL file as shown in the figure.



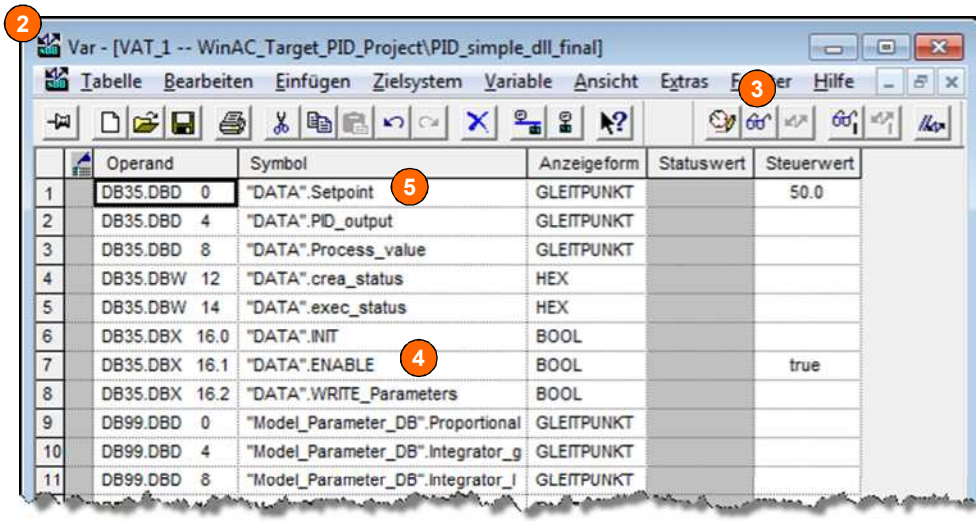
8.2.4 Commissioning

Once you have downloaded the program to WinAC RTX, perform the following steps for commissioning.

Table 8-3: Commissioning with STEP 7 V5.5

Step	Instruction
1.	Set WinLC to RUN mode.
2.	Open the "**VAT_1**" variable table (see Figure 8-2).
3.	Click the "**Monitor Variable**" icon.
4.	Change the value of the "**ENABLE**" variable to "**TRUE**" and click the "**Modify Variable**" icon.
5.	Change the value of the "**Setpoint**" variable to the desired value, e.g. 50, and click the "**Modify Variable**" icon. Due to the setpoint change, the PID controller will output an output value. The process will change accordingly until it matches the "**Setpoint**".

Figure 8-2: Variable table



Note

The following variables allow you to diagnose errors.

The "**crea_status**" variable corresponds to **SFB65001.Status**
The "**exec_status**" variable corresponds to **SFB65002.Status**

For the error codes, please refer to the WinAC ODK manual.
http://support.automation.siemens.com/WW/view/en/35948966

8.3 STEP 7 V11/V12 (TIA Portal)

8.3.1 WinAC ODK Library

For STEP 7 V11 or higher, all necessary ODK blocks have already been integrated in the engineering.

8.3.2 Contents of the sample program

The "„**…\04_STEP7_V11_Project**" folder contains the sample project for STEP7 V11 with the following contents:

- **PID_Simple**
  This example provides a prepared program for integration by WinAC Target. The following blocks have already been created:
  - **OB100 (Complete Restart**) is called when starting the PLC. The PID controller is initialized in this OB.
  - **OB35 (CYC_INT5)** as a cyclic OB with a 100 ms cycle. After integration, the simulated plant and the PID controller (PID_Controller) are called in this block.
  - **DB35 (Data)** as a global data block. It includes all necessary variables:
    - **Setpoint [Real]**,
    - **PID_output [Real]**,
    - **Process_value [Real]**,
    - **crea_status [Word]** (provides the status via CREA_COM),
    - **exec_status[Word]** (provides the status via EXEC_COM),
    - **initialize [Bool]**
    - **enable [Bool]**
    - **Model_param [Any]**
    - **Write_model_param [Bool]**
  - **FB100 (PROC_C)** with **instance DB100**
    Simulated PT3 process (parameterized like the process in Simulink)
  - **SFB65001 (CREA_COM)** to initialize the DLL/RTDLL file
  - **SFB65002 (EXEXC_COM)** to run the DLL/RTDLL file
  - **Watch table_1**
    Variable table with the variables of DB35 (DATA) and DB1 (Model_Param_DB)

- **PID_Simple_dll_final**
  This program includes ready-to-use programming with integration of the PID controller from Simulink by WinAC Target. This program uses a DLL call.

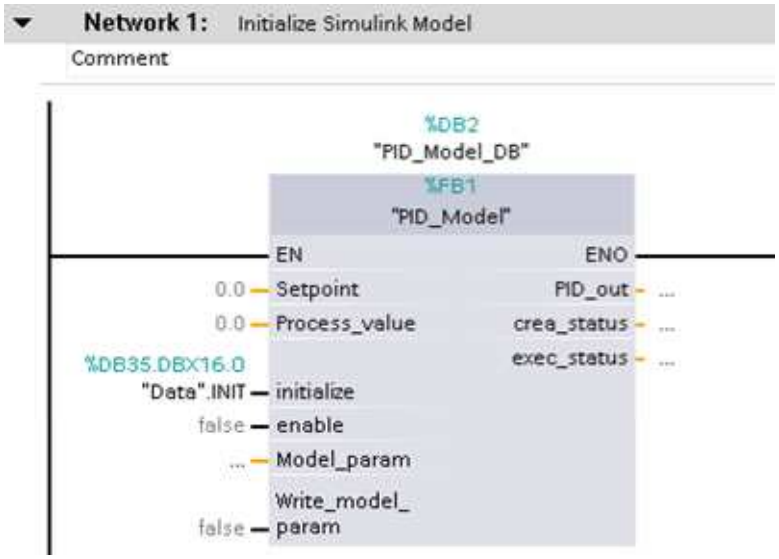
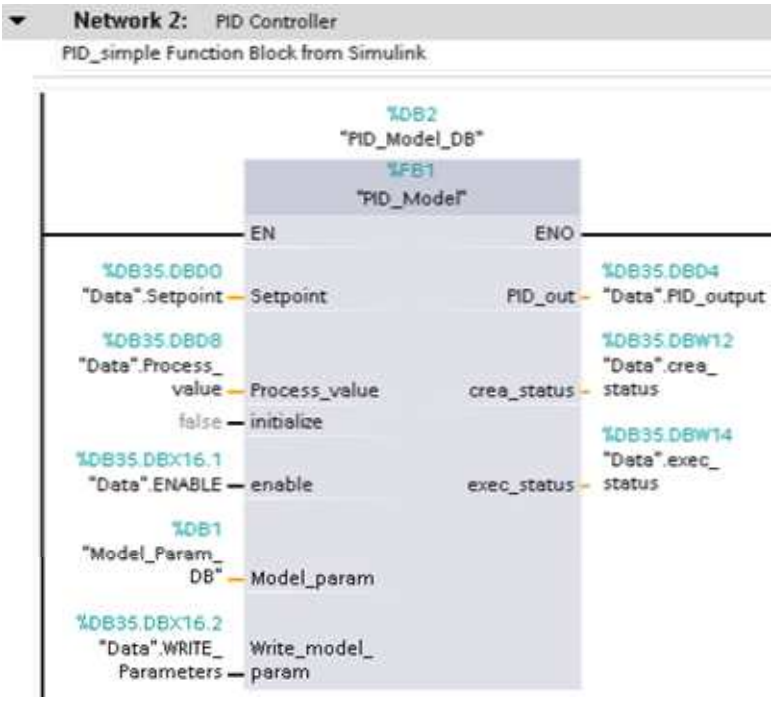
- **PID_Simple_rtdll_final**
  This program includes ready-to-use programming with integration of the PID controller from Simulink by WinAC Target. This program uses an RTDLL call.

8.3.3 Integrating the Simulink model into STEP 7 V11/V12

The following steps complete the "**PID_simple**" program with the "**PID_Model**" block from the Simulink model:

Table 8-4: Integrating the Simulink model into STEP 7 V11 / V12

Step	Instruction
1.	Open TIA Portal V11.
2.	Open the "**WinAC_Target_PID_Project**" sample project in the "**...\04_STEP7_V11_Project**" directory.
3.	Go to the "**Project View**" (bottom left).
4.	Expand the tree under "**PID_Simple [IPC427C PN]**".
5.	Expand the tree under "**WinAC RTX [CPU]**".
6.	Double-click to select "**Add new external file**" under the "**External source files**" folder.
7.	Select "**SCL Sources (*.scl)**".
8.	If you have already generated the SCL source and the DLL/RTDLL for STEP 7 V11 using WinAC Target, navigate to the directory where the SCL source is located. Alternatively, open the SCL source under "**…\02_ WinAC_Target_Code\PID_Model_V11_dll**" for the DLL call or "**…\02_WinAC_Target_Code\PID_Model_V11_rtdll**" for the RTDLL call
9.	The SCL source is now located in the STEP 7 V11 project. Right-click the "**PID_Model.scl**" SCL source.
10.	In the context menu, select "**Generate Blocks**". A function block is generated from the SCL source and stored in the "**Program Blocks**" folder.
11.	Open "**COMPLETE RESTART [OB100]**" and use drag and drop to insert the generated FB from the Project tree into network 1.
12.	Select "**OK**" to confirm the data block generation.

Step	Instruction
13.	Interconnect the inputs and outputs as shown in the figure. 
14.	Open “**CYC_INT5 [OB35]**” and use drag and drop to insert the generated FB from the Project tree into network 2.
15.	Select “**OK**” to confirm the data block generation.
16.	Interconnect the inputs and outputs as shown in the figure. 
17.	Download the program to WinAC RTX.

Step	Instruction
18.	Now copy the **DLL (PID_Model.dll)** or **RTDLL (PID_Model.rtdll)** to the **C:** directory of the PC system with WinAC RTX. RTDLL files have to be registered (see notes below).
19.	Set WinLC RTX to RUN mode.

Note

If you want to use a different path to store the DLL/RTDLL file, you have to modify the path also in the SCL source and regenerate the function block.

Variable: "**DLL_name**"

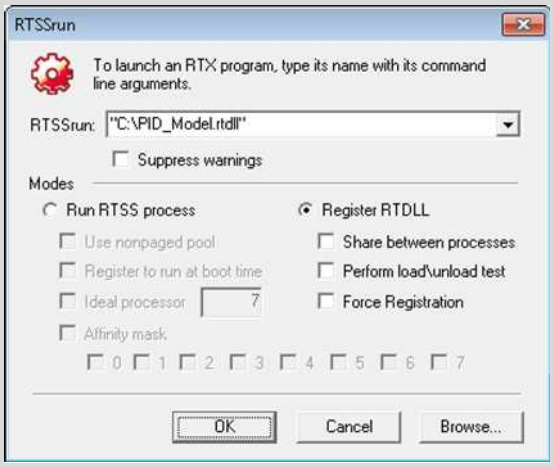
Alternatively, you can also change the variable directly in the generated function block

"**Interface – Static – DLL_name**"

Note

If an RTDLL is to be called in WinAC RTX, it must be registered in IntervalZero Runtime after copying to the runtime system.

In the PC system, select "**Start - Programs - IntervalZero - RTX 2009 - Tools – RtssRun**". Register the RTDLL file as shown in the figure.



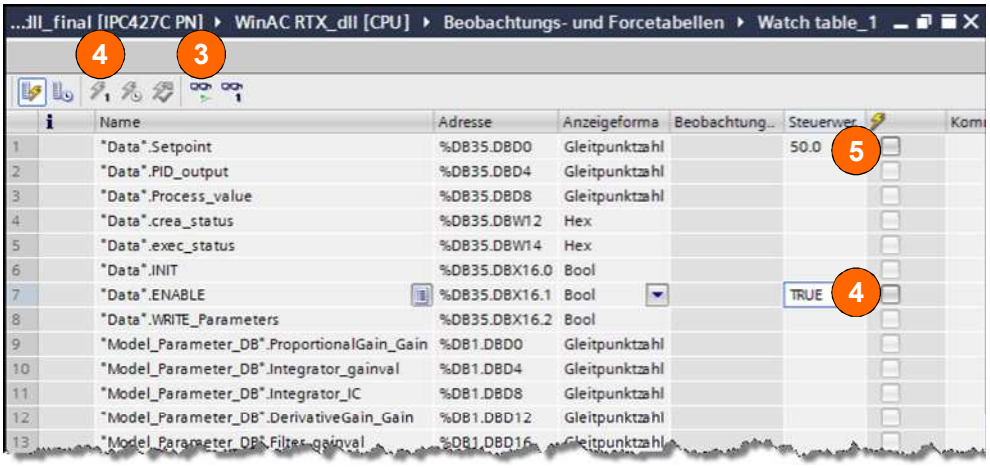
8.3.4 Commissioning

Once you have downloaded the program to WinAC RTX and copied the DLL/RTDLL file, perform the following steps for commissioning.

Table 8-5: Commissioning with STEP 7 V11/V12

Step	Instruction
1.	Set WinLC to RUN mode.
2.	In “**Watch and force table**”, open the “**Watch table_1**” variable table.
3.	Click the “**Monitor all**” icon
4.	Change the value of the “**enable**” variable to “**True**” and click the “**Modify all selected values once and now**” icon.
5.	Change the value of the “**Setpoint**” variable to the desired value, e.g. 50.0, and click the “**Modify all selected values once and now**” icon. Due to the setpoint change, the PID controller will output an output value. The process will change accordingly until it matches the set “**Setpoint**”.

Figure 8-3: Variable table



Note

The following variables allow you to diagnose errors.

The “**crea_status**” variable corresponds to **SFB65001.Status**
The “**exec_status**” variable corresponds to **SFB65002.Status**

For the error codes, please refer to the WinAC ODK manual.
http://support.automation.siemens.com/WW/view/en/35948966

9 External mode

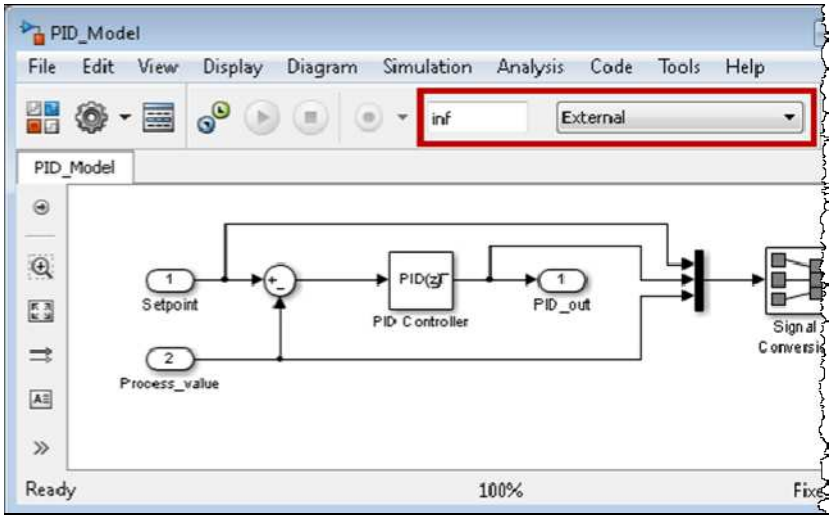
Note If you want to monitor and change parameters with External mode, you must have parameterized WinAC Target accordingly before encoding (see 7.2 Activating External mode).

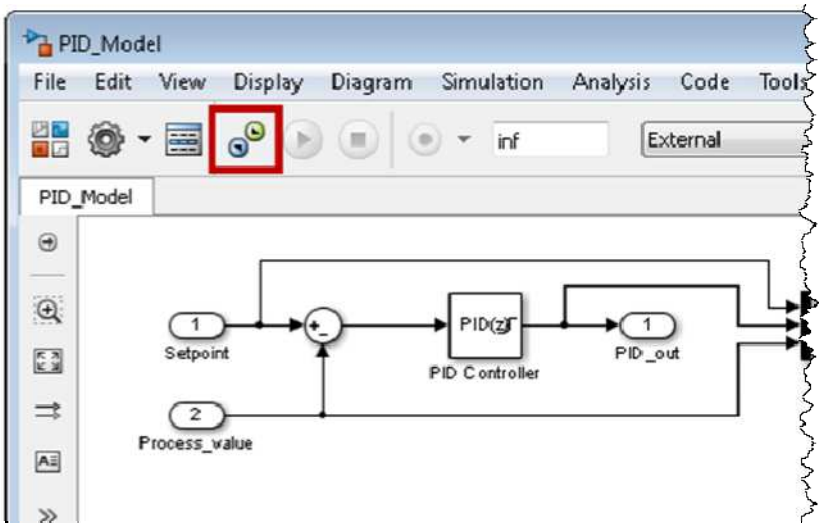
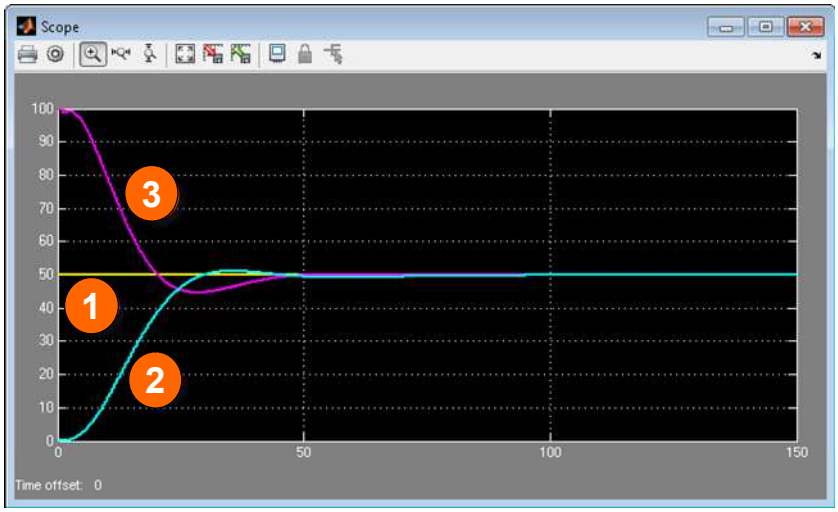
Note If you want to use External mode in conjunction with an RTDL file, additionally start the RTDLL Communication Service (see 9.3 RTDLL Communication Service for External mode).

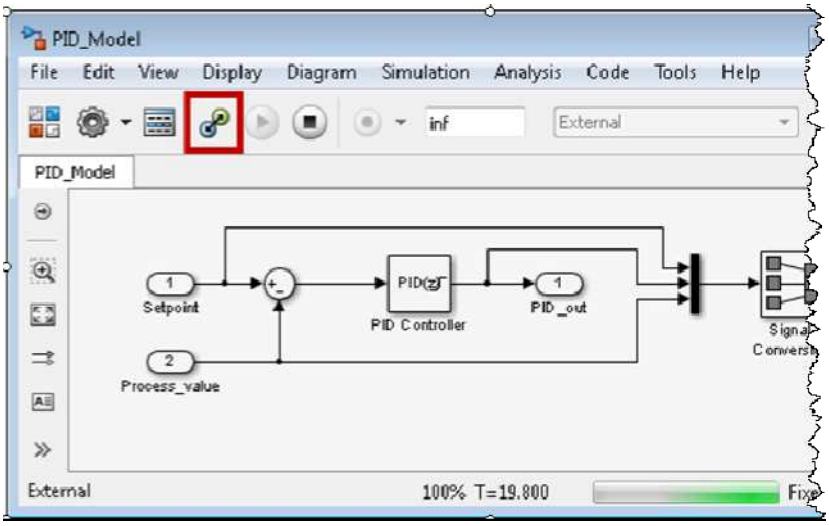
Note To use External mode, port 17725 must be enabled on the runtime system (here: IPC427C) when a firewall is enabled.

9.1 Monitoring

Table 9-1: Monitoring with External mode

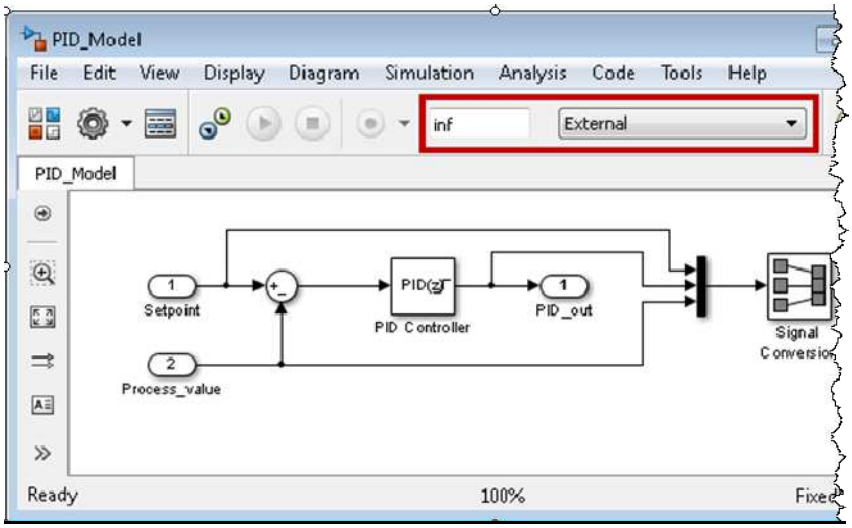
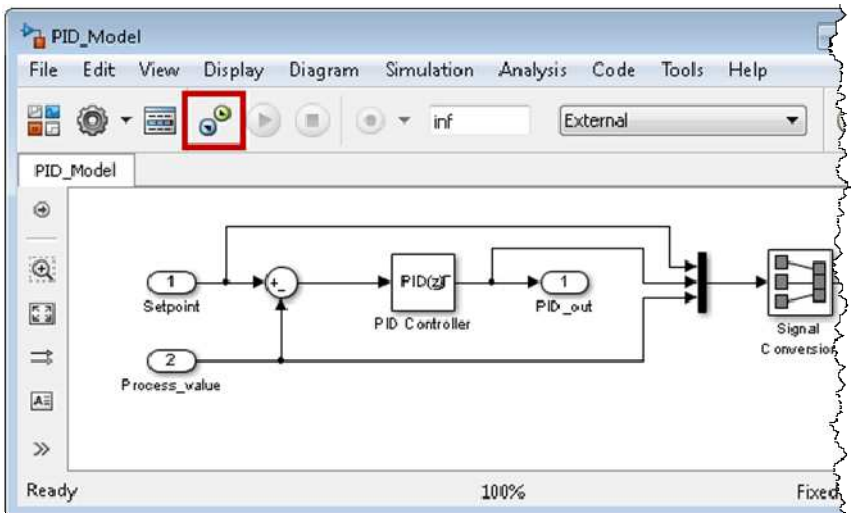
Step	Instruction
1.	Open the "**PID_Model.mdl**" model created with WinAC Target and transferred to WinAC RTX
2.	Set "**Simulation stop time**" to "**inf**" and "**Simulation Mode**" to "**External**". 

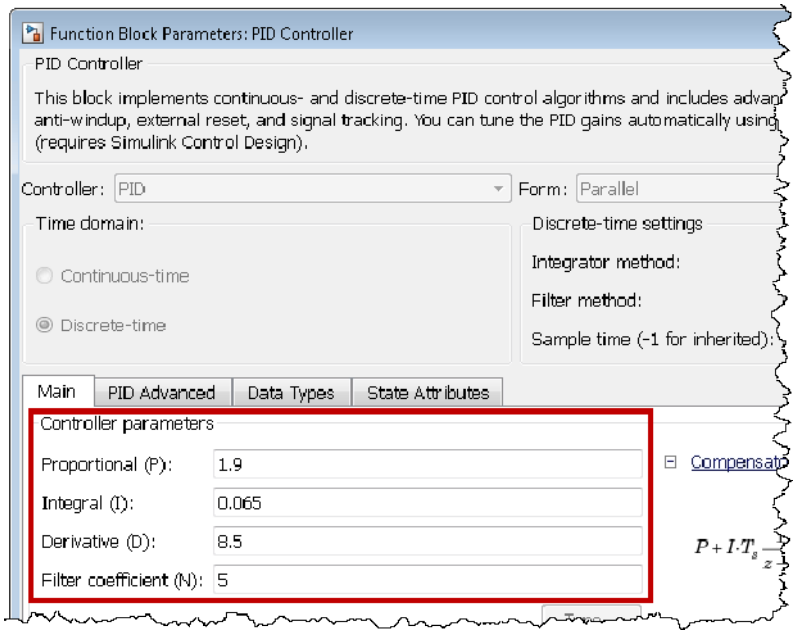
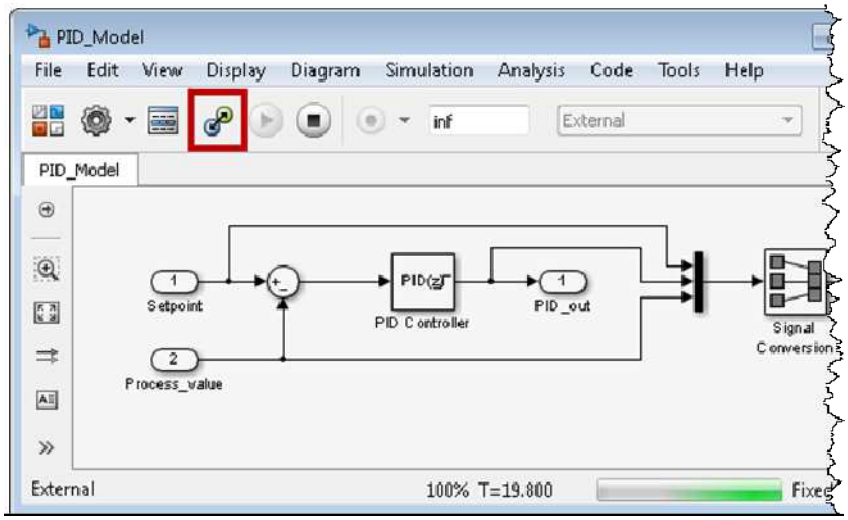
Step	Instruction
3.	Use the "**Connect To Target**" button to connect Simulink to the controller model on WinAC RTX 
4.	In the model, double-click to open the "**Scope**" function to graphically display the controller block signals.  1. Setpoint 2. Process_value 3. PID_out

Step	Instruction
5.	To disconnect the connection to the block in WinAC RTX, click the "**Disconnect From Target**" button. 

9.2 Changing internal parameters with Simulink

Table 9-2: Changing parameters with External mode

Step	Instruction
1.	Open the "**PID_Model.mdl**" model created with WinAC Target and transferred to WinAC RTX
2.	Set "**Simulation stop time**" to "**inf**" and "**Simulation Mode**" to "**External**". 
3.	Use the "**Connect To Target**" button to connect Simulink to the controller model on WinAC RTX. 

Step	Instruction
4.	To change controller block parameters, double-click to open "**PID Controller**".  In this dialog box, you can adjust the controller parameters.
5.	To disconnect the connection to the block in WinAC RTX, click the "**Disconnect From Target**" button. 
6.	While WinLC RTX is in **RUN** mode, you can connect to the block in the PLC with External mode in the Simulink model. As soon as a general WinLC RTX reset is performed, the communication port of the PLC will be closed for External mode.

Note

Parameter changes with External mode are valid only until the next restart of WinAC RTX.

If you want to permanently save parameter changes, re-encode the DLL/RTDLL and save it on the runtime system. In this case, it is not necessary to reintegrate the SCL source into the S7 program as the model interfaces have not changed.

Note

A general WinLC RTX reset closes the port for communication with the Simulink model. This operation may take longer. As soon as the general reset has been completed and the port has been closed, the STOP LED lights up.

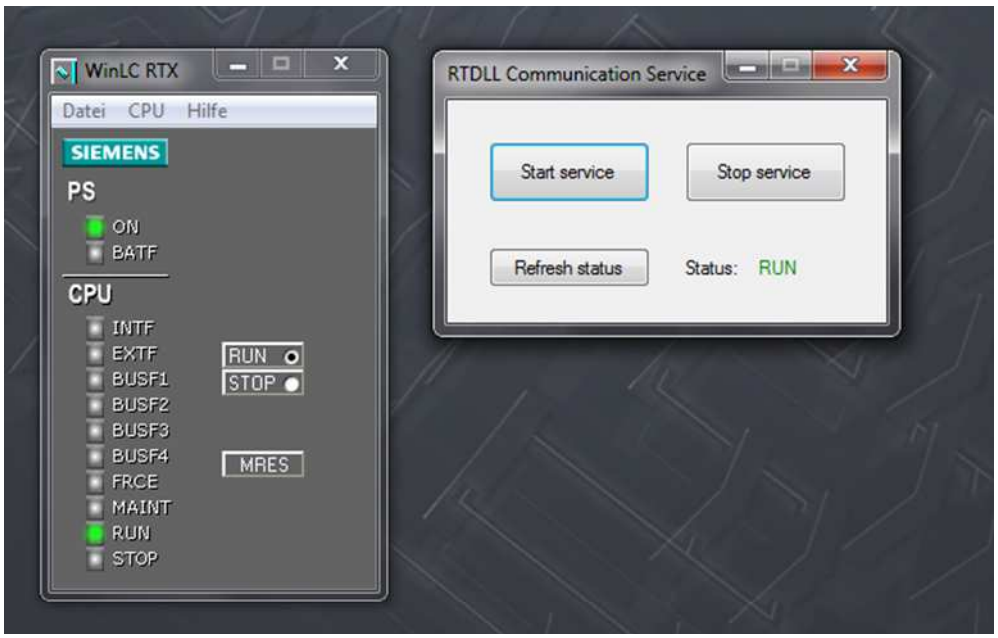
9.3 RTDLL Communication Service for External mode

If you are using External mode in conjunction with an RTDLL file, you have to start the RTDLL Communication Service on the runtime system before going online with External mode. This program enables a service that implements the communication between the IntervalZero real-time environment and Windows. This allows data exchange between Simulink and the RTDLL file. To enable the RTDLL Communication Service, perform the following steps.

Table 9-3: External mode with an RTDLL file

Step	Instruction
1.	Click the engineering PC where WinAC Target is installed: Windows 7 "**Start->All Programs -> SIMATIC -> WinAC Target -> RTDLL Communication Service**" Windows XP: "**Start -> SIMATIC -> WinAC Target -> RTDLL Communication Service**"
2.	Copy the entire contents of the open window to the runtime system.
3.	Run the "**RTDLL_Com_Service.exe**" file.
4.	Select the "**Start service**" button to start the service.

Figure 9-1: WinLC RTX panel / RTDLL Communication Service for External mode

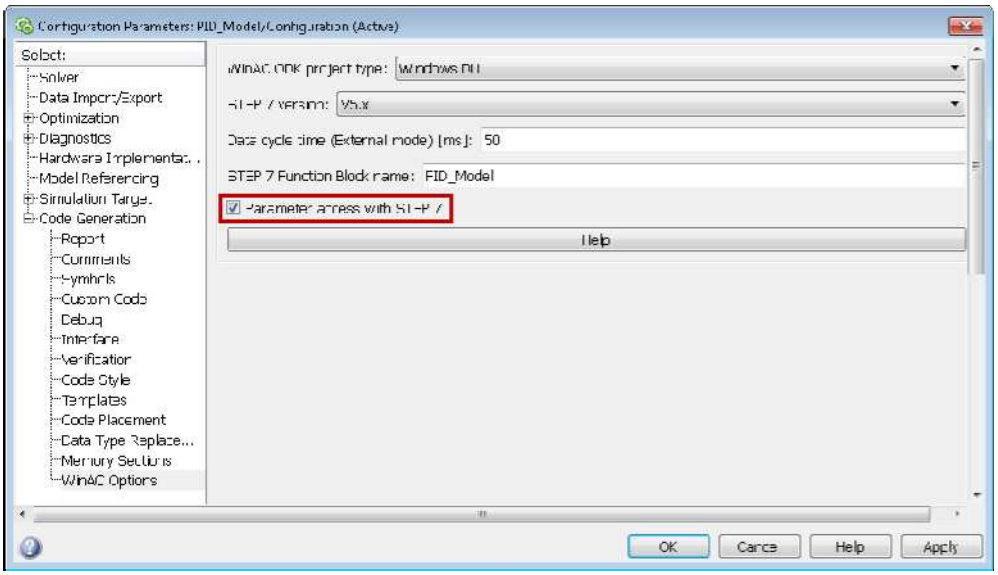


10 Manipulating model parameters using STEP 7

10.1 Enabling parameter access for STEP 7

To allow access to the internal parameters of the Simulink models in the STEP 7 user program, parameter access for STEP 7 must be enabled in WinAC Target before generating the DLL/RTDLL file.

Figure 10-1: WinAC Options



Note The maximum length for a data block comment in STEP 7 V5.x is 79 characters. When composing the comment from the Simulink model information, this limit may be exceeded.
Compiling the SCL source generates all blocks and reduces the relevant location in the comment to the maximum number. The complete comment is retained in the SCL source.
A respective warning is output when compiling the SCL source.

Note In data blocks for STEP 7 V5.x, the maximum length of variable names is limited to 24 characters. The original parameter names from the Simulink model are used for the variable names in STEP 7. If these names are too long for STEP 7 V5.x, they will be automatically truncated. For security, the original parameter name is in the respective comment.

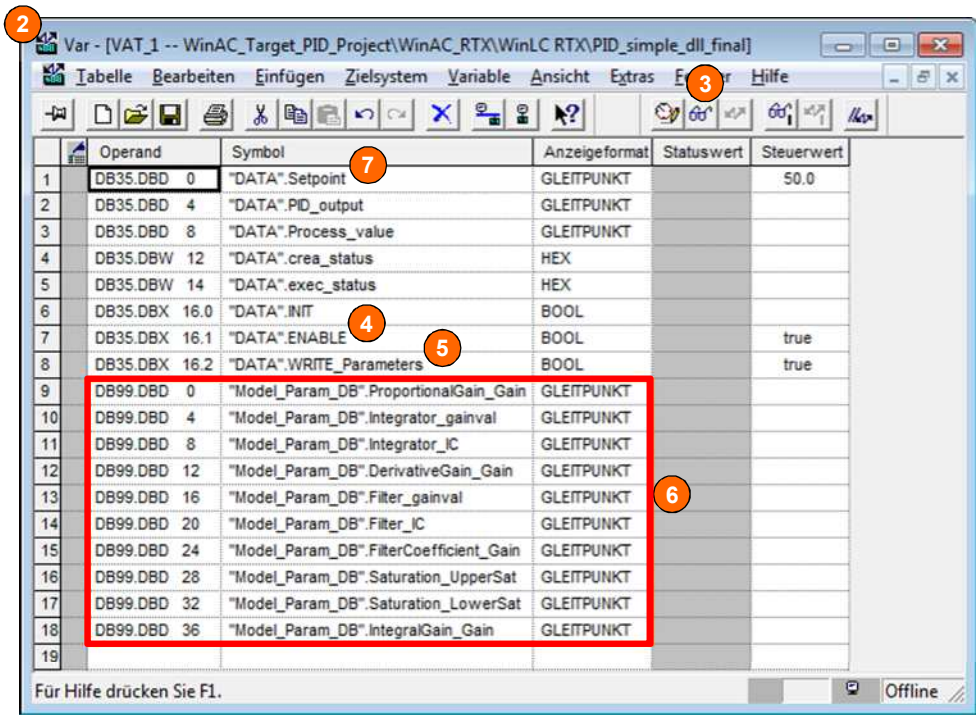
10.2 Manipulating internal model parameters using STEP 7 V5.5

Once you have downloaded the program to WinAC RTX, perform the following steps for manipulation.

Table 10-1: Commissioning with STEP 7 V5.5

Step	Instruction
1.	Set WinLC to RUN mode.
2.	Open the "**VAT_1**" variable table (see Figure 8-2).
3.	Click the "**Monitor Variable**" icon.
4.	Change the value of the "**ENABLE**" variable to "**TRUE**" and click the "**Modify Variable**" icon. The Simulink model parameters are read and saved in the "**Model_Param_DB**" data block.
5.	Change the value of the "**WRITE_Parameters**" variable to "**TRUE**" and click the "**Modify Variable**" icon.
6.	Change the values of "**Model_Param_DB**" to the desired values and click the "**Modify Variable**" icon. The PID controller is run with these changed parameter values.
7.	Change the value of the "**Setpoint**" variable to the desired value, e.g. **50**, and click the "**Modify Variable**" icon. Due to the setpoint change, the PID controller will output an output value. The process will change accordingly until it matches the "**Setpoint**".

Figure 10-2: Variable table



Note

Parameter changes with STEP 7 are valid only until the next restart of WinAC RTX.

If you want to permanently save parameter changes, re-encode the DLL/RTDLL and save it on the runtime system. In this case, it is not necessary to reintegrate the SCL source into the S7 program as the model interfaces have not changed.

NOTICE

When manipulating Simulink model parameters using STEP 7, these changes will not be updated when using External mode in Simulink. Furthermore, changes to parameters cannot be made via External mode as long as the "WRITE_Parameters" variable is set to "TRUE".

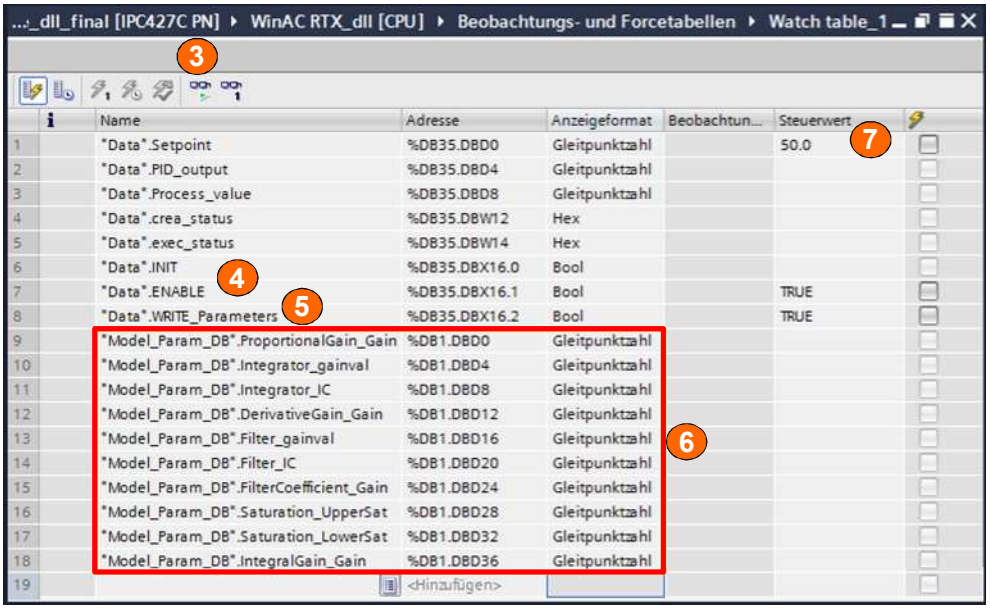
10.3 Manipulating internal model parameters using STEP 7 V11/V12

Once you have downloaded the program to WinAC RTX and copied the DLL/RTDLL file, perform the following steps for commissioning.

Table 10-2: Commissioning with STEP 7 V11/V12

Step	Instruction
1.	Set WinLC to RUN mode.
2.	In "**Watch and force table**", open the "**Watch table_1**" variable table.
3.	Click the "**Monitor all**" icon.
4.	Change the value of the "**ENABLE**" variable to "**TRUE**" and click the "**Modify all selected values once and now**" icon. The Simulink model parameters are read and saved in the "**Model_Param_DB**" data block.
5.	Change the value of the "**WRITE_Parameters**" variable to "**TRUE**" and click the "**Modify all selected values once and now**" icon.
6.	Change the values of "**Model_Param_DB**" to the desired values and click the "**Modify all selected values once and now**" icon. The PID controller is run with these changed parameter values.
7.	Change the value of the "**Setpoint**" variable to the desired value, e.g. 50.0, and click the "**Modify all selected values once and now**" icon. Due to the setpoint change, the PID controller will output an output value. The process will change accordingly until it matches the set "**Setpoint**".

Figure 10-3: Variable table



Note

Parameter changes with STEP 7 are valid only until the next restart of WinAC RTX.

If you want to permanently save parameter changes, re-encode the DLL/RTDLL and save it on the runtime system. In this case, it is not necessary to reintegrate the SCL source into the S7 program as the model interfaces have not changed.

NOTICE

When manipulating Simulink model parameters using STEP 7, these changes will not be updated when using External mode in Simulink. Furthermore, changes to parameters cannot be made via External mode as long as the "WRITE_Parameters" variable is set to "TRUE".

10.4 Parameters in Simulink and STEP 7

To allow exact assignment between the parameters in the Simulink model und the parameters in STEP 7, the "**Model_Param_DB**" data block was provided with comments.

The comments are structured as follows:

Table 10-3: Structure of "Model_Param_DB" comments

Step	Instruction
1.	Name of the subsystem where the parameter is located
2.	Name of the block where the parameter is located
3.	Complete internal name of the Simulink model parameter

Figure 10-4: "Model_Param_DB" data block in STEP 7

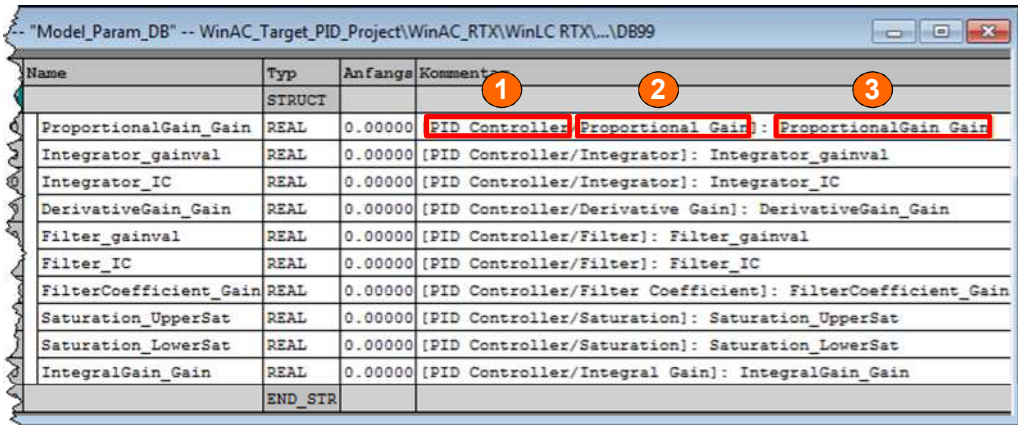
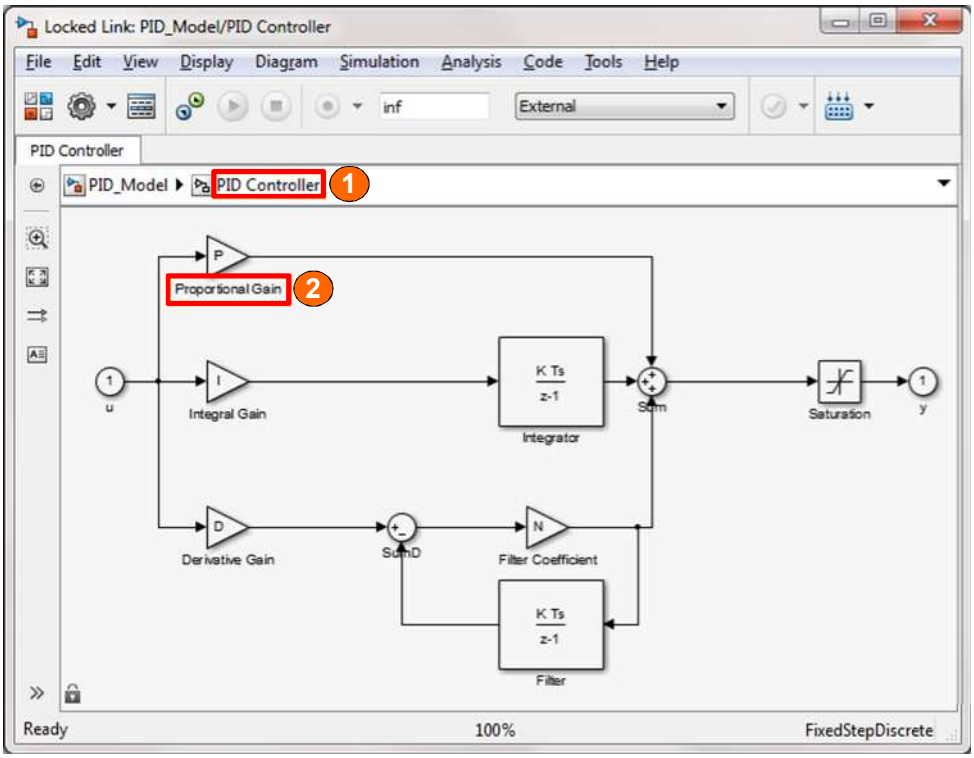


Figure 10-5: Simulink model parameters



Note

The maximum length for a data block comment in STEP 7 V5.x is 79 characters. When composing the comment from the Simulink model information, this limit may be exceeded.
Compiling the SCL source generates all blocks and reduces the relevant location in the comment to the maximum number. The complete comment is retained in the SCL source.
A respective warning is output when compiling the SCL source.

Note

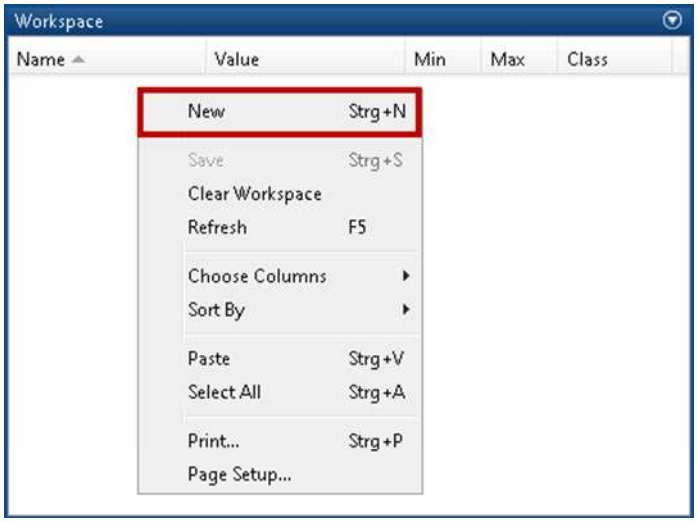
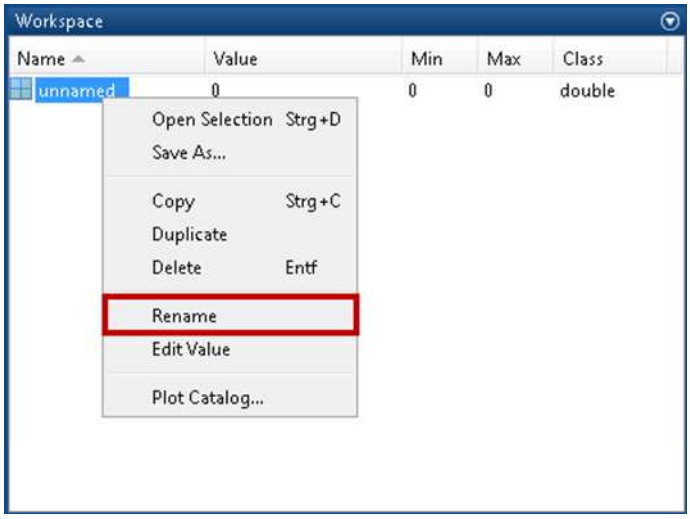
In data blocks for STEP 7 V5.x, the maximum length of variable names is limited to 24 characters. The original parameter names from the Simulink model are used for the variable names in STEP 7. If these names are too long for STEP 7 V5.x, they will be automatically truncated. For security, the original parameter name is in the respective comment.

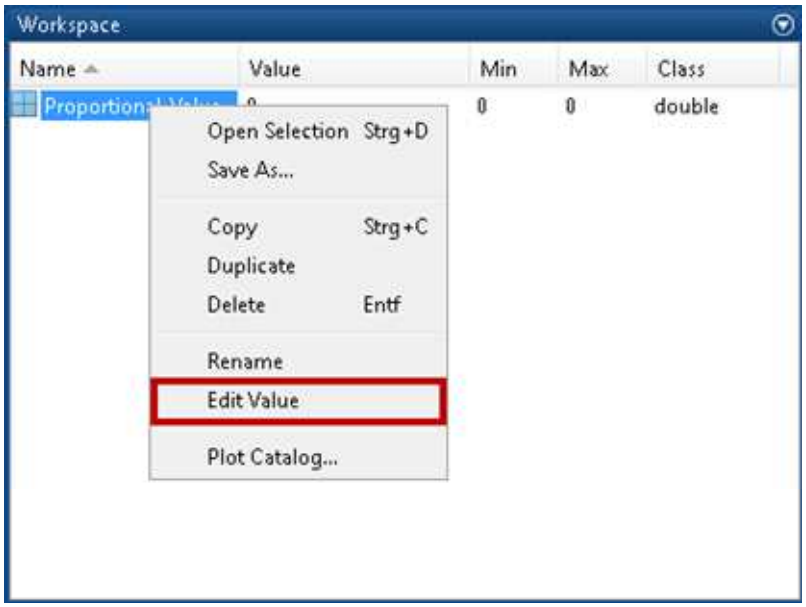
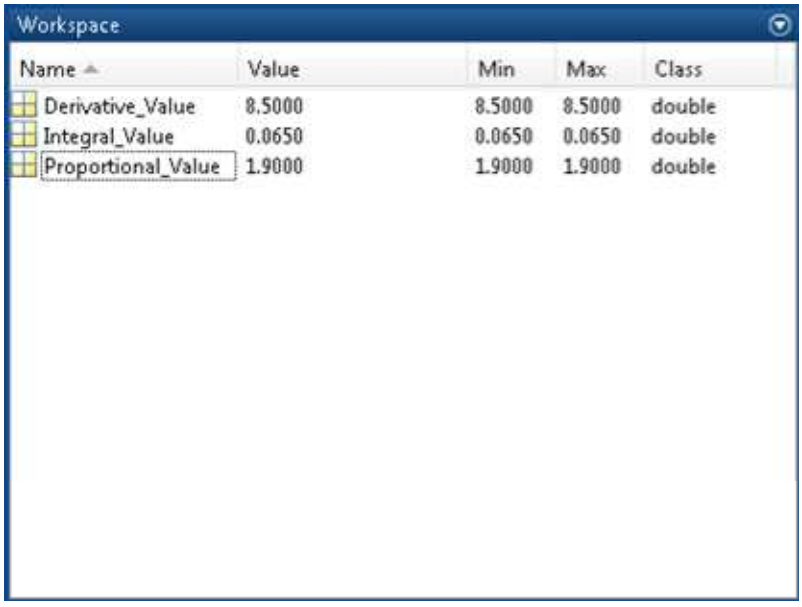
11 Manipulating User-Defined Parameters with STEP 7

In MATLAB, you can create variables and interconnect them as parameters in the Simulink model. WinAC Target and STEP 7 then enable you to manipulate these user-defined parameters.

11.1 Creating user-defined parameters

To allow access to the user-defined parameters of the Simulink models in the STEP 7 user program, the MATLAB variables must be created in the MATLAB workspace before generating the DLL/RTDLL file.
Table 11-1: Creating user-defined parameters

Step	Instruction
1.	Right-click on a free space in the MATLAB workspace and click on **"New"** to create a new MATLAB variable. 
2.	Right-click on the MATLAB variable you have created and select **"Rename"**. Change the name to **"Proportional_Value"**. 

Step	Instruction
3.	Right-click on the MATLAB variable you have created and select **"Edit Value"**. Change the value to **"1.9"**. 
4.	Perform these steps for the **"Integral_Value"** and **"Derivative_Value"** variables and assign the values as follows. 

Note

The created MATLAB variables are not saved in the Simulink model. After restarting MATLAB, they are no longer available. The MATLAB variables need to be created again.

You can export MATLAB variables. For more information, please refer to the MATLAB Help.

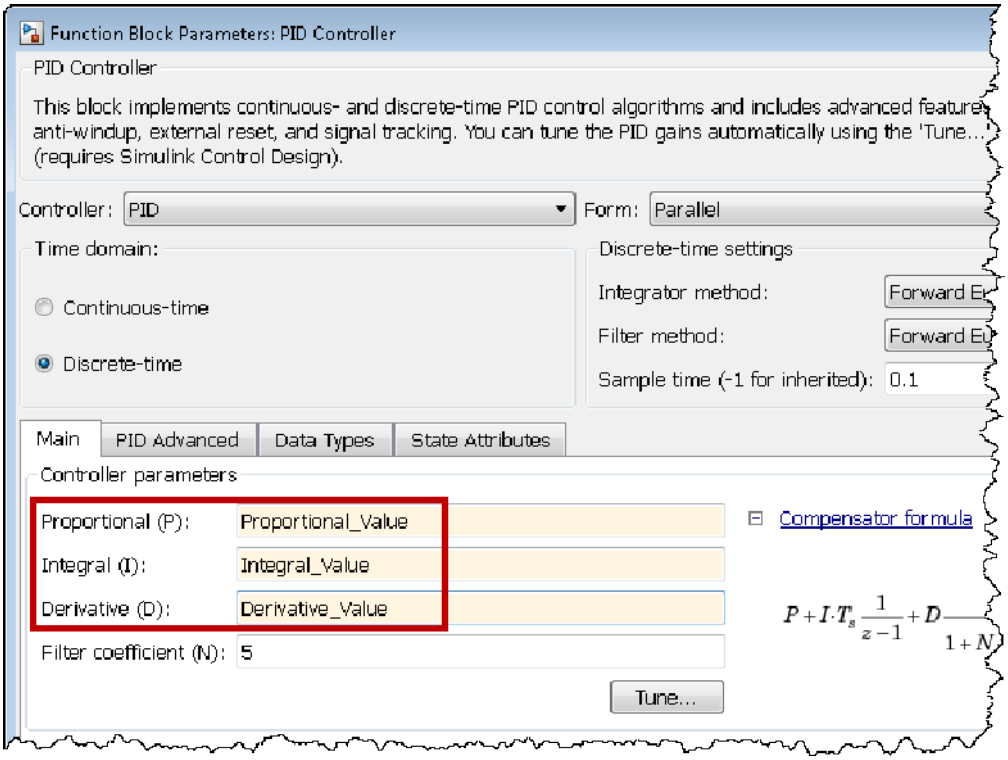
NOTE External mode is not available with structures as user-defined parameters in a Simulink model.

Simulink resp.External Mode does not support uploading of signal with structure type.

11.2 Interconnecting user-defined parameters

The figure below shows the interconnection of the created MATLAB variables with the block parameters of “PID Controller”.

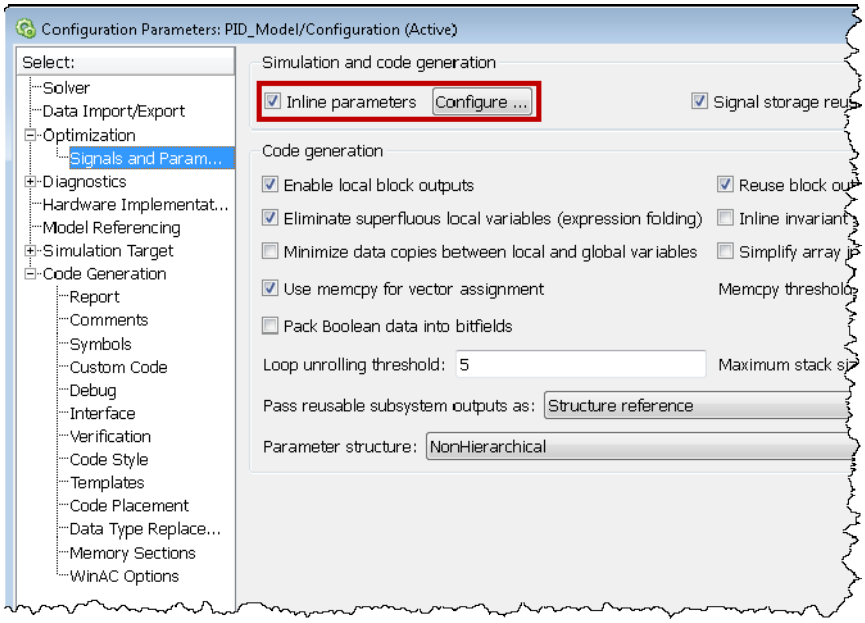
Figure 11-1: Interconnecting user-defined parameters

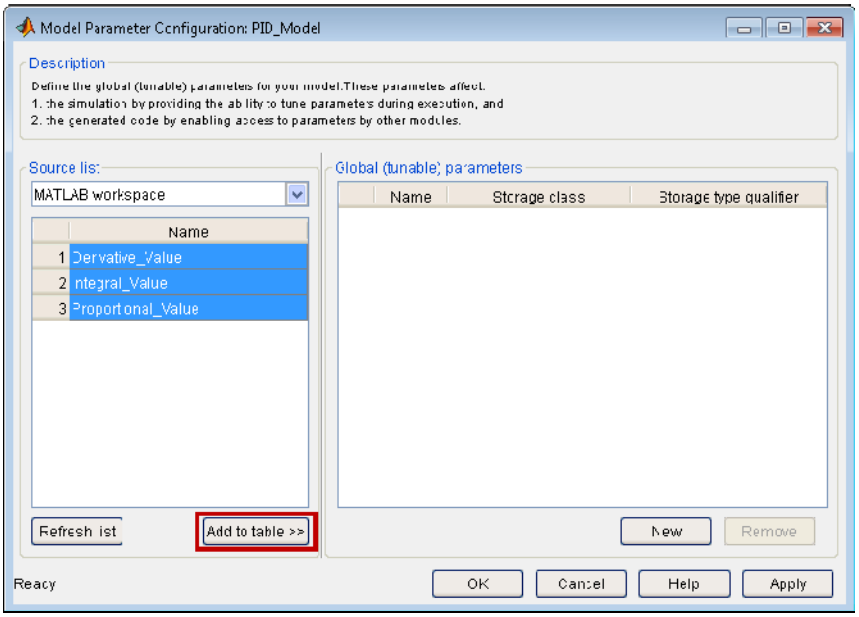


11.3 Configuring user-defined parameters

To allow access to the user-defined parameters of the Simulink models in the STEP 7 user program, the MATLAB variables must be configured as model parameters before generating the DLL/RTDLL file.

Table 11-2: Configuring user-defined parameters

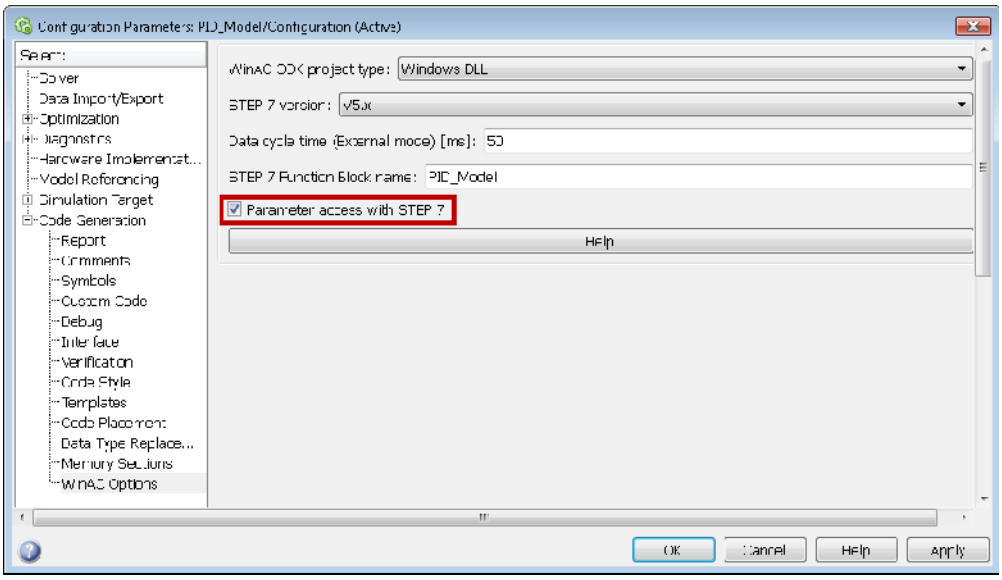
Step	Instruction
5.	In the Simulink model, open **"Code Generation Options"**
6.	Open the **"Optimization"** navigation item and select the **"Signals and Parameters"** item
7.	Check **"Inline Parameters"** and click on **"Configure …"** 

Step	Instruction
8.	Select the listed variables from the **"MATLAB workspace"** and click on **"Add to table >>"** 
9.	Close the dialog with "OK".

11.4 Enabling parameter access for STEP 7

To allow access to the user-defined parameters of the Simulink models in the STEP 7 user program, parameter access for STEP 7 must be enabled with WinAC Target before generating the DLL/RTDLL file.

Figure 11-2: Enabling parameter access for STEP 7



Note

In data blocks of STEP 7 V5.x, the maximum length of variable names is limited to 24 characters. The original parameter names from the Simulink model are used for the variable names in STEP 7. If these names are too long for STEP 7 V5.x, they will be automatically truncated. For security, the original parameter name is in the respective comment.

Note

The maximum length for a data block comment in STEP 7 V5.x is 79 characters. When composing the comment from the Simulink model information, this limit may be exceeded.
Compiling the SCL source generates all blocks and reduces the relevant location in the comment to the maximum number. The complete comment is retained in the SCL source.
A respective warning is output when compiling the SCL source.

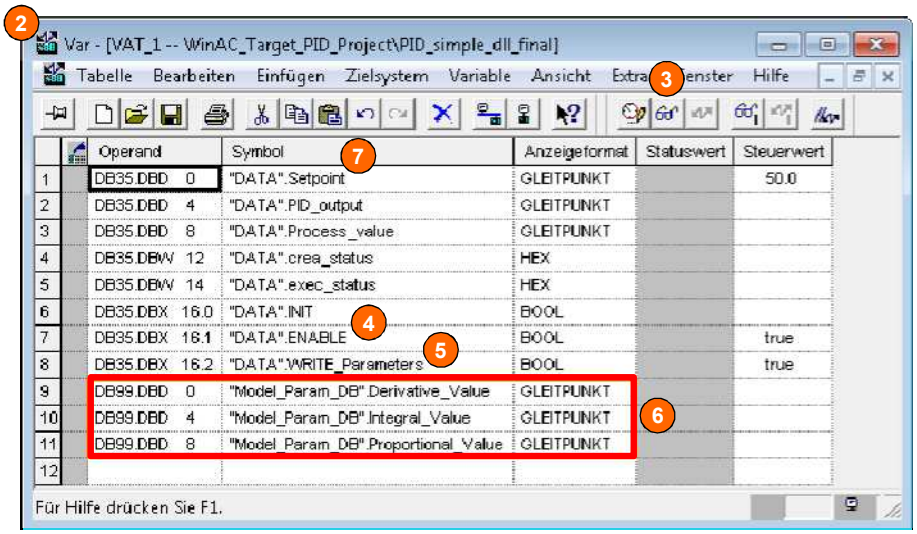
11.5 Manipulating User-Defined Parameters with STEP 7 V5.5

Once you have downloaded the program to WinAC RTX, perform the following steps for manipulation.

Table 11-3: Commissioning with STEP 7 V5.5

Step	Instruction
1.	Set WinLC to RUN mode.
2.	Open the "**VAT_1**" variable table.
3.	Click on the "**Monitor Variable**" icon.
4.	Change the value of the "**ENABLE**" variable to "**TRUE**" and click on the "**Modify Variable**" icon. The Simulink model parameters are read and saved in the "**Model_Param_DB**" data block.
5.	Change the value of the "**WRITE_Parameters**" variable to "**TRUE**" and click on the "**Modify Variable**" icon.
6.	Change the values of "**Model_Param_DB**" to the desired values and click on the "**Modify Variable**" icon. The PID controller is run with these changed parameter values.
7.	Change the value of the "**Setpoint**" variable to the desired value, e.g. **50**, and click on the "**Modify Variable**" icon. Due to the setpoint change, the PID controller will output an output value. The process will change accordingly until it matches the "**Setpoint**".

Figure 11-3: Variable table



Note

Parameter changes with STEP 7 are valid only until the next restart of WinAC RTX.

If you want to permanently save parameter changes, re-encode the DLL/RTDLL file and save it on the runtime system. In this case, it is not necessary to reintegrate the SCL source into the S7 program as the model interfaces have not changed.

NOTICE

When manipulating Simulink model parameters using STEP 7, these changes will not be updated when using External mode in Simulink. Furthermore, changes to parameters cannot be made via External mode as long as the "WRITE_Parameters" variable is set to "TRUE".

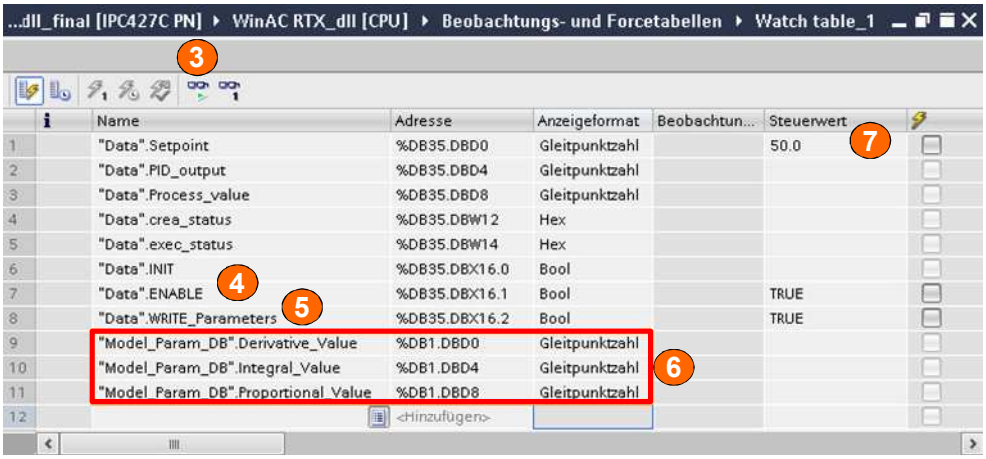
11.6 Manipulating User-Defined Parameters with STEP 7 V11/V12

Once you have downloaded the program to WinAC RTX and copied the DLL/RTDLL file, perform the following steps for commissioning.

Table 11-4: Commissioning with STEP 7 V11/V12

Step	Instruction
1.	Set WinLC to RUN mode.
2.	In "**Watch and force table**", open the "**Watch table_1**" variable table.
3.	Click on the "**Monitor all**" icon.
4.	Change the value of the "**ENABLE**" variable to "**TRUE**" and click on the "**Modify all selected values once and now**" icon. The Simulink model parameters are read and saved in the "**Model_Param_DB**" data block.
5.	Change the value of the "**WRITE_Parameters**" variable to "**TRUE**" and click on the "**Modify all selected values once and now**" icon.
6.	Change the values of "**Model_Param_DB**" to the desired values and click on the "**Modify all selected values once and now**" icon. The PID controller is run with these changed parameter values.
7.	Change the value of the "**Setpoint**" variable to the desired value, e.g. 50.0, and click on the "**Modify all selected values once and now**" icon. Due to the setpoint change, the PID controller will output an output value. The process will change accordingly until it matches the set "**Setpoint**".

Figure 1111-4: Variable table



Note

Parameter changes with STEP 7 are valid only until the next restart of WinAC RTX.

If you want to permanently save parameter changes, re-encode the DLL/RTDLL file and save it on the runtime system. In this case, it is not necessary to reintegrate the SCL source into the S7 program as the model interfaces have not changed.

NOTICE

When manipulating Simulink model parameters using STEP 7, these changes will not be updated when using External mode in Simulink. Furthermore, changes to parameters cannot be made via External mode as long as the "WRITE_Parameters" variable is set to "TRUE".

11.7 User-defined parameters in Simulink and STEP 7

To allow exact assignment between the parameters in the Simulink model and the parameters in STEP 7, the "**Model_Param_DB**" data block was provided with comments.

The comments are structured as follows:

Table 11-5: Structure of "Model_Param_DB" comments

Step	Instruction
1.	Name of the subsystem where the parameter is located
2.	Name of the block where the parameter is located
3.	Complete internal name of the user-defined Simulink model parameter

Figure 11-5: "Model_Param_DB" data block in STEP 7

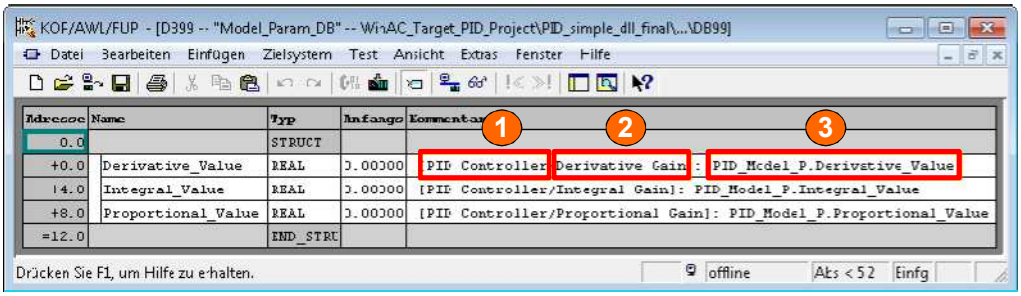
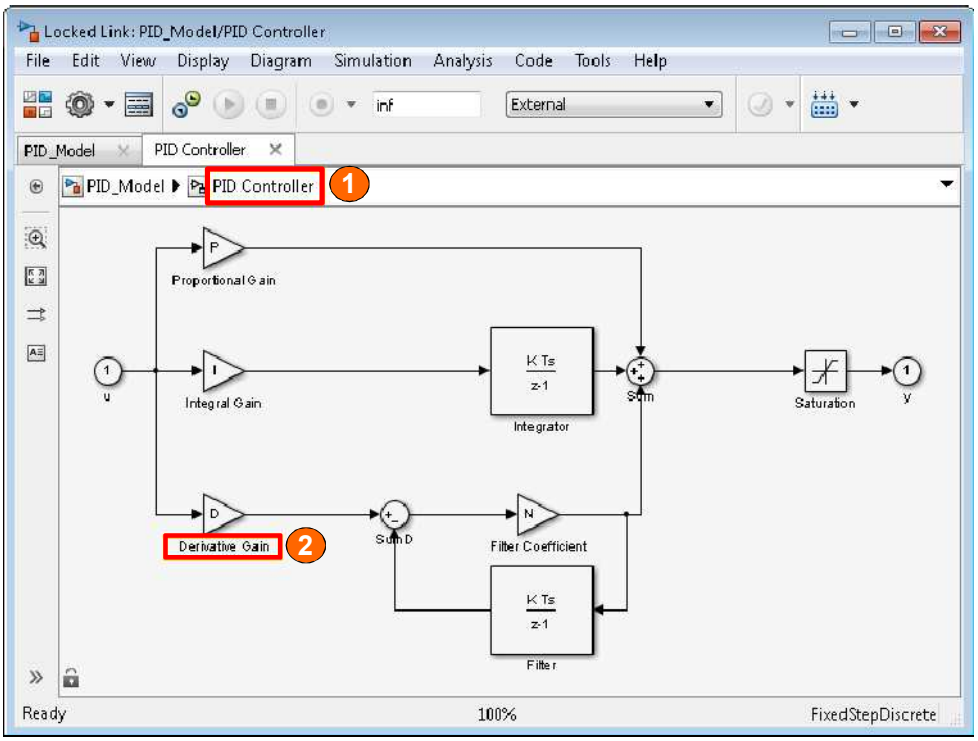


Figure 11-6: Simulink model parameter



Note

In data blocks in STEP 7 V5.x, the maximum length of variable names is limited to 24 characters. The original parameter names from the Simulink model are used for the variable names in STEP 7. If these names are too long for STEP 7 V5.x, they will be automatically truncated. For security, the original parameter name is in the respective comment.

Note

The maximum length for a data block comment in STEP 7 V5.x is 79 characters. When composing the comment from the Simulink model information, this limit may be exceeded.
Compiling the SCL source generates all blocks and reduces the relevant location in the comment to the maximum number. The complete comment is retained in the SCL source.
A respective warning is output when compiling the SCL source.

12 Error diagnostics

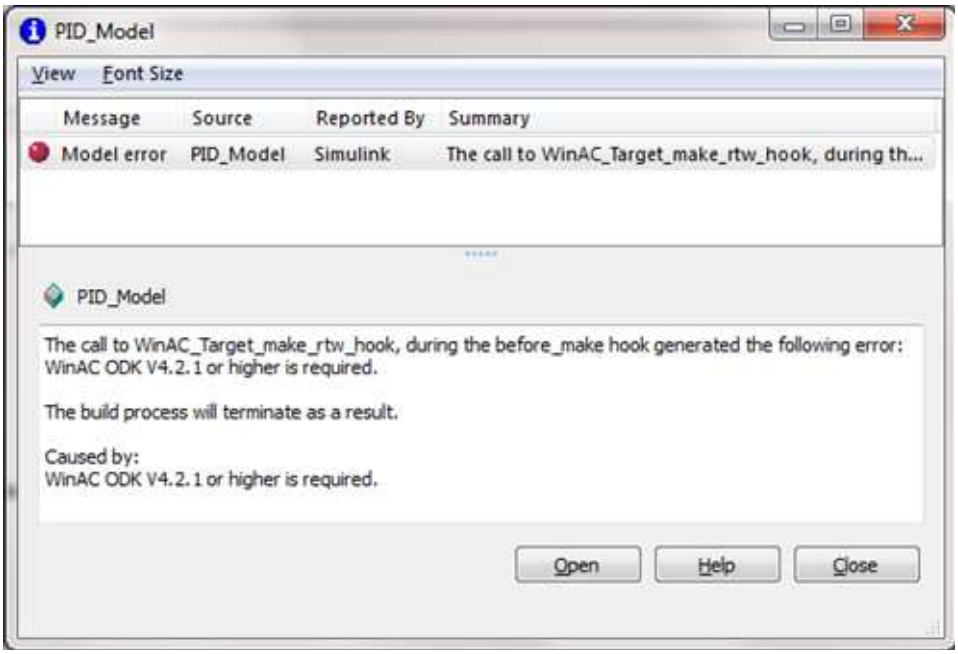
Different problems may occur when using WinAC Target. The aim of this chapter is to help you with diagnostics.

12.1 Encoding with WinAC Target

When generating codes with WinAC Target, the following error reports may occur.

Error screen

Figure 12-1: Error screen



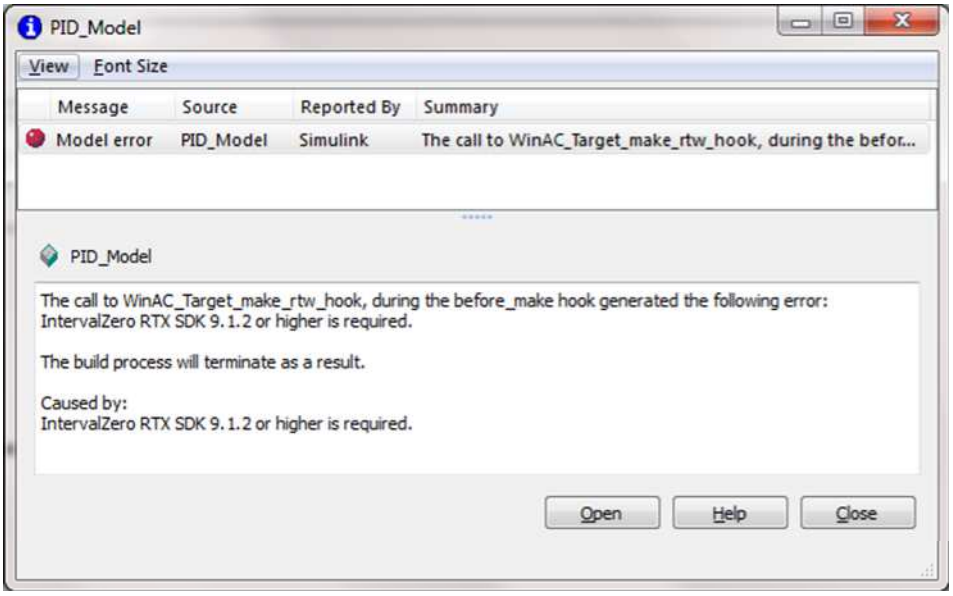
Error diagnostics

This error message appears during code generation when WinAC ODK V4.2.1 is not installed.

Install the missing software.

Error screen

Figure 12-2: Error screen



Error diagnostics

This error message appears during code generation when IntervalZero RTX SDK 9.1.2 is not installed and the “Real-Time DLL (RTDLL)” option was selected in WinAC Target.

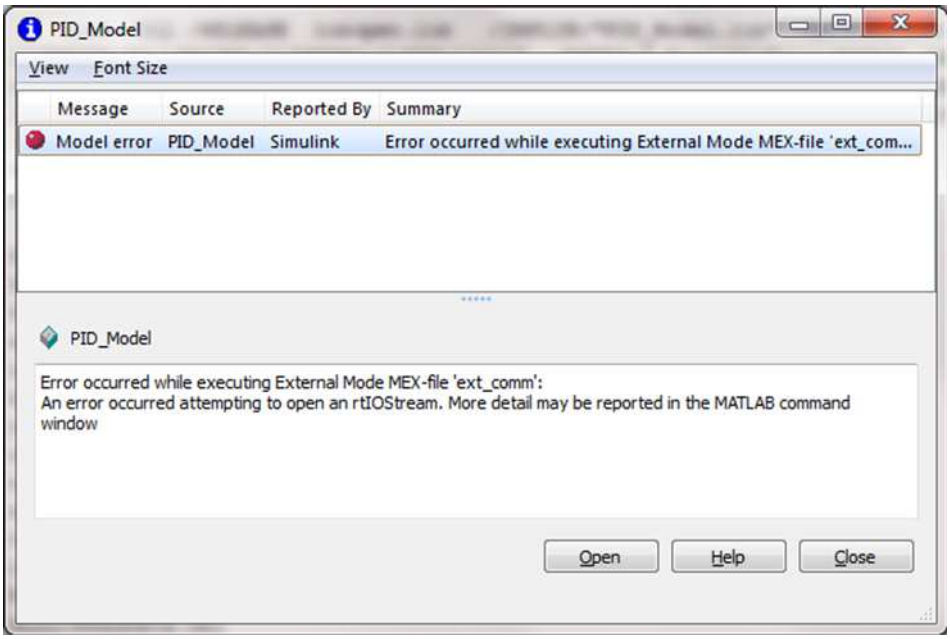
Install the missing software or select “**DLL**” in the WinAC Target options.

12.2 Connecting to External mode

When connecting to External mode, the following error messages may occur.

Error screen

Figure 12-3: Error screen



Error diagnostics

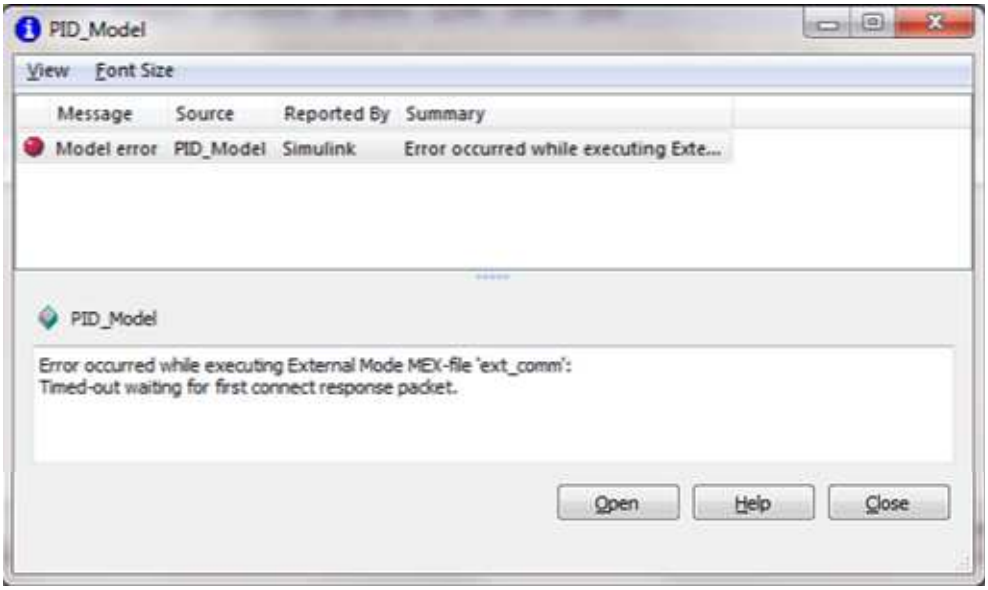
This error message indicates that Simulink could not connect to the set communication partner.

Table 12-1: Error diagnostics

No.	Instruction
1.	Make sure that you have set the correct IP address in the code generation options. (See Chapter 7.3.1)
2.	Make sure that the DLL runs with compiled External mode in WinAC RTX.
3.	When using a real-time DLL (RTDLL), make sure that the Communication Service is started.
4.	Make sure that port 17725 of your firewall is enabled on the runtime system.

Error screen

Figure 12-4: Error screen



Error diagnostics

Table 12-2: Error diagnostics

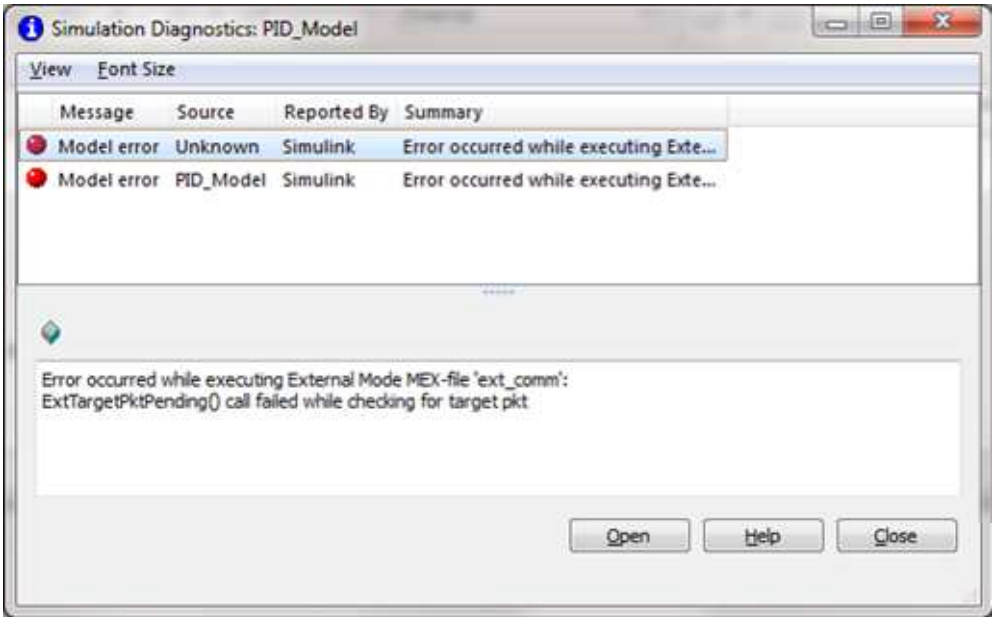
No.	Instruction
1.	Make sure that the S7 program has been downloaded to the PLC.
2.	Make sure that the DLL/RTDLL has been initialized.
3.	Make sure that WinLC is in RUN mode and that the controller is running.

12.3 During operation

During operation, the following error messages may occur.

Error screen

Figure 12-5: Error screen



Error diagnostics

If this error message is displayed when running External mode, make sure that the development computer and the target system are connected to one another.

13 References

This list is not complete and only presents a selection of relevant information.

Table 13-1: Internet links

	Topic	Title
1	Reference to the entry	http://support.automation.siemens.com/WW/view/en/56969417
2	Windows Automation Center Open Development Kit (WinAC ODK) V4.1	http://support.automation.siemens.com/WW/view/en/19256518
3	SIMATIC Embedded Automation SIMATIC IPC427C Bundles, SIMATIC HMI IPC477C Bundles	http://support.automation.siemens.com/WW/view/en/37954208
4	MathWorks home page	http://www.mathworks.com/
5	Microsoft home page	http://www.microsoft.com/
6	IntervalZero home page	http://www.mathworks.com/
7	PLC programming software for the entire SIMATIC controller range	http://www.automation.siemens.com/mcms/simatic-controller-software/en/Pages/Default.aspx
8	SIMATIC PC-based Controller	http://www.automation.siemens.com/mcms/programmable-logic-controller/en/software-plc/Pages/Default.aspx
9	SIMATIC Industry PC SIMATIC IPC427C	http://support.automation.siemens.com/WW/view/en/37028954
10	STEP 7 – manuals	http://support.automation.siemens.com/WW/view/en/10805384/133300
11	SIMATIC WinAC ODK – manuals	http://support.automation.siemens.com/WW/view/en/12840073/133300
12	SIMATIC WinAC RTX – manuals	http://support.automation.siemens.com/WW/view/en/10805641/133300
13	Windows Automation Center RTX Open Development Kit (WinAC ODK) – manual	http://support.automation.siemens.com/WW/view/en/35948966
14	PC-based Automation: Basics for the Solution of Automation Tasks Based on WinAC RTX	http://support.automation.siemens.com/WW/view/en/21004765
15	PC-Based Automation: Linking Windows Applications to WinAC RTX with WinAC ODK Using the Example of SIMATIC Vision Sensors	http://support.automation.siemens.com/WW/view/en/21572937
16	WinAC ODK Tips and Tricks	http://support.automation.siemens.com/WW/view/en/67217690

14 History

Table 14-1: Document versions

Version	Date	Modification
V1.0	04/2013	First version
V1.1	06/2013	Update: Chapter 11 Manipulating User-Defined Parameters with STEP 7