SIEMENS

Application description • 12/2013

Block for SIMOTION SCOUT for Monitoring 24V-Branches

SIMOTION CPU / SITOP PSE200U with Single Channel Message

http://support.automation.siemens.com/WW/view/en/82555461

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.

Any form of duplication or distribution of these Application Examples or excerpts hereof is prohibited without the expressed consent of Siemens Industry Sector.

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 Item-ID 50203404.

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

Table of contents

1 Overview of ST Source “myPseDiag”

What will I get?

ST source “myPseDiag” (ST = Structured Text) contains the FBPseDiag function block.

The FBPseDiag function block monitors the 24V branches using SITOP PSE200U with single channel message and SIMOTION CPUs.

The document on hand describes function block FBPseDiag. The function block provides you with tested code with clearly defined interfaces. You can use it as the basis for the task you wish to realize.

A key concern of this document is to describe:

- the FBPseDiag function block contained in ST source “myPseDiag”.
- the functionality implemented through this block.

The present documentation furthermore illustrates possible applications, and the included step-by-step instructions help you to integrate the block into your SIMOTION SCOUT project.

1.1 Different user scenarios

Possible applications for using the FBPseDiag block of ST source “myPseDiag”

The SITOP PSE200U electronic selectivity module is designed to be connected to a stabilized 24 V DC power supply with an output current up to 40 A (e.g. SITOP). SITOP PSE200U allows the 24 V DC output voltage generated by a stabilized power supply to be split between four load circuits. For each output, the rated current can be set individually with a potentiometer in the range from 0.5 A to 3 A or in the range from 3 A to 10 A, depending on the type. When the rated current is exceeded, the output will be disabled after a defined period of time and can be re-enabled using buttons or remote reset after a waiting time.

The states of all four load circuits are serially coded via the STATE output of the PSE200U module with single channel message.

The FBPseDiag block of ST source “myPseDiag” evaluates the serial code of the STATE output in the SIMOTION CPU.

The following section shows a scenario for the possible use of the FBPseDiag block of ST source “myPseDiag”:

Scenario

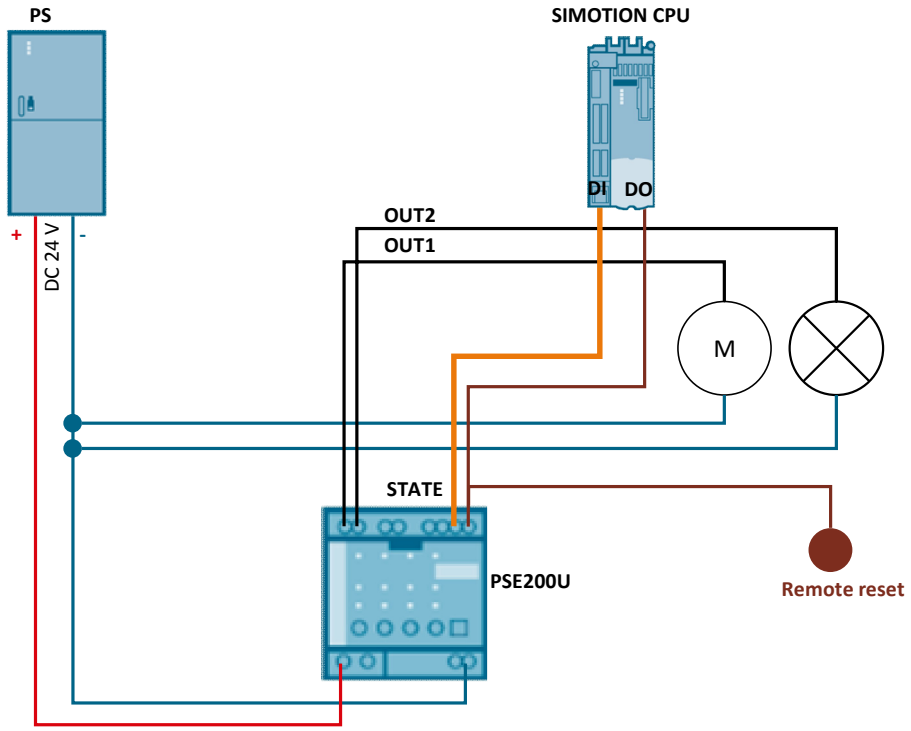
Via a digital input, the signal of the STATE output is read in and evaluated by the SIMOTION CPU. This allows you to monitor the state of channels OUT 1 to OUT 4 in the user program of the CPU.

The SIMOTION CPU detects, for example, whether the motor connected to load circuit OUT 1 has generated an overload.

The SIMOTION CPU detects, for example, if the lighting connected to load circuit OUT 2 has caused a short circuit.

1.1 Different user scenarios

Figure 1-1



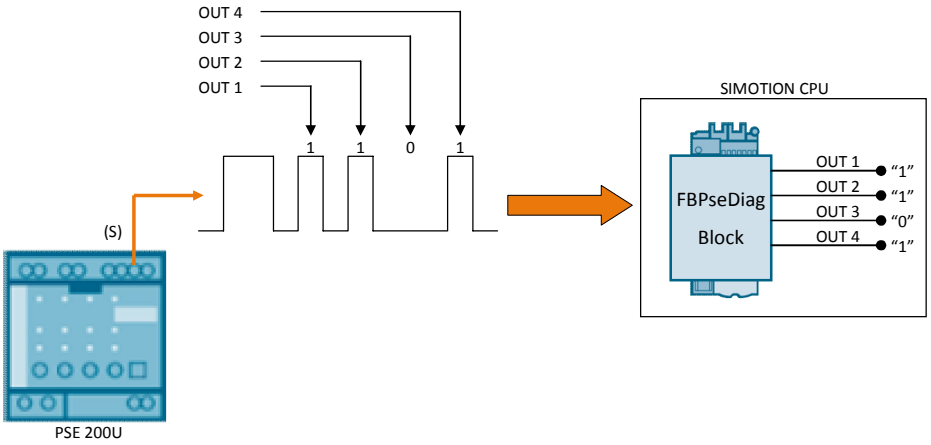
1.2 Workflow

The diagrammatic representation below shows the sequence of the function.

The STATE output of the PSE200U selectivity module with single channel message provides a signal that serially codes all four outputs OUT 1 to OUT 4.

To evaluate the signal of the STATE output, integrate ST source “myPseDiag” into the SIMOTION CPU. At its output, the contained FBPseDiag block indicates the state of the four outputs of the PSE200U module.

Figure 1-2



1.3 Hardware and software requirements

Requirements for this library

To make use of the full functionality of the ST source described here, the hardware and software requirements listed below must be met:

Hardware

Table 1-1

No.	Component	Order number	Qty.	Alternative
1.	SIMOTION D410-2 DP	6AU410-2AA00-0AA0	1	any SIMOTION CPU
2.	SITOP PSE200U 3A	6EP1961-2BA31	1	SITOP PSE200U 10A, order no. 6EP1961-2BA41

Software

Table 1-2

No.	Component	Order number	Qty.
1.	Configuration software SIMOTION SCOUT V4.0 or higher	6AU1810-0BA40-0XA0	1

1.4 Library resources

What will I learn here?

The overview below shows the main memory occupancy by the blocks of ST source “myPseDiag”.

Overall occupancy

The overall size of all blocks of ST source “myPseDiag” in the main memory is 3956 bytes (3.96 Kbytes).

Occupancy of the individual blocks

Table 1-3

Block	Size of main memory
FBPseDiag	3956 bytes

Code attribute of ST source “PseDiag”

The following table shows the code attributes of the ST source from which ST source “myPseDiag” is generated.

Table 1-4

ST source	Dynamic data	Retain data	Interface data	Code size
PseDiag	0 bytes	0 bytes	0 bytes	3956 bytes

2 Program Organization Units of ST Source “myPseDiag”

What will I learn here?

This chapter explains all blocks of the ST source “myPseDiag”. Before that, however, the block essentially involved in the implementation of the functionality is discussed in detail.

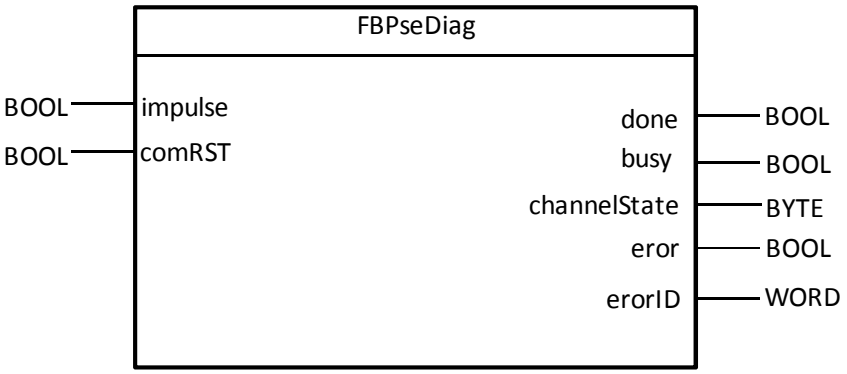
2.1 Explanation of the blocks

This chapter explains the FBPseDiag block belonging to ST source “myPseDiag”.

2.1.1 FBPseDiag block

Diagram

Figure 2-1



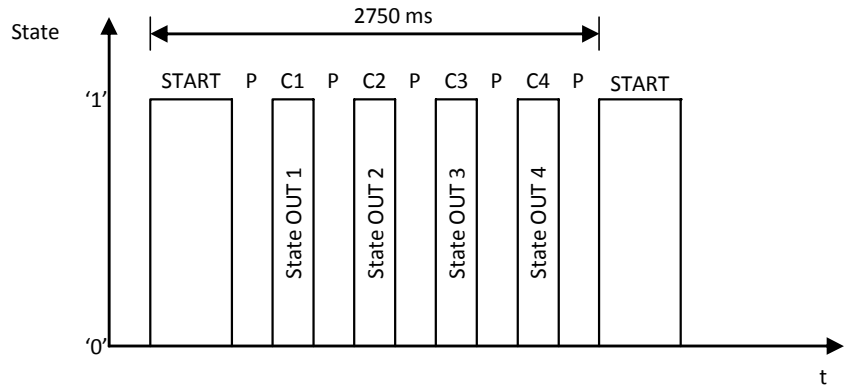
Principle of operation of the FBPseDiag block

The FBPseDiag block reads the STATE output of the PSE200U module with single channel message via the “impulse” input to evaluate the signal characteristic of the STATE output, and display the state of outputs OUT 1 to OUT 4 of the PSE200U module with single channel message at the channelState output of the FBPseDiag block.

A message frame of the signal consists of one start bit and four channel bits that are separated by pause bits. The start bit is always “1” and the pause bits are always “0”. The channel bits signal the state of channels OUT1 to OUT4.

Figure 2-2 shows the signal characteristic of the STATE output of the PSE200U module with single channel message.

Figure 2-2



Principle of operation of the PSE200U module with single channel message

Table 2-1 shows which status causes channels OUT 1 to OUT 4 to go to the “1” or “0” state.

Table 2-1

Status	LED of the PSE200U module	State of OUT 1 to OUT 4
Device starting up, supply voltage missing	Off	0
Output enabled	Green	1
Output current > rated current	Green flashing	1
Output was automatically disabled	Red	0
Automatic disable can be reset	Red flashing	0
Output manually disabled	Orange flashing	0
Output defective (internal fuse has tripped)	Off	0
Device over temperature	Red running light	0

Function characteristics

Figure 2-3 provides a graphical representation of the functional sequences of the FBPseDiag block.

Figure 2-3

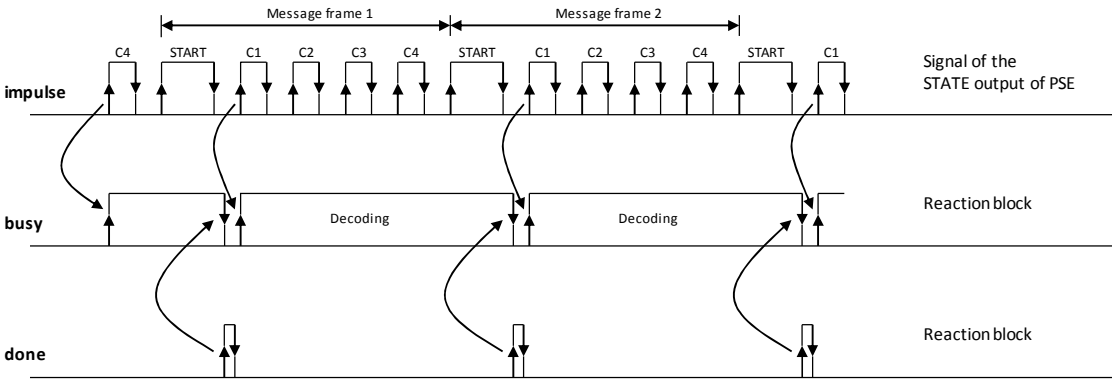
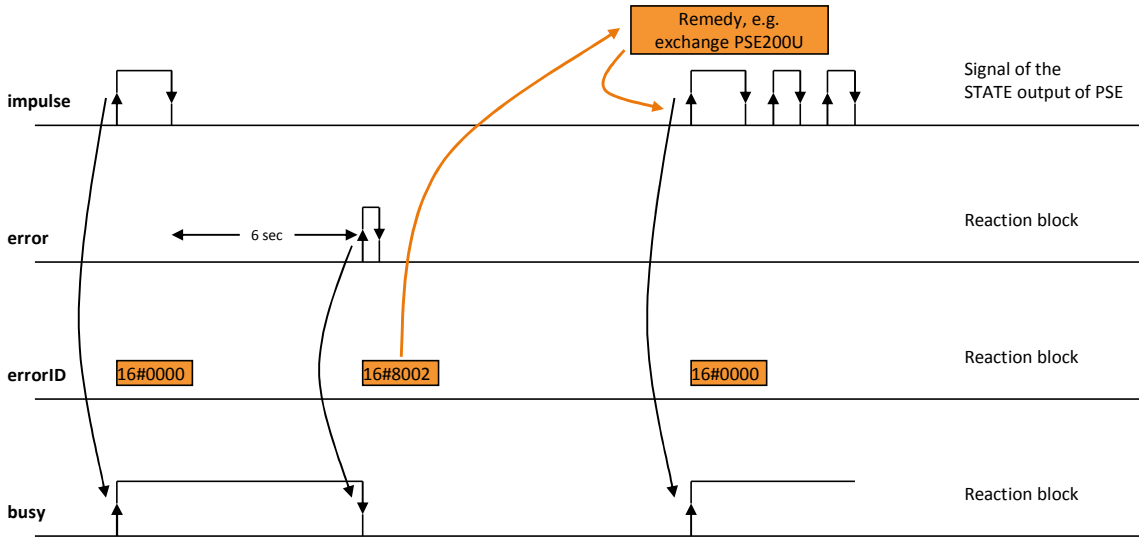


Figure 2-4 provides a graphical representation of the functional sequences of the FBPseDiag block in the event of an error, for example, if the PSE200U module is defective and does not provide a signal at the STATE output. Therefore, there will be no signal change at the “impulse” input of the FBPseDiag block. If a signal change is overdue for more than 6 seconds, the ERROR output will be set to TRUE for one cycle and value W#16#8002 is displayed at the errorID output. For as long as the FBPseDiag block does not detect a signal change at “impulse” input, the ERROR output is set to TRUE every 6 seconds for one cycle, and value W#16#8002 is displayed at the errorID output.

Figure 2-4



Call environment of the FBPseDiag block

The FBPseDiag block is accessed via instances. This enables evaluating several SITOP PSE200U modules using a block.

The program organization units, in which the FBPseDiag is instanced, need to be called cyclically, which can be performed in Background Task, for example.

The following table shows the properties and elements of ST source “PseDiag”.

Table 2-2

Properties and elements of ST source “PseDiag”	
Name	PseDiag
Programming language	ST
Know-how protection	No
Library	-
Function / function block / program	FBPseDiag

The following table shows the properties of the FBPseDiag block.

Table 2-3

No.	Properties of the FBPseDiag block
1.	Can be called in all cyclic programs

No.	Properties of the FBPseDiag block
2.	Adjustment to the application necessary: no

Note

To be able to evaluate the coded signals of the PSE200U correctly, the cycle time must not exceed 100ms.

If the cycle time exceeds 100ms, the FBPseDiag block will display an error with the value W#16#8001 at the errorID output.

Inputs

Table 2-4

Parameter	Data type	Description
impulse	BOOL	Input via which the signal of the STATE output of the PSE200U module is read in Figure 2-2 shows the time characteristic of the signal using an example
comRst	BOOL	When there is a positive edge, a reset will be triggered. All parameters (static variables and outputs of the FBPseDiag block will be reset.

Outputs

Parameter	Data type	Description
done	BOOL	TRUE, if a message frame was evaluated completely and without errors, and the state of outputs OUT 1 to OUT 4 of the PSE200 module is displayed at the channelState output of the FBPseDiag block. Only TRUE for one cycle.
busy	BOOL	TRUE, if the FBPseDiag block is active. BUSY is set to FALSE if a message frame was successfully evaluated and the data of the channelState output can be adopted.
channelState	BYTE	State of channels OUT 1 to OUT 4: Bit 0 = 1 if channel OUT 1 has state 0 Bit 0 = 0 if channel OUT 1 has state 1 Bit 1 = 1 if channel OUT 2 has state 0 Bit 1 = 0 if channel OUT 2 has state 1 Bit 2 = 1 if channel OUT 3 has state 0 Bit 2 = 0 if channel OUT 3 has state 1 Bit 3 = 1 if channel OUT 4 has state 0 Bit 3 = 0 if channel OUT 1 has state 1 Bit 4: not assigned Bit 5: not assigned Bit 6: not assigned Bit 7: not assigned Table 2-1 shows an overview of possible channel states
errorID	WORD	Status, if error = TRUE **Note** Only for one cycle.

Parameter	Data type	Description
error	BOOL	TRUE if an error occurs when executing the routine. Only TRUE for one cycle. Default value: FALSE

Status and error displays

Table 2-5

Status	Meaning	Remedy / Notes
W#16#8000	No errors	-
W#16#8001	Cycle time of 100ms exceeded.	Call the FBPseDiag in a cyclic program with a cycle time of max. 100ms.
W#16#8002	No signal change detected at the “impulse” input for at least 6 seconds.	Check whether the STATE output of the PSE200U module is connected to the digital input of the CPU. Check whether the power supply has been connected to the PSE200U module.

3 Working with the ST Source

What will I learn here?

This chapter includes instructions for integrating ST source “PseDiag” into your SIMOTION SCOUT project as well as instructions for using the FBPseDiag block.

3.1 Integrating the ST source into a SIMOTION SCOUT project

The actions listed below define how to integrate ST source “PseDiag” into your SIMOTION SCOUT project. Subsequently, you can use the FBPseDiag block.

Note The following section assumes that a SIMOTION SCOUT project exists.

In this example, the “myProjPseDiag” project was created.

Note The following section assumes that a configured SIMOTION CPU exists.

In this example, SIMOTION CPU “mySIMOTION” was created.

Table 3-1

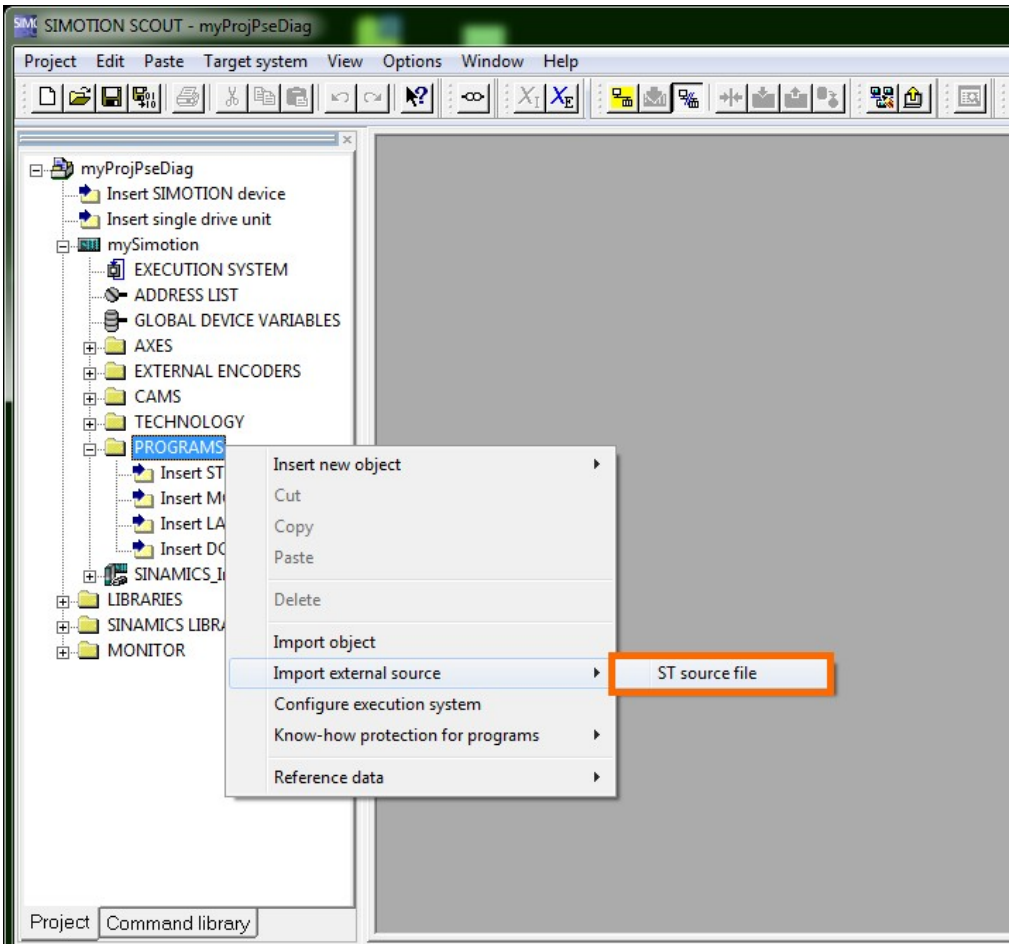
No.	Action
1.	The ST source is available on the HTML page from which you downloaded this document. Save ST source “ST_PseDiag.zip” on your hard drive.
2.	Extract the ST source.
3.	After the ST source has been extracted, open it in SIMOTION SCOUT.

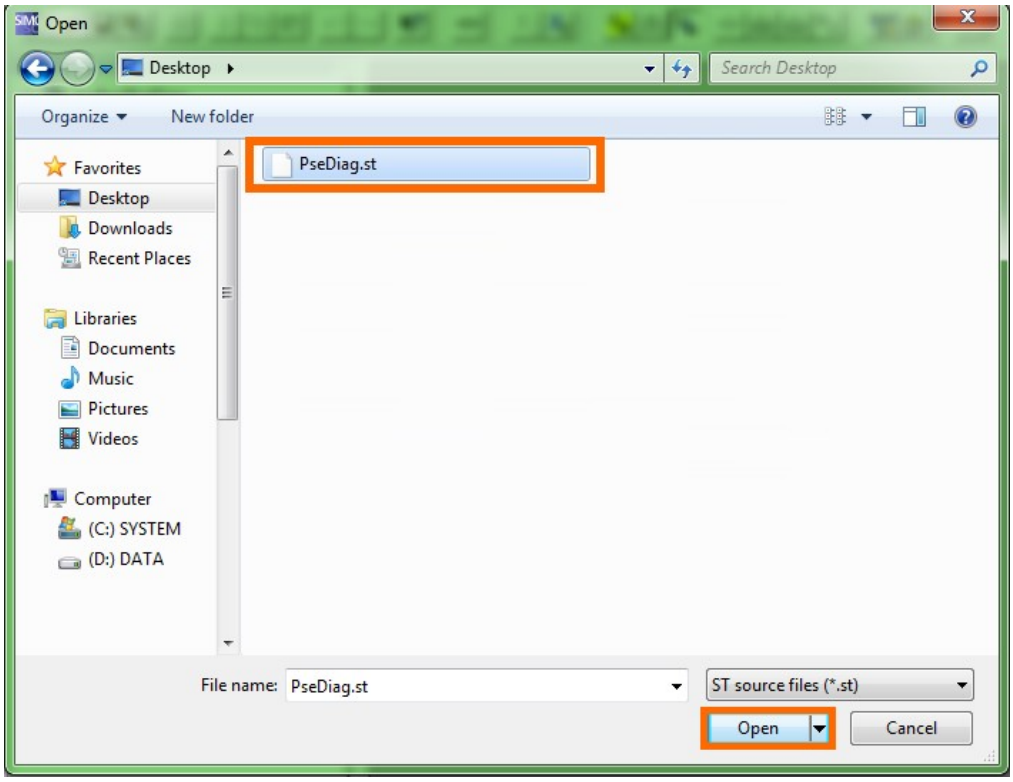
Integrating the ST source into SIMOTION SCOUT

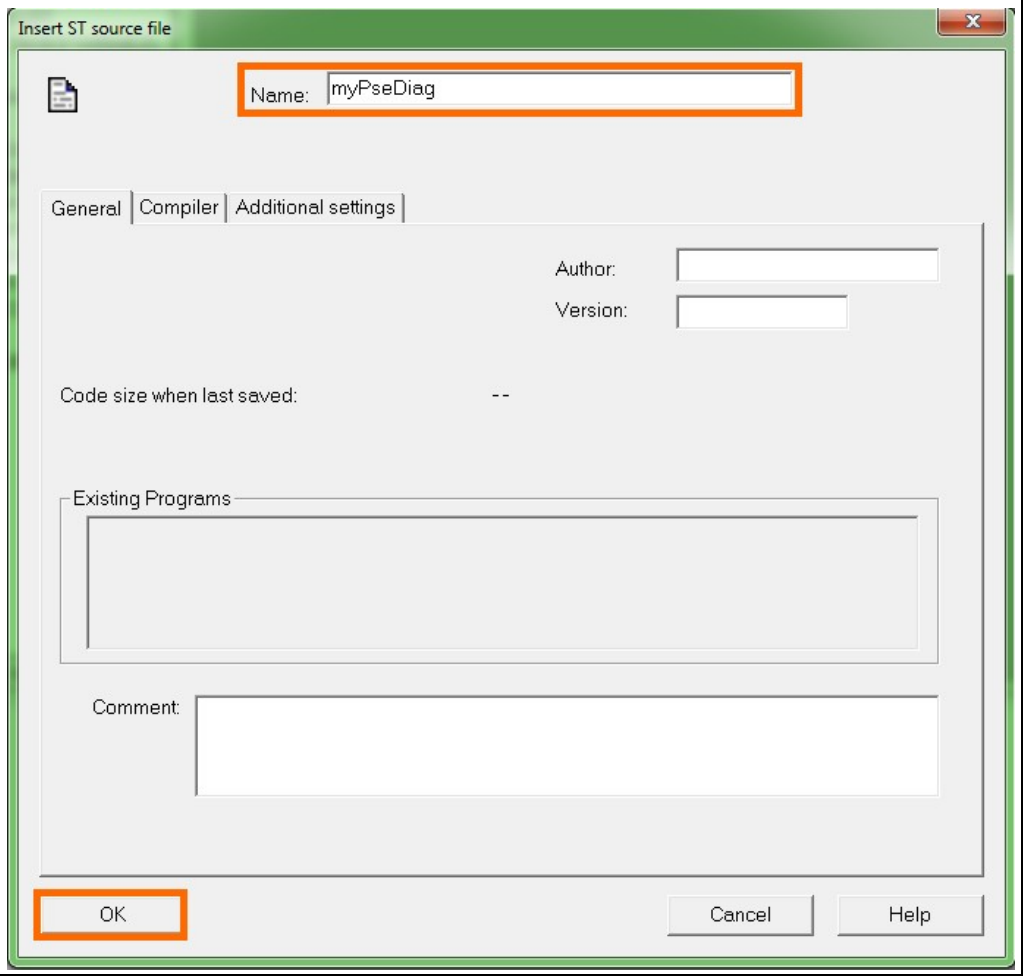
In order to integrate ST source “ST_PseDiag.zip” into SIMOTION, please proceed as follows:

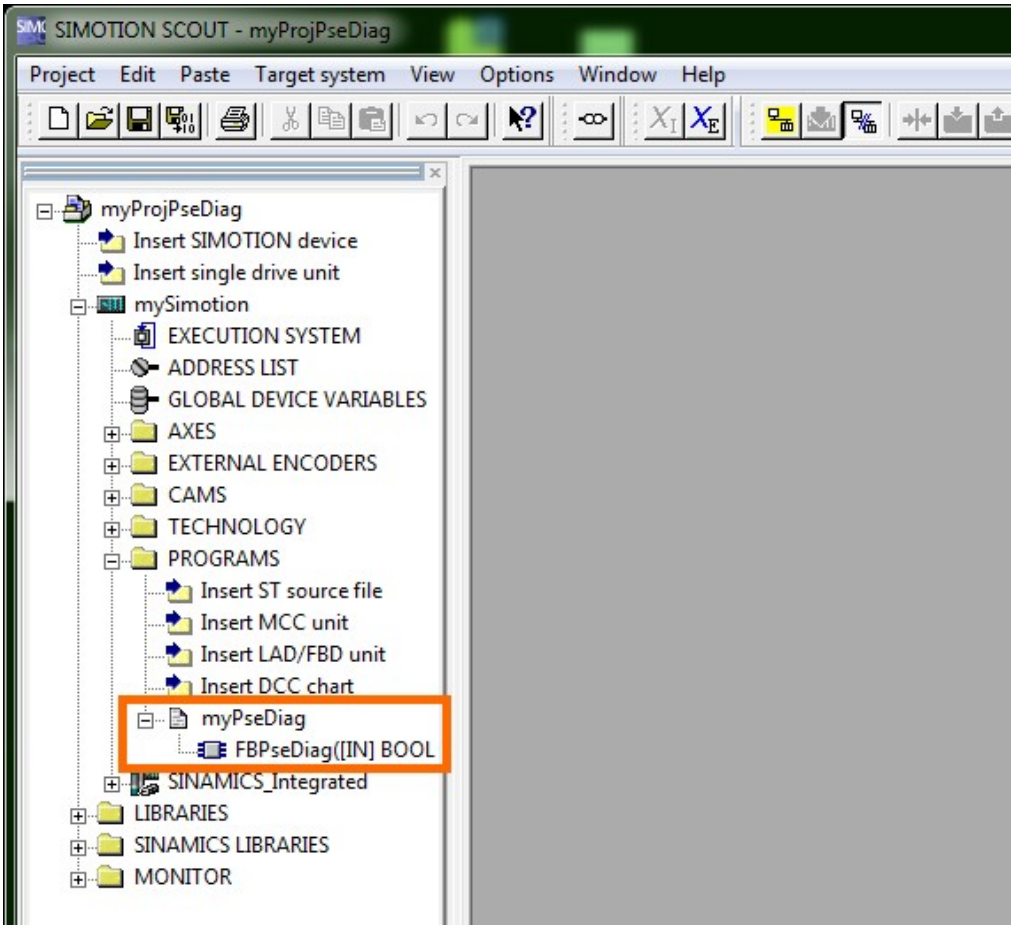
3.1 Integrating the ST source into a SIMOTION SCOUT project

Table 3-2

No.	Action
1.	In the project navigation you right-click onto the “PROGRAMS” folder of your SIMOTION CPU and select the “Import external source > ST source file” menu. 

No.	Action
2.	Select ST source “PseDiag” and click on the “Open” button. The “Insert ST source file” dialog box opens.  **Note** You can identify the source file by the file name extension “st”.

No.	Action
3.	Assign a name to the ST source, for example “myPseDiag”, and click on the “OK” button. 

No.	Action
4.	Upon importing ST source “PseDiag”, an ST source, for example myPseDiag, is added to the “PROGRAMS” folder of your SIMOTION CPU. 
5.	Save the SIMOTION SCOUT project.

3.2 Instancing the FBPseDiag block

The table below lists steps for using an instance of the FBPseDiag block for evaluating a PSE200U module.

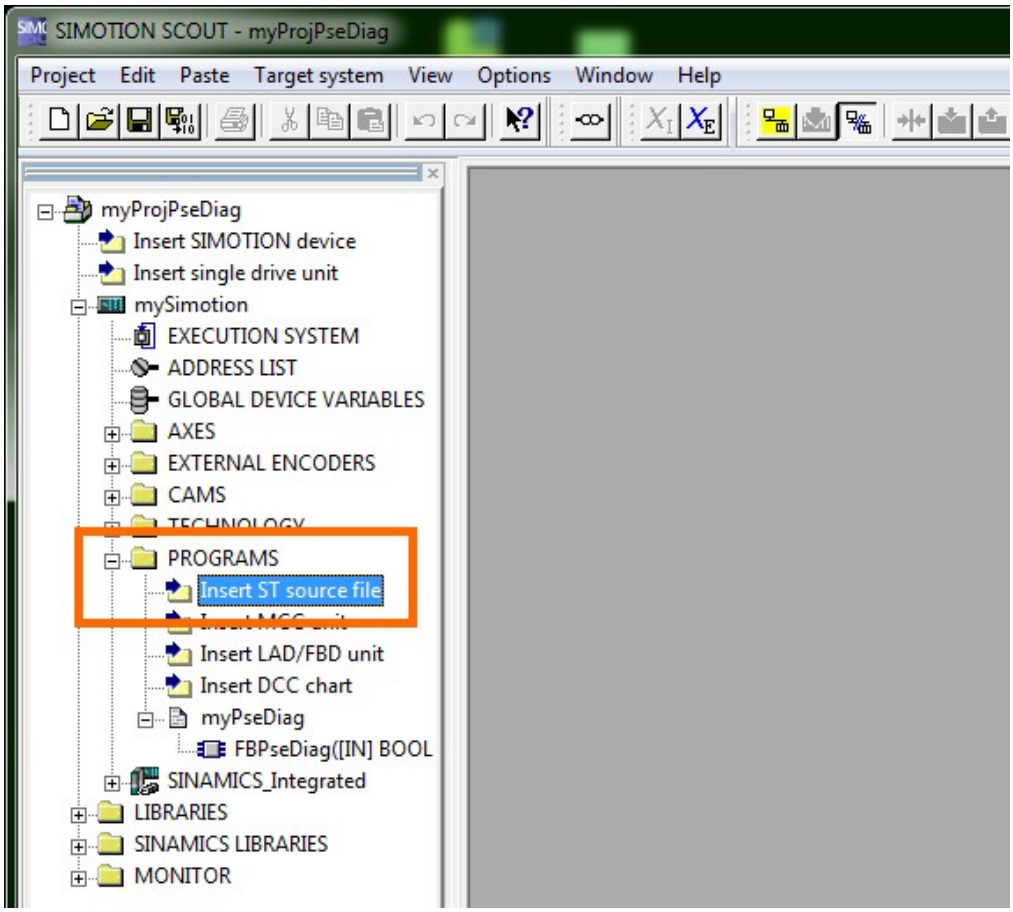
Note

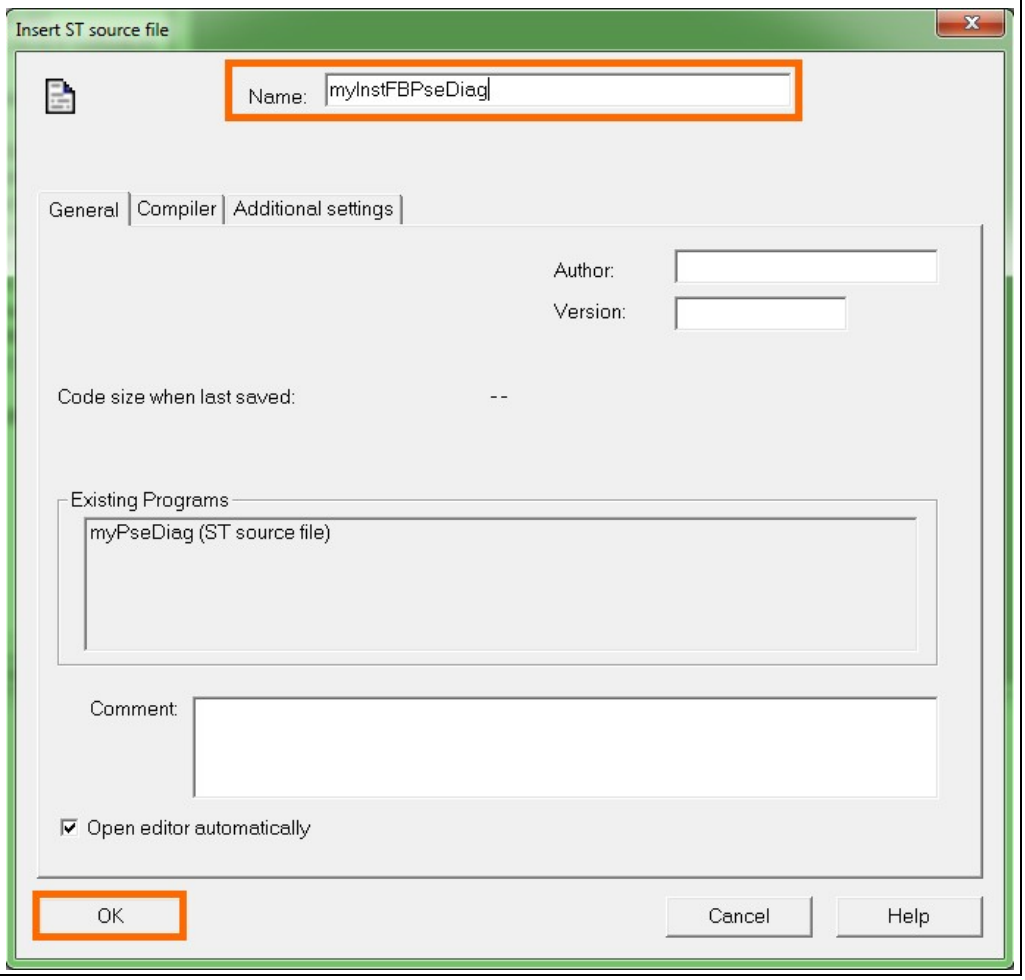
The S (STATE) output must be wired to a digital input of the SIMOTION CPU.

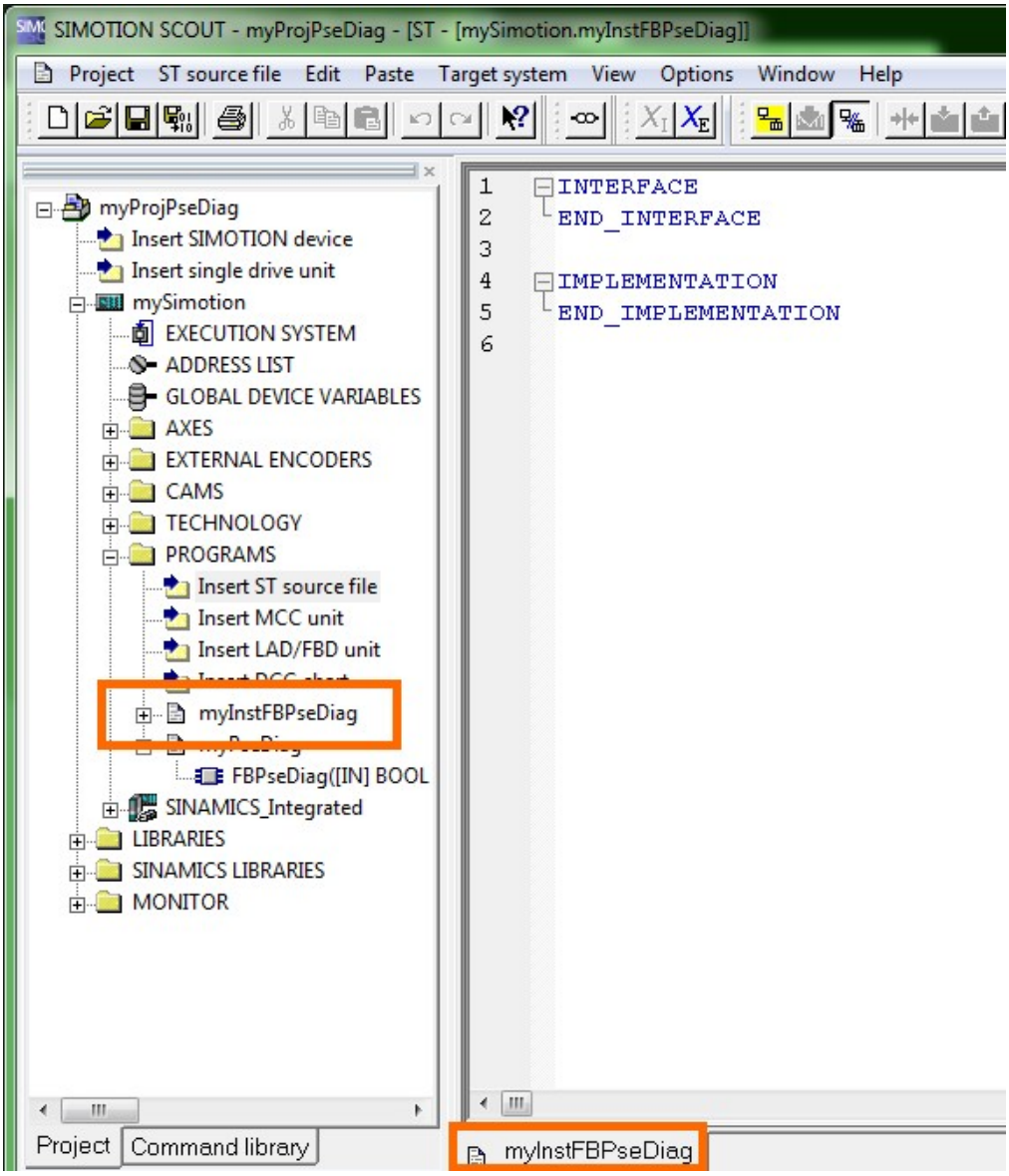
Transferring the input signal from the SINAMICS Integrated CU into the ADDRESS LIST of the SIMOTION is assumed.

3.2 Instancing the FBPseDiag block

Table 3-3

No.	Action
1.	In the “PROGRAMS” folder of your SIMOTION CPU you double-click on the “Insert ST source file” entry. The “Insert ST source file” dialog box opens. 

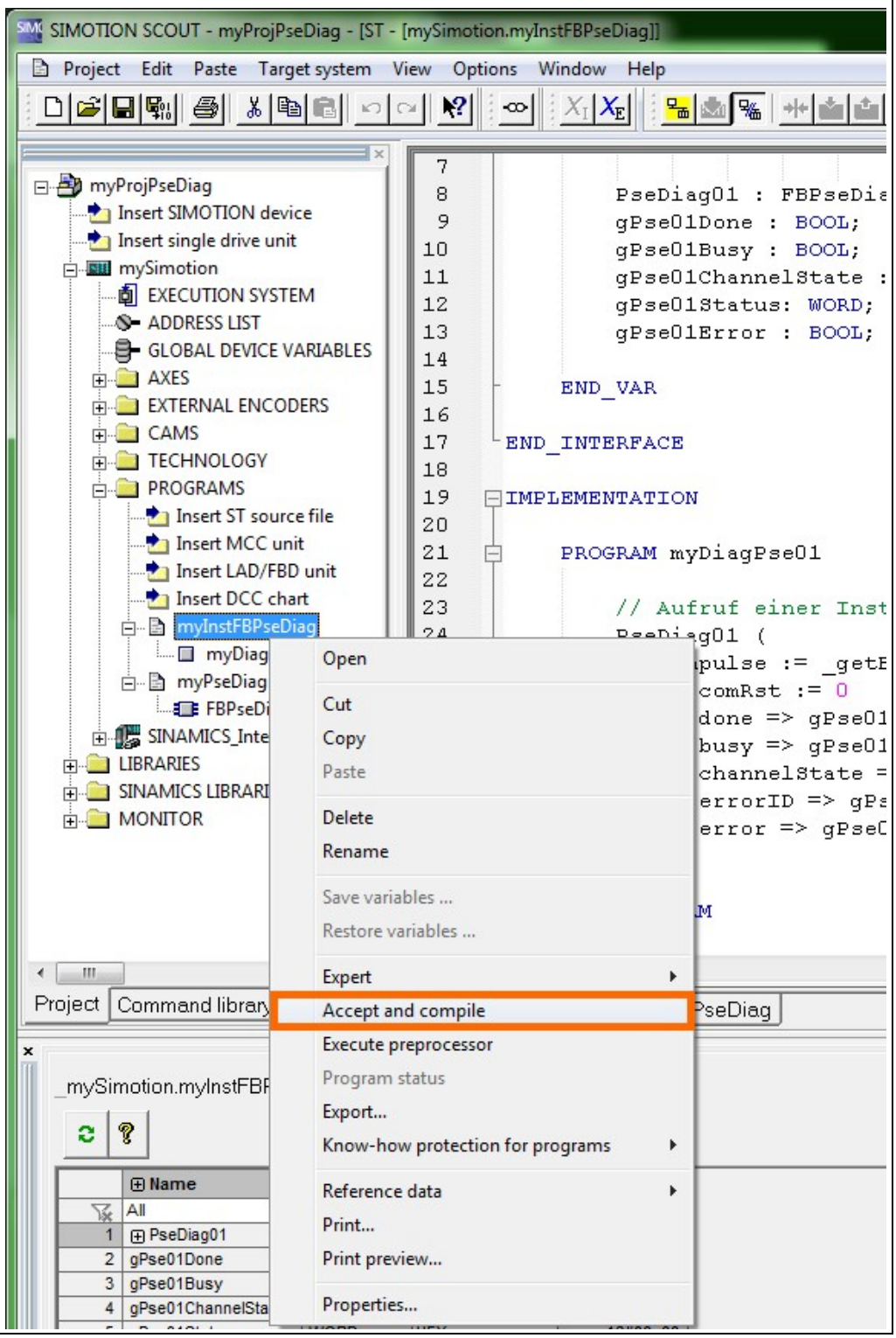
No.	Action
2.	Enter a name for the new ST source, e.g. “myInstFPPseDiag”. 

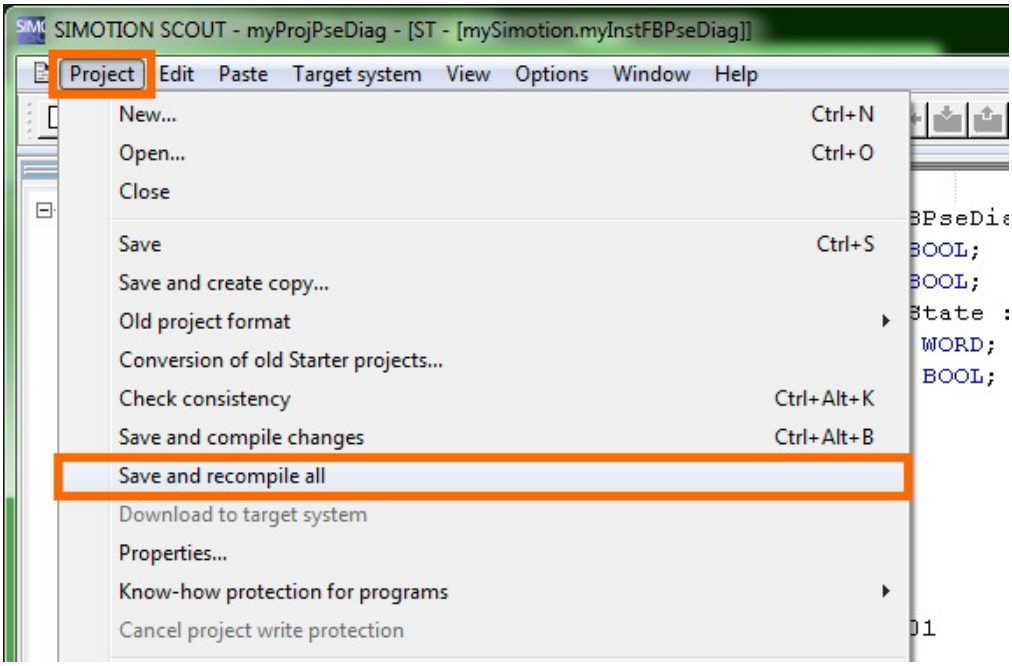
No.	Action
3.	In the “PROGRAMS” folder of your SIMOTION CPU, an empty ST source is created. 

No.	Action
4.	Define the INTERFACE interface of the empty ST source “myInstFBPseDiag”. • Import the ST source with the FBPseDiag block with the instruction: - USES myPseDiag • Declare the program for calling the instance of the FBPseDiag with the instruction: - PROGRAM myDiagPse01 • Assign the FBPseDiag block to the “PseDiag01” instance and declare global variables for interconnecting the interface of the block instance: VAR_GLOBAL PseDiag01: FBPseDiag; gPse01Done: BOOL; gPse01Busy: BOOL; gPse01ChannelState: BYTE; gPse01Status: WORD; gPse01Error: BOOL; END_VAR **Note** The declaration of global variables enables using values of the block instance outside of the ST source. 1 INTERFACE 2 3 USES myPseDiag; 4 PROGRAM myDiagPse01; 5 6 VAR_GLOBAL 7 8 PseDiag01 : FBPseDiag; 9 gPse01Done : BOOL; 10 gPse01Busy : BOOL; 11 gPse01ChannelState : BYTE; 12 gPse01Status: WORD; 13 gPse01Error : BOOL; 14 15 END_VAR 16 17 END_INTERFACE

No.	Action
5.	Create an instance of the FBPseDiag within a program, e.g. myDiagPse01. PROGRAM myDiagPse01 PseDiag01 (impulse:= _getBit(iab16Cu[1],0 , comRst:= 0 , done => gPse01Done , busy => gPse01Busy , channelState => gPse01ChannelState , errorID => gPse01Status , error => gPse01Error); END_PROGRAM **Note** “iab16Cu[1],0” corresponds to the digital input of the CPU, to which the S (STATE) output of the PSE200U module is wired. 19 IMPLEMENTATION 20 21 PROGRAM myDiagPse01 22 23 24 PseDiag01 (25 impulse := _getBit(iab16Cu[1],0) 26 , comRst := 0 27 , done => gPse01Done 28 , busy => gPse01Busy 29 , channelState => gPse01ChannelState 30 , errorID => gPse01Status 31 , error => gPse01Error 32); 33 34 END_PROGRAM 35 36 END_IMPLEMENTATION

No.	Action
6.	In the project navigation you right-click on the created ST source “myInstFBPseDiag” and select the “Accept and compile” menu. ST source “myInstFBPseDiag” is compiled.



No.	Action
7.	In the SIMOTION SCOUT you select the “Project > Save and recompile all” menu. The “myProjPseDiag” project is saved and recompiled. 

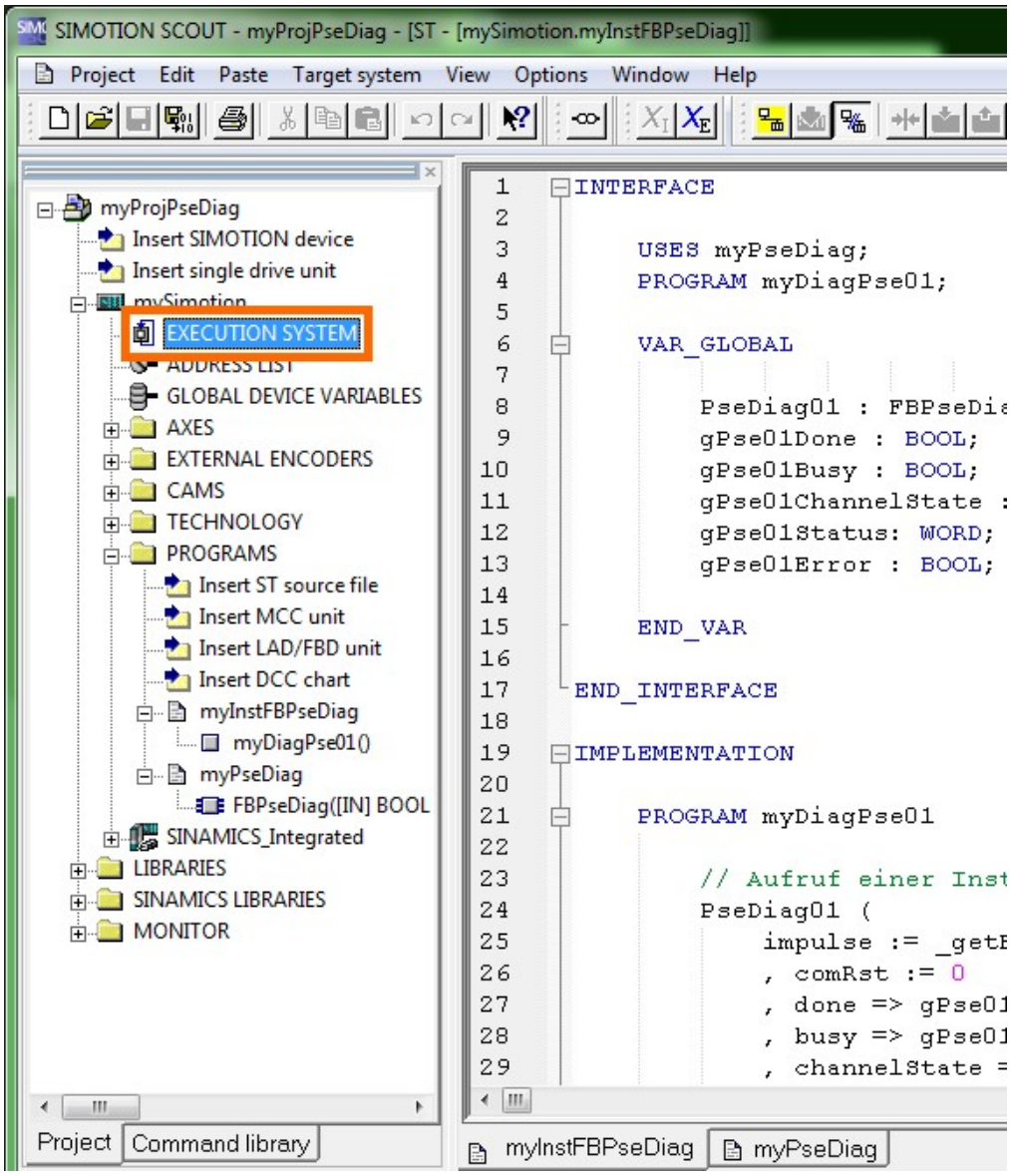
3.3 Assigning the program in EXECUTION SYSTEM

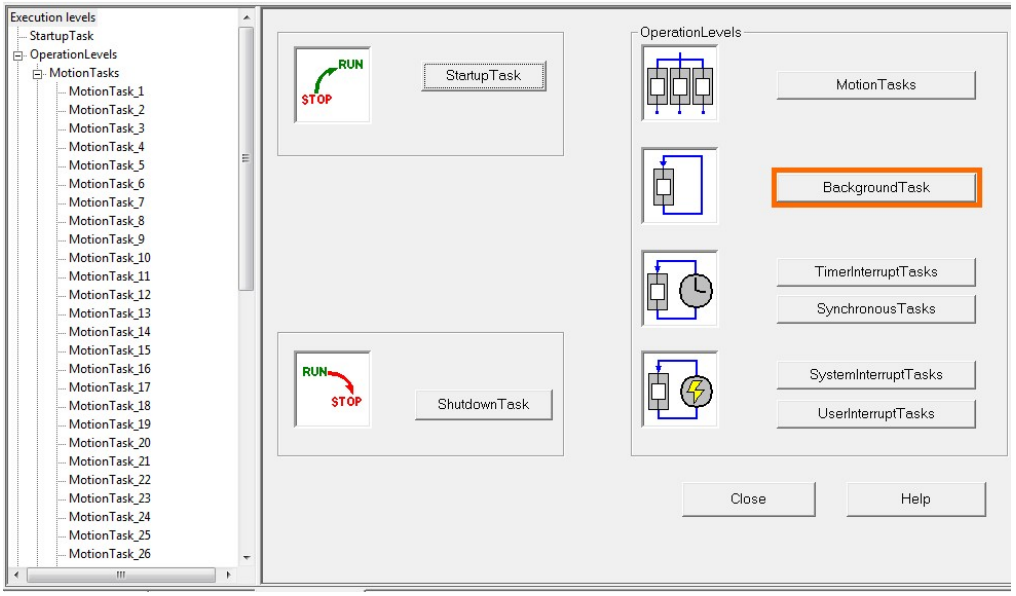
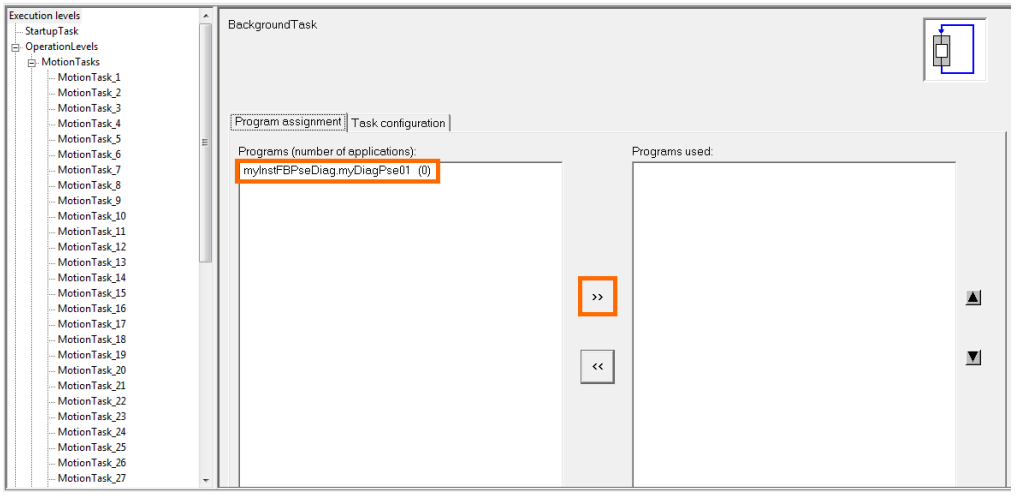
The table below lists the steps for running the created program in a cyclical task.

Note The project must have been compiled without errors.

3.3 Assigning the program in EXECUTION SYSTEM

Table 3-4

No.	Action
1.	To open the EXECUTION SYSTEM of your SIMOTION CPU, double-click on the EXECUTION SYSTEM entry in the project navigation. 

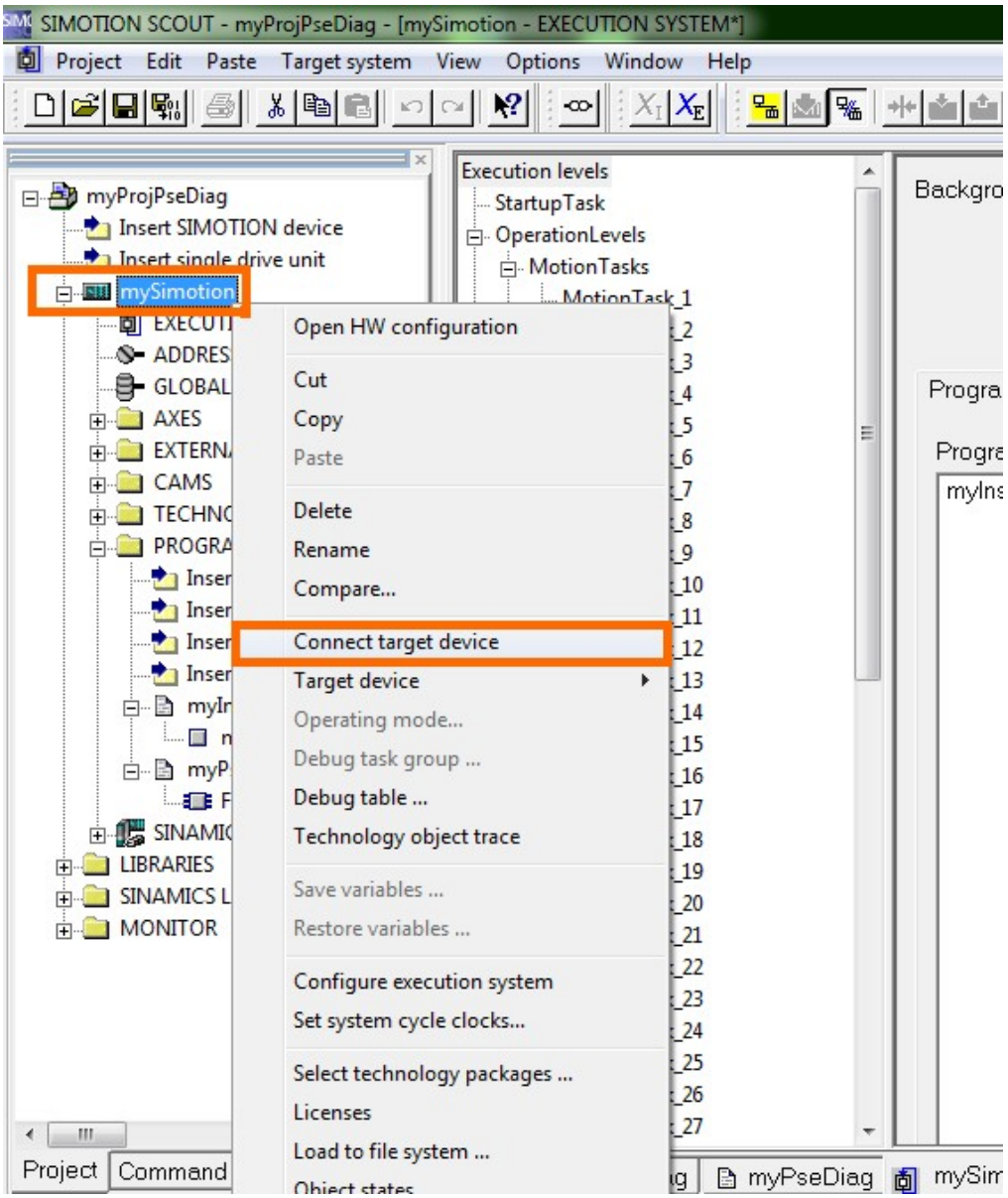
No.	Action
2.	In the Editor window you click on the “BackgroundTask” button. The program allocation for Background Task opens. 
3.	Select the created “myDiagPse01” program and click on the “>>” button, so the “myDiagPse01” program is assigned to the used programs and hence to the BackgoundTask. 
4.	Save the project.

3.4 Loading the program into the SIMOTION CPU

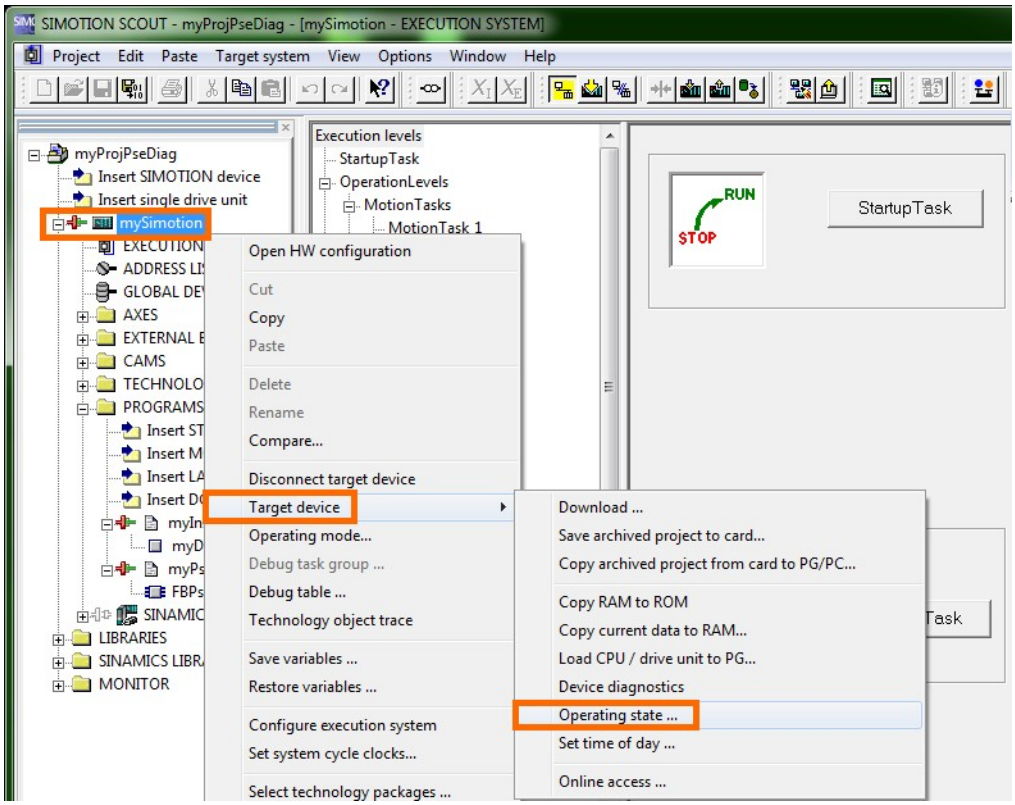
The table below lists the steps for loading your user program into the CPU.

3.4 Loading the program into the SIMOTION CPU

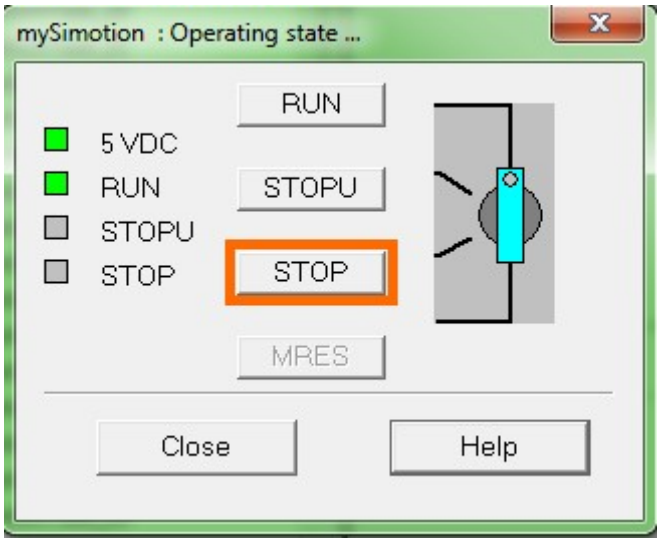
Table 3-5

No.	Action
1.	In the project navigation you right-click on SIMOTION CPU “mySimotion” and select the “Connect target device” menu to create a connection with the SIMOTION CPU. 

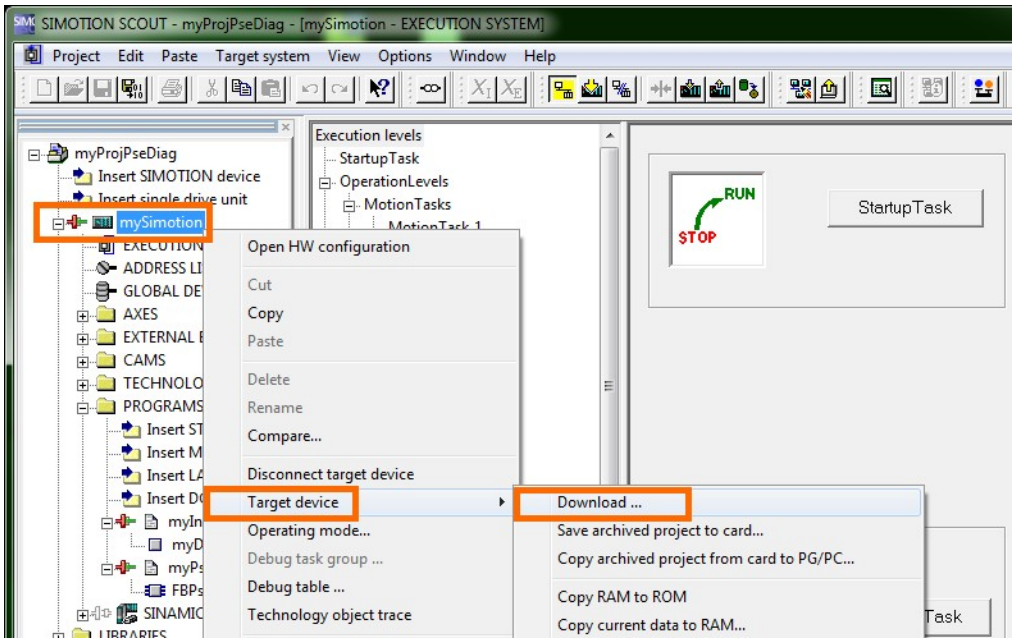
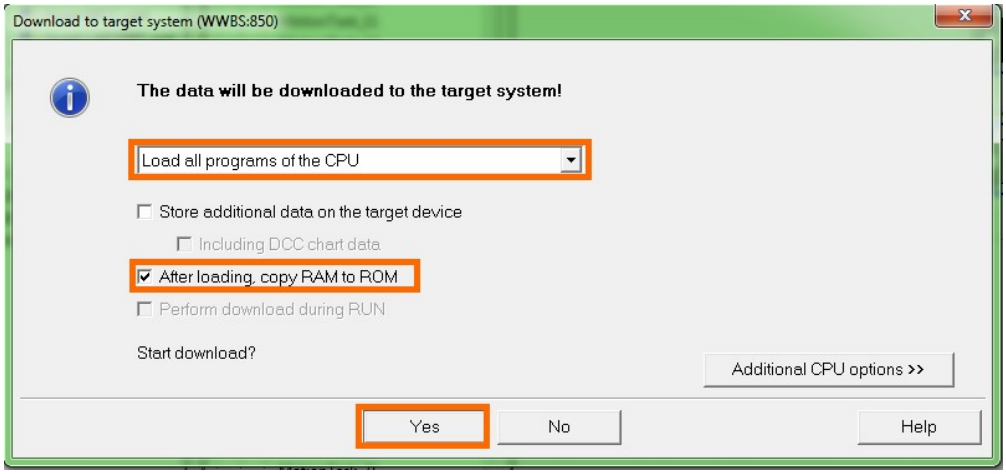
No.	Action
2.	In the Project tree, right-click on SIMOTION CPU “mySimotion” and select the “Target device > Operating state…” menu. The “Operating state” dialog box opens.



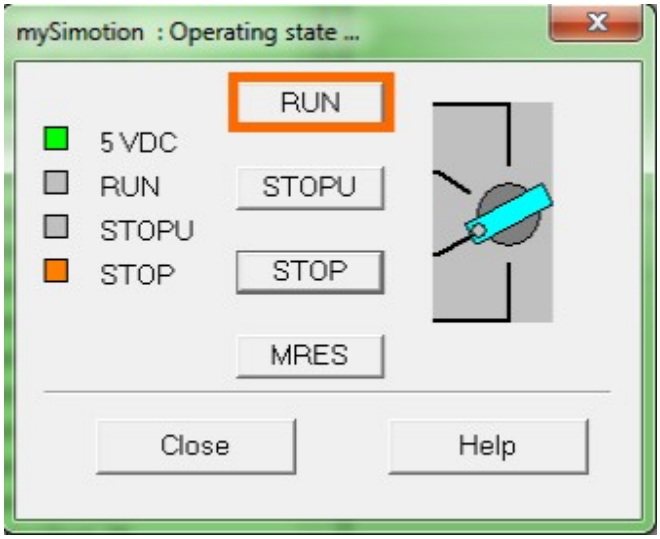
Click the "STOP” button to stop the SIMOTION CPU.



3.4 Loading the program into the SIMOTION CPU

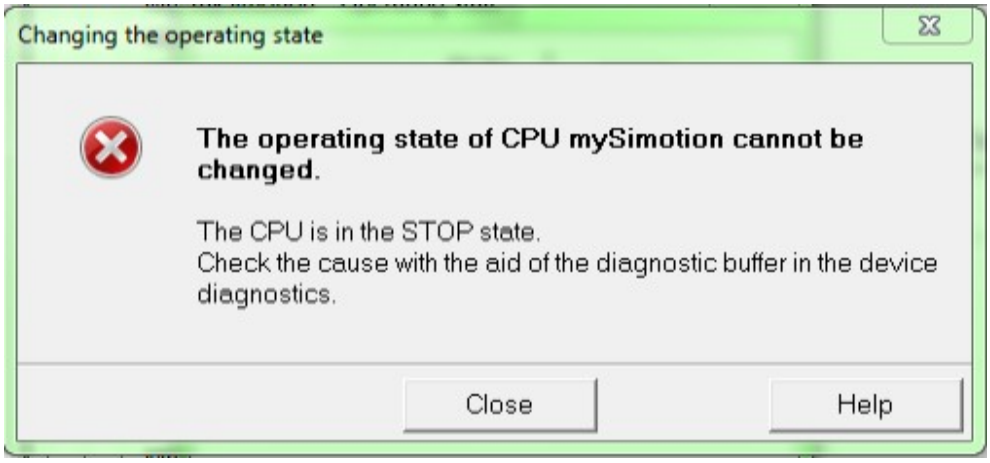
No.	Action
3.	Load all programs into the SIMOTION CPU. In the Project tree, right-click on the SIMOTION CPU and select the “Target device > Download…” menu. The “Download to target system” dialog box opens.  Select the “Load all programs of the CPU” entry and activate the “After loading, copy RAM to ROM” checkbox. Click on the “Yes” button to start the download. 

No.	Action
4.	In the “Operating state” dialog box, click on the “RUN” button to start the SIMOTION CPU.



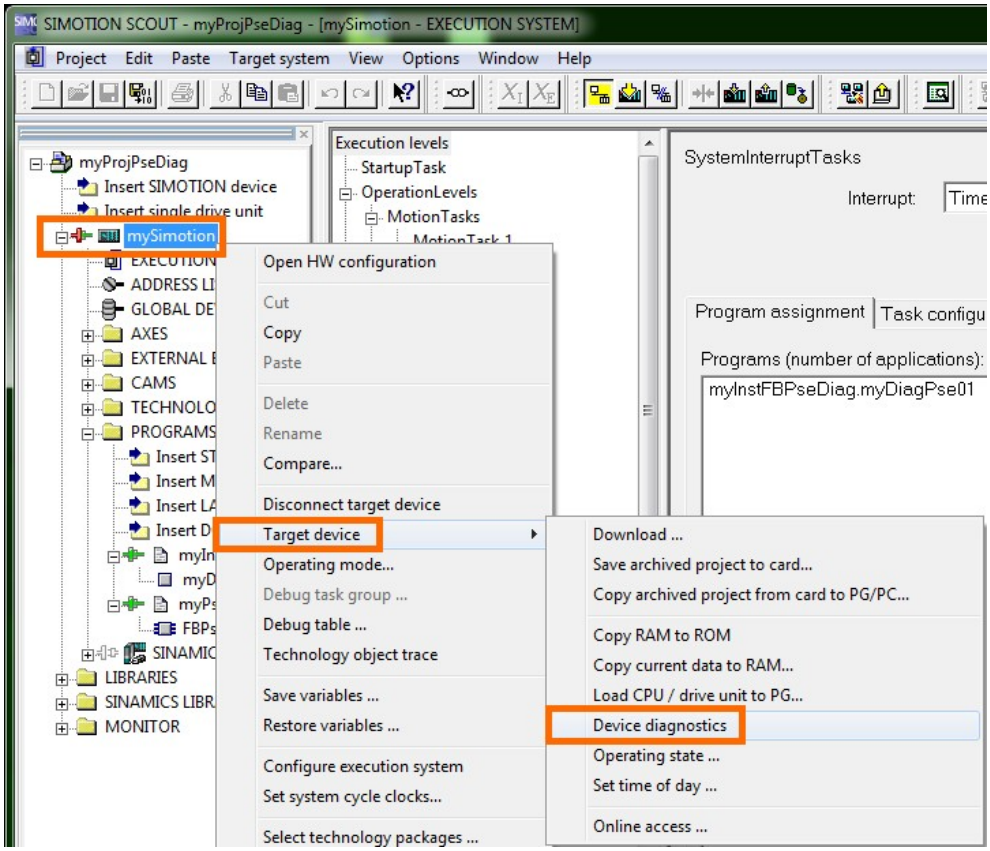
If when starting the SIMOTION CPU an error message stating that the operating state of the CPU cannot be changed is output, open the device diagnostics for the SIMOTION CPU.

Figure 3-1



To open the device diagnostics, right-click on the SIMOTION CPU and select the “Target device > Device diagnostics” menu.

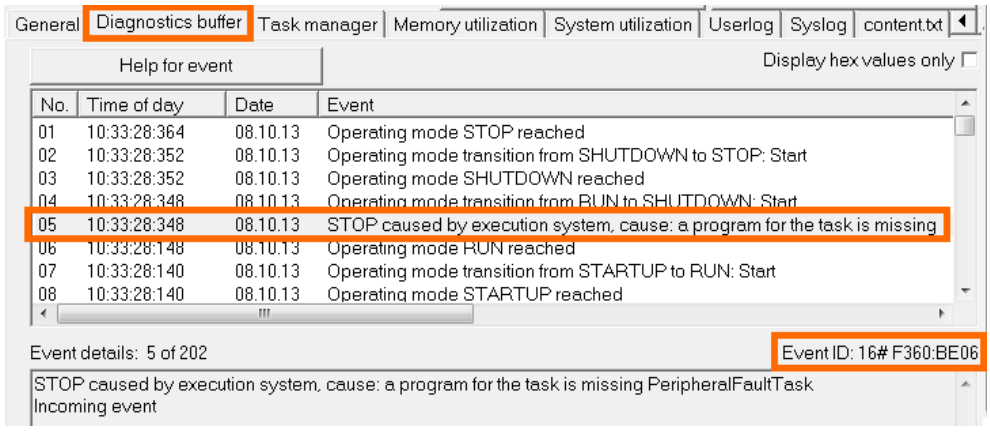
Figure 3-2



In the device diagnostics, open the “Diagnostics buffer”.

Search for the last operating mode transition from RUN to STOP.

Figure 3-3



If the event with ID 1F360BE06 is responsible for the transition from RUN to STOP, perform the remedy described in **Fehler! Verweisquelle konnte nicht gefunden werden.**.

Event F360BE06 occurs when a program for the PeripheralFault Task is missing.

Note

The described remedy only tests the created user program.

It is up to the user to verify the measure in a concrete application case.

The following description assumes an existing connection with the target device.

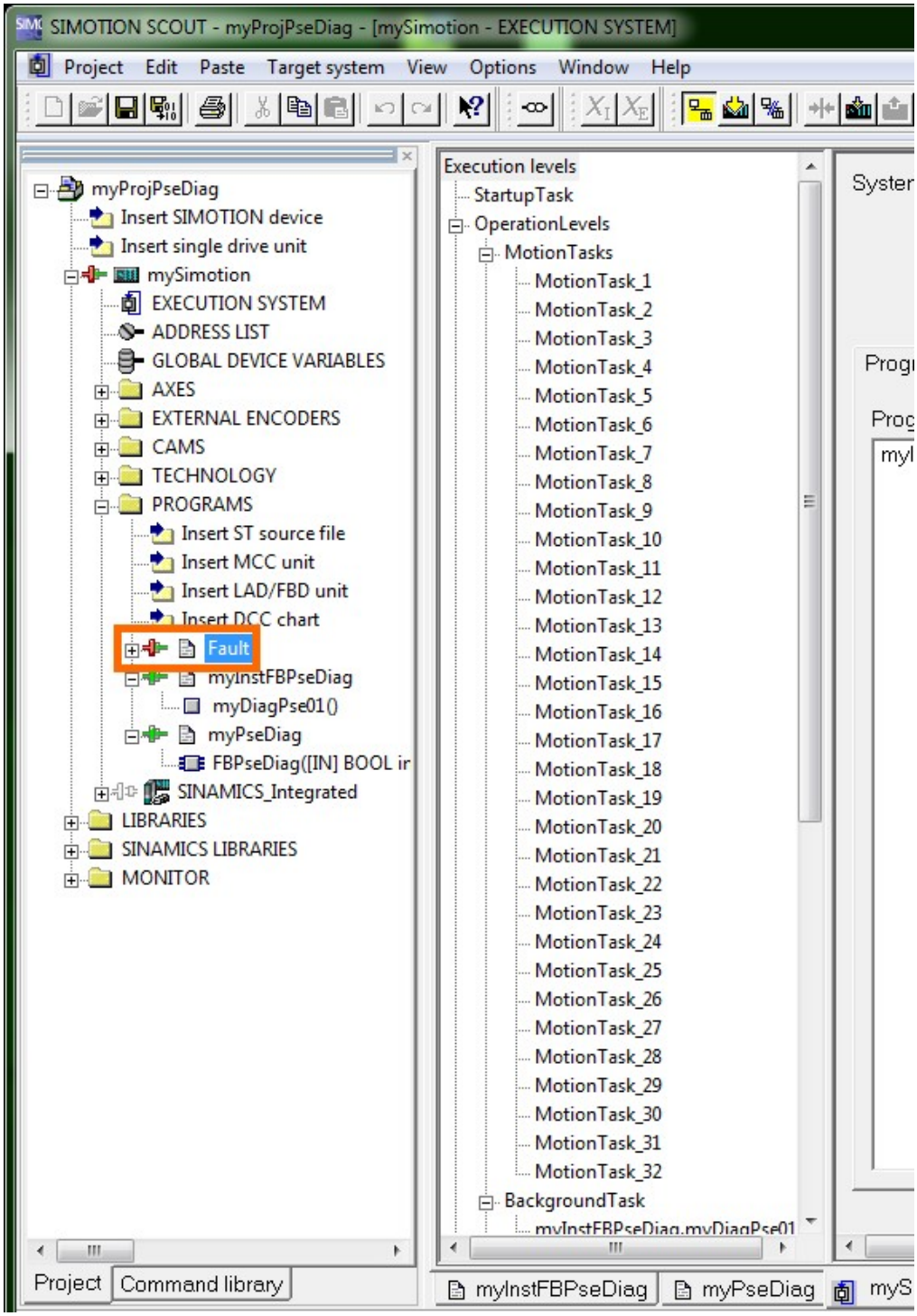
Procedure when event F360BE06 occurs

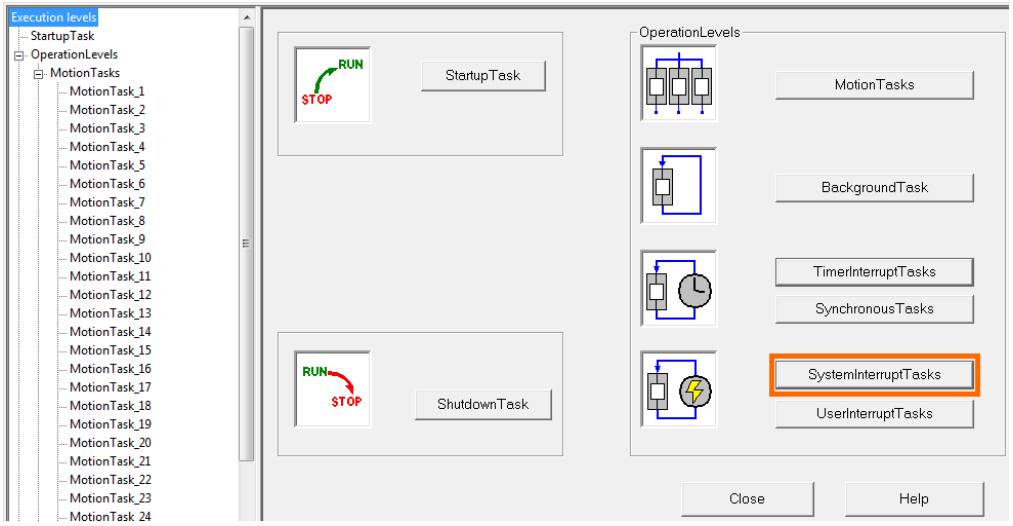
Table 3-6 describes the procedure for an occurred F360BE06 event.

3.4 Loading the program into the SIMOTION CPU

Table 3-6

No.	Action
1.	Insert a new ST source, e.g. Fault (see Table 3-3, steps 1-3).



No.	Action
2.	In this new ST source you create a program without instruction, e.g. myEmptyProg. 1 INTERFACE 2 3 PROGRAM myEmptyProg; 4 5 END_INTERFACE 6 7 8 IMPLEMENTATION 9 10 PROGRAM myEmptyProg 11 ; 12 END_PROGRAM 13 14 END_IMPLEMENTATION 15
3.	Compile ST source “Fault” (see Table 3-3, step 6).
4.	Compile the project. (see Table 3-3, step 7).
5.	Open the EXECUTION SYSTEM of the SIMOTION CPU (see Table 3-4, step 1).
6.	In the Editor window you click on the “SystemInterruptTask” button. The program allocation for SystemInterruptTask opens. 
7.	Select the “PeripheralFaultTask” interrupt. Select the created “myEmptyProg” program and click on the “>>” button, so the “myEmptyProg” program is assigned to the used programs and hence to the SystemInterruptTask.
8.	Save the project.
9.	Load the user program to the SIMOTION CPU. (See Table 3-5).

4 Literature

Table 4-1

	Subject	Title
\1\	Siemens Industry Online Support	http://support.automation.siemens.com
\2\	Download page of the entry	http://support.automation.siemens.com/WW/view/en/82555461
\3\	SIMOTION SCOUT ST Structured Text, Programming and Operation Manual	http://support.automation.siemens.com/WW/view/en/61056268
\4\	SITOP PSE200U 3A	http://support.automation.siemens.com/WW/view/en/42248945
\5\	SITOP PSE200U 10A	http://support.automation.siemens.com/WW/view/en/42248587
\6\	SIMOTION SCOUT V4.3.1	http://support.automation.siemens.com/WW/view/en/61005675

5 History

Table 5-1

Version	Date	Modifications
V1.0	12/2013	First version