Documentation

Modbus/TCP connection S7-1200/1500 ↔ SE-7xx

using the new S7 Modbus interface

Documentation: 15th November 2017
valid for: Version 1.4

Author: Lukas Jolbej

In this documentation the S7 Modbus connection to the STANGE SE-7xx device is explained.

Used devices:

- STANGE SE-702
- Siemens S7-1212C DC/DC/DC (6ES7 212-1AE40-0XB0)

connected via a 100 MBit/s switch

Used software:

- Siemens TIA Portal V13 SP1 Update 9
- Windows 7 SP1
- Device version 7.0.1.10 for the SE-702
- Firmware version 4.1 for the S7-1212C

Requirements:

- The feature must be licensed in the SE-7xx
- IP address and Modbus Unit ID of the SE-7xx must be known

corresponding TIA Portal templates:

- se7xx-1200-1500-scl-mb (template project), version 1.4
- se7xx-1200-1500-scl-mb-library (library), version 1.4

Table of contents

Getting started

Function overview

What are the features?

- Control and query programmer
- Control and query control zone
- Query setvalue and its status
- Control and query alarmhandler
- Generate alarm and query alarm status
- Control and query datalogger
- Query process step status
- Query digital track status
- Query tolerance status, enable tolerance (if configured as external activatable)
- Query limit status
- Set digital variable in SE-7xx (starting at FE 2000)
- Query digital variable from SE-7xx (starting at FA 2000)
- Set analog variable in SE-7xx (values 41-80)
- Query analog variable from SE-7xx (values 1-40)
- Set actual value in SE-7xx, force Overflow/Underflow/Break status
- Query actual value from SE-7xx, query error status of actual value

What data is transferred from the SE-7xx to the S7 (status data)?

- Boolean data
  - ➔ Control zone status, Setvalue status, Actual value status, Tolerance status, Limit status, Programmer status, Process step status, Digital track status, Digital output variables, Alarmhandler status, Alarm status, Datalogger status
- 32 Bit floating point values (REAL)
  - ➔ Control zone outputs (Y values), Setvalues, Analog variables 1-40, Actual values

What data is transferred from the S7 to the SE-7xx (control data)?

- Boolean data
  - ➔ Programmer control, Control zone control, Tolerance enabling, Digitale input variables, Alarmhandler control, Alarm inputs, Datalogger control
- 32 Bit floating point values (REAL)
  - ➔ Analog variables 41-80, Actual values (correction points, mean values etc. can be configured via SE-7xx)

Check licensing status

The correct licensing status of the S7 Modbus connection can be checked in the SE-7xx.
Configuration, *Hardware Test*, *License Information*, *Siemens Modbus Connection* will show the current status.
In case of a missing license this entry shows “No” and a license alarm will be activated.

Activate Siemens Modbus connection in the SE-7xx

First the S7 interface is enabled in the SE-7xx device. This takes place under *Configuration*, *Standard Settings*, *Siemens Modbus Connection*. Change the setting *Modbus enabling* to *Enabled*.

Datalogger configuration (PLC statement list)

For proper functionality of the datalogger when enabling the S7 interface, the following two lines must be present in the STANGE SE-7xx PLC statement list:

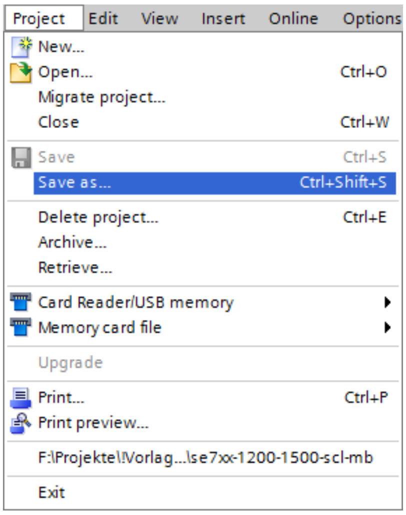
```
L FO 768
R FO 1311
```

The PLC statement list can be found at *Configuration*, *Functions*, *PLC statement list*. After adding those two lines apply the changes by selecting *Apply* (*Take Over*) and then save the changes with *Back*.

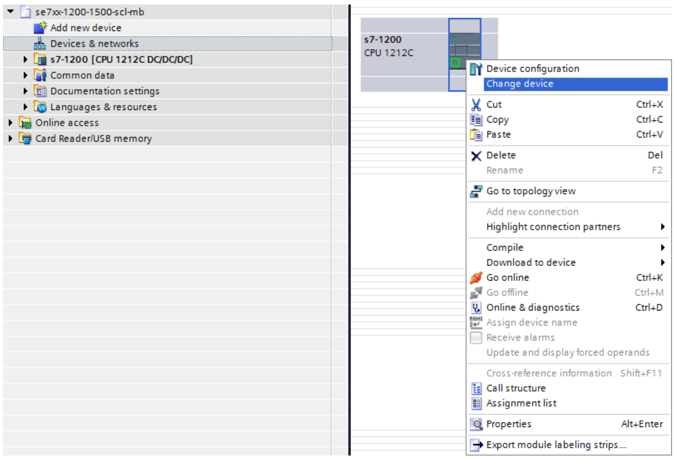
For more information on configuring the datalogger see the corresponding documentation.

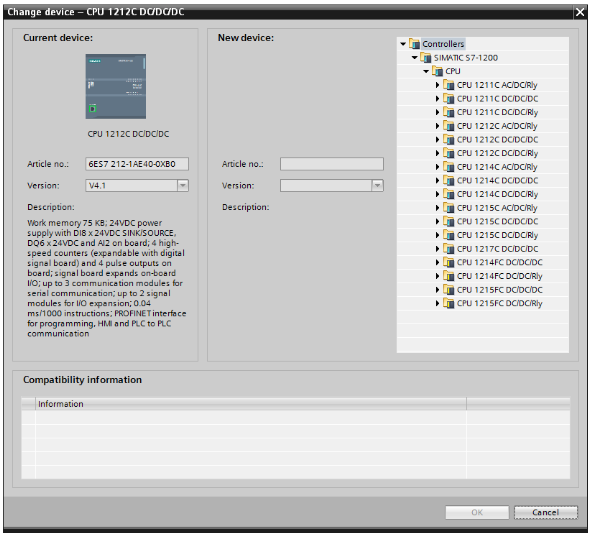
Using the project as a template

The project *se7xx-1200-1500-scl-mb* can be used as a template. It is loaded into the TIA Portal and can be saved directly via *Project > Save as* as a copy with a new name. This enables to use the template again.



In the template project, there are projected one S7-1212C DC/DC/DC and one S7-1513-1 PN, respectively. Both contain the same modules. The project must be adapted if another PLC than S7-1212C DC/DC/DC or S7-1513-1 PN is used: under ***Devices & networks*** the S7 must be selected by right mouse click and ***Change device*** opens the Change device dialog. Now the used device can be selected. The S7 which is not used can be deleted from the project by right mouse click.





CPU 1212C DC/DC/DC

Article no.: 6ES7 212-1AE40-0XB0

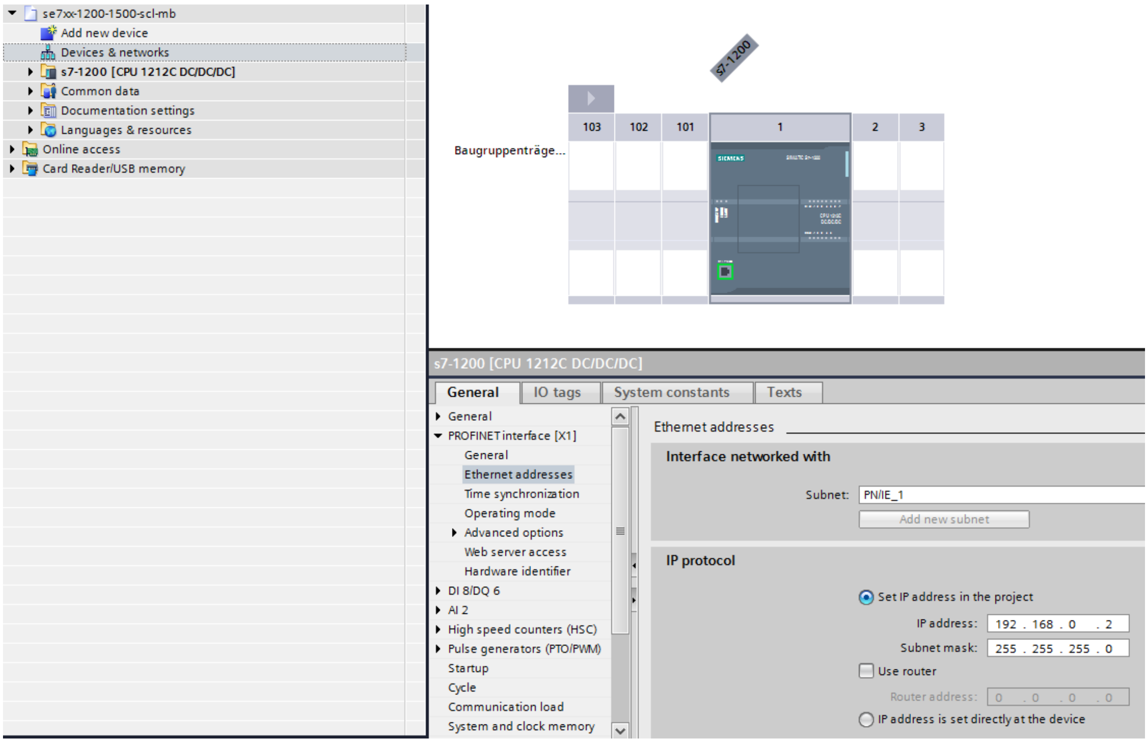
Version: V4.1

Description:

Work memory 75 KB; 24VDC power supply with DI8 x 24VDC SINK/SOURCE, DQ6 x 24VDC and AI2 on board; 4 high-speed counters (expandable with digital signal board) and 4 pulse outputs on board; signal board expands on-board I/O; up to 3 communication modules for serial communication; up to 2 signal modules for I/O expansion; 0.04 ms/1000 instructions; PROFINET interface for programming, HMI and PLC to PLC communication

To change the IP address of the S7, select ***Devices & networks*** and double-click on the S7. Then double-click on the S7 again. Under the category ***PROFINET interface*** the IP address, Subnet mask and Router address (if needed) of the S7 can be set.

Then the S7 can be assigned to an existing „Subnet“ by selecting the correct entry or you can create a new subnet by clicking „Add new subnet“. Either way, this step of assigning a subnet is necessary so TIA Portal can connect to the S7. Finally, the blocks need to be compiled and loaded onto the S7.



Subnet: PN/IE_1

● Set IP address in the project

IP address: 192 . 168 . 0 . 2

Subnet mask: 255 . 255 . 255 . 0

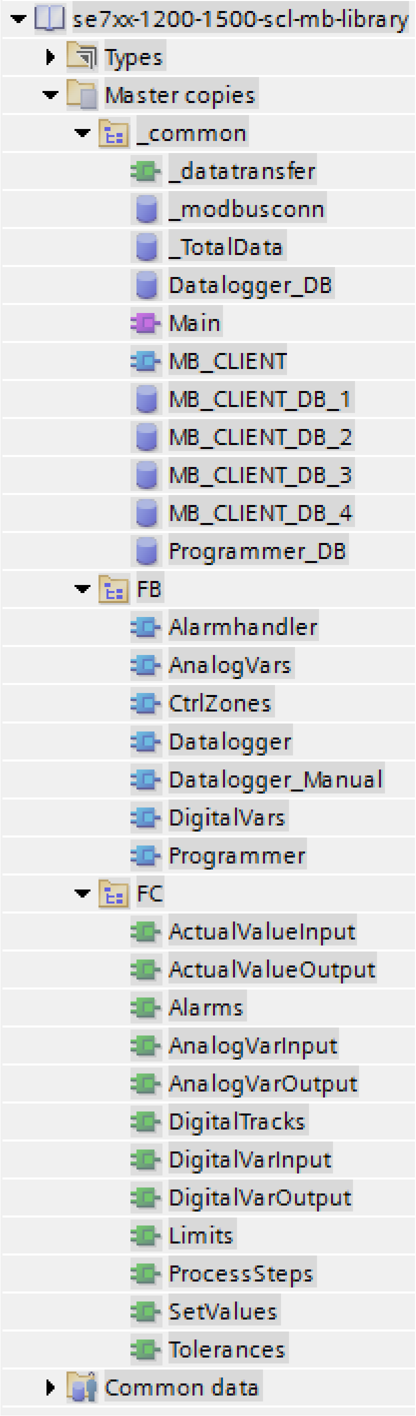
□ Use router

Router address: 0 . 0 . 0 . 0

○ IP address is set directly at the device

Using the library modules in an existing project

The provided library *se7xx-1200-1500-scl-mb-library* can be used if a S7 project already exists in the TIA Portal and only the Modbus communication modules shall be added to the project. The following screenshot shows an overview of the contained modules:



- se7xx-1200-1500-scl-mb-library
  - Types
  - Master copies
    - _common
      - _datatransfer
      - _modbusconn
      - _TotalData
      - Datalogger_DB
      - Main
      - MB_CLIENT
      - MB_CLIENT_DB_1
      - MB_CLIENT_DB_2
      - MB_CLIENT_DB_3
      - MB_CLIENT_DB_4
      - Programmer_DB
    - FB
      - Alarmhandler
      - AnalogVars
      - CtrlZones
      - Datalogger
      - Datalogger_Manual
      - DigitalVars
      - Programmer
    - FC
      - ActualValueInput
      - ActualValueOutput
      - Alarms
      - AnalogVarInput
      - AnalogVarOutput
      - DigitalTracks
      - DigitalVarInput
      - DigitalVarOutput
      - Limits
      - ProcessSteps
      - SetValues
      - Tolerances
  - Common data

The following modules are needed at least in order that the Modbus communication works. They must be copied into the project (***Program blocks***):

- _datatransfer and _modbusconn
- _TotalData
- Datalogger [FB114] (when using the datalogger)
- MB_CLIENT
- MB_CLIENT_DB[1-4]

The project and the library use the system library “S7 Open user communication” in version 4.0.

The blocks need System Flags (Merker). In the device configuration, select “System and clock memory” and check “Enable the use of system memory byte”.

The remaining FC/FB can be integrated to the project as required. But they only work if the above described modules are available in the project. ***_datatransfer*** is the module that performs the actual communication of both devices. It must be integrated via OB1. An example OB1 can be found in the library.

If the datalogger is configured as active in the SE-7xx, the FB ***Datalogger*** must be called via OB1 and be served with an IDB. This step is mandatory, otherwise the SE-7xx datalogger will not work. For full flexibility, the FB ***Datalogger_Manual*** can be used as an alternative (see corresponding chapter).

If both the Programmer block and the Datalogger block are used in the project, the input ***ProgStart*** of the Programmer block must be connected with the bit ***Start_Programmer*** of the Datalogger IDB by an OR operation, otherwise the programmer will not start. If ***ProgStart*** is not connected at all, no changes are needed.

To avoid problems when using the Programmer block and the Datalogger block at the same time, the Programmer block shall be called before the Datalogger block is called. Otherwise the programmer of the SE-7xx may not start. It is not advised to insert ***Programmer*** and ***Datalogger*** more than one time each. To avoid further problems, only one of the two Datalogger FBs should be used.

If the datalogger is not used, the FB ***Datalogger*** (or ***Datalogger_Manual***) do not have to be imported into the project.

Finally, the blocks need to be compiled and loaded onto the S7.

If a compile error occurs (“Network 1: Tag ***transfer_error*** not defined”), a variable tag shall be defined in OB1, Network 1 by clicking with the right mouse button on ***transfer_error*** and selecting “Define tag”. Then the blocks have to be recompiled and reloaded onto the S7.

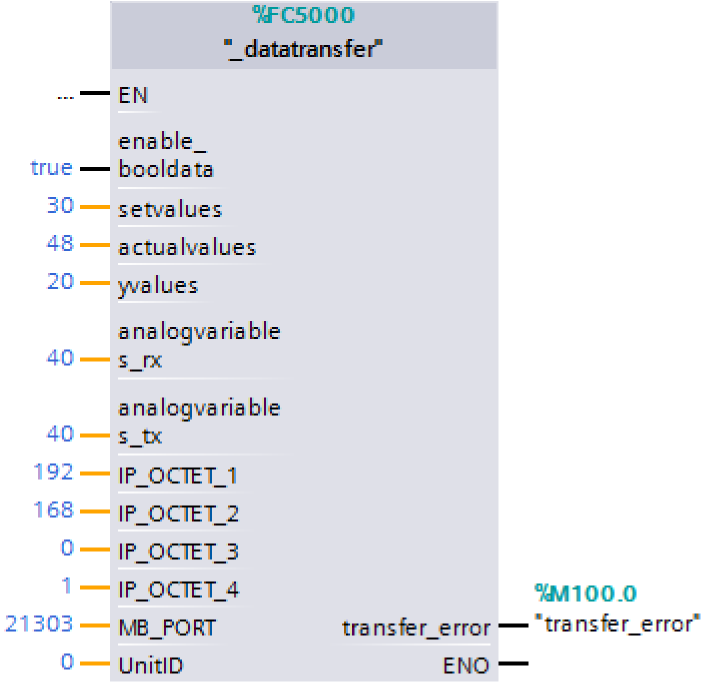
Description of the functionality

General

The most important module is ***Main*** [OB1]. The content is cyclically called and contains at least ***_datatransfer*** [FC5000] and when using the datalogger ***Datalogger*** [FB114]. ***_datatransfer*** controls the general data exchange between both the S7 and the SE-7xx. Without this FC no Modbus communication is possible.

_datatransfer [FC5000]	
Parameter	**Description**
enable_booldata	Transfer boolean control/status values [*true/false*]
setvalues	Number of setvalues [0..30]
actualvalues	Number of actual values [0..48]
yvalues	Number of controller Y values [0..20]
analogvariables_rx	Number of analog variables from SE-7xx [0..40]
analogvariables_tx	Number of analog variables to SE-7xx [0..40]
IP_OCTET_[1..4]	IP address of the SE-7xx (default: 192.168.0.1)
MB_PORT	Modbus/TCP port of the SE-7xx (default: 21303)
UnitID	Modbus Unit ID of the SE-7xx (default: 0)

Please set the IP address and port of the SE-7xx. An error at the Modbus transfer is signaled at output ***transfer_error*** where required. It can be set to a flag or a Bool variable in a DB.



%FC5000
"_datatransfer"
... — EN
true — enable_booldata
30 — setvalues
48 — actualvalues
20 — yvalues
40 — analogvariables_rx
40 — analogvariables_tx
192 — IP_OCTET_1
168 — IP_OCTET_2
0 — IP_OCTET_3
1 — IP_OCTET_4
21303 — MB_PORT
0 — UnitID
transfer_error — %M100.0 "transfer_error"
ENO

The respective blocks correspond to the components of the STANGE device. They can be easily dragged into OB1 or a self-created FC/FB. Then they are integrated by their inputs/outputs into the program sequence.

InstanceNo describes the number of the instance of the function; for example, digital track 4 or limit value 2. Thereby a limit check takes place; i.e. for an InstanceNo outside of the valid range (for instance setvalue 23 in case of a maximum of 20 possible values) the value is set to the maximum possible instance; for values equal to or less than 0 instance 1 is selected.

The number of insertable blocks is not limited. For each inserted FB a separate IDB (Instance DB) is created. Not used inputs/outputs of FCs can be set to an unused flag or variable in a DB. The sequence of instances of a FC/FB does not make any difference; however, each new call of a block with an already used instance number overwrites each previous call of this block with this instance. Not used blocks can be deactivated via input EN (set to false).

Overview FCs/FBs

Name	Block	Function
SetValues	FC101	Reads setvalue and setvalue status
Alarms	FC103	Generates alarm, reads alarm status
ProcessSteps	FC104	Reads process step status
DigitalTracks	FC105	Reads digital track status
Tolerances	FC106	Reads tolerance status, external tolerance activation
Limits	FC107	Reads limit status
DigitalVarInput	FC108	Sets digital input variable in SE-7xx (starting at FI 2000)
DigitalVarOutput	FC109	Reads digital output variable from SE-7xx (starting at FO 2000)
AnalogVarInput	FC110	Sets analog input variable in SE-7xx (values 41-80)
AnalogVarOutput	FC111	Reads analog output variable from SE-7xx (values 1-40)
ActualValueInput	FC112	Sets actual value in SE-7xx, force Overflow/Underflow/Break status
ActualValueOutput	FC113	Reads actual value from SE-7xx, reads actual value error status
Programmer	FB100	Controls Programmer and gets status
CtrlZones	FB102	Controls controlzone and gets status
Alarmhandler	FB103	Controls alarmhandler and gets status
DigitalVars	FB108	Writes and reads digital variables
AnalogVars	FB110	Writes and reads analog variables
Datalogger	FB114	Controls datalogger and gets status (automatic mode)
Datalogger_Manual	FB115	Controls datalogger and gets status (manual mode)

General structure of the FCs/FBs

Inputs: InstanceNo [instance number] and respective function inputs

Outputs: Function outputs

Temp: instno_tmp: Copy of InstanceNo; used for limit check

Constant: entries: contains maximum number of instances; used for limit check

FC5000: _datatransfer and DB5000: _TotalData

The FC5000 ***_datatransfer*** is responsible for sending and receiving data via Modbus TCP. Therefor the system block ***mb_client*** is being used. Four TCP connections are used, each connection multiplexes two data transfers. Each data transfer of a connection is thereby assigned each one of two ***timeslices***. The timeslices are processed one after another and each connection separately.

	Data transmission	
Connection	**c[1-4]_1**	**c[1-4]_2**
1	Booldata.rx	Booldata.tx
2	Setvalues.rx	Actualvalues.rx
3	Yvalues.rx	Actualvalues.tx
4	Analogvariables.rx	Analogvariables.tx

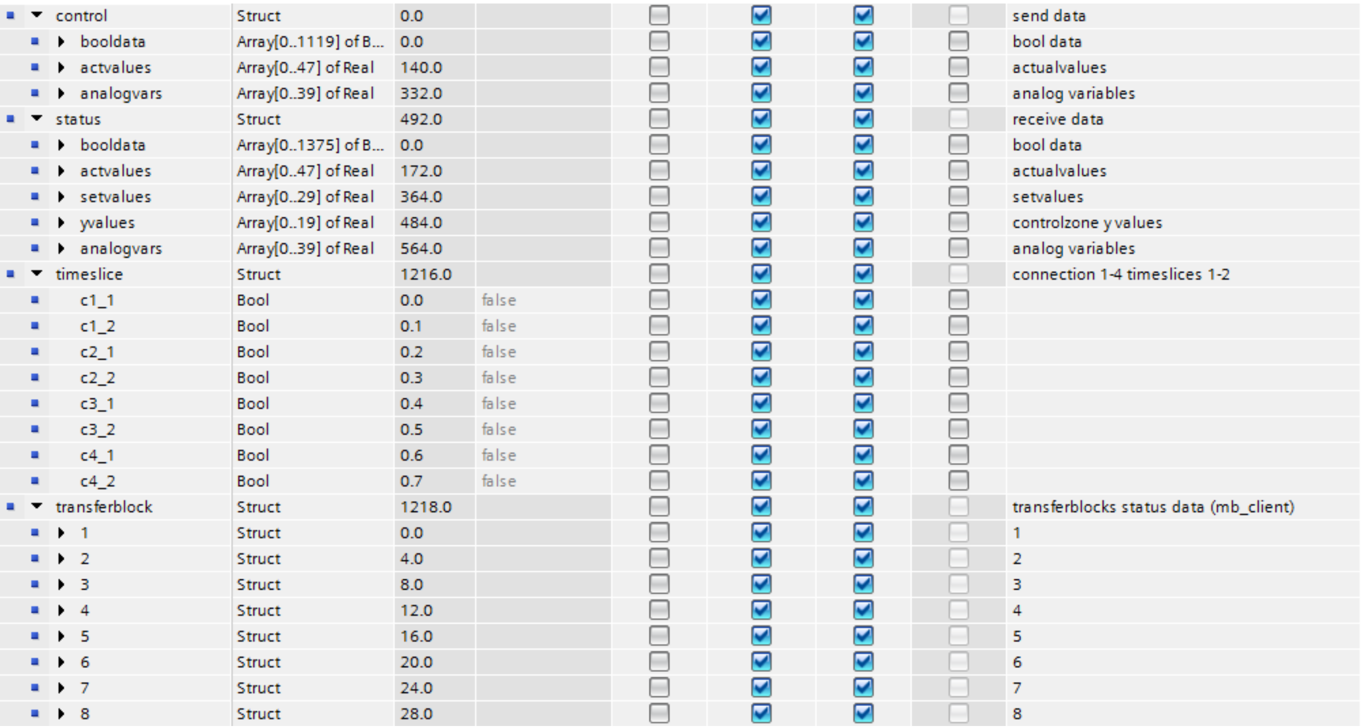
Locally all the data is buffered in the DB ***_TotalData***. All the FC/FB just access this DB when reading or writing data. Data from ***control*** are sent to the SE-7xx, data from SE-7xx are stored in ***status***. It is wise to not use any direct connections with entries from ***_TotalData***, but to use the interface of the FC/FB.

The internal mapping of the Modbus registers to the functions of the SE-7xx is done automatically. No settings or configuration changes are needed here.

There may be some circumstances where the Modbus Unit ID has to be changed in the SE-7xx. In this case this value needs to be applied at the input parameter UnitID of ***_datatransfer***. Otherwise the Modbus connection fails. By default, this value is 0.

- Connection 1:
  - Receive BoolData (172 Byte) status.booldata c1_1
  - Send BoolData (140 Byte) control.booldata c1_2
- Connection 2:
  - Receive setvalues (120 Byte[1]) status.setvalues c2_1
  - Receive actual values (192 Byte[1]) status.actvalues c2_2
- Connection 3:
  - Receive Y values (80 Byte[1]) status.yvalues c3_1
  - Send actual values (192 Byte[1]) control.actvalues c3_2
- Connection 4:
  - Receive analog variables (160 Byte[1]) status.analogvars c4_1
  - Send analog variables (160 Byte[1]) control.analogvars c4_2

[1] when transferring all values (4 Byte per value)



control	Struct	0.0		□	☑	☑	□	send data
booldata	Array[0..1119] of B...	0.0		□	☑	☑	□	bool data
actvalues	Array[0..47] of Real	140.0		□	☑	☑	□	actualvalues
analogvars	Array[0..39] of Real	332.0		□	☑	☑	□	analog variables
status	Struct	492.0		□	☑	☑	□	receive data
booldata	Array[0..1375] of B...	0.0		□	☑	☑	□	bool data
actvalues	Array[0..47] of Real	172.0		□	☑	☑	□	actualvalues
setvalues	Array[0..29] of Real	364.0		□	☑	☑	□	setvalues
yvalues	Array[0..19] of Real	484.0		□	☑	☑	□	controlzone y values
analogvars	Array[0..39] of Real	564.0		□	☑	☑	□	analog variables
timeslice	Struct	1216.0		□	☑	☑	□	connection 1-4 timeslices 1-2
c1_1	Bool	0.0	false	□	☑	☑	□	
c1_2	Bool	0.1	false	□	☑	☑	□	
c2_1	Bool	0.2	false	□	☑	☑	□	
c2_2	Bool	0.3	false	□	☑	☑	□	
c3_1	Bool	0.4	false	□	☑	☑	□	
c3_2	Bool	0.5	false	□	☑	☑	□	
c4_1	Bool	0.6	false	□	☑	☑	□	
c4_2	Bool	0.7	false	□	☑	☑	□	
transferblock	Struct	1218.0		□	☑	☑	□	transferblocks status data (mb_client)
1	Struct	0.0		□	☑	☑	□	1
2	Struct	4.0		□	☑	☑	□	2
3	Struct	8.0		□	☑	☑	□	3
4	Struct	12.0		□	☑	☑	□	4
5	Struct	16.0		□	☑	☑	□	5
6	Struct	20.0		□	☑	☑	□	6
7	Struct	24.0		□	☑	☑	□	7
8	Struct	28.0		□	☑	☑	□	8

_TotalData.control.booldata

Array [0..1119] of Bool.

Contains all Bools which should be sent to the SE-7xx.

_TotalData.status.booldata

Array [0..1375] of Bool.

Contains all Bools received from the SE-7xx.

_TotalData.control.actvalues/analogvars

and

_TotalData.status.actvalues/setvalues/yvalues/analogvars

Array [0..47/39] of Real and Array [0..47/29/19/39] of Real.

Activation by values greater than zero at FC5000. Contains 32-Bit actual values(/setvalues/Y values)/analog variables.

When sending actual values they must not be configured as “unassigned” in the SE-7xx.
The sent analog variables 1-40 are mapped in the SE-7xx as analog variables 41-80.

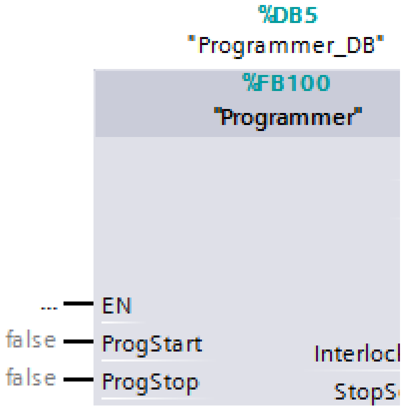
FB100: Programmer and FB114: Datalogger

When the S7 interface is activated, the datalogger only works if the *Datalogger* block is programmed into the program sequence via OB1. This *Datalogger* block contains the logic for the job control of the datalogger. The inputs and outputs of the *Datalogger* FB may be wired in the project, but this is not necessary.

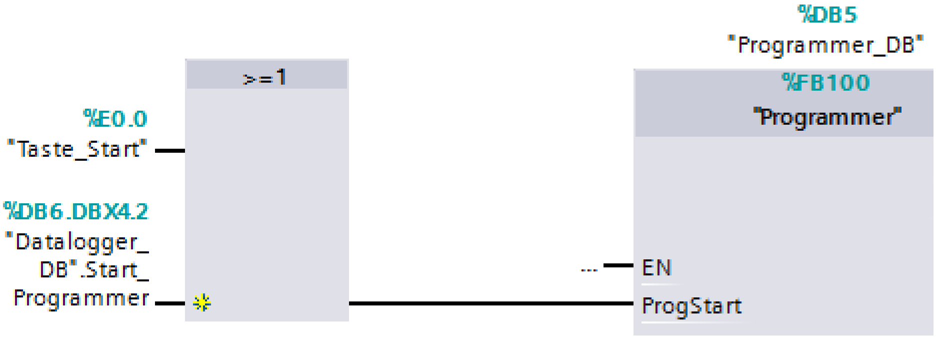
To start the datalogger (and the programmer) via the S7, set an impulse to the input *ProcessStart*. After five seconds the programmer will be started automatically. As soon as the programm has reached the end or the operator selected END or RESET, the logger stops automatically. The recorded log data can be viewed via the log list of the datalogger in the SE-7xx. When the logging is started from the S7, the user "plc" will be displayed in the charge details, otherwise the name of the currently logged in user.

To only start the programmer without the datalogger, you can set an impulse to the input *ProgStart* of *Programmer*.

Because a *Programmer* block may overwrite the programmer start event, its input (*ProgStart*) must be supplied with the bit *Start_Programmer* of the *Datalogger* IDB by an OR block. This step is not necessary if there is no wiring at *ProgStart* at all.



Programmer without wiring at ProgStart



Programmer with wiring at ProgStart

FB114: Datalogger and FB115: Datalogger_Manual

The FB ***Datalogger*** works in “automatic mode”. This means it contains the logic to detect when the user wants to start the programmer via the graphical interface of the SE-7xx and then to finally start the datalogger and the programmer. This logic is necessary since most of the PLC statement list lines got obsolete with the S7 Modbus connection.
Therefore, the FB must just be called via OB1; connections to its inputs/outputs are not necessary. When needed, the datalogger and the programmer can also be started from the S7. Normally this functionality is adequate for most use cases.

But when the user wants full flexibility in controlling the datalogger, ***Datalogger_Manual*** can be used. It contains no logic, but offers no limits in processing the control and status signals.
To avoid problems, only one of both FB should be used in the program.

When the datalogger is enabled in the SE-7xx configuration, the Start button on the Programmer page only creates a process start event (and does not start the programmer yet). This event is displayed on the output ***ProcessstartActive***. Also, the input ***ProcessStart*** creates a process start event. The event mainly sets the right user name in the batch list of the SE-7xx (“plc” if started from the S7).

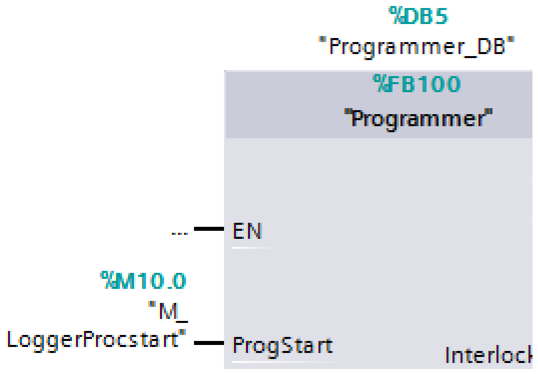
ProcessstartActive can be used to trigger ***LogStart*** to start the datalogger. Also, the input ***ProgStart*** of ***Programmer*** gets the signal to start the programmer. ***ProcessstartActive*** will be reset automatically when the programmer is running (these are the two lines which must be inserted into the SE-7xx PLC statement list).

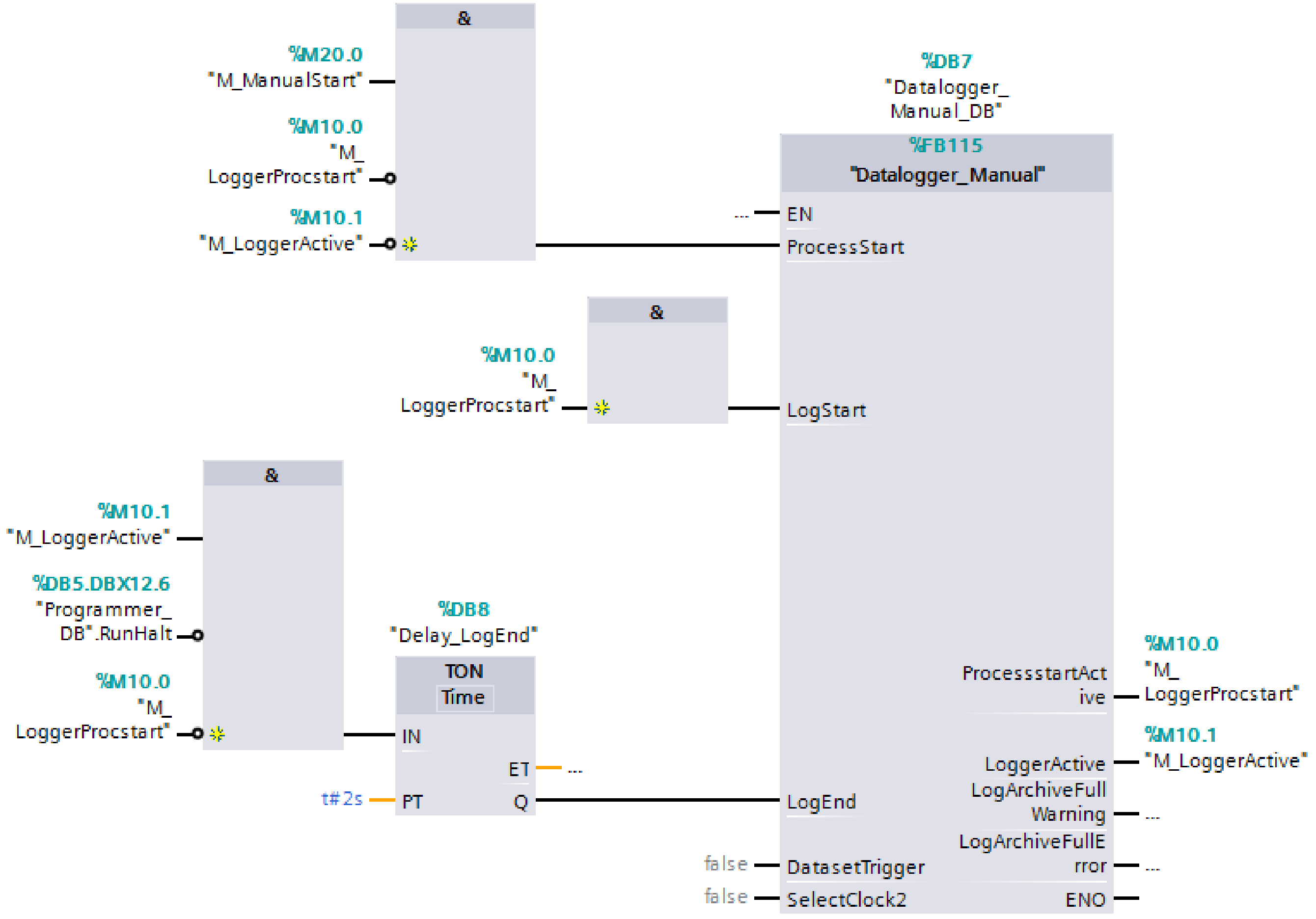
LoggerActive shows that the datalogger is recording. Using ***LogEnd*** the recording will be finished.

On the following page there is an example of using ***Datalogger_Manual***.

The process start event can either be created locally (***M_ManualStart***); under the condition that no other event exists and the datalogger is not active. Or the event can be created by the SE-7xx (Programmer).
ProcessstartActive outputs a signal through the event. This signal (stored in ***M_LoggerProcstart***) can then be used to start the datalogger recording using ***LogStart***. The flag ***M_LoggerActive*** then shows the active status of the datalogger.
This flag will then be used to end the datalogger recording as soon as the programmer reaches the program end or RESET.

Starting of the datalogger and the programmer can be achieved by just applying an impulse to ***M_ManualStart***. The status outputs can be processed in the program if necessary.
As an alternative, the FB ***Datalogger*** can be used. It already contains all the logic.


%DB5
"Programmer_DB"
%FB100
"Programmer"
... EN
%M10.0
"M_
LoggerProcstart" ProgStart
Interlock


&
%M20.0
"M_ManualStart"
%M10.0
"M_
LoggerProcstart"
%M10.1
"M_LoggerActive"
%DB7
"Datalogger_
Manual_DB"
%FB115
"Datalogger_Manual"
... EN
ProcessStart
&
%M10.0
"M_
LoggerProcstart"
LogStart
&
%M10.1
"M_LoggerActive"
%DB5.DBX12.6
"Programmer_
DB".RunHalt
%M10.0
"M_
LoggerProcstart"
%DB8
"Delay_LogEnd"
TON
Time
IN
ET ...
t#2s PT
Q
LogEnd
ProcessstartActive
%M10.0
"M_
LoggerProcstart"
LoggerActive
%M10.1
"M_LoggerActive"
LogArchiveFullWarning ...
LogArchiveFullError ...
false DatasetTrigger
false SelectClock2
ENO

FC108/FC109, FB108: DigitalVarInput, DigitalVarOutput, DigitalVars (Digital variables)

Digital input variables of the SE-7xx can be set from the S7 with the blocks ***DigitalVarInput*** and ***DigitalVars***. These will be mapped to Function inputs (FI) 2000 to 2199 of the SE-7xx and can be used for status displays in the Visualisation, for example.
These variables can only be written and not be read by the S7.

Digital output variables of the SE-7xx can be read from the S7 with the blocks ***DigitalVarOutput*** and ***DigitalVars***. These are mapped to Function outputs (FO) 2000 to 2199 of the SE-7xx and can be used for buttons in the Visualisation, for example.
These variables can only be read and not be written by the S7.

FC110/FC111, FB110: AnalogVarInput, AnalogVarOutput, AnalogVars (Analog variables)

Analog input variables of the SE-7xx can be set from the S7 with the blocks ***AnalogVarInput*** and ***AnalogVars***. These will be mapped to analog variables 41-80 of the SE-7xx and can be used for status displays in the Visualisation or as substituting control zone setvalues, for example.
These variables can only be written and not be read by the S7.

Analog output variables of the SE-7xx can be read from the S7 with the blocks ***AnalogVarOutput*** and ***AnalogVars***. These are mapped to analog variables 1-40 of the SE-7xx and can be used for input fields in the Visualisation, for example.
These variables can only be read and not be written by the S7.

Analog variables must be configured as IEEE-Float before using them (configuration in SE-7xx).

FC112/FC113: ActualValueInput, ActualValueOutput (Actual values)

Actual values of the SE-7xx can be set from the S7 with the block ***ActualValueInput***. These can be used there as a controller actual value, for example. They can also be configured with correction points, mean values, etc.

When sending actual values they must not be configured as “unassigned” in the SE-7xx, but at least “linear“.

Special values according to IEEE 754 can trigger an actual value alarm in the SE-7xx:

Actual value	IEEE 754 description	SE-7xx
0x7F800000	positive infinity	Overflow
0xFF800000	negative infinity	Underflow
0x7F800001 ff.	signalling NaN	Break
0xFF800001 ff.	signalling NaN	Break
0x7FC00000 ff.	quiet NaN	Break
0xFFC00000 ff.	quiet NaN	Break

The respective status can be generated in the SE-7xx with the inputs ***ForceOverflow***/***ForceUnderflow***/***ForceBreak***.

Actual values of the SE-7xx can be read from the S7 with the block ***ActualValueOutput***. The output ***ActValueError*** will change to true if the actual value has an error (for example break).

How To

External setvalue supply by S7

Although the SE-7xx has the ability to store programm recipes which result into setvalue definitions over time, setvalues also may be gathered remotely by an S7. This makes the programmed setvalue definition worthless as it will not come into effect. The SE-7xx will then calculate Y values with these external setvalues ignoring setvalues coming directly from the programmer.

A direct modification of the internal programmer setvalues is not possible due to technical reasons. However, there is a way to control which setvalue a control zone gets. The idea is to provide the control zone a substituting setvalue. This substituting setvalue is supplied by an analog variable from the S7. On the S7 this is simply done by inserting a block providing the desired setvalue as an input parameter. Finally, the control zone will be configured to activate the substituting setvalue permanently.

This needs a one-time configuration in the SE-7xx. In the new S7 Modbus interface the 80 analog variables of the SE-7xx are divided into 40 read values (1-40) and 40 write values (41-80). Therefore, there must be at least 41 analog variables configured so the S7 can at least write one analog variable to the SE-7xx.

➔ Configuration > Functions > Analog Variables > Param. (button on the left side)
Please configure at least **41** values. After touching "Back" the variables list is shown again.
For example, analog variable 41 will be configured.
After selecting variable 41 and "Edit" the configuration page is displayed. A meaningful description will help finding the variable later. The variable type must be configured to **IEEE Float**. The display format allows setting the decimal places (the more decimal places, the less digits left of the decimal point). A maximum of two decimal places is advised. Below the low and high limit of the analog variable – the external setvalue – can be changed. You can just set them to their maximum, **-99999.9** or **99999.9**, respectively.
The control system address is not applicable here. The initialization mode changes the behavior of the analog variable after a reset. It can be changed if needed. The default value is **None**.

The configuration of the analog variable is now done.

➔ Configuration > Functions > Control Zones
Here all the control zones are listed. Select the respective control zone which shall be supplied with the external setvalue and choose "Edit". Scroll down to "Subst. SV Type" and select **Variable**.
The substituting setvalue number corresponds to the number of the analog variable which will contain the external setvalue, for example: **41**. The description of the analog variable is shown in parantheses.

The configuration can now be saved by exiting. This completes the configuration in the SE-7xx.

In the S7, the ***AnalogVarInput*** block can be used to send the analog variable to the SE-7xx. At the input parameter ***Value***, the setvalue to be sent is specified in REAL format (float). ***InstanceNo*** = **1** then corresponds to analog variable 41 in the SE-7xx, which was configured above as a substitute setvalue. Similarly, other alternative setvalues can also be defined, which are then transferred to the SE-7xx as analog variables 42, etc.

The ***CtrlZones*** block is used to enable the substitute setvalue at the selected control zone. ***InstanceNo*** specifies the number of the control zone (e. g. **1**). By setting **true** at the ***EnableSubstSV*** input, the alternative setvalue configured in the SE-7xx is finally activated. The current setvalue can then be displayed on the "Controller" page.

The SE-7xx controls independently of the status of the programmer. Using the ***Disable*** input of ***CtrlZones***, the control zone can be deactivated if necessary, i. e. the Y controller output can be set to 0.0.

To obtain a Start/Stop signal from the SE-7xx, a button can be defined in the Visualization, whose status can be read out via ***DigitalVarOutput***. The other possibility would be to create a pseudo-program recipe in order to start and stop the programmer as usual. This would also have the advantage that the integrated datalogger could then create batch records.

Description of the interface (FC/FB)

_datatransfer [FC5000]

Data transfer between datablock _TotalData and SE-7xx.

Input	Format	Function
enable_booldata	Bool	Enable to transfer bool data between SE-7xx and S7
setvalues	UInt	Set the number of Setvalues to receive from SE-7xx
actualvalues	UInt	Set the number of Actualvalues to exchange with SE-7xx
yvalues	UInt	Set the number of Yvalues to receive from SE-7xx
analogvariables_rx	UInt	Set the number of Analogvariables to receive from SE-7xx
analogvariables_tx	UInt	Set the number of Analogvariables to send to SE-7xx
IP_OCTET_[1..4]	UInt	First/second/third/fourth part of IPv4 address of SE-7xx
MB_PORT	UInt	Port number of SE-7xx (default: 21303)
UnitID	UInt	Modbus Unit ID of SE-7xx (default: 0)

Output	Format	Function
transfer_error	Bool	Error flag: one or more transfer failed

ActualValueInput [FC112]

Sends a float value as an actual value to the SE-7xx. The configured actual value must not be “unassigned”.
InstanceNo: 1..48

Input	Format	Function
InstanceNo	Int	Number of Actualvalue
Input	Real	Actualvalue input
ForceOverflow	Bool	Force Overflow signal on this Actualvalue
ForceUnderflow	Bool	Force Underflow signal on this Actualvalue
ForceBreak	Bool	Force Break signal on this Actualvalue

ActualValueOutput [FC113]

Reads an actual value and its error condition from the SE-7xx.
InstanceNo: 1..48

Input	Format	Function
InstanceNo	Int	Number of Actualvalue

Output	Format	Function
Value	Real	Actualvalue output (value)
ActValueError	Bool	Actualvalue has an error

Alarms [FC103]

Generates an alarm in SE-7xx (1-200) and reads current alarm status (1-240) from SE-7xx.
System alarms (201-240) can only be read and AlarmInput will be ignored.
InstanceNo: 1..240

Input	Format	Function
InstanceNo	Int	Number of alarm
AlarmInput	Bool	Generate selected alarm

Output	Format	Function
AlarmOutput	Bool	Current alarm status

AnalogVarInput [FC110]

Sends a float value as an analog variable to the SE-7xx (analog variables 41-80).
Analog input value 1 will be mapped as analog variable 41.
InstanceNo: 1..40

Input	Format	Function
InstanceNo	Int	Number of analog variable input
Value	Real	Value of analog variable input

AnalogVarOutput [FC111]

Reads an analog variable from the SE-7xx (analog variables 1-40).
InstanceNo: 1..40

Input	Format	Function
InstanceNo	Int	Number of analog variable output

Output	Format	Function
Value	Real	Value of analog variable output

DigitalTracks [FC105]

Reads current status of the selected digital track from the SE-7xx.
InstanceNo: 1..64

Input	Format	Function
InstanceNo	Int	Number of digital track

Output	Format	Function
State	Bool	Digital track active

DigitalVarInput [FC108]

Sends a digital variable to the SE-7xx. They are mapped as FI 2000-2199.
InstanceNo: 1..200

Input	Format	Function
InstanceNo	Int	Number of digital input
State	Bool	State of selected digital input

DigitalVarOutput [FC109]

Reads digital variable from the SE-7xx. They are mapped as FO 2000-2199.
InstanceNo: 1..200

Input	Format	Function
InstanceNo	Int	Number of digital output

Output	Format	Function
State	Bool	State of selected digital output

Limits [FC107]

Reads current status of the selected limit from the SE-7xx.
InstanceNo: 1..40

Input	Format	Function
InstanceNo	Int	Number of limit

Output	Format	Function
Crossed	Bool	Limit crossed

ProcessSteps [FC104]

Reads current status of the selected process step from the SE-7xx.
InstanceNo: 1..50

Input	Format	Function
InstanceNo	Int	Number of process step

Output	Format	Function
State	Bool	Process step active

SetValues [FC101]

Returns value and status of setvalue from the SE-7xx.
InstanceNo: 1..30

Input	Format	Function
InstanceNo	Int	Number of setvalue

Output	Format	Function
Value	Real	Value of setvalue
ManualSVEnabled	Bool	Manual setvalue setting enabled
SVRising	Bool	Setvalue is rising
SVConst	Bool	Setvalue is constant
SVFalling	Bool	Setvalue is falling
SVRampsection	Bool	Setvalue is currently in ramp section

Tolerances [FC106]

Enables tolerance (if configured as external) and returns status of the selected tolerance from the SE-7xx.
InstanceNo: 1..40

Input	Format	Function
InstanceNo	Int	Number of tolerance
EnableTol	Bool	Enable tolerance

Output	Format	Function
PlusTolCrossed	Bool	Upper tolerance crossed
MinusTolCrossed	Bool	Lower tolerance crossed

Alarmhandler [FB103]

Controls the alarmhandler of the SE-7xx and returns its status.

Input	Format	Function
AckAcoustic	Bool	Acknowledge acoustic alarm
AckOptical	Bool	Acknowledge optical common alarm
AlarmComing_bcdbin	Bool	Alarm is coming; using BCD/binary notation for alarm selection
AlarmGoing_bcdbin	Bool	Alarm is going; using BCD/binary notation for alarm selection
ClearAll	Bool	Clear all alarms
Lock209	Bool	Lock or unlock Alarm 209 (void actualvalues)

Output	Format	Function
AcousticAck	Bool	Acoustic alarm has been acknowledged
OpticalAck	Bool	Optical alarm has been acknowledged
AcousticOut	Bool	Acoustic alarm output
OpticalOut	Bool	Optical alarm output
CommonOut	Bool	Common alarm output
FeedbackCommonack	Bool	Feedback for common acknowledging (acknowledging all alarms)
FeedbackSingleack	Bool	Feedback for single acknowledging (acknowledging one alarm)
AlarmnrReceived_bcdbin	Bool	The alarm number in BCD/binary format has been received
Priority1	Bool	Priority 1 alarm active
Priority2	Bool	Priority 2 alarm active
Priority3	Bool	Priority 3 alarm active
Priority4	Bool	Priority 4 alarm active
Priority5	Bool	Priority 5 alarm active
Priority6	Bool	Priority 6 alarm active
Priority7	Bool	Priority 7 alarm active
Priority8	Bool	Priority 8 alarm active

AnalogVars [FB110]

Sends and reads multiple analog variables to/from the SE-7xx.
Analog input variables are written to analog variables 41-80 of the SE-7xx.
Analog output variables are read from analog variables 1-40 of the SE-7xx.

Input	Format	Function
Input1	Real	Value of analog variable input
Input2	Real	Value of analog variable input
[…]	[…]	[…]
Input40	Real	Value of analog variable input

Output	Format	Function
Output1	Real	Value of analog variable output
Output2	Real	Value of analog variable output
[…]	[…]	[…]
Output40	Real	Value of analog variable output

CtrlZones [FB102]

Controls control zone settings of the SE-7xx and returns its status.
InstanceNo: 1..20

Input	Format	Function
InstanceNo	Int	Number of controller
PIDselect	Int	Number of PID parameter set (1-8)
Disable	Bool	Disable controller
EnableYlimit	Bool	Enable Y limiter for controller
EnableSubstSV	Bool	Enable substituting setvalue for controller
EnableSubstAV	Bool	Enable substituting actual value for controller
EnableYhandConstVal	Bool	Enable Y-HAND constant value
EnableXtrack	Bool	Enable X-Tracking for controller
EnableYtrack	Bool	Enable Y-Tracking for controller

Output	Format	Function
Value	Real	Y-value for controller
Heating	Bool	Controller is heating
Cooling	Bool	Controller is cooling
AVVoidalarm	Bool	Alarm: Value is broken
AVTolerancealarm	Bool	Alarm: Value is out of tolerance
YhandActive	Bool	Y-HAND is active
XtrackAct	Bool	X-Tracking is active
YtrackAct	Bool	Y-Tracking is active
MinusTolCrossed	Bool	Value is lower than lower tolerance
PlusTolCrossed	Bool	Value is higher than upper tolerance
LowLimCrossed	Bool	Value is lower than lower limit
HighLimCrossed	Bool	Value is higher than upper limit

DigitalVars [FB108]

Sends and reads multiple digital variables to/from the SE-7xx.
Digital inputs are written to FI 2000-2199. Digital outputs are read from FO 2000-2199.
Shift10 can be used to set the focus on which values to write/read; e.g. if Shift10 is 5, digital variables 51-60 are written/read.
Shift10: 0..19

Input	Format	Function
Shift10	Int	Offset multiplicated by 10 to access all 200 inputs/outputs
Input1	Bool	Digital input
Input2	Bool	Digital input
[…]	[…]	[…]
Input10	Bool	Digital input

Output	Format	Function
Output1	Bool	Digital output
Output2	Bool	Digital output
[…]	[…]	[…]
Output10	Bool	Digital output

Programmer [FB100]

Controls the programmer of the SE-7xx and returns its status.

Input	Format	Function
ProgStart	Bool	Program control: START program (without datalogger)
ProgStop	Bool	Program control: STOP program
ProgReset	Bool	Program control: RESET program
ProgInterlock	Bool	Program control: INTERLOCK program
JumpNextSect	Bool	Program control: Jump to next section
JumpProgEnd	Bool	Program control: Jump to program end
StopSectEndEnable	Bool	Program control: Stop at section end [static]
ContSectEnd	Bool	Program control: Continue (if section end reached) [impulse]
SetNoProg	Bool	Program control: Set current program to "no program"
SetProg	Bool	Program control: Select program using SetProgNr (integer)
SetProgNr	Int	Program control: Set program number
JumpAV	Bool	Program control: Feature "Jump to actual value"
JumpAVDest	Int	Program control: Selection of controlzone for feature “Jump to actual value"

Output	Format	Function
ProgNr	Int	Program status: Current program number
SectNr	Int	Program status: Current section number
Reset	Bool	Program status: RESET
Run	Bool	Program status: RUN
Stop	Bool	Program status: STOP
InterlockActive	Bool	Program status: INTERLOCK active
StopSectEnd	Bool	Program status: STOP after reaching section end
ProgEnd	Bool	Program status: Program END
RunHalt	Bool	Program status: Program in RUN or STOP
PwrFailStop	Bool	Program status: STOP after power failure
AVNotfound	Bool	Program status: Provided actual value not found
NewSectLoaded	Bool	Program status: New program section loaded
ProgSelected	Bool	Program status: Program selected
ProgNotfound	Bool	Program status: Program not found
CurrentProgChanged	Bool	Program status: Current program changed
StarttimeEnabled	Bool	Program status: Program start at specific time/date enabled

Datalogger [FB114]

Controls the SE-7xx datalogger and returns its status („automatic mode“).
Must be inserted if the datalogger is used (after the Programmer block).
Do not use with **Datalogger_Manual [FB115]**.

Input	Format	Function
ProcessStart	Bool	Starts the datalogger and after 5 seconds starts the programmer
DatasetTrigger	Bool	Trigger dataset
SelectClock2	Bool	Select clock 2 instead of clock 1 for data logging

Output	Format	Function
ProcessstartActive	Bool	Received a local (PLC) or remote (SE-7xx) process start event
LoggerActive	Bool	Datalogger active
LogArchiveFullWarning	Bool	Log archive nearly full
LogArchiveFullError	Bool	Log archive completely full

Datalogger_Manual [FB115]

Controls the SE-7xx datalogger and returns its status („manual mode“).
Must be inserted if the datalogger is used (after the Programmer block).
Do not use with **Datalogger [FB114]**.

Input	Format	Function
ProcessStart	Bool	Generates a process start event
LogStart	Bool	Starts the current datalogger recording
LogEnd	Bool	Stops the current datalogger recording
DatasetTrigger	Bool	Trigger dataset
SelectClock2	Bool	Select clock 2 instead of clock 1 for data logging

Output	Format	Function
ProcessstartActive	Bool	Received a local (PLC) or remote (SE-7xx) process start event
LoggerActive	Bool	Datalogger active
LogArchiveFullWarning	Bool	Log archive nearly full
LogArchiveFullError	Bool	Log archive completely full