



P2M2HBVN (PROFINET) FUNCTION BLOCK(S) FOR SIEMENS 1200/1500 PLC SET UP GUIDE

PREFACE

This Set Up Guide is designed to help integrate Parker Hannifin's P2M Industrial Ethernet for Profinet valve module into a Siemens S7 1200/1500 PLC environment. The sample code and Function Blocks (FB) were developed in TIA Portal v15.

The "FB_TIA_P2M2HBVN_PRD_Rx" FB facilitates the call-up of the acyclic service data from the P2M Industrial Ethernet for Profinet module utilizing RDREC and WRREC functions from the standard Siemens library.

The "FB_TIA_P2M2HBVN_PD_Rx" FB facilitates communication and handling of process data between PLC and the P2M Industrial Ethernet Profinet module utilizing GETIO and SETIO functions from the standard Siemens library.

*NOTE the standard Siemens Function blocks used in these function blocks are only compatible with S7 1200 and 1500 PLC's.

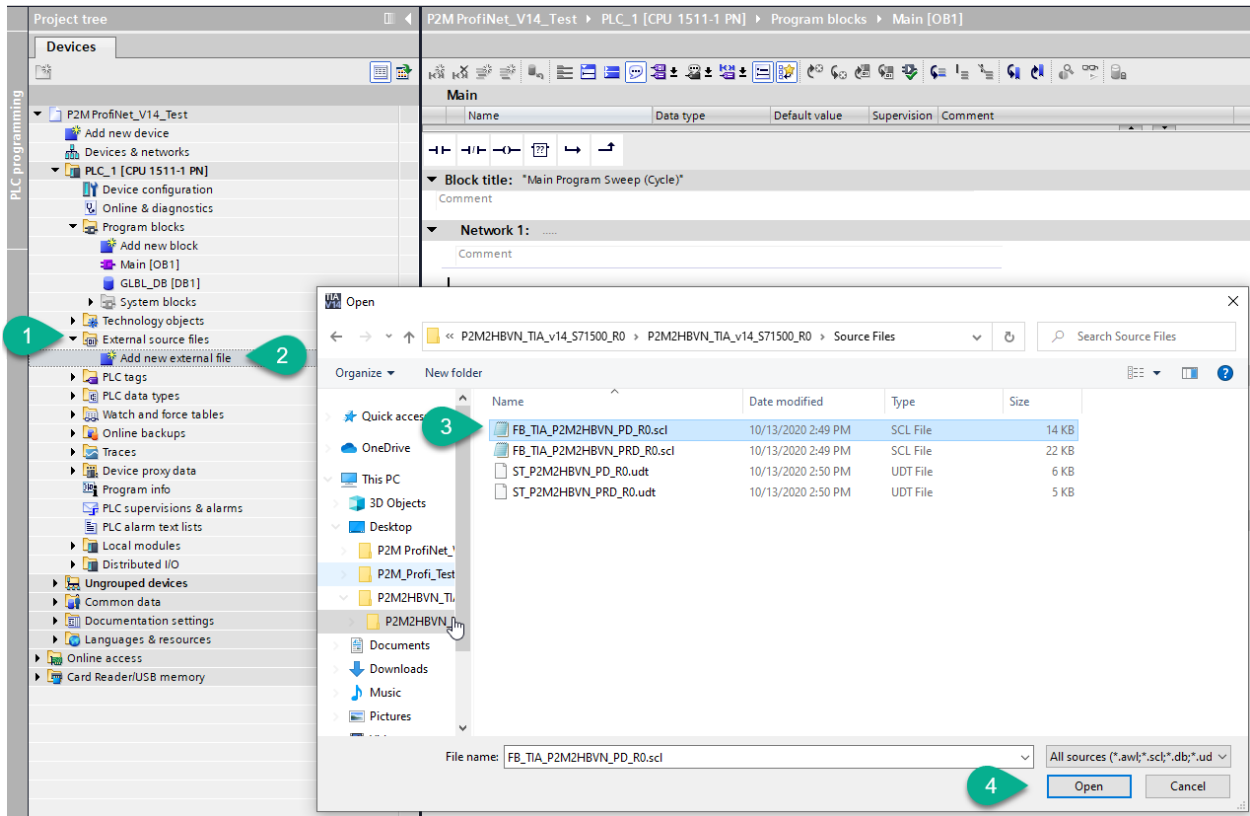
You can download resources such as the GSDML configuration file and the full user manual here:

<https://ph.parker.com/us/17571/en/p2m-ethernet-node>

CONTENTS

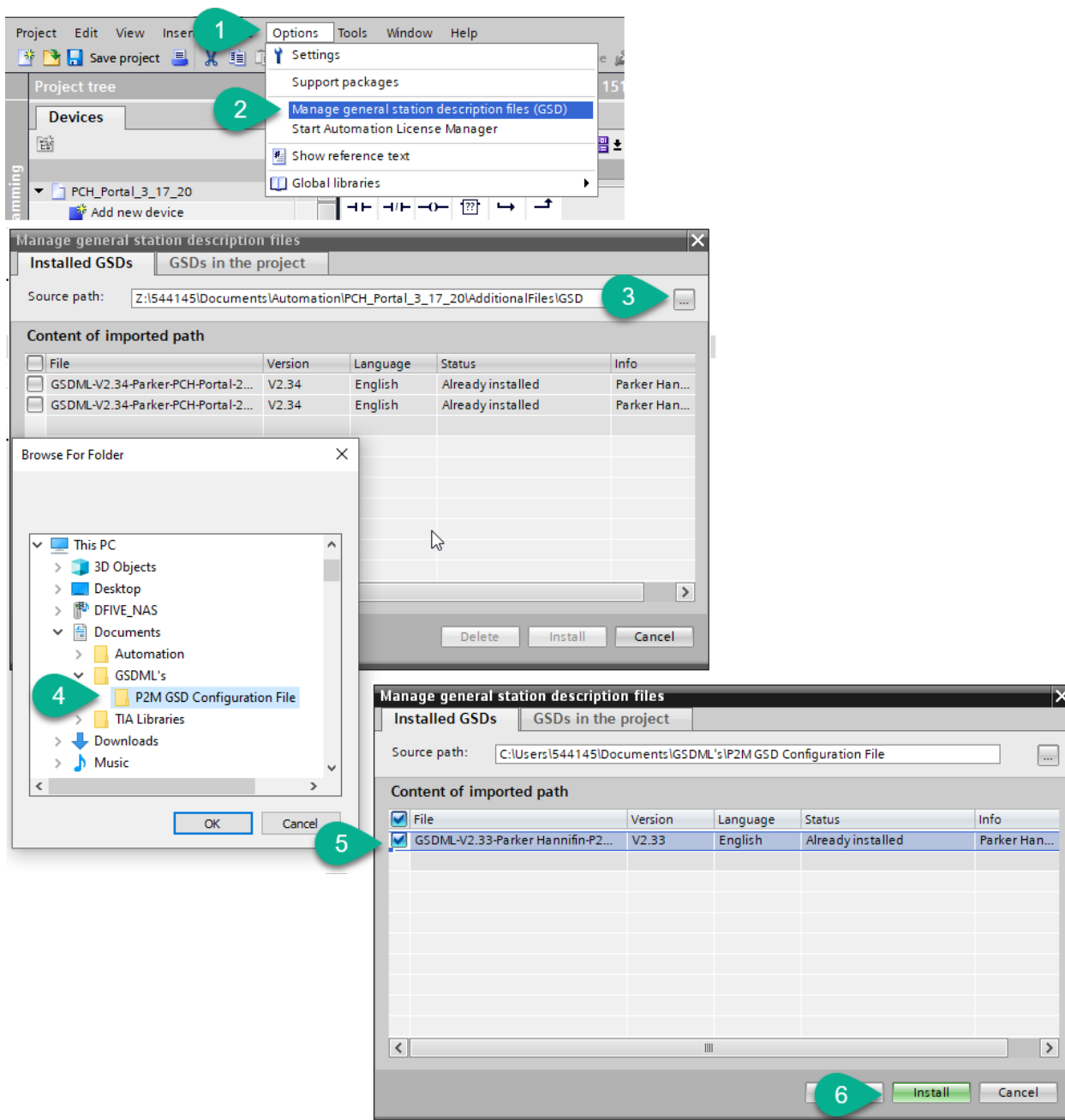
PREFACE	2
Importing P2M ProFINET GSDML.....	5
Adding the P2m to the program	6
IMPORTING THE FUNCTION BLOCK LIBRARY.....	7
USING Process Data Function Block.....	8
Setting up the Process Data Function Block	11
Descripton of the other FB FUNCTIONS.....	12
“FB_TIA_P2M2HBVN_PD_R0” Error Codes	13
Using P2M Parameter data function block.....	15
Setting up the PARAMETER Data Function Block	18
Reading Data using Parameter function block	19
Writing Data using Parameter function block	21
Program examples of using the Function Blocks	27
Example of Process Data function Block.....	27
Example of firing solenoids	28
Example of system command function.....	29
Example of clearing all counts	30
Example of clearing individual cycle count	31
Example of writing aux voltage low and high setpoint.....	32
Example of writing solenoid state behavior	33
Example of using the parameter function block.....	34
How to import blocks using the source files.....	35
Adding External Source Files in TIA.....	36

P2M2HBVN TIA Portal S7 1200/1500 Function Blocks



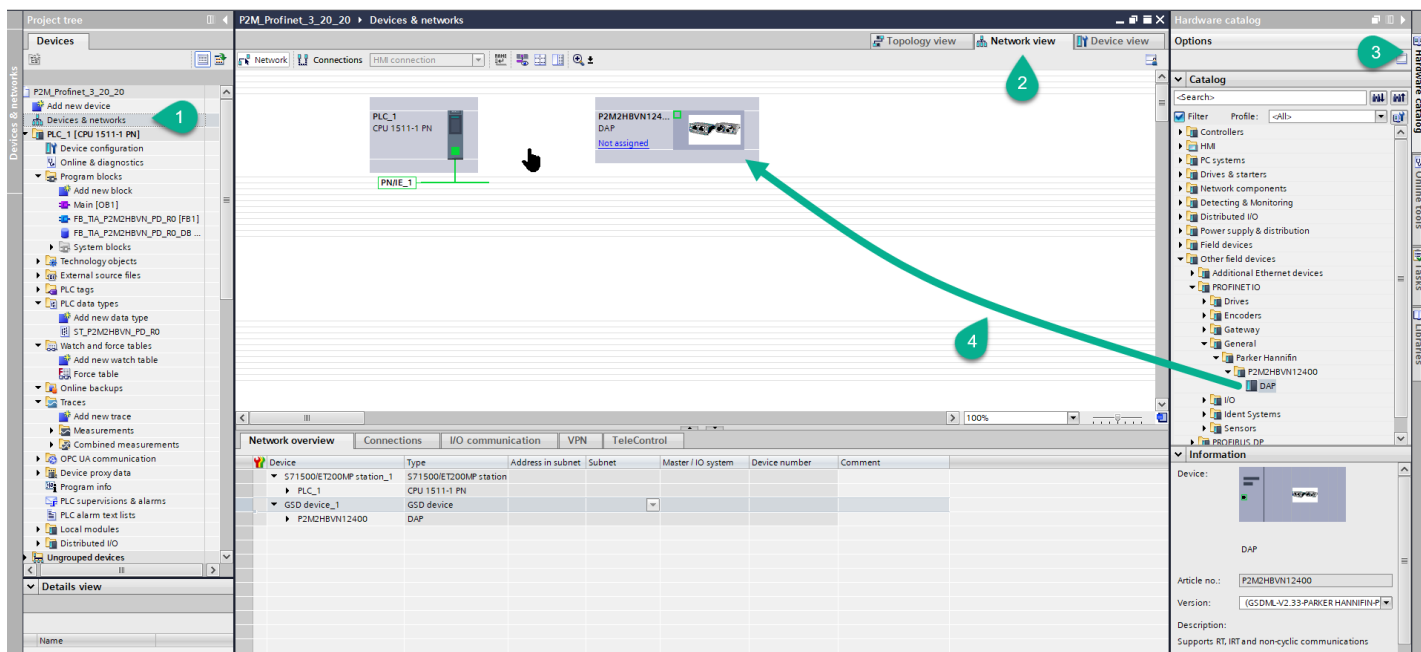
IMPORTING P2M PROFINET GSDML

1. Go to Options
2. Select Manage general station description files (GSD)
3. Click 3 dots to find the directory of where the GSD file is saved
4. Select the directory and click OK
5. Select the P2M GSD file
6. Click install




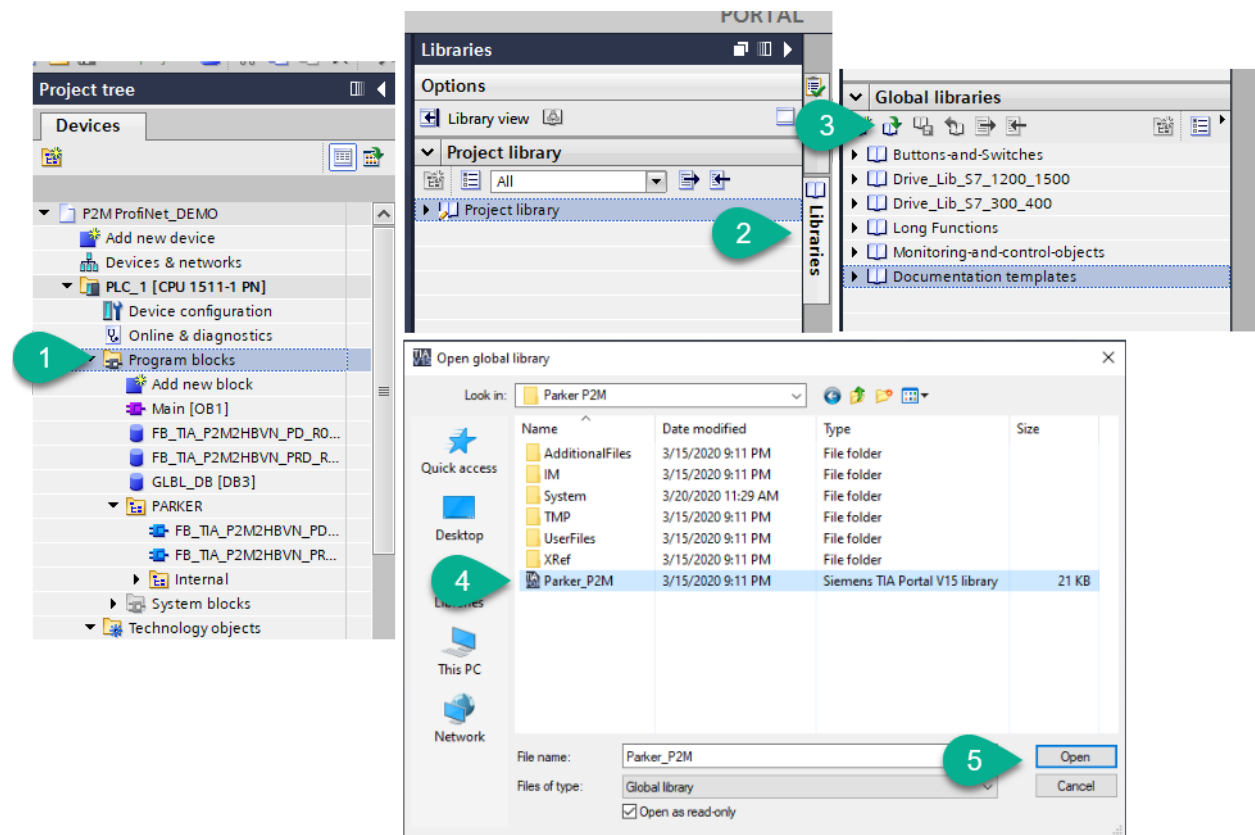
ADDING THE P2M TO THE PROGRAM

1. Select devices and networks in Project tree.
2. Select the Network view tab.
3. Select Hardware catalog tab.
4. Find the P2M DAP (Device Access Point) in the hardware catalog: Other field devices -> PROFINET IO -> General -> Parker Hannifin ->P2M2HBVN12400 . Then select the DAP and drag it to the Network view tab.



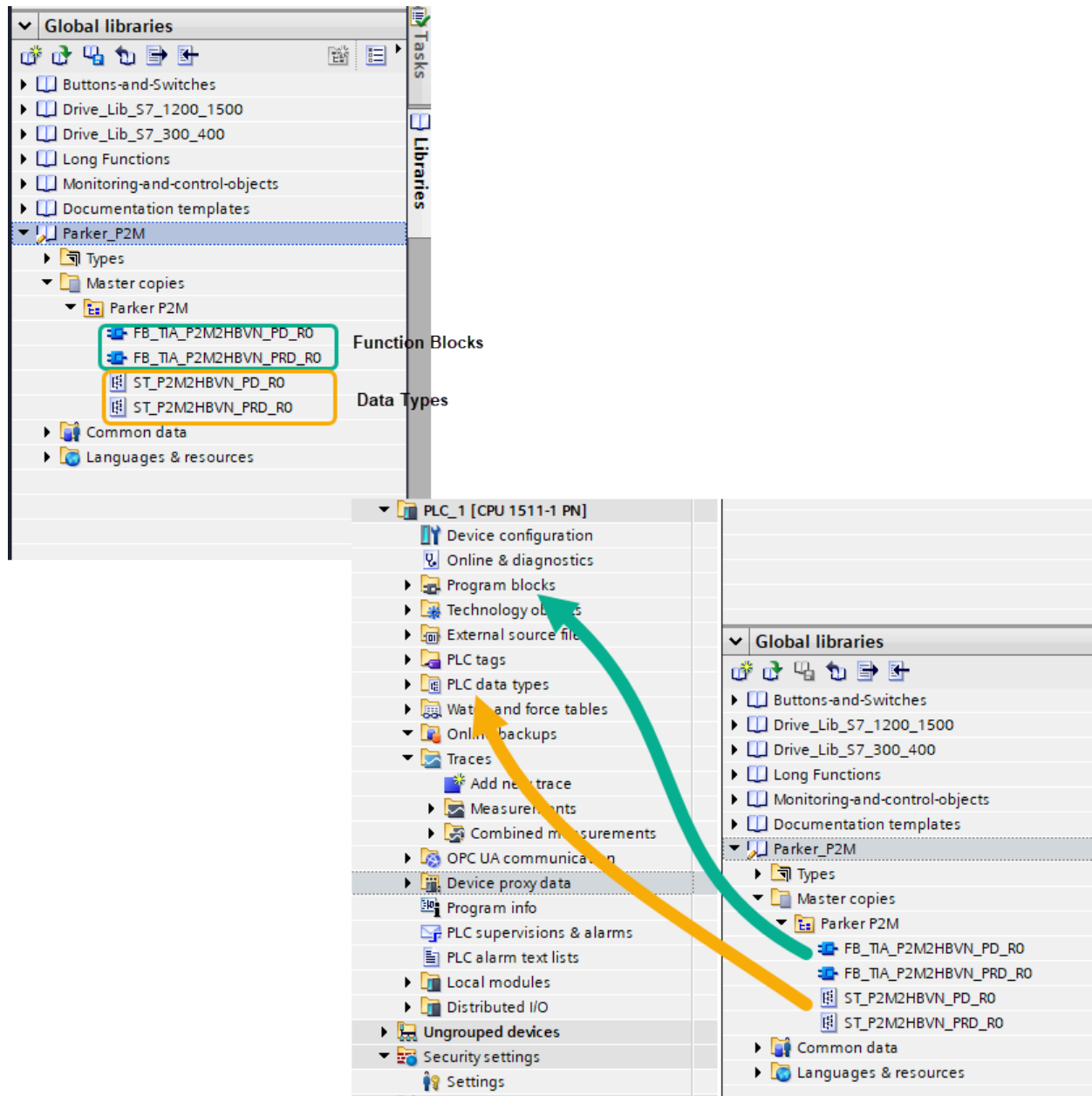
IMPORTING THE FUNCTION BLOCK LIBRARY

1. Select Program blocks in the project tree.
2. Go to the right side and click the libraries tab.
3. Select Open Global Library .
4. Go to the directory where the Parker_P2M is and select it.
5. Click OPEN.



P2M2HBVN TIA Portal S7 1200/1500 Function Blocks

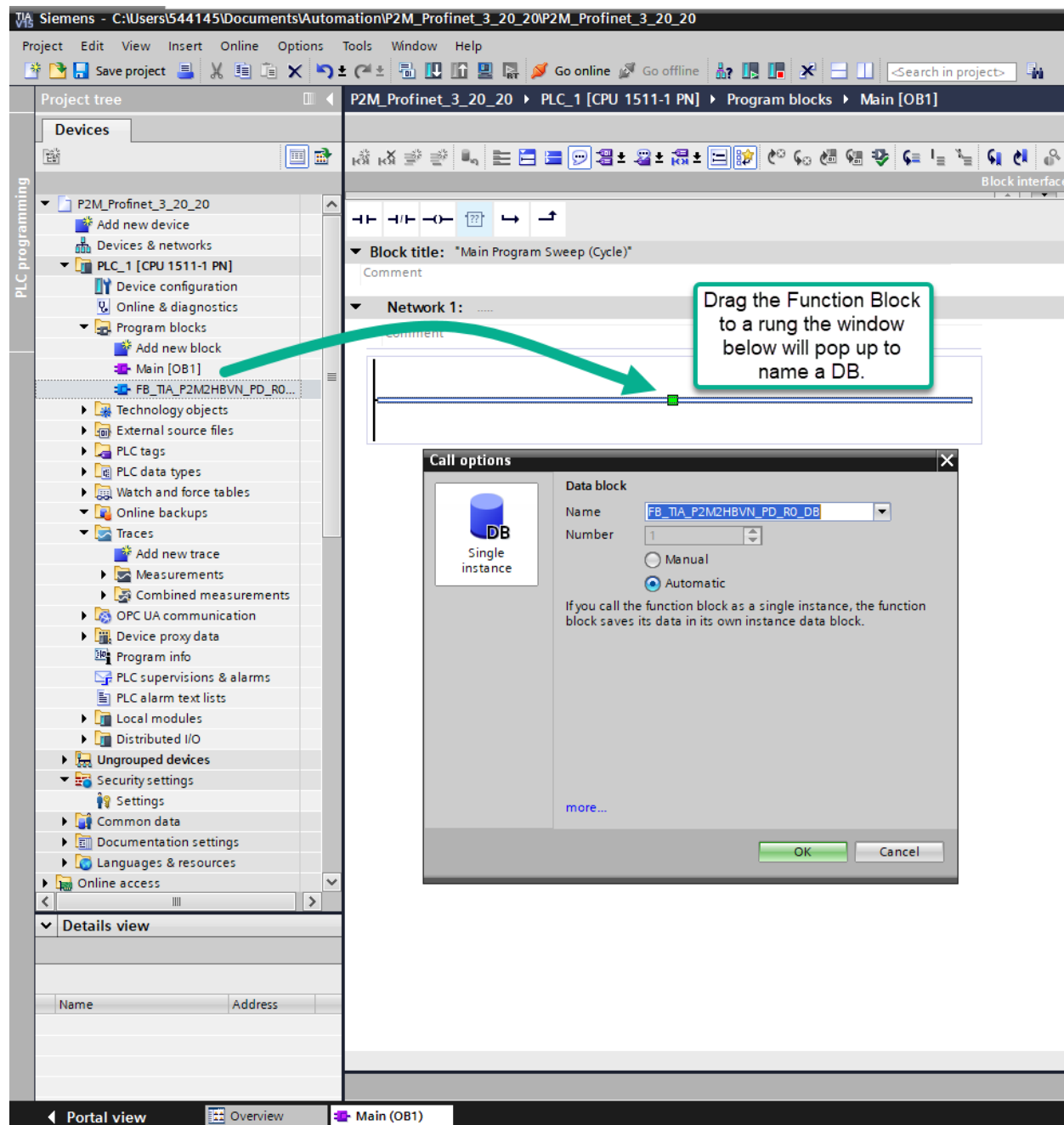
Once this is completed you can now open the library to get to the function block files by opening
->Master copies ->Parker P2M. Now drag the Function Blocks and Data Types into the Project tree to be used.



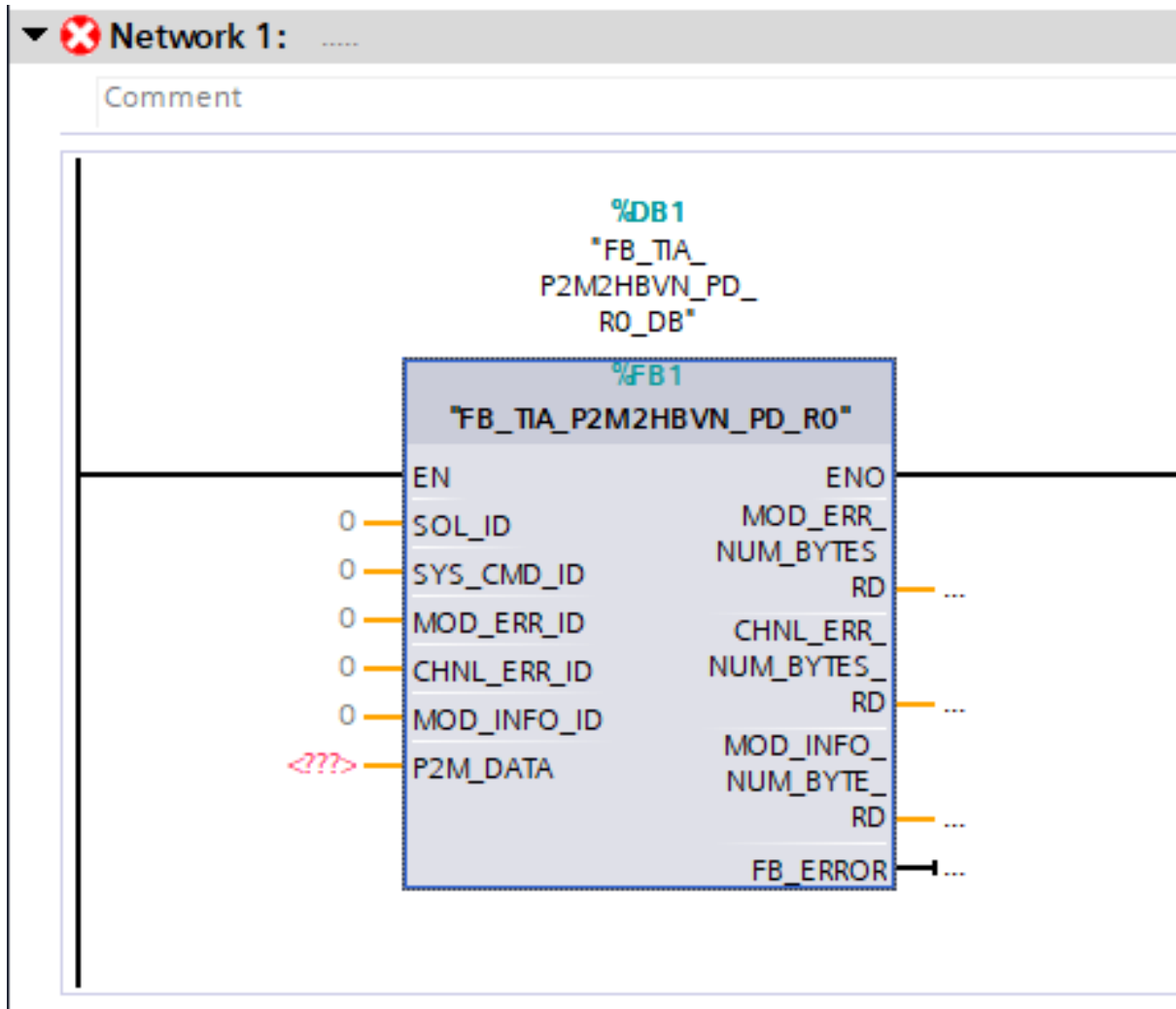
USING PROCESS DATA FUNCTION BLOCK

P2M2HBVN TIA Portal S7 1200/1500 Function Blocks

The “FB_TIA_P2M2HBVN_PD_Rx” FB simplifies the usage of P2M Profinet with Siemens 1200/1500 PLCs. data is mapped to user-friendly control and diagnostic structure within TIA Portal. Once the blocks and data types are in the Program blocks folder you need to create the DB to go with the function block.




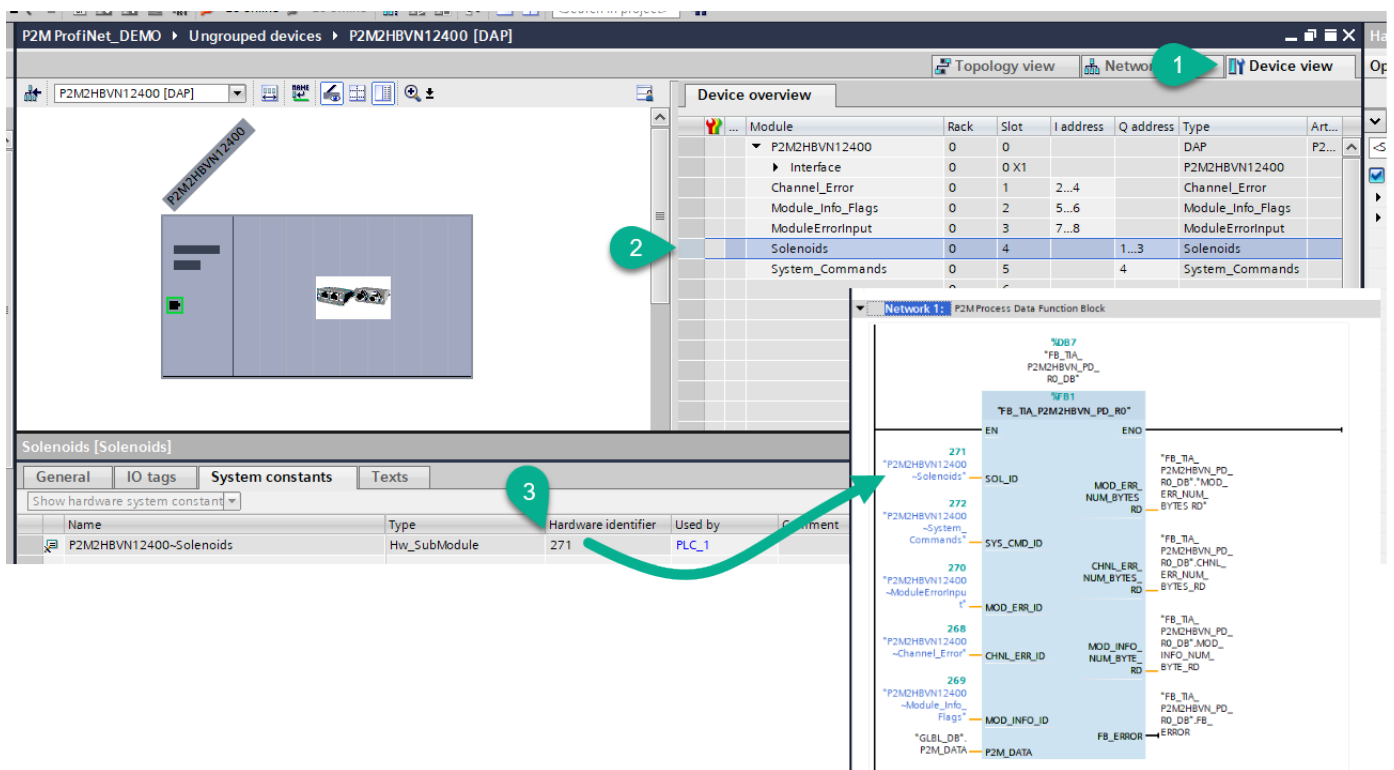
Once you name the Data Block and click OK the block and DB will be created see below. There needs to be one function block and data block per P2M module.



SETTING UP THE PROCESS DATA FUNCTION BLOCK

After adding the P2M to the program the block needs to be set up with the appropriate hardware ID's

1. Go to Device view tab click this button  to get to Device Overview tab.
2. Click on the module section you would like to get the Hardware ID number in this example it is for the Solenoids.
3. Go to the Properties tab -> System constants tab.
4. You must assign the Hardware ID for the rest of the slots
 - a. SOL_ID = Slot 4
 - b. SYS_CMD_ID = Slot 5
 - c. MOD_ERR_ID = Slot 3
 - d. CHNL_ERR_ID = Slot 1
 - e. MOD_INFO_ID = Slot 2



The screenshot displays the TIA Portal interface for configuring the P2M2HBVN12400 function block. The 'Device overview' table lists the modules and their hardware IDs. The 'System constants' tab shows the mapping of hardware IDs to function block inputs. The 'Network 1' diagram shows the function block with its inputs and outputs.

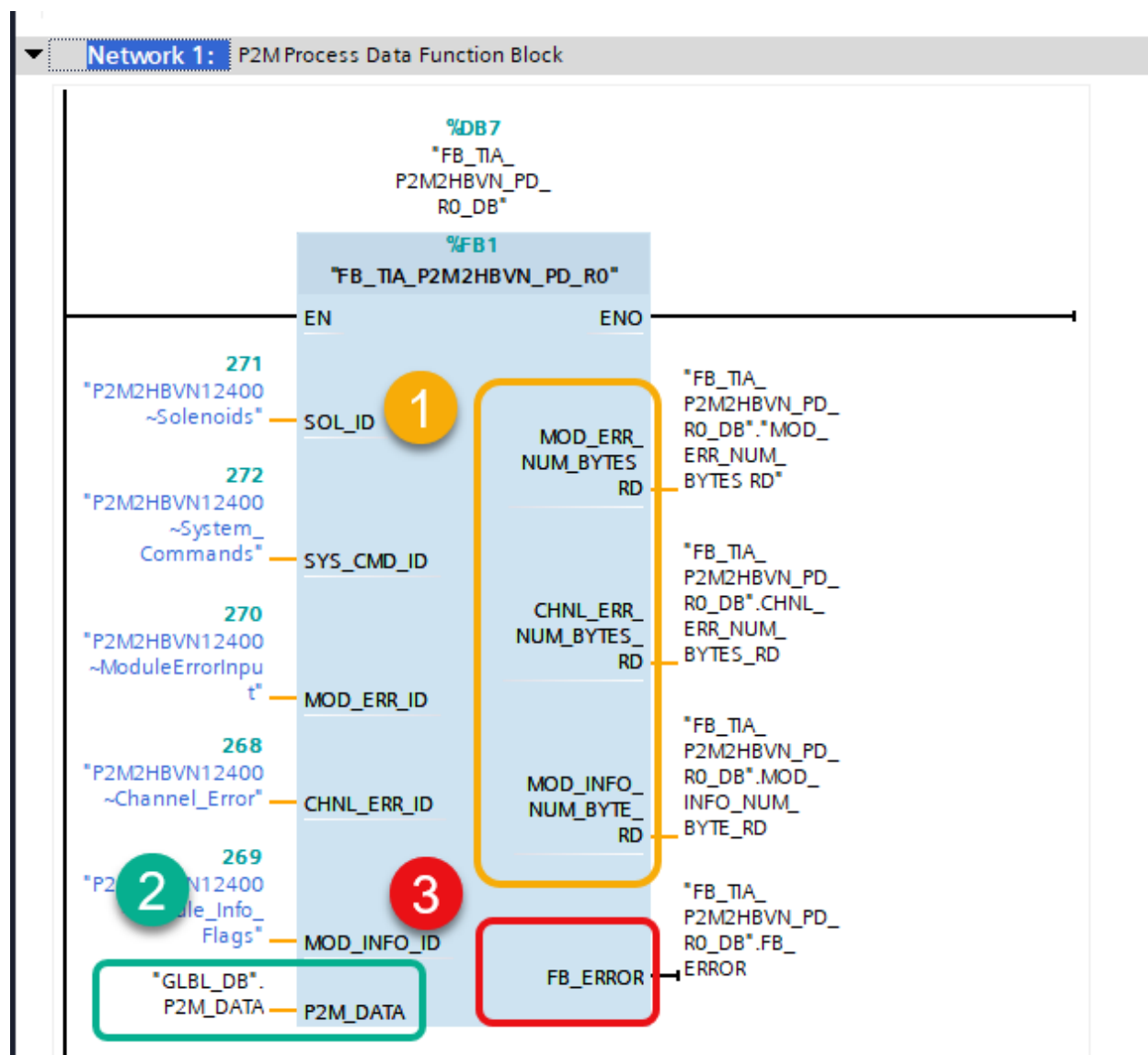
Module	Rack	Slot	I address	Q address	Type	Art...
P2M2HBVN12400	0	0			DAP	P2...
Interface	0	0 X1			P2M2HBVN12400	
Channel_Error	0	1	2...4		Channel_Error	
Module_Info_Flags	0	2	5...6		Module_Info_Flags	
ModuleErrorInput	0	3	7...8		ModuleErrorInput	
Solenoids	0	4		1...3	Solenoids	
System_Commands	0	5		4	System_Commands	

Name	Type	Hardware identifier	Used by	Comment
P2M2HBVN12400-Solenoids	Hw_SubModule	271	PLC_1	

Network 1: P2M Process Data Function Block

```

graph LR
    EN[EN] --> FB[P2M2HBVN12400]
    FB --> ENO[ENO]
    FB --> SOL_ID[SOL_ID]
    FB --> MOD_ERR_NUM_BYTES[MOD_ERR_NUM_BYTES]
    FB --> SYS_CMD_ID[SYS_CMD_ID]
    FB --> CHNL_ERR_NUM_BYTES[CHNL_ERR_NUM_BYTES]
    FB --> MOD_ERR_ID[MOD_ERR_ID]
    FB --> CHNL_ERR_ID[CHNL_ERR_ID]
    FB --> MOD_INFO_NUM_BYTES[MOD_INFO_NUM_BYTES]
    FB --> MOD_INFO_ID[MOD_INFO_ID]
    FB --> FB_ERROR[FB_ERROR]
    FB --> P2M_DATA[P2M_DATA]
  
```



DESCRIPTION OF THE OTHER FB FUNCTIONS

1. The bytes read just inform the user how many bytes of data was read from the P2M.
2. P2M_DATA is where the tag with the P2M "ST_P2M2HBVN_PD_R0" data type. In this case it is defined in a Global Data Block (The type of DB depends on the how your program is constructed).
3. FB_Error this is to let the user know that there is an error in the GETIO or SETIO call within the block. Please see the tables later in this document. If there is an error, the the P2M module may not function properly.

[illegible]

“FB_TIA_P2M2HBVN_PD_R0” ERROR CODES

GETIO ERROR CODES

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
8090	<ul style="list-style-type: none"> You have not configured a module for the specified hardware identifier or You have ignored the restriction concerning the length of consistent data, or you have not specified a hardware identifier as an address at parameter LADDR .
8092	A data type other than (Array of) bit string or integer was specified at the RECORD parameter.
8093	No DP module/PROFINET IO device from which you can read consistent data exists for the hardware identifier specified in LADDR . This error code also occurs if the module addressed by means of LADDR does not have inputs.
80A0	An access error was detected when accessing the I/O.
80B1	The length of the specified target range at parameter RECORD is shorter than the configured user data length.
80C0	The data have not been read yet.
General error information	See also: GET_ERR_ID: Get error ID locally
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

SETIO ERROR CODES

Parameter RET_VAL

Error code* (W#16#...)	Explanation
0000	No error occurred.
8090	<ul style="list-style-type: none"> You have not configured a module for the specified hardware identifier or You have ignored the restriction concerning the length of consistent data, or you have not specified a hardware identifier at parameter LADDR .
8092	A data type other than (Array of) bit string or integer was specified at the RECORD parameter.
8093	No DP module /PROFINET IO device to which you can write consistent data exists at the HW ID specified in LADDR . This error code also occurs if the DP standard slave / PROFINET IO device addressed via LADDR does not have outputs.
80A1	Access error detected while I/O devices were being accessed.
80B1	The length of the specified source range at the RECORD parameter is shorter than the outputs of the configured DP standard slave / PROFINET IO device.
80C1	The data of the previous write job have not yet been processed by the DP standard slave / PROFINET IO device.
General error information	See also: GET_ERR_ID: Get error ID locally
* The error codes in the program editor are displayed as integer or hexadecimal values. For information on switching the display formats, refer to "See also".	

****PLEASE NOTE** that the “FB_TIA_P2M2HBVN_PD_R0” function block uses GETIO (read process image) and SETIO (write process image). Do not use the IO tags that are created after importing the GSDML file. Because the parameter data function block read and writes the image tables in P2M using GETIO and SETIO. If you use both the function block and IO tags it will cause issues of overwriting each other. It is recommended to disable process image “automatic update” to “none”. If this is done it will only allow the FB to read and write the process image. The function block reads and writes the process image to and from the “ST_P2M2HBVN_PD_R0” data type that is created and assigned on the P2M_DATA pin.

1. Go to Device view tab select the row to change the process image update setting
2. Select automatic updates
3. Select the three dots to the right and select none. This should be done for each Input and Output slot that has %I and %Q assigned.

Device overview

Module	Rack	Slot	I address	Q address	Type	Art...
P2M2HBVN12400	0	0			DAP	P2...
Interface	0	0 X1			P2M2HBVN12400	
Channel_Error	0	1	2...4		Channel_Error	
Module_Info_Flags	0	2	5...6		Module_Info_Flags	
ModuleErrorInput	0	3	7...8		ModuleErrorInput	
Solenoids	0	4		1...3	Solenoids	
System_Commands	0	5		4	System_Commands	
	0	6				
	0	7				
	0	8				
	0	9				

I/O addresses

Output addresses

Start address: 1
End address: 3
Organization block: --- (Automatic update)
Process image: Automatic update

Solenoids [Solenoids]

I/O addresses

Output addresses

Start address: 1
End address: 3
Organization block: --- (None)
Process image: None

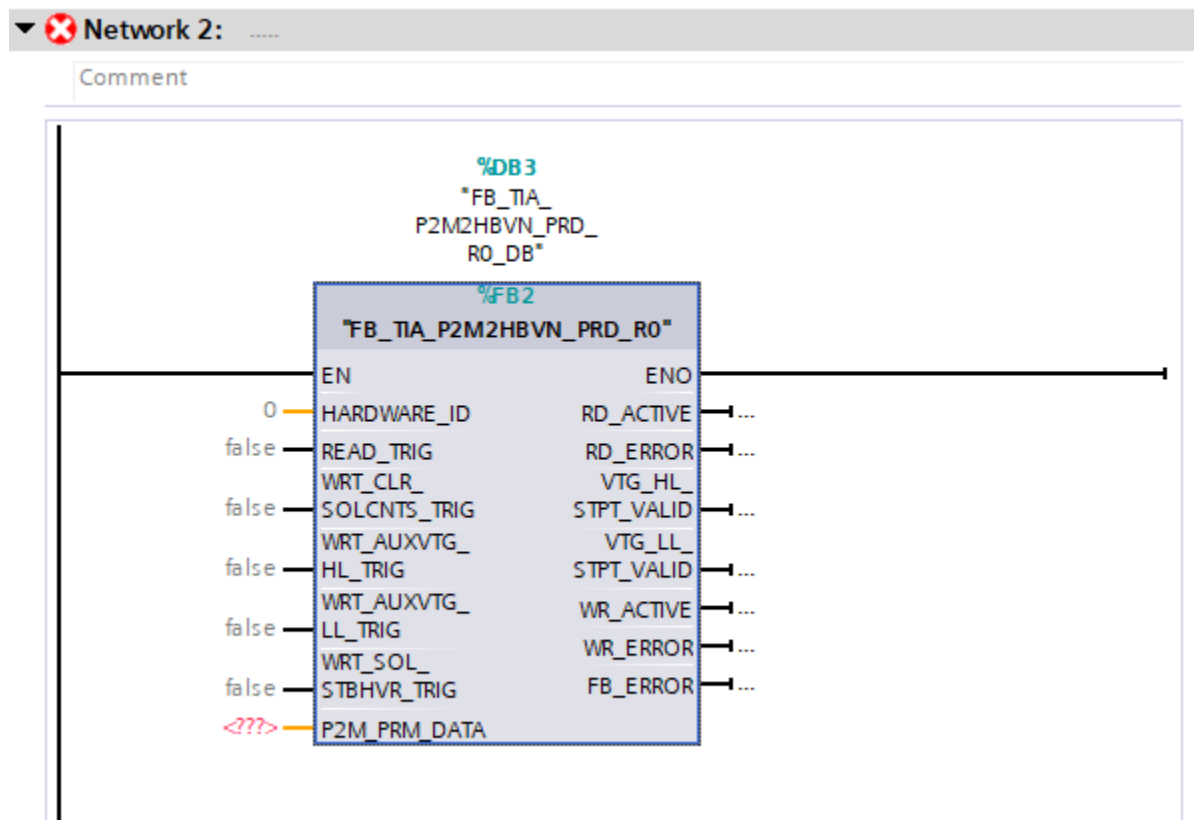
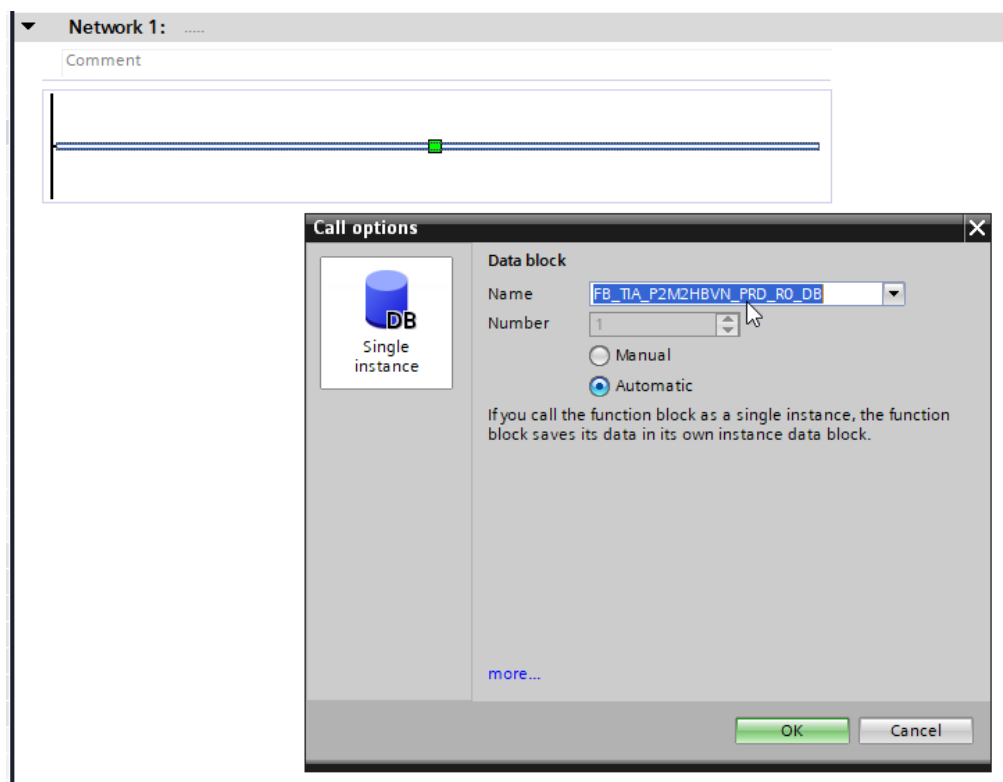
P2M “ST_P2M2HBVN_PD_R0” data structure contains all the process data for the P2M module. In an easy to follow format for turning on solenoids and monitoring any faults of the P2M module.

GLBL_DB										
	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Supervis...	Comment
1	▼ Static									
2	▼ P2M_DATA	*ST_P2M2HBVN_PD_R0*			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Data Structure for P2M Profinet P2M2HBVN1240
3	SYS_CMD_BYTE	Byte	16#0		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			System Command Byte
4	CHNL_ERR	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Channel Error for Solenoids
5	FB_ERR_STAT	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Function block error codes for GETIO and SETIO
6	MOD_INFO_FLAGS	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Module Info flags
7	SOLENOID	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Solenoid Outputs
8	MOD_ERR_FLAGS	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Error Flags for P2M
7	▼ SOLENOID	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Solenoid Outputs
8	EV1	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Solenoid 1
9	EV2	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Solenoid 2
10	EV3	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Solenoid 3
11	EV4	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Solenoid 4
32	▼ MOD_ERR_FLAGS	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Error Flags for P2M
33	ACK_Required	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Set if a major fault requiring acknowledgment i
34	AUX_VTG_WARN	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Set if the Auxiliary Voltage is outside Normal ra
35	AUX_VTG_ERR	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Set if the Auxiliary Voltage is outside Warning a
36	TEMP_WARN	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Set if a temperature warning (in one of the out)
37	OUTPUT_DRVR...	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Set if an over current / short-circuit error has oc
38	MODULE_ERR	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Depending on the source of the fault, this error
39	OUTPUT_STG...	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Set if no Auxiliary Power is available. No acknov
4	▼ CHNL_ERR	Struct			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Channel Error for Solenoids
5	EV1	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Channel Error EV1
6	EV2	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Channel Error EV2
7	EV3	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Channel Error EV3
8	EV4	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Channel Error EV4
9	EV5	Bool	false		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>			Channel Error EV5

USING P2M PARAMETER DATA FUNCTION BLOCK

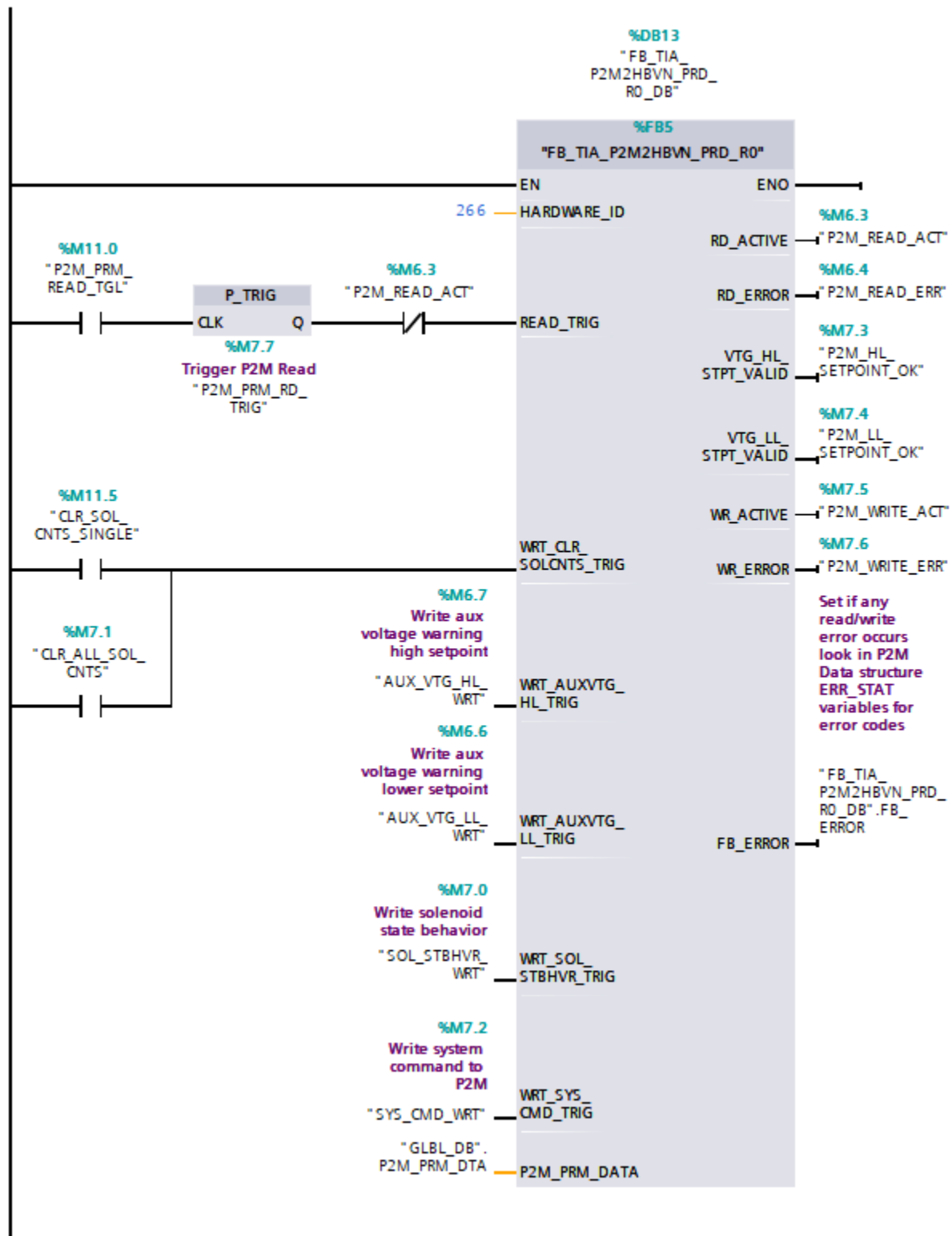
The “FB_TIA_P2M2HBVN_PRD_Rx” FB simplifies the usage of P2M Profinet with Siemens 1200/1500 PLCs. Data is mapped to user-friendly control and diagnostic structure within TIA Portal. Once the blocks and data types are in the Program blocks folder you need to create the DB to go with the function block.

P2M2HBVN TIA Portal S7 1200/1500 Function Blocks



Network 11: P2M Parameter Data Function Block

Demonstration of Parameter reading of the P2M Profinet Block.



P2M2HBVN TIA Portal S7 1200/1500 Function Blocks

The “FB_TIA_P2M2HBVN_PRD_R0” function block read and writes all parameter data the P2M is capable of.

Device overview

Module	Rack	Slot	I address	Q address	Type
P2M2HBVN12400	0	0			DAP
Interface	0	0 X1			P2M2HBVN12400
Channel_Error	0	1	2...4		Channel_Error
Module_Info_Flags	0	2	5...6		Module_Info_Flags
ModuleErrorInput	0	3	7...8		ModuleErrorInput
Solenoids	0	4		1...3	Solenoids
System_Commands	0	5		4	System_Commands
	0	6			
	0	7			

Interface [Interface]

Name	Type	Hardware identifier	Used by
P2M2HBVN12400~Interface~Port_1	Hw_Interface	267	PLC_1
P2M2HBVN12400~Interface~Port_2	Hw_Interface	261	PLC_1
P2M2HBVN12400~Interface	Hw_Interface	266	PLC_1

Network 1:

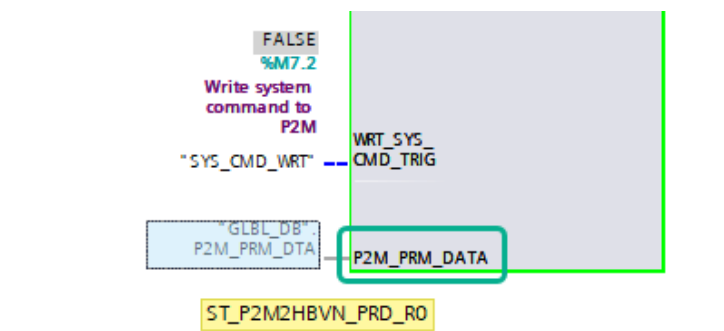
```

%DB1
"FB_TIA_P2M2HBVN_PRD_R0"
%FB5
"FB_TIA_P2M2HBVN_PRD_R0"
EM
0
HARDWARE_ID
RD_ACTIVE
RD_ERROR
VTG_HL
STPT_VALID
VTG_LL
STPT_VALID
WR_ACTIVE
WR_ERROR
FB_ERROR
RD_ACTIVE
RD_ERROR
VTG_HL
STPT_VALID
VTG_LL
STPT_VALID
WR_ACTIVE
WR_ERROR
FB_ERROR
false
READ_TRIG
WRT_CLR
SOLCNTS_TRIG
WRT_AUXVTG_
HL_TRIG
WRT_AUXVTG_
LL_TRIG
WRT_SOL_
STBHRV_TRIG
WRT_SYS_
CMD_TRIG
P2M_PRM_DATA
  
```

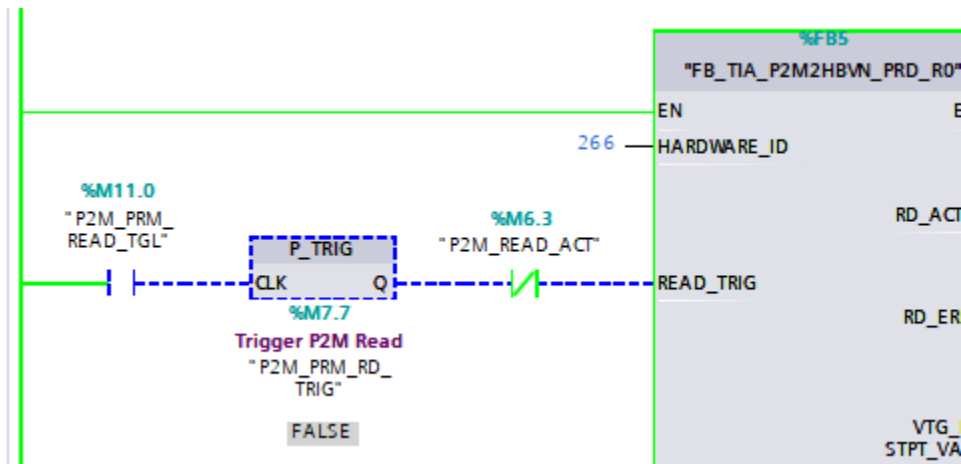
SETTING UP THE PARAMETER DATA FUNCTION BLOCK

After adding the P2M to the program the block needs to be set up with the appropriate hardware ID's

1. Go to Device view tab click this button to get to Device Overview tab. Click on interface.
2. Go to the Properties tab -> System constants tab.
3. Then assign the hardware id for P2M interface in this example it is “266” each P2M will have a different hardware id #.
4. P2M_PRM_DATA is where the tag with the P2M “ST_P2M2HBVN_PRD_R0” data type is assigned. In this case it is defined in a Global Data Block (The type of DB depends on the how your program is constructed).































































READING DATA USING PARAMETER FUNCTION BLOCK



The **READ_TRIG** pin will trigger the read instructions to start. The block will read all the data the P2M offers in parameter data all at once. Below is a list of all the data the function block reads.

1. All 24 Solenoid counts
2. AUX voltage
3. AUX Voltage High Level warning setpoint
4. AUX Voltage Low Level warning setpoint
5. Solenoid state behavior upon loss of communication

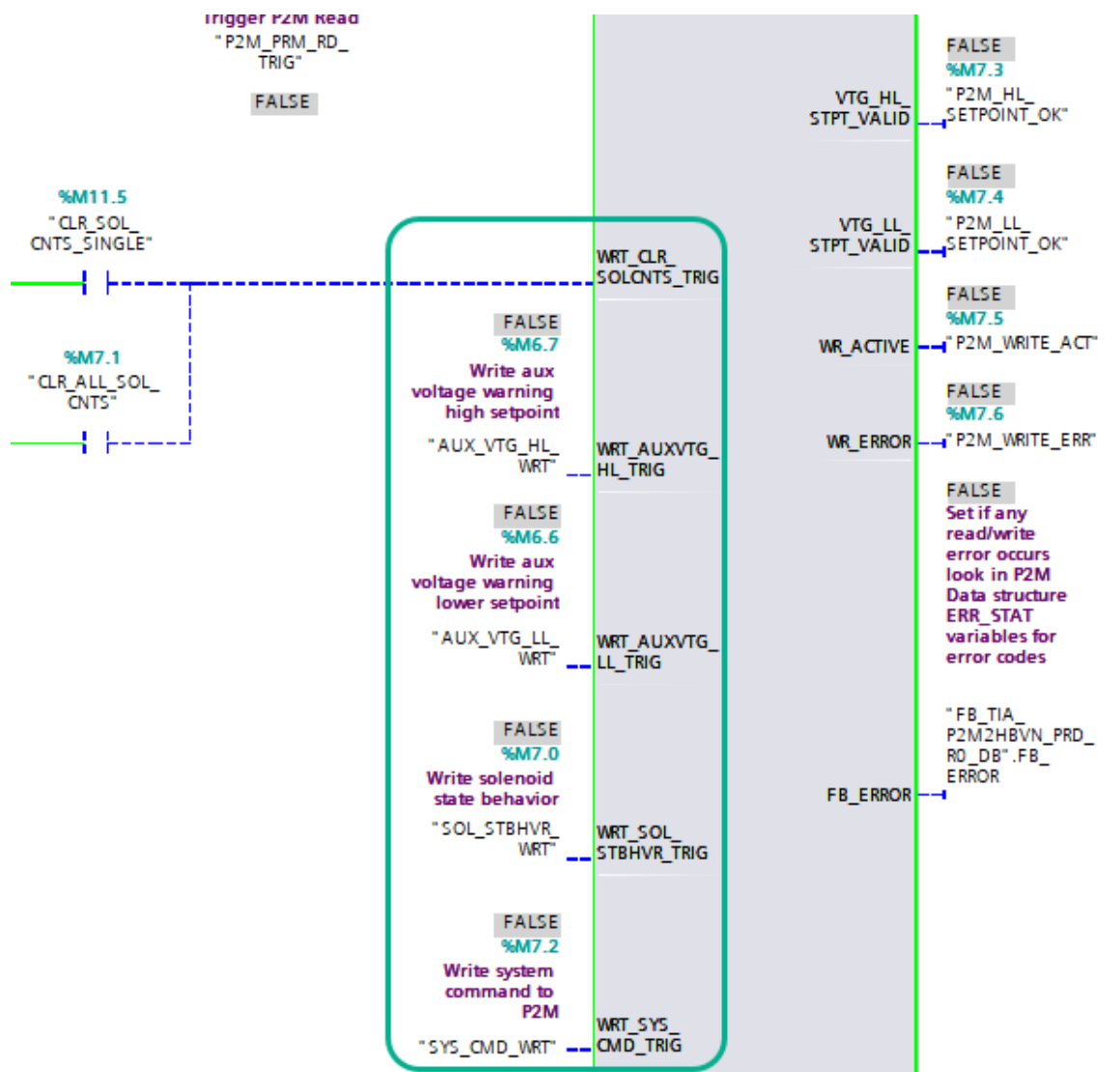
While the block is executing a read function, it cannot be triggered again. It is important to note that acyclic data (parameter data) is not meant to be read continuously at a high cycle rate it. All the data will get put into the data structure below as the read instructions process.

			P2M_PRM_DTA	*ST_P2M2HBVN_PR...		
			R_SOL_CNTS	Struct		
			EV1	DInt	0	151021
			EV2	DInt	0	151021
			EV3	DInt	0	151021
			EV4	DInt	0	151021
			EV5	DInt	0	0
0			EV6	DInt	0	0
1			EV7	DInt	0	0
2			EV8	DInt	0	0
3			EV9	DInt	0	0
4			EV10	DInt	0	0
5			EV11	DInt	0	0
6			EV12	DInt	0	0
7			EV13	DInt	0	0
8			EV14	DInt	0	0
9			EV15	DInt	0	0
0			EV16	DInt	0	0
1			EV17	DInt	0	0
2			EV18	DInt	0	0
3			EV19	DInt	0	0
4			EV20	DInt	0	0
5			EV21	DInt	0	0
6			EV22	DInt	0	0
7			EV23	DInt	0	0
8			EV24	DInt	0	0
9			R_AUX_VTG	UInt	0	24130
0			R_AUX_VTG_HL	UInt	0	26000
1			R_AUX_VTG_LL	UInt	0	23000
2			R_SOL_ST_BHVR	Byte	16#0	16#01

WRITING DATA USING PARAMETER FUNCTION BLOCK

Each parameter that can be written has to be triggered independently. The parameters that can be written the P2M:

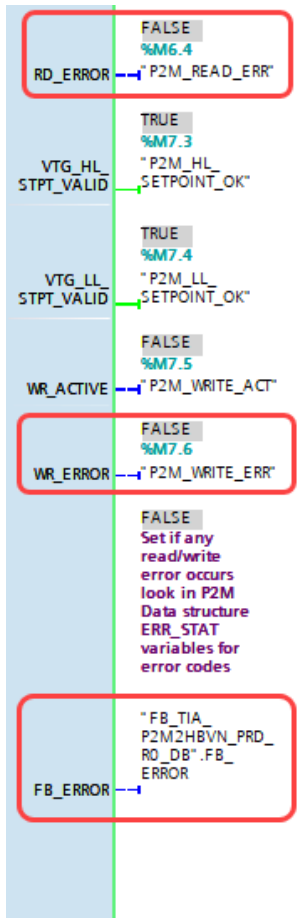
1. Clear Solenoid counts
2. Aux voltage High level set point range is 24 to 28.5 VDC
 - a. VTG_HL_STPT_VALID output validates that the range is correct on this setting
3. Aux voltage Low level set point range 18 to 24 VDC
 - a. VTG_LL_STPT_VALID output validates that the range is correct on this setting
4. Solenoid state behavior setting
5. System command



The function block reads can be triggered at any time the block will coordinate one write command at a time. Once the write command is executed it will take the data put into tag assigned on the P2M_PRM_DATA pin and write it to the P2M module. For example, if you set W_CLR_SOL_CNTS.EV1 to "TRUE" and trigger the block to write it will reset the count on EV1.

10		▼	W_CLR_SOL_CNTS	Struct		
11		■	EV1	Bool	false	FALSE
12		■	EV2	Bool	false	FALSE
13		■	EV3	Bool	false	FALSE
14		■	EV4	Bool	false	FALSE
15		■	EV5	Bool	false	FALSE
16		■	EV6	Bool	false	FALSE
17		■	EV7	Bool	false	FALSE
18		■	EV8	Bool	false	FALSE
19		■	EV9	Bool	false	FALSE
20		■	EV10	Bool	false	FALSE
21		■	EV11	Bool	false	FALSE
22		■	EV12	Bool	false	FALSE
23		■	EV13	Bool	false	FALSE
24		■	EV14	Bool	false	FALSE
25		■	EV15	Bool	false	FALSE
26		■	EV16	Bool	false	FALSE
27		■	EV17	Bool	false	FALSE
28		■	EV18	Bool	false	FALSE
29		■	EV19	Bool	false	FALSE
30		■	EV20	Bool	false	FALSE
31		■	EV21	Bool	false	FALSE
32		■	EV22	Bool	false	FALSE
33		■	EV23	Bool	false	FALSE
34		■	EV24	Bool	false	FALSE
35		■	W_SOL_ST_BHVR	Byte	16#0	16#01
36		■	W_AUX_VTG_HL	UInt	0	25000
37		■	W_AUX_VTG_LL	UInt	0	21000
38		▶	WR_FB_ERR	Struct		
39		■	FB_FAULT	Bool	false	FALSE

To detect read and write errors there are two outputs on the function block RD_ERROR and WR_ERROR. These outputs will turn true if there is a fault on RDREC or WRREC block that is used inside the function block. The FB_ERROR output is a rollup of these two fault conditions.



RD_FB_ERR	Struct		
R_SOLCNTS_FB...	DWord	16#0	16#0000_0000
R_AUXVTG_FB...	DWord	16#0	16#0000_0000
R_AUXVTG_HL...	DWord	16#0	16#0000_0000
R_AUXVTG_LL...	DWord	16#0	16#0000_0000
R_SOL_STBHV...	DWord	16#0	16#0000_0000
WR_FB_ERR	Struct		
W_CLR_SOLCN...	DWord	16#0	16#0000_0000
W_SOL_STBHV...	DWord	16#0	16#0000_0000
W_AUXVTG_HL...	DWord	16#0	16#0000_0000
W_AUXVTG_LL...	DWord	16#0	16#0000_0000
FB_FAULT	Bool	false	FALSE

The below tables are the potential error codes that can be generated by the RDREC and WRREC blocks that are used inside the function block.

Parameter STATUS

Description

The STATUS output parameter contains error information. If it is interpreted as ARRAY[1...4] of BYTE, the error information has the following structure:

Field element	Name	Meaning
STATUS[1]	Function_Num	<ul style="list-style-type: none"> B#16#00, if no error Function ID from DPV1-PDU: In the event of an error, B#16#80 is output (in the event of an error reading a data record B#16#DE and writing a data record B#16#DF). If no DPV1 protocol element is used, then B#16#C0 will be output.
STATUS[2]	Error_Decode	Location of the error ID
STATUS[3]	Error_Code_1	Error ID
STATUS[4]	Error_Code_2	Manufacturer-specific extension of the error ID

Field element STATUS[2]

STATUS[2] can have the following values:

Error_Decode (B#16#...)	Source	Meaning
00 to 7F	CPU	No error or no warning
80	DPV1	Error according to IEC 61158-6
81 to 8F	CPU	B#16#8x shows an error in the xth call parameter of the instruction.
FE, FF	DP profile	Profile-specific error

Parameter STATUS

Field element STATUS[3]

STATUS[3] can have the following values:

Error_Decode (B#16#...)	Error_Code_1 (B#16#...)	Explanation according to DVP1	Meaning
00	00		No error, no warning
70	00	reserved, reject	Initial call; no active data record transfer
	01	reserved, reject	Initial call; data record transfer has started
	02	reserved, reject	Intermediate call; data record transfer already active
80	81		The system data type at parameter TINFO does not match the call environment of the instruction. The used system data type must match the organization block in the user program (example: for a time-delay interrupt OB you need the system data type TI_Delay).
	90	reserved, pass	Invalid logical start address
	92	reserved, pass	Illegal type for VARIANT pointer
	93	reserved, pass	The DP component addressed via ID or F_ID is not configured.
	96		The "RALRM" does not supply the OB start information, management information, header information, or additional interrupt information. For OBs 4x, 55, 56, 57, 82 and 83, you use the "DPNRM_DG" instruction to read the current diagnostic frame of the relevant DP slave asynchronously (address information from OB start information).
	A0	read error	Negative acknowledgment when reading the module
	A1	write error	Negative acknowledgment when writing to the module
	A2	module failure	DP protocol error at layer 2 (e.g., slave failure or bus problems)
	A3	reserved, pass	General communication error or IO device / DP slave is unreachable
	A4	reserved, pass	Communication on PBUS+ disrupted
	A5	reserved, pass	-
	A7	reserved, pass	DP slave or module is occupied (temporary error)
	A8	version conflict	DP slave or module reports non-compatible versions
	A9	feature not supported	Function is not supported by DP slave or module
	AA to AF	user specific	DP slave or module reports a manufacturer-specific error in its application. Please check the documentation from the manufacturer of the DP slave or module.
	B0	invalid index	Data record not known in module Illegal data record number ≥ 256
	B1	write length error	Error at length specification: <ul style="list-style-type: none"> With "RALRM": length error in AINFO With "RDREC": length error in MLEN With "WRREC": length error in LEN
	B2	invalid slot	<ul style="list-style-type: none"> The configured slot is not occupied. For PROFINET IO and PROFIBUS DP: IO device / DP slave is unreachable
	B3	type conflict	Actual module type does not match specified module type
	B4	invalid area	DP slave or module reports access to an invalid area
	B5	state conflict	DP slave or module not ready
	B6	access denied	DP slave or module denies access
	B7	invalid range	DP slave or module reports an invalid range for a parameter or value
	B8	invalid parameter	DP slave or module reports an invalid parameter
	B9	invalid type	DP slave or module reports an invalid type With "RDREC": buffer too small (subsets cannot be read) With "WRREC": buffer too small (subsets cannot be written)
	BA to BF	user specific	DP slave or module reports a manufacturer-specific error when accessing. Please check the documentation from the manufacturer of the DP slave or module.

P2M2HBVN TIA Portal S7 1200/1500 Function Blocks

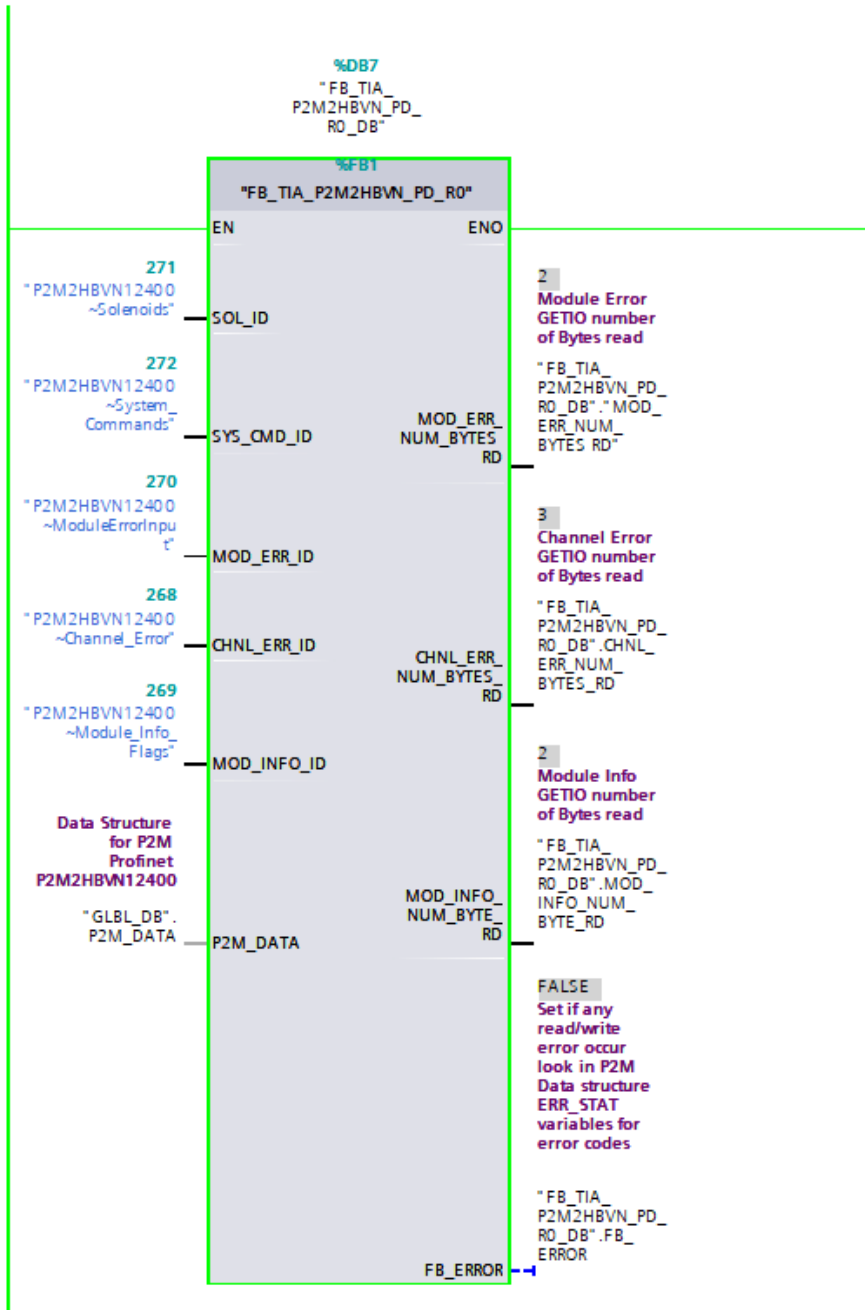
	C0	read constrain conflict	With "WRREC" : The data is only written when the CPU is in STOP mode. Note: this means that writing by the user program is not possible. You only write the data online with PG/PC. With "RDREC" : the module routes the data record, but either no data is present or the data can only be read when the CPU is in STOP mode. Note: if data can only be read when the CPU is in STOP mode, then an evaluation by the user program is not possible. Therefore, it is only possible to read the data online with PG/PC.
	C1	write constrain conflict	The data of the previous write job on the module for the same data record has not yet been processed by the module.
	C2	resource busy	The module is currently processing the maximum possible number of jobs for a CPU.
	C3	resource unavailable	The required operating resources are currently occupied.
	C4		Internal temporary error. Job could not be executed. Repeat the job. If this error occurs often, check your installation for sources of electrical interference.
	C5		DP slave or module not available.
	C6		Data record transfer was canceled due to priority class cancellation
	C7		Job aborted due to warm or cold restart on the DP master
	C8 to CF		DP slave or module reports a manufacturer-specific resource error. Please check the documentation from the manufacturer of the DP slave or module.
	Dx	user specific	DP slave specific. Refer to the description of the DP slave.
	Ex	user specific	Manufacturer-specific. Please check the documentation of the module.
81	00 to FF		Error in the initial call parameter (with "RALRM" : MODE)
	00		Illegal operating mode
82	00 to FF		Error in the second call parameter
:	:		:
88	00 to FF		Error in the eighth call parameter (with "RALRM" : TINFO)
	01		Wrong syntax ID
	23		Configuration limits exceeded or destination area too small
	24		Wrong range ID
	32		DB/DI no. out of user range
	3A		DB/DI no. is NULL for area ID DB/DI or specified DB/DI does not exist
89	00 to FF		Error in the ninth call parameter (with "RALRM" : AINFO)
	01		Wrong syntax ID
	23		Configuration limits exceeded or destination area too small
	24		Wrong range ID
	32		DB/DI no. out of user range
	3A		DB/DI no. is NULL for area ID DB/DI or specified DB/DI does not exist
8A	00 to FF		Error in the 10th call parameter
:	:		:
8F	00 to FF		Error in the 15th call parameter
FE, FF	00 to FF		Profile-specific error

PROGRAM EXAMPLES OF USING THE FUNCTION BLOCKS

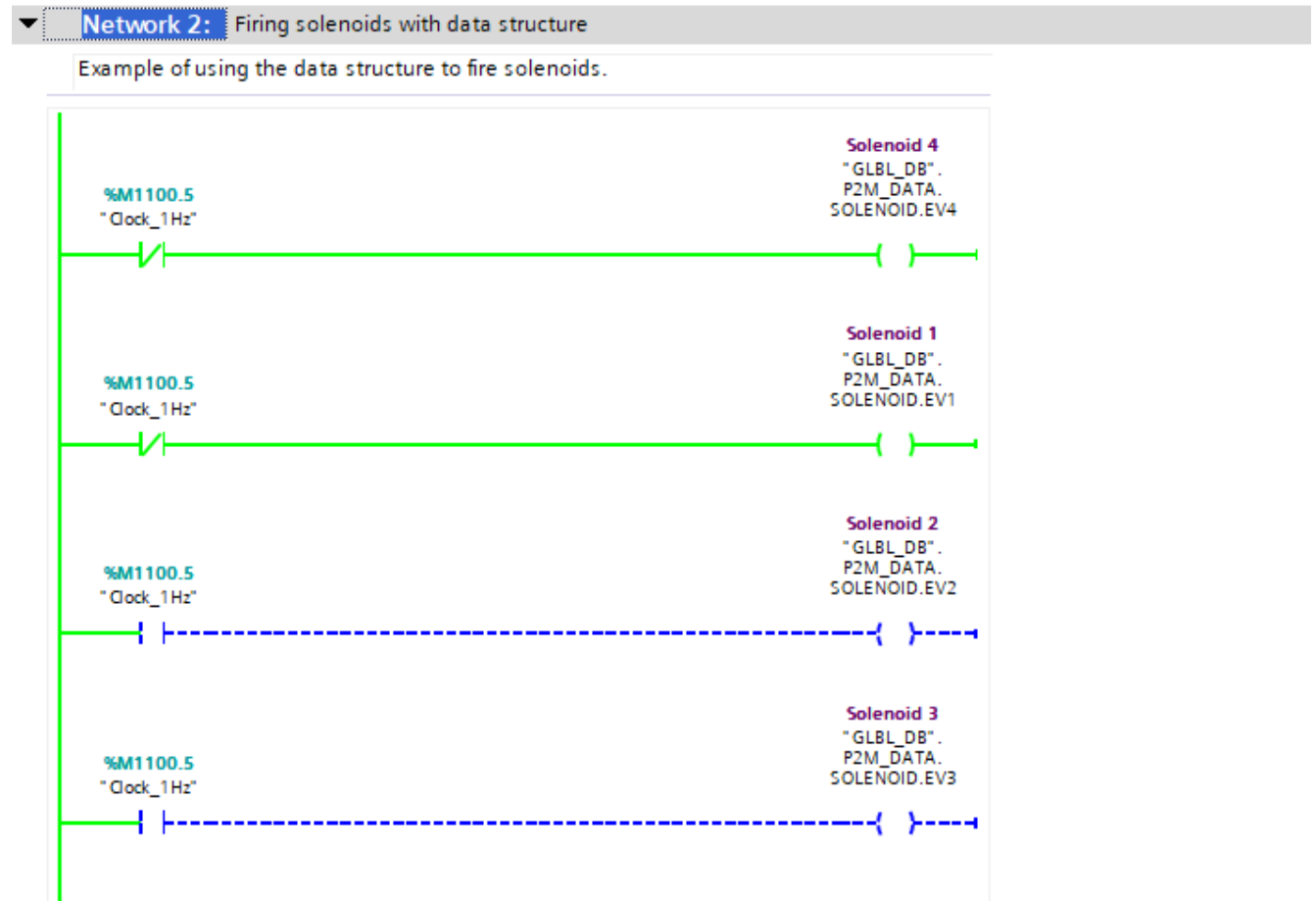
EXAMPLE OF PROCESS DATA FUNCTION BLOCK

Network 1: P2MProcess Data Function Block

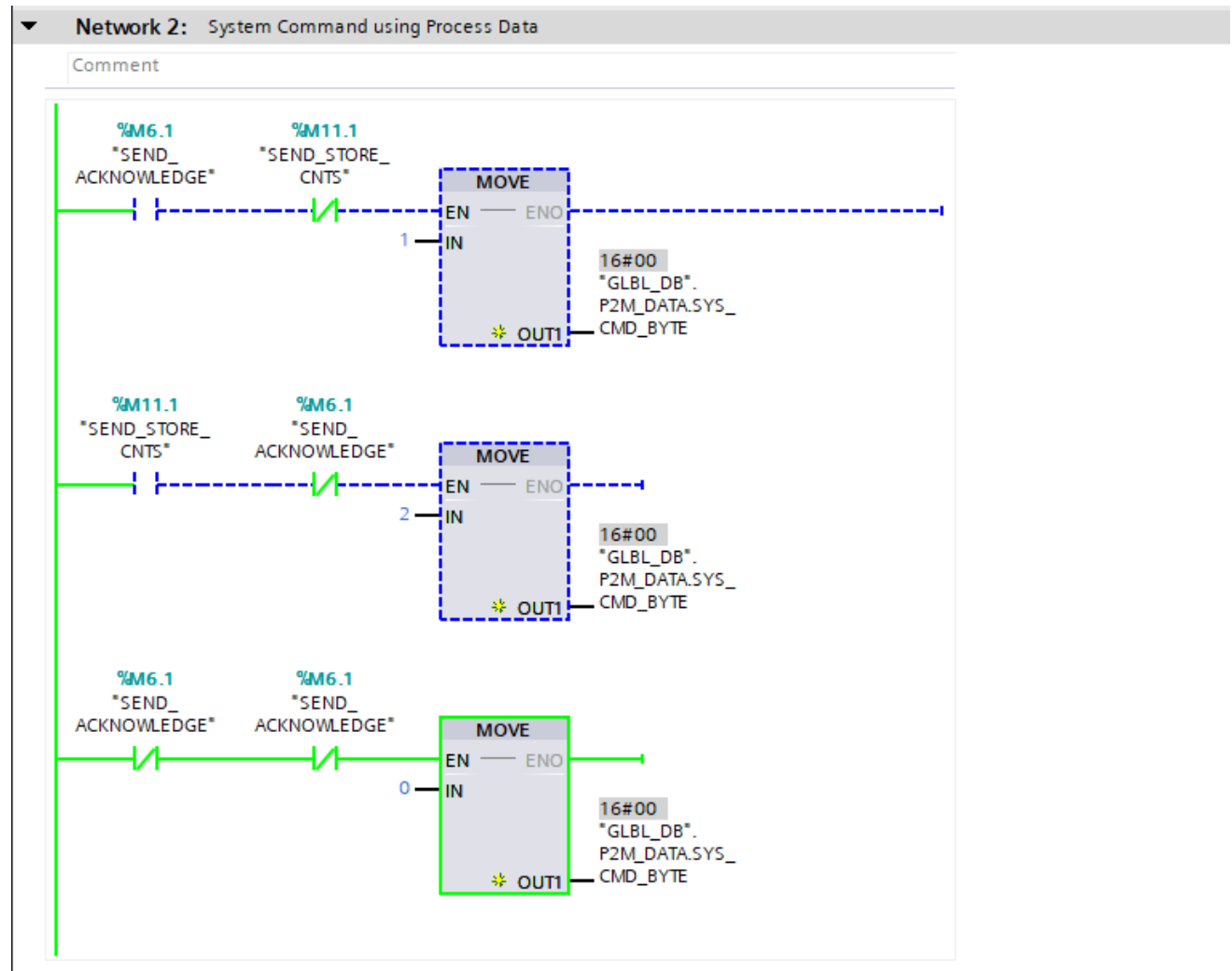
- ▼ This Process data block it uses GETIO and SETIO to read the P2M process image tables. This block is called every scan cycle. It reads and writes the Inputs and Outputs from the P2M_DATA pin on the block. **IF YOU USE THIS BLOCK YOU CANNOT ALSO USE THE ADDRESSES %I or %Q etc.. GENERATED FROM THE GSDML FILE.



EXAMPLE OF FIRING SOLENOIDS



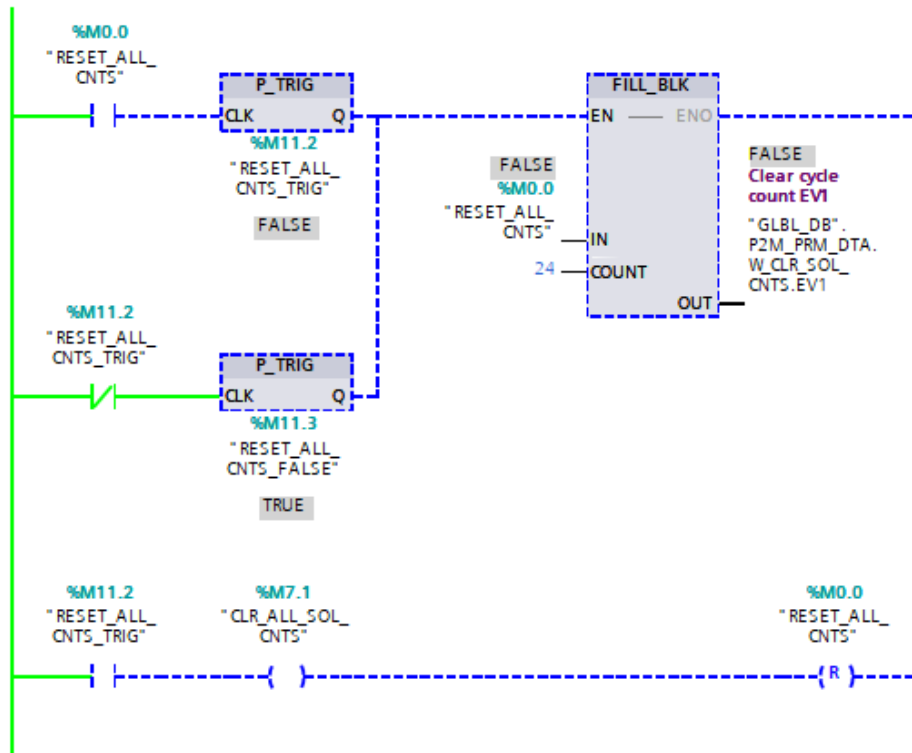
EXAMPLE OF SYSTEM COMMAND FUNCTION



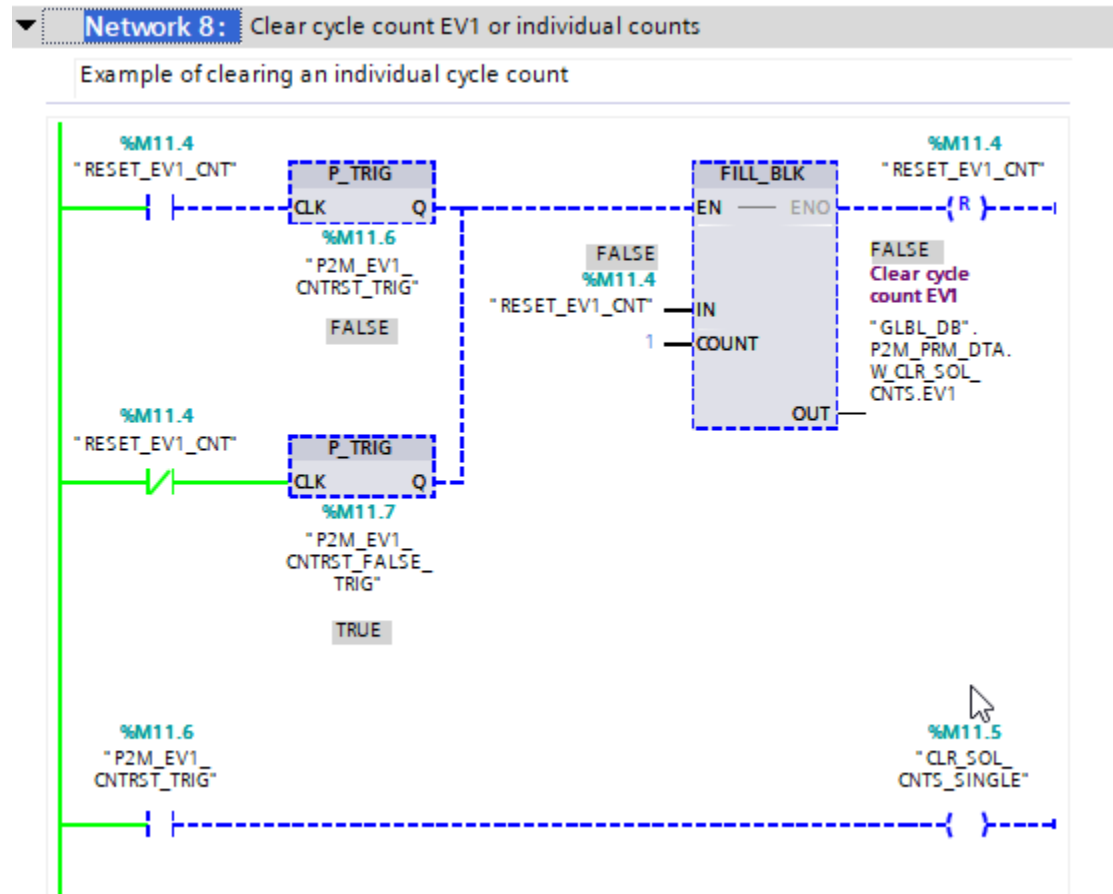
EXAMPLE OF CLEARING ALL COUNTS

▼ **Network 7:** Clear all counts

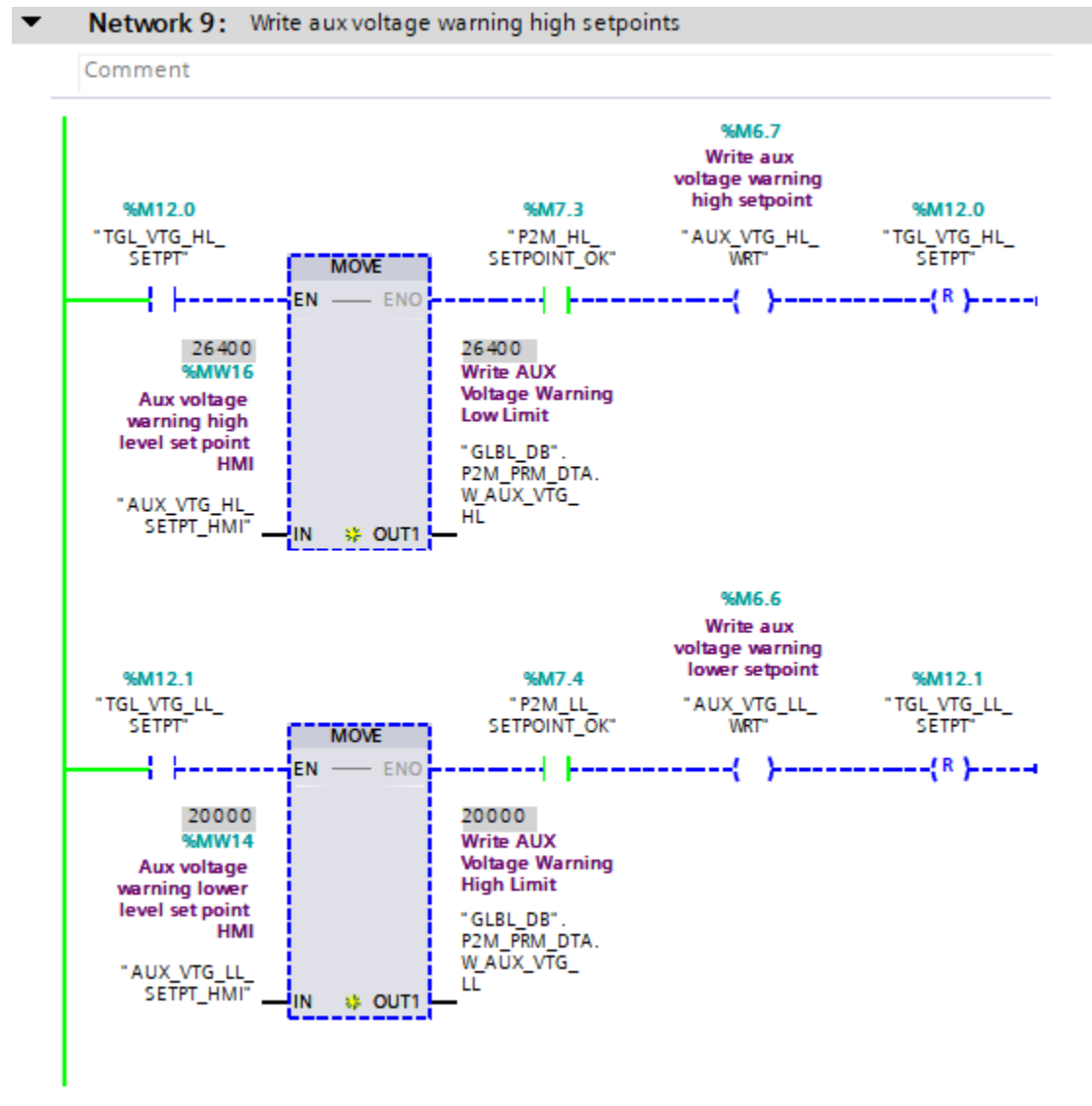
- ▼ Demonstration of resetting all counts of P2M at once. This example uses the fill block move to set them all to "true" then sets them all back to the default state of "false".



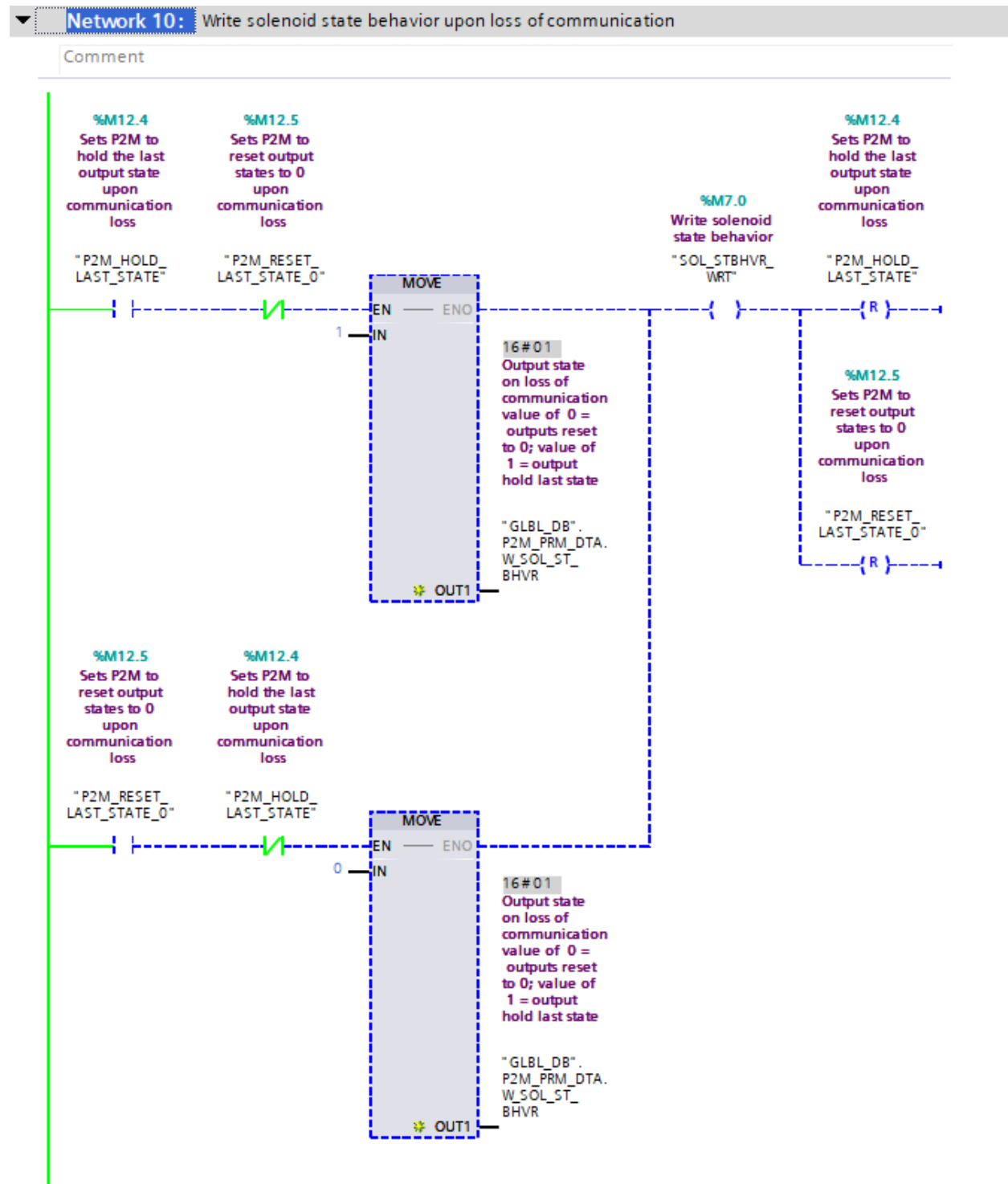
EXAMPLE OF CLEARING INDIVIDUAL CYCLE COUNT



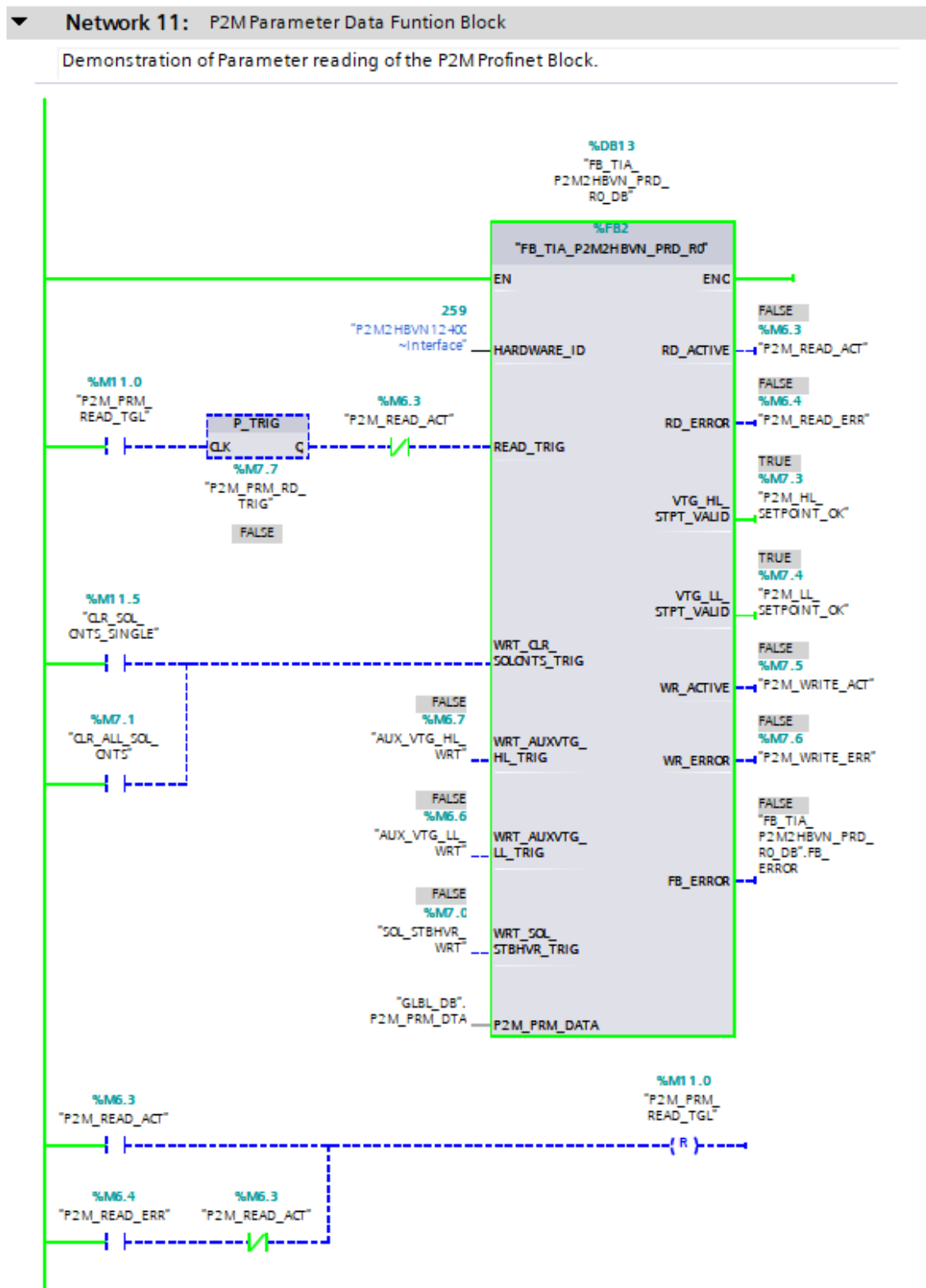
EXAMPLE OF WRITING AUX VOLTAGE LOW AND HIGH SETPOINT



EXAMPLE OF WRITING SOLENOID STATE BEHAVIOR



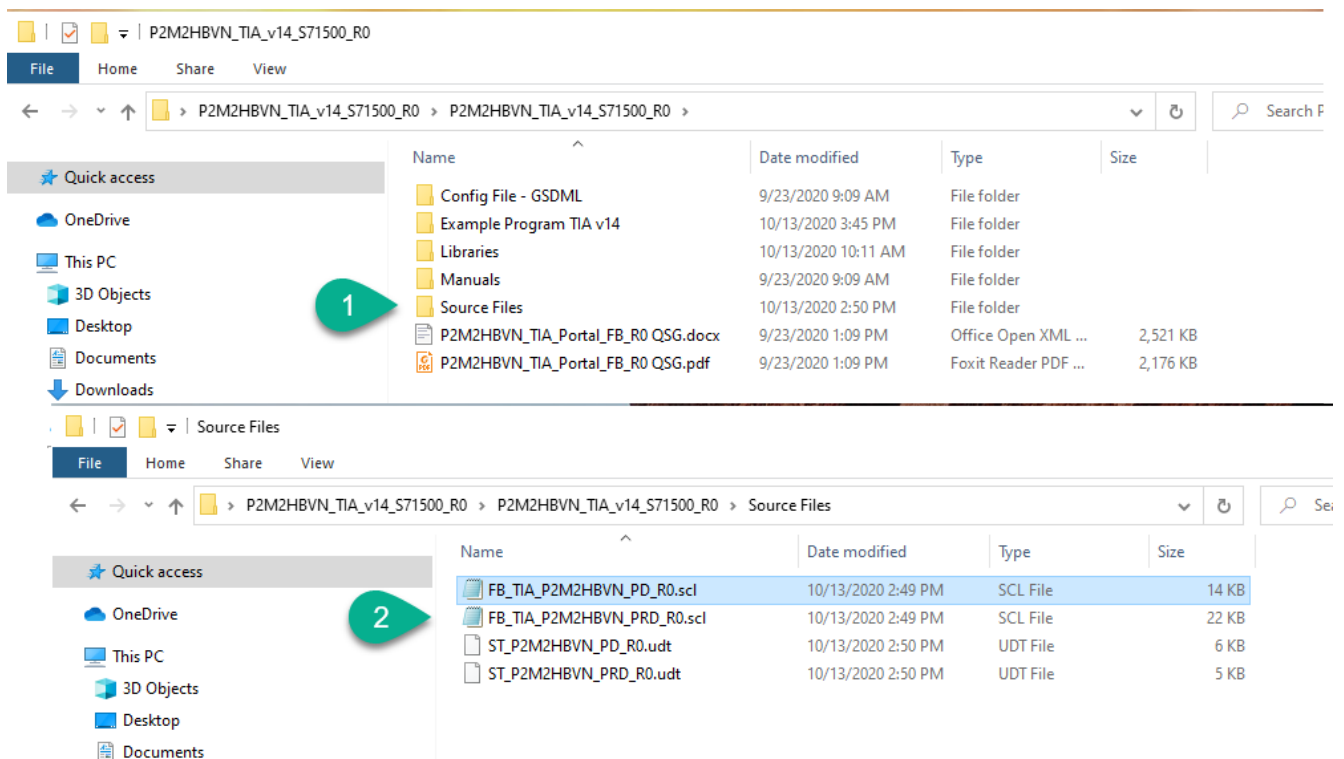
EXAMPLE OF USING THE PARAMETER FUNCTION BLOCK



HOW TO IMPORT BLOCKS USING THE SOURCE FILES

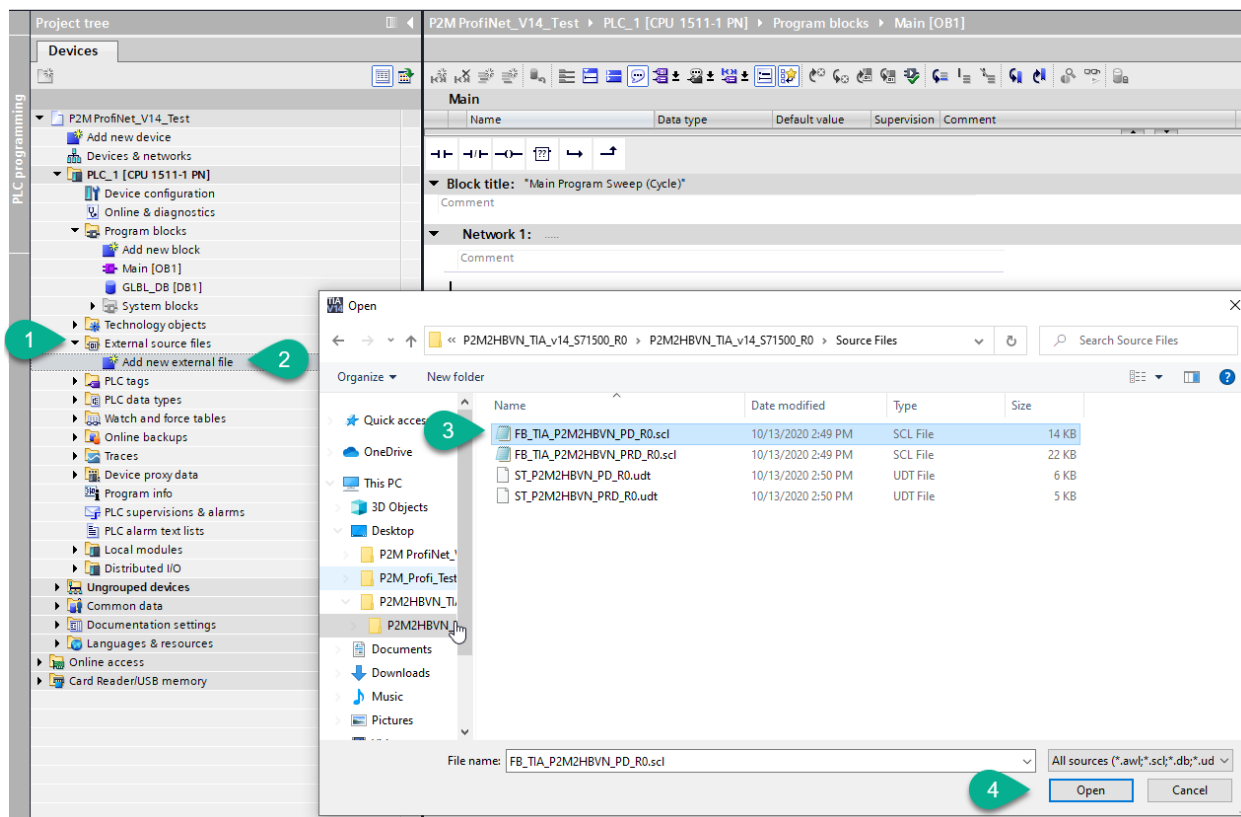
In TIA Portal you can also import blocks from source files instead of using a library. Open the “P2M2HBVN_TIA_v14_S71500_R0” folder.

1. Navigate to the folder “Source Files” open folder.
2. Inside you see source files for the function blocks and user defined datatypes.

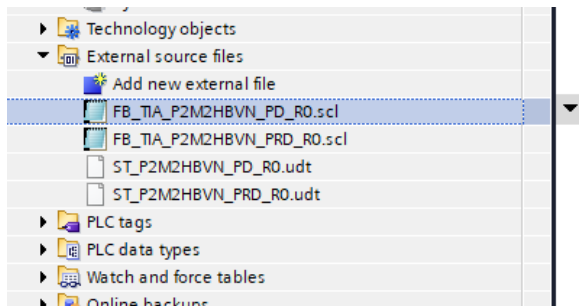


ADDING EXTERNAL SOURCE FILES IN TIA

1. Open a TIA Project go to the external source files folder in the project tree.
2. You can either right click on the folder and select “Add new external file” or double click “Add new external file”
3. An open folder dialog box will open you can select all of the files or just the ones you want.
 - a. ** Note you will need two files for each block to work “.scl and .udt” both files that have “PD” (Parameter Data) work together and both files that have “PRD” (Process Data) work together.
4. When you select your files you click open.



Once you have imported the files it will look like the below image.



P2M2HBVN TIA Portal S7 1200/1500 Function Blocks

5. You can highlight all the files or just one.
6. Right click on the highlighted files and select generate blocks from source. You will see a warning just click ok and the generation of the block will begin.
7. When the import is done you will see the function blocks and user defined data types were created.

