



Function Blocks for PSD-4__ and PSC-4__ with PROFINET interface

halstrup-walcher GmbH

Stegener Straße 10
D-79199 Kirchzarten

Phone: +49 (0) 76 61/39 63-0
Fax: +49 (0) 76 61/39 63-99

E-Mail: info@halstrup-walcher.de
Internet: www.halstrup-walcher.de

Table of Contents

1	Safety precautions	4
1.1	Appropriate use	4
1.2	Symbols	4
2	Data Structure DRIVE_DATA	4
3	Data Component “Antriebsdaten”	6
4	Error Description (Error ID)	7
5	Description and use of the function blocks	8
5.1	Opening the library.....	8
5.2	Copying of function blocks into the user project	9
5.3	Generating instance data blocks	10
5.4	Generating global data.....	10
5.5	Commonalities of all function blocks	10
5.6	Lockings between the function blocks	11
5.7	Example.....	11
5.8	MC_Communication_S7-1200_1500 and _S7-300 (FC101)	13
5.9	MC_Move (FB110).....	15
5.10	MC_Error (FB111)	19
5.11	MC_Error_ID (FC100).....	22
5.12	MC_ReadParameter (FB112)	25
5.13	MC_WriteParameter (FB113).....	29
5.14	MC_Parametrization (FB114).....	32
5.15	MC_PosParametrization (FB115)	42

Purpose of instruction manual

This instruction manual describes the function blocks for the PSD-4__-PN and PSC-4__-PN (with PROFINET interface).

All images and explanations in this manual refer to a PSD-4__-PN device example project, they are also valid for PSC-4__-PN devices.

Improper use of these devices or failure to follow these instructions may cause injury or equipment damage. Every person who uses the devices must therefore read the manual and understand the possible risks. The instruction manual, and in particular the safety precautions contained therein, must be followed carefully. **Contact the manufacturer if you do not understand any part of this instruction manual.**

The manufacturer reserves the right to continue developing these function blocks without documenting such development in each individual case. The manufacturer will be happy to determine whether this manual is up-to-date.

© 2021

The manufacturer owns the copyright to this instruction manual. It must not be copied either wholly or in part or made available to third parties.

1 Safety precautions

1.1 Appropriate use

The positioning systems PSD-4__-PN and PSC-4__-PN are especially suitable for automatically setting tools, stops or spindles for wood-processing equipment, packing lines, printing equipment, filling units and other types of special machines.

PSD-4__-PN and PSC-4__-PN positioning systems are not stand-alone devices and may only be used if coupled to another machine.

1.2 Symbols

The symbols given below are used throughout this manual to indicate instances when improper operation could result in the following hazards:



WARNING!

This warns you of a potential hazard that could lead to bodily injury up to and including death if the corresponding instructions are not followed.



CAUTION!

This warns you of a potential hazard that could lead to significant property damage if corresponding instructions are not followed.



INFORMATION!

This indicates that the corresponding information is important for operating the function blocks properly.

2 Data Structure DRIVE_DATA

For each drive there's a data structure, in which some data of the drive is deposited. For each drive a global instance of this structure is required. This instance must be provided to each FB that operates on the corresponding drive. Hereby it shall be prevented for example, that two FBs accesses the parameter interface of a single drive. Furthermore, in this data structure the addresses for the input and output data of the corresponding drive has to be deposited.

Parameter name	Data type	Written by	Description
PdAddressIn	INT (S7-1200 + S7-1500) DWORD (S7-300)	User	Address process data Device → Controller
PdAddressOut	INT (S7-1200 + S7-1500) DWORD (S7-300)	User	Address process data Controller → Device
AxisName	STRING[16]	User (optional)	Name of the axis
AxisDescription	STRING[32]	User (optional)	Description (e.g. function, task of this axis)
State	DINT	Function blocks	Actual state
Communication_error	BOOL	Function blocks	Communication fault to IO Device
Private	STRUCT	Function blocks	Data structure for internal use

The following diagram shows how in the TIA Portal the given process data addresses may be checked:

The screenshot shows the Siemens TIA Portal interface. On the left, the 'Project tree' displays the configuration of a PLC 1 (CPU 1511-1 PN) and its connection to a PSD4xx module. The 'Overview of addresses' table is visible, showing the following data:

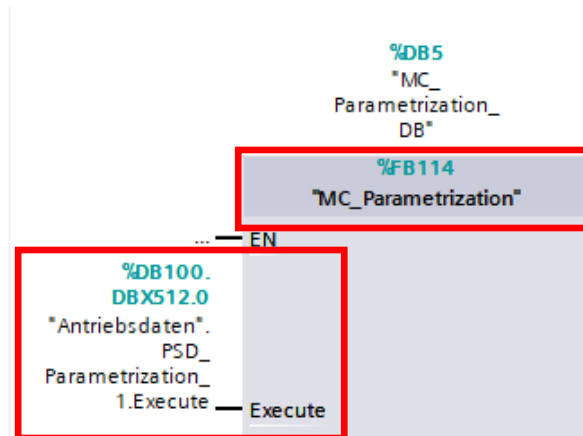
Type	Addr. fr...	Addr. to	Size	Module	Rack	Slot	Device name	Devic...	Master/IO system	PIP	OB
I	0	15	16 Byte	16 Byte Eingang_1	0	1	PSD411 [PSD]	1	PROFINET-System [100]	Automatic update	-
Q	0	13	14 Byte	14 Byte Ausgang_1	0	2	PSD411 [PSD]	1	PROFINET-System [100]	Automatic update	-

(Addresses of process data in Step 7 TIA)

3 Data Component “Antriebsdaten”

This data component serves as a template for connecting the inputs and outputs of the function blocks. The data component provides the variables which are necessary to connect each input and output of all available function blocks. Each variable is present two times, so two positioning systems may be controlled.

Example:



In case the data component is taken over into the project, it is advisable to delete the variables which are not used (in order that PLC memory is not reserved needlessly) and to adjust the data component to the number of drives which are actually present.

The data component uses the data types “Parameter” and “ParameterEnable” by default, so these types have to be taken over of the library into the project also.

4 Error Description (Error ID)

Subsequently the error codes are shown, which are displayed by the function blocks:

ErrorID (hex)	Description
16xF000 (mask)	FB
16#1xxx	Error in MC_Move
16#2xxx	Error in MC_Error
16#3xxx	Error in MC_ReadParameter
16#4xxx	Error in MC_WriteParameter
16#5xxx	Error in MC_Parametrization
16#6xxx	Error in MC_PosParametrization
16#0F00 (mask)	Internal FB and PD errors
16#x1xx	Error in state machine or other FB internal error
16#x2xx	Invalid PD input address
16#x3xx	Invalid PD output address
16#x4xx	Error while reading PD
16#x5xx	Error while writing PD
16#x6xx	Unallowed input data change
16#00F0 (mask)	Parameter errors
16#xx1x	Parameter: communication timeout (1000 ms)
16#xx2x	Parameter: invalid parameter number
16#xx3x	Parameter: value is read only
16#xx4x	Parameter: lower or upper limit exceeded
16#xx5x	Parameter: faulty sub-index
16#xx6x	Parameter: not an array
16#xx7x	Parameter: incorrect data type
16#xx8x	Parameter: setting not allowed (resetting only)
16#xx9x	Parameter: request cannot be processed due to operating state
16#xxAx	Other error
16#000F (mask)	Drive errors
16#xxx1	Drag error
16#xxx2	Under- or overvoltage motor supply
16#xxx3	Positioning run aborted
16#xxx4	Temperature exceeded
16#xxx5	Absolute measuring system error
16#xxx6	Block or overcurrent error
16#xxx7	Manual displacement
16#xxx8	Incorrect target value
16#xxx9	Under- or overvoltage during run
16#xxxA	Lower position limit exceeded
16#xxxB	Upper position limit exceeded

The errors "Drive Errors" are a copy of the error bits in the status word of the PSx.

Examples:

- Run command (MC_Move) with incorrect target value → ErrorID = 16#1008
- Writing a parameter (MC_WriteParameter) with invalid parameter number → ErrorID = 16#4020

5 Description and use of the function blocks

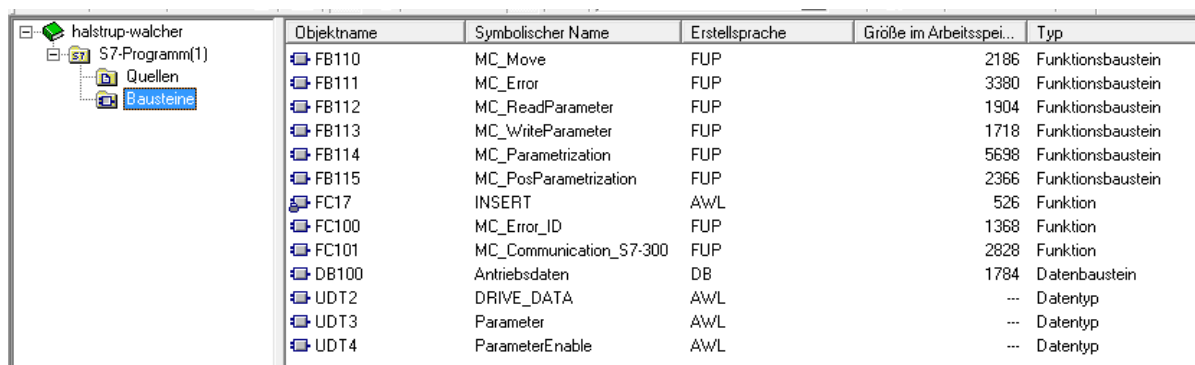
Initially the function blocks have to be included in an own user project. This happens by opening the desired library and copying the desired function blocks.

The following libraries are available in the respective current version:

- "library_Ps4xx_PN_S7_300"
→ for CPU S7-300 with Step 7 Classic V5.5
- "library_Ps4xx_PN_TIA13_300"
→ for CPU S7-300 in the TIA Portal (V13)
- "library_Ps4xx_PN_TIA13_1200-1500"
→ for CPU S7-1200/1500 in the TIA Portal (V13)
- "library_Ps4xx_PN_TIA14_300"
→ for CPU S7-300 in the TIA Portal (V14)
- "library_Ps4xx_PN_TIA14_1200-1500"
→ for CPU S7-1200/1500 in the TIA Portal (V14)
- "library_Ps4xx_PN_TIA16_300"
→ for CPU S7-300 in the TIA Portal (V16)
- "library_Ps4xx_PN_TIA16_1200-1500"
→ for CPU S7-1200/1500 in the TIA Portal (V16)

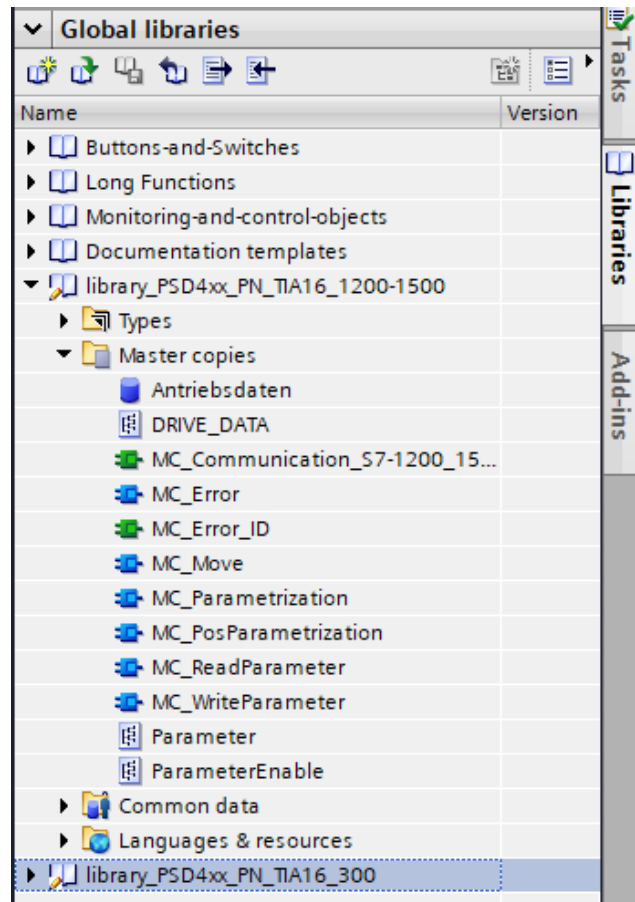
5.1 Opening the library

After opening the library the function blocks present itself in the following way:



Objektname	Symbolischer Name	Erstelsprache	Größe im Arbeitsspei...	Typ
FB110	MC_Move	FUP	2186	Funktionsbaustein
FB111	MC_Error	FUP	3380	Funktionsbaustein
FB112	MC_ReadParameter	FUP	1904	Funktionsbaustein
FB113	MC_WriteParameter	FUP	1718	Funktionsbaustein
FB114	MC_Parametrization	FUP	5698	Funktionsbaustein
FB115	MC_PosParametrization	FUP	2366	Funktionsbaustein
FC17	INSERT	AWL	526	Funktion
FC100	MC_Error_ID	FUP	1368	Funktion
FC101	MC_Communication_S7-300	FUP	2828	Funktion
DB100	Antriebsdaten	DB	1784	Datenbaustein
UDT2	DRIVE_DATA	AWL	---	Datentyp
UDT3	Parameter	AWL	---	Datentyp
UDT4	ParameterEnable	AWL	---	Datentyp

(View in Step 7 Classic V5.5)



(View in Step 7 TIA)

5.2 Copying of function blocks into the user project

The following elements of the library (independently of the actually used function blocks “MC_...” and the number of drives) in any case have to be copied into the project:

Copy into “Program blocks” of the desired CPU:

- Data type “DRIVE_DATA” (“UDT2” in case of Step 7 Classic V5.5)
- for S7-1200 und S7-1500: MC_Communication_S7-1200_1500
alternatively for S7-300: MC_Communication_S7-300
- MC_Error_ID

Additionally when using the library version for Step 7 Classic V5.5:

Copy into “S7 Program → Blocks”:

- FC17 (“INSERT”) ... a Siemens system function

These elements serve for the communication to the drive. The functions and function blocks of this group must NOT be called in the program.

Besides the desired function blocks “MC_...” have to be copied into the user project, e.g. all or a choice of the following blocks:

- MC_Move
- MC_Error
- MC_ReadParameter
- MC_WriteParameter
- MC_Parametrization
- MC_PosParametrization
- Data type “Parameter”
- Data type “ParameterEnable”

Additionally the data component “Antriebsdaten” may be taken over as a template for connecting the inputs and outputs of the function blocks. In it’s standard implementation, the data types “Parameter” and “ParameterEnable” also have to be taken over (see chapter 3).

5.3 Generating instance data blocks

Per axis and per desired function block an instance data block has be generated.

5.4 Generating global data

Per axis one global variable of the type DRIVE_DATA has to be generated (size: 132 byte in Classic Step 7, 92 byte in TIA Portal). Additionally those variables have to be generated which are connected to the inputs and outputs of the particular blocks.

If applicable, it’s reasonable to generate an additional data block of the type “global” for this data (e.g. DB100), in case of the TIA Portal also the template “Antriebsdaten” may be used (see chapter 3).

It’s not necessary to connect all inputs and outputs. If parts of a block are not used, the associated inputs may stay unconnected, then the respective initial value for this input is valid. Outputs not used also may stay open.

5.5 Commonalities of all function blocks

The function blocks are inserted in a part of the program that is called cyclically (e.g. in the OB1) and immediately be linked with their respective instance data blocks.

To the input “Drive” (type IN_OUT) the variable of the type “DRIVE_DATA” has to be connected which is provided for this axis.

The input EN and the output ENO of each function block may stay unconnected.

The inputs and outputs “Drive”, “EN” and “ENO” are not listed any more separately in the following descriptions of the particular function blocks.

5.6 Lockings between the function blocks

The function blocks partly are locked against each other. Thereby it's ensured e.g. that not two accesses out of different function blocks can be executed simultaneously on the parameter channel of an axis.

The following rules apply:

- In case the input "Release" of MC_Move is set, the function blocks MC_Parametrization and MC_PosParametrization cannot be activated (setting of "Execute" results in error 16#x100).
- In contrast it is possible to call MC_ReadParameter or MC_WriteParameter during a movement (e.g. in order to read out the actual value of the torque or to change the target speed during a run).
- In case the input "Execute" of MC_Parametrization or MC_PosParametrization is set, the function block MC_Move cannot be activated (setting of "Release" results in error 16#1100).
- Always only one of the function blocks MC_ReadParameter, MC_WriteParameter, MC_Parametrization or MC_PosParametrization can be active. If the input "Execute" of one of these function blocks is already set and the input "Execute" of a further function block is set, this one will report error 16#x100.
- The function block MC_Error can always be activated, independently of the other function blocks.

5.7 Example

The following example shall explain the procedure of involving the function blocks:

In the project three drives are located. Each drive shall be controlled by a function block MC_Move, additionally it shall be possible to determine the state of each drive with MC_Error and it shall be possible for each drive to execute any read and write access.

According chapter 5.2 for this purpose we copy the following blocks out of the library into the project of the user:

Copy into "Program blocks" of the desired CPU:

- Data type "DRIVE_DATA" ("UDT2" in case of Step 7 Classic V5.5)
- FC101 ("MC_Communication_S7-1200_1500" or "MC_Communication_S7-300")
- FC100 ("MC_Error_ID")
- FB110 ("MC_Move")
- FB111 ("MC_Error")
- FB112 ("MC_ReadParameter")
- FB113 ("MC_WriteParameter")

Additionally when using the library version for Step 7 Classic V5.5:

Copy into "S7 Program → Blocks":

- FC17 ("INSERT") ... a Siemens system function

In the next step according chapter 5.3 we generate the following instance data blocks:

- three instance data blocks of the type FB110 ("MC_Move"), e.g.
 - DB1101, symbolic name = "PSD_1_FB110"
 - DB1102, symbolic name = "PSD_2_FB110"
 - DB1103, symbolic name = "PSD_3_FB110"
- three instance data blocks of the type FB111 ("MC_Error"), e.g.
 - DB1111, symbolic name = "PSD_1_FB111"
 - DB1112, symbolic name = "PSD_2_FB111"
 - DB1113, symbolic name = "PSD_3_FB111"
- three instance data blocks of the type FB112 ("MC_ReadParameter"), e.g.
 - DB1121, symbolic name = "PSD_1_FB112"
 - DB1122, symbolic name = "PSD_2_FB112"
 - DB1123, symbolic name = "PSD_3_FB112"
- three instance data blocks of the type FB113 ("MC_WriteParameter"), e.g.
 - DB1131, symbolic name = "PSD_1_FB113"
 - DB1132, symbolic name = "PSD_2_FB113"
 - DB1133, symbolic name = "PSD_3_FB113"

After that according chapter 5.4 we generate another data block of the type "global" with the name DB1000 (we recommend using the "Antriebsdaten" template in the TIA Portal). In that data block we generate three variables of the type "DRIVE_DATA", e.g. with the following identifiers:

- "PSD_1"
- "PSD_2"
- "PSD_3"

Additionally there we generate those variables which control the particular blocks, e.g.:

- "PSD_1_Move_DINT_TargetPosition",
- "PSD_2_Move_DINT_TargetPosition",
- "PSD_1_Read_BOOL_Execute",
- "PSD_2_Read_BOOL_Execute",
- ...

Now all preparations are done in order to use the function blocks. In the editor "Program blocks" the function blocks "MC_..." appear in the section "FB blocks". So in our example the function blocks MC_Move, MC_Error, MC_ReadParameter and MC_WriteParameter are inserted each three times in a part of the program that is called cyclically (e.g. in the OB1) and immediately be linked with their respective instance data blocks. Additionally MC_Communication_S7-1200_1500 and MC_Error_ID are used, where MC_Error_ID is called in the FBs mentioned above. MC_Communication_S7-1200_1500 is called separately cyclically for each drive.

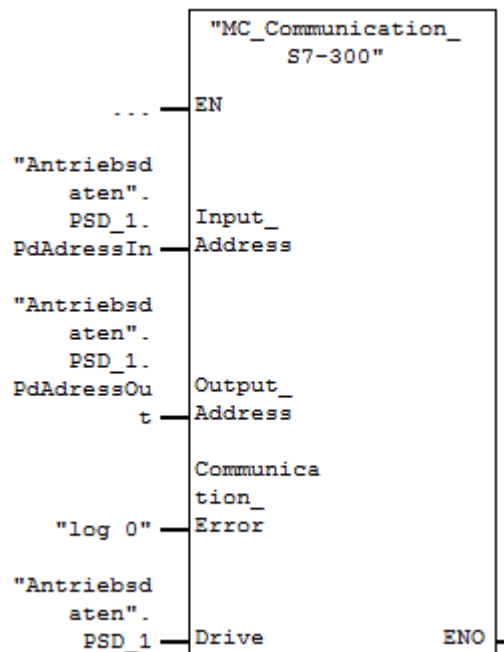
To the input "Drive" we connect the variable of the type "DRIVE_DATA" which is provided for this axis (thus "PSD_1", "PSD_2" and "PSD_3").

After that we connect the rest of the required inputs and outputs with those variables which are provided for this purpose (thus "PSD_1_Move_DINT_TargetPosition", ...).

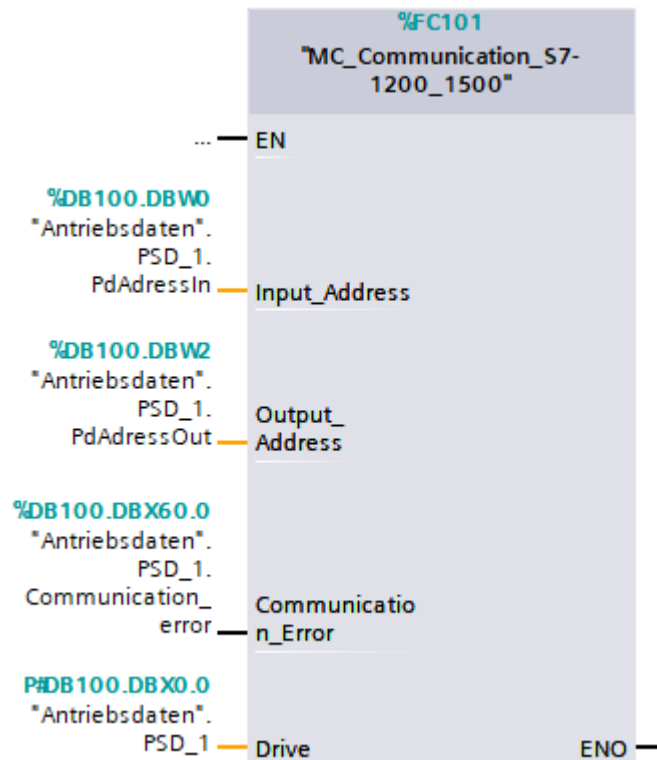
5.8 MC_Communication_S7-1200_1500 and _S7-300 (FC101)

In case of the IO Controller S7-300, this component is called "MC_Communication_S7-300". In case of the IO Controllers S7-1200 and S7-1500, this component is called "MC_Communication_S7-1200_1500". The functionality is identical in both cases, below the variant for S7-1200 and S7-1500 is described.

This FC is used for the communication with the drive (IO Device). It performs the communication centrally for all following FBs (MC_Move, MC_ReadParameter, MC_WriteParameter and MC_Error). The input and output module is filed in the data type DRIVE_DATA. With the help of this interface the other FBs are able to access the drive's data.



(View in Step 7 Classic V5.5)



(View in Step 7 TIA)

Input_Address

Address of the input module of the drive

- Type: INT (in case of S7-1200 and S7-1500), DWORD (in case of S7-300)
- Direction: INPUT

Output_Address

Address of the output module of the drive

- Type: INT (in case of S7-1200 and S7-1500), DWORD (in case of S7-300)
- Direction: INPUT

Communication_Error

Communication error to the IO device (the drive)

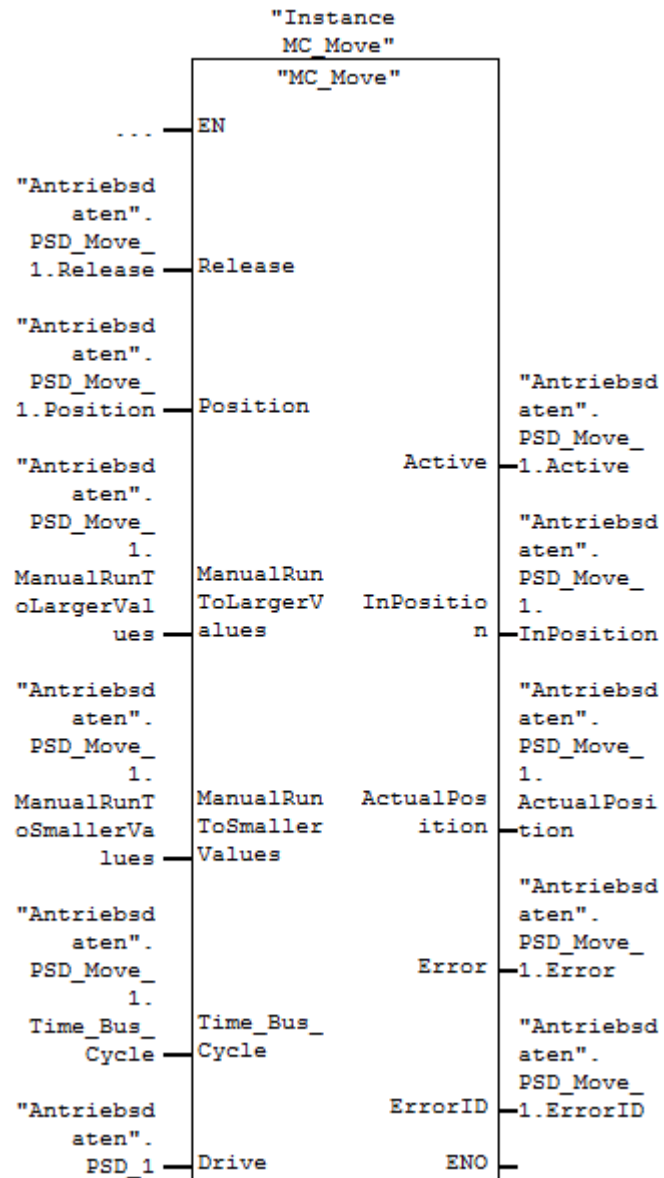
- Type: BOOL
- Direction: INPUT

Description:

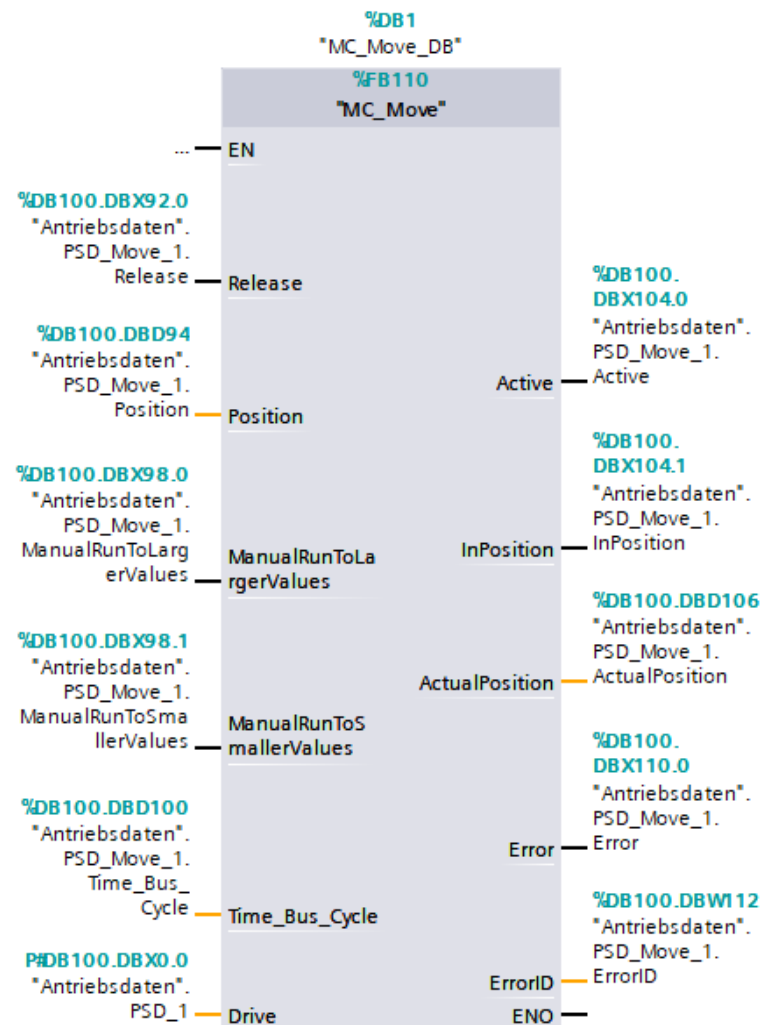
- The diagnosis of the component status of the IO devices usually is managed centrally in the PLC, e.g. in OB86 or with the help of the command "DeviceStates".
- When the corresponding IO Device fails (e.g. here the drive), this input has to be set to "TRUE". As long as the communication is not disturbed, the input has to be set to "FALSE".
- In case of communication faults, run commands started with the help of MC_Move are aborted. As well, running parameter accesses (MC_ReadParameter, MC_WriteParameter) are aborted.

5.9 MC_Move (FB110)

This FB serves to send run commands to the drive.



(View in Step 7 Classic V5.5)



(View in Step 7 TIA)

Release

Release of the drive

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

Description:

- Run commands will only be executed if this bit is set.
- This input directly controls the release bit (bit 4) in the control word. If this input stays activated and e.g. the readjustment in the drive is activated, the drive readjusts automatically.
- If the input is activated and the target position is changed, the drive immediately moves to that position. An edge is not necessary.
- If the input is deasserted during the run, the drive stops.

Position

Target position to be approached

- Type: DINT
- Initial value: 0

- Direction: INPUT

Description:

- If during a run a new target position is sent, this target position is approached immediately.
- If the release bit is still set after the end of a run and the target position is changed, the drive immediately approaches to that position.



INFORMATION!

In order to move to the same target position e.g. after a blocking condition, the release has to be deasserted and asserted again.

ManualRunToLargerValues

Manual run to larger values

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

Description:

- Manual run to larger values, finishing at the positive range limit.
- Additionally the input "Release" has to be on resp. set.



CAUTION!

When deasserting the input "ManualRunToLargerValues", additionally the release input has to be deasserted. Otherwise the drive will move to the target position (FB input "Position").

ManualRunToSmallerValues

Manual run to smaller values

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

Description:

- Manual run to larger values, finishing at the negative range limit.
- Additionally the input "Release" has to be on resp. set.



CAUTION!

When deasserting the input "ManualRunToSmallerValues", additionally the release input has to be deasserted. Otherwise the drive will move to the target position (FB input "Position").

Time_Bus_Cycle

Cycle time of the Profinet bus cycle

- Type: TIME
- Initial value: 40 ms
- Direction: INPUT

Description:

- In case of long cycle times of the Profinet bus cycle, it might appear that the PLC doesn't receive a change of the bit "drive is running". This is the case when the time for the run command is shorter than the bus cycle.

- To face this situation, the parameter “Time_Bus_Cycle” has been implemented. The default value is set to 40 ms for one bus cycle. For this time period, after beginning a run command the output “Active” is asserted in any case and the output “InPosition” is deasserted. After this time period, depending of the response of the status word, these outputs adopt the corresponding states (bit “drive is running” and bit “target position reached”).
- In case of longer cycle times of the bus cycle, it is recommended to enter the actual duration of the cycle multiplied by 2.
- If a shorter bus cycle time can be kept for sure, the time for a positioning is very short and after a completed positioning the higher-level sequence shall be continued quickly, the value for Time_Bus_Cycle also might be reduced.

Active

Run command or run is active

- Type: BOOL
- Direction: OUTPUT

This output is asserted, if:

- the release bit is set from 0 to 1 (rising edge)
- the release is already present and the target position is changing
- the bit “drive is running” in the status word of the drive is set (e.g. when the drive is readjusting its position)

This output is deasserted, if:

- at the end of a run the bit “drive is running” in the status word of the drive is no longer set
- a communication error occurs

InPosition

Target position reached

- Type: BOOL
- Direction: OUTPUT

This output is a copy of the status bit “target position reached”. If a communication error occurs, it will be deasserted.

Actual position

Actual value of the position

- Type: DINT
- Direction: OUTPUT

This value is a copy of the actual position. If a communication error occurs, the value will be set to 0.

Error

Error while executing the FB or error in drive

- Type: BOOL
- Direction: OUTPUT

The error bit also might be set during a move of the drive (e.g. drag error).

ErrorID

Error code

- Type: WORD
- Direction: OUTPUT

The error bit also might be set during a move of the drive (e.g. drag error). In case of no error, the value is set to 0.



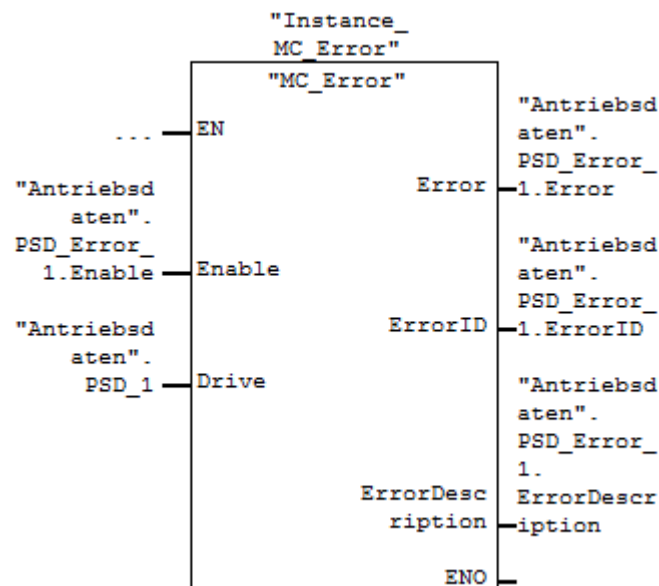
The outputs “Error” and “ErrorID” of MC_Move are always updated – also if the input “Release” is not set.

If the drive reports multiple errors, the ErrorID with the highest priority is shown. This priority corresponds to the order in the following table (highest priority has 16#x1xx):

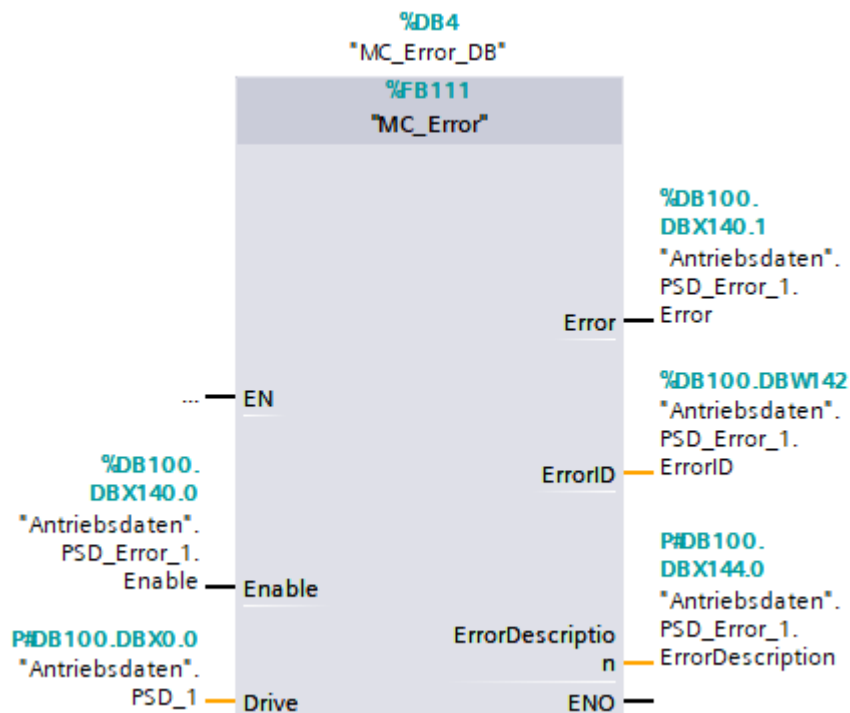
ErrorID	Description
16#x1xx	FB internal error
16#x2xx	Invalid PD input address
16#x3xx	Invalid PD output address
16#x4xx	Error while reading PD
16#x5xx	Error while writing PD
16#xxx2	Under- or overvoltage motor supply
16#xxx4	Temperature exceeded
16#xxx5	Absolute measuring system error
16#xxx8	Incorrect target value
16#xxx9	Under- or overvoltage during run
16#xxx6	Block or overcurrent error
16#xxx7	Manual displacement
16#xxxA	Lower position limit exceeded
16#xxxB	Upper position limit exceeded
16#xxx3	Positioning run aborted
16#xxx1	Drag error

5.10 MC_Error (FB111)

This FB reports the state of the drive and the FB as error bit, error code (“ErrorID”) and as text. In the case that MC_Error is activated, the error code is always the same as the error code of the function block MC_Move.



(View in Step 7 Classic V5.5)



(View in Step 7 TIA)

Enable

The outputs Error, ErrorID and ErrorDescription permanently are updated by the drive, as long as Enable is set. If Enable is deasserted, these outputs switch to their default values.

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

Error

Error while executing the FB or error in drive

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

ErrorID

Error code (see following table "ErrorID")

- Type: WORD
- Default value: 0
- Direction: OUTPUT

ErrorDescription

Error Description as text

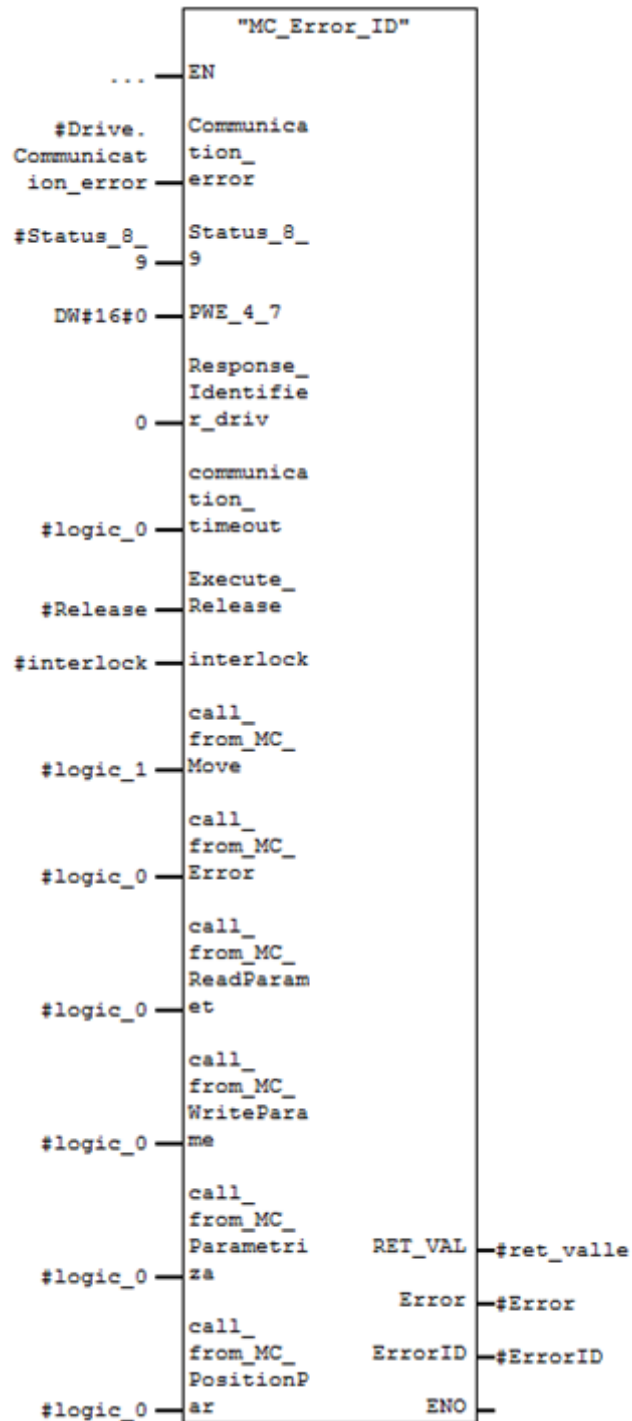
- Type: STRING
- Default value: ""
- Direction: OUTPUT

The priority corresponds to the order in the following table (highest priority has 16#x1xx).

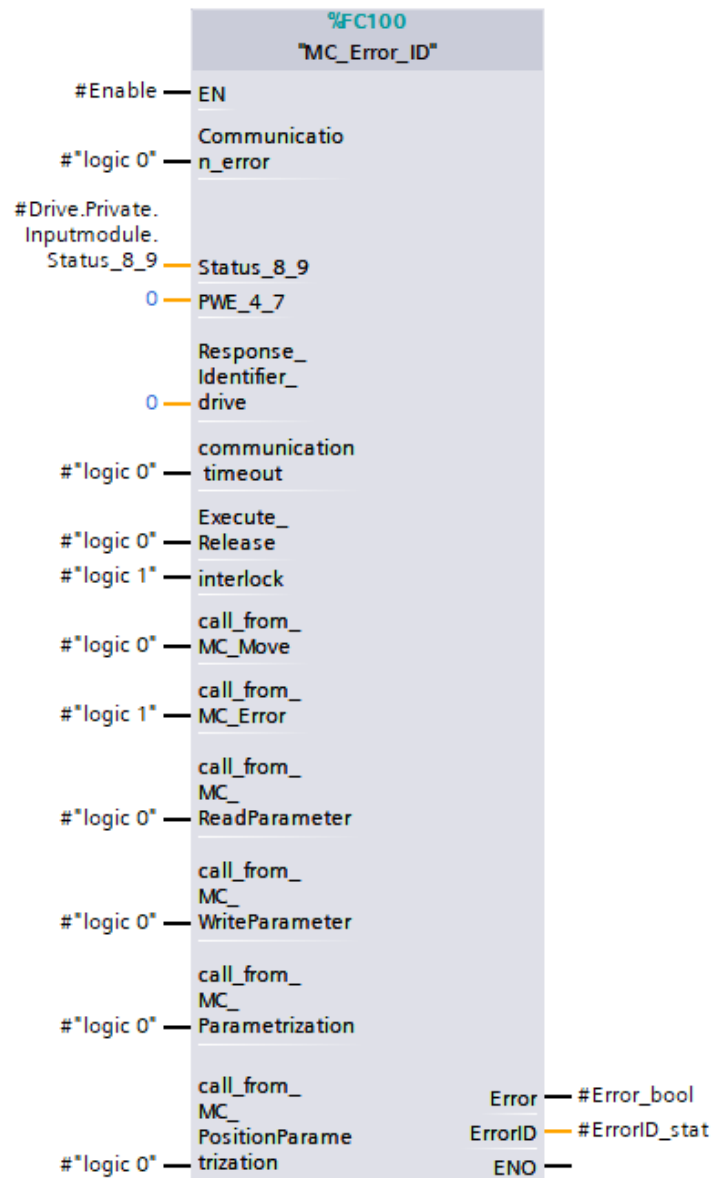
ErrorID	ErrorDescription
16#x1xx	FB internal error
16#x2xx	Invalid PD input address
16#x4xx	Error while reading PD
16#xxx2	Under- or overvoltage motor supply
16#xxx4	Temperature exceeded
16#xxx5	Absolute measuring system error
16#xxx8	Incorrect target value
16#xxx9	Under- or overvoltage during run
16#xxx6	Block or overcurrent error
16#xxx7	Manual displacement
16#xxxA	Lower position limit exceeded
16#xxxB	Upper position limit exceeded
16#xxx3	Positioning run aborted
16#xxx1	Drag error

5.11 MC_Error_ID (FC100)

This FC is used internally as a sub function of the FBs “MC_Move”, “MC_ReadParameter”, “MC_WriteParameter” and “MC_Error”. It just has to be copied from the library into the application program. The setting of the inputs is done completely internally.



(View in Step 7 Classic V5.5)



(View in Step 7 TIA)

Communication error

In case of communication faults, the boolean output “Error” and its corresponding “ErrorID” are set.

- Type: BOOL
- Direction: INPUT

Status_8_9

Evaluation of the status word of the drive

- Type: STRUCT
- Direction: INPUT

PWE_4_7

Evaluation of the parameter value

- Type: DWORD
- Direction: INPUT

Response Identifier drive

Evaluation of the response identifier of the drive

- Type: INT
- Direction: INPUT

communication timeout

Evaluation of a communication timeout

- Typ: BOOL
- Art: INPUT

Execute Release

Is used for the evaluation of internal lockings between the function blocks. According to the FB calling MC_Error_ID, this input is connected to "Execute" or "Release".

- Type: BOOL
- Direction: INPUT

interlock

Is used for the evaluation of internal lockings between the function blocks.

- Type: BOOL
- Direction: INPUT

Call from MC_Move

FC (sub function) is called in the FB "MC_Move". According to the FB calling MC_Error_ID, the first digit of the Error ID is set.

- Type: BOOL
- Direction: INPUT

Call from MC_Error

FC (sub function) is called in the FB "MC_Error". According to the FB calling MC_Error_ID, the first digit of the Error ID is set.

- Type: BOOL
- Direction: INPUT

Call from MC_ReadParameter

FC (sub function) is called in the FB "MC_ReadParameter". According to the FB calling MC_Error_ID, the first digit of the Error ID is set.

- Type: BOOL
- Direction: INPUT

Call from MC_WriteParameter

FC (sub function) is called in the FB "MC_WriteParameter". According to the FB calling MC_Error_ID, the first digit of the Error ID is set.

- Type: BOOL
- Direction: INPUT

Call from MC Parametrization

FC (sub function) is called in the FB "MC_Parametrization". According to the FB calling MC_Error_ID, the first digit of the Error ID is set.

- Type: BOOL
- Direction: INPUT

Call from MC PosParametrization

FC (sub function) is called in the FB "MC_PosParametrization". According to the FB calling MC_Error_ID, the first digit of the Error ID is set.

- Type: BOOL
- Direction: INPUT

Error

If an error occurs, this bit is set.

- Type: BOOL
- Direction: OUTPUT

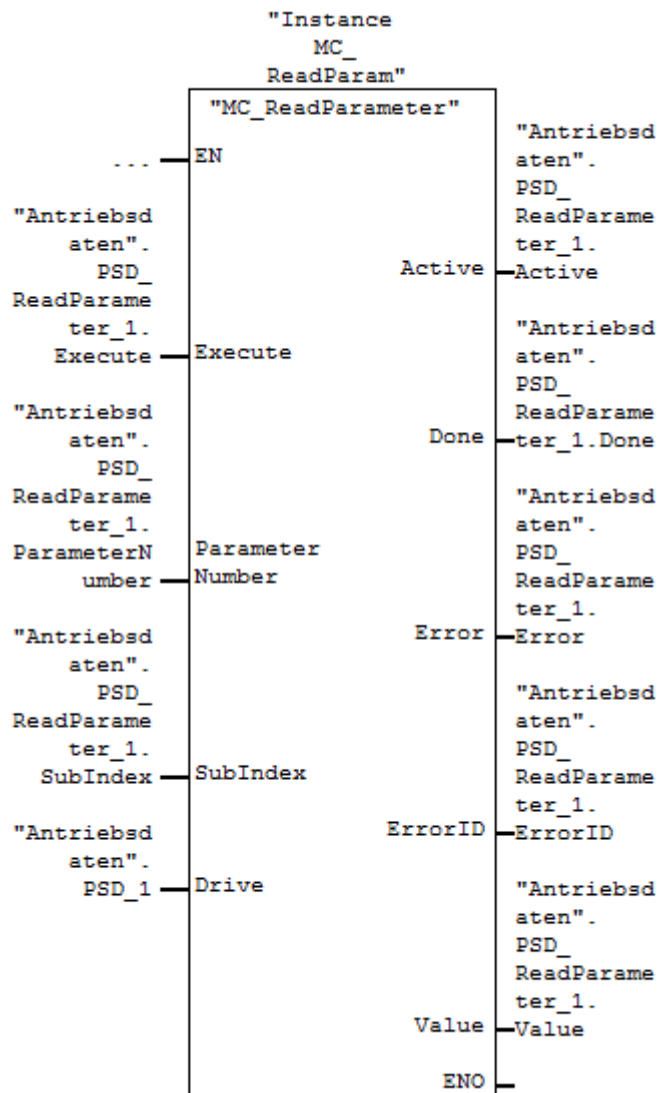
ErrorID

Depending on the inputs, "ErrorID" is set.

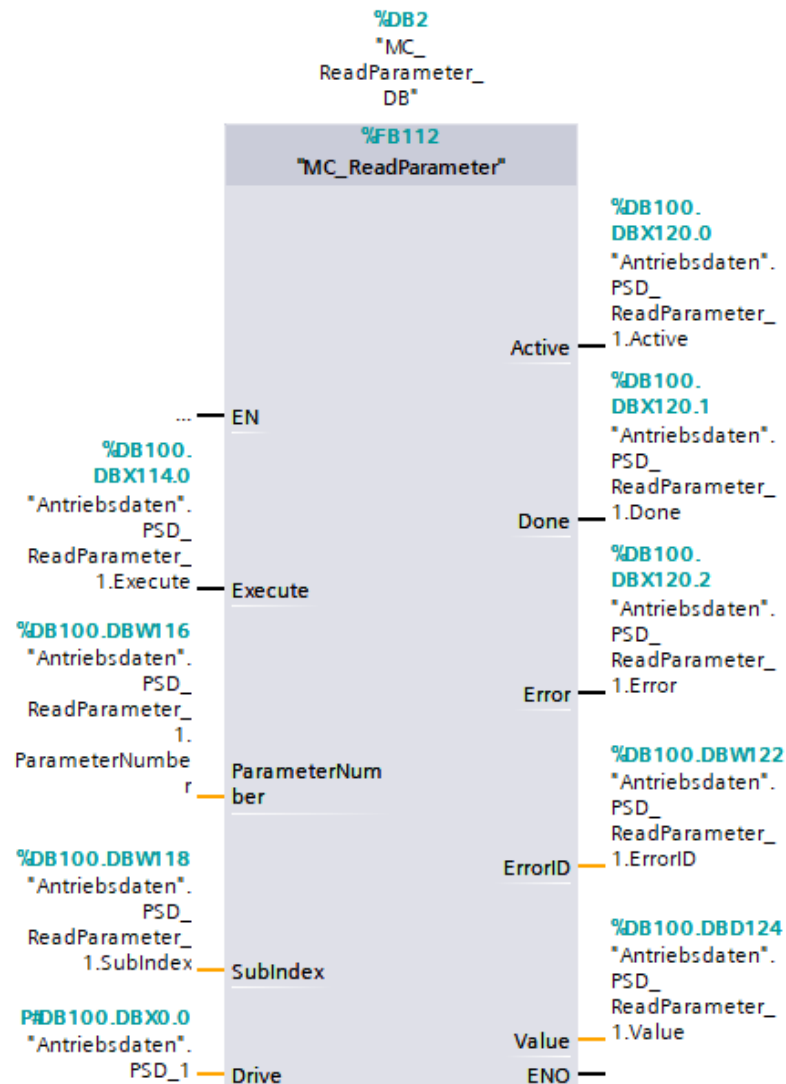
- Type: WORD
- Direction: OUTPUT

5.12 MC_ReadParameter (FB112)

With this FB values of parameters can be read by the drive.



(View in Step 7 Classic V5.5)



(View in Step 7 TIA)

Execute

Start of a reading process

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

Description:

When issuing a rising edge, a reading process of the parameter which is specified by "ParameterNumber" and "Subindex" is started. For a new reading process, a new rising edge has to be generated. When deasserting the bit, the outputs fall back to their specified default value.

ParameterNumber

Parameter number of the parameter to be read

- Type: INT
- Initial value: 0
- Direction: INPUT

SubIndex

Array subindex of the parameter

- Type: INT
- Initial value: 0
- Direction: INPUT

Active

Bit is set as long as the reading process is active

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

The bit is being deasserted as soon as the value has been read or an error occurred.

Done

Bit is set as soon as the parameter has been read successfully and is available in "Value"

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

The bit is being deasserted when starting a reading process.

Error

Bit is set if an error occurred during the execution of the FB

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

ErrorID

Error code (see table "ErrorID" in chapter 4)

- Type: WORD
- Default value: 0
- Direction: OUTPUT

Drive errors are not considered when reading a parameter.

Value

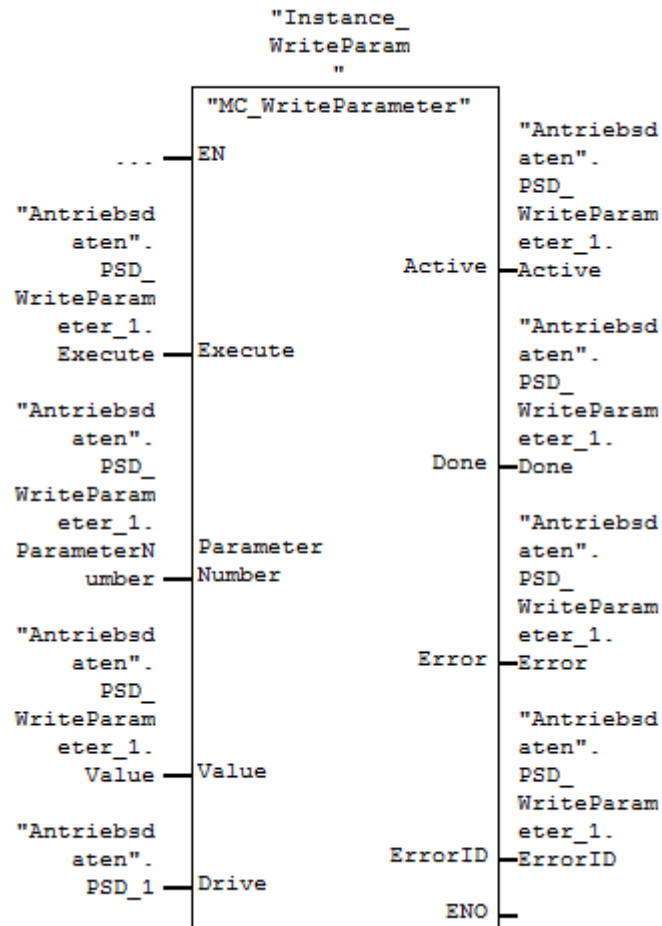
Actual value of the parameter which has been read

- Type: DINT
- Default value: 0
- Direction: OUTPUT

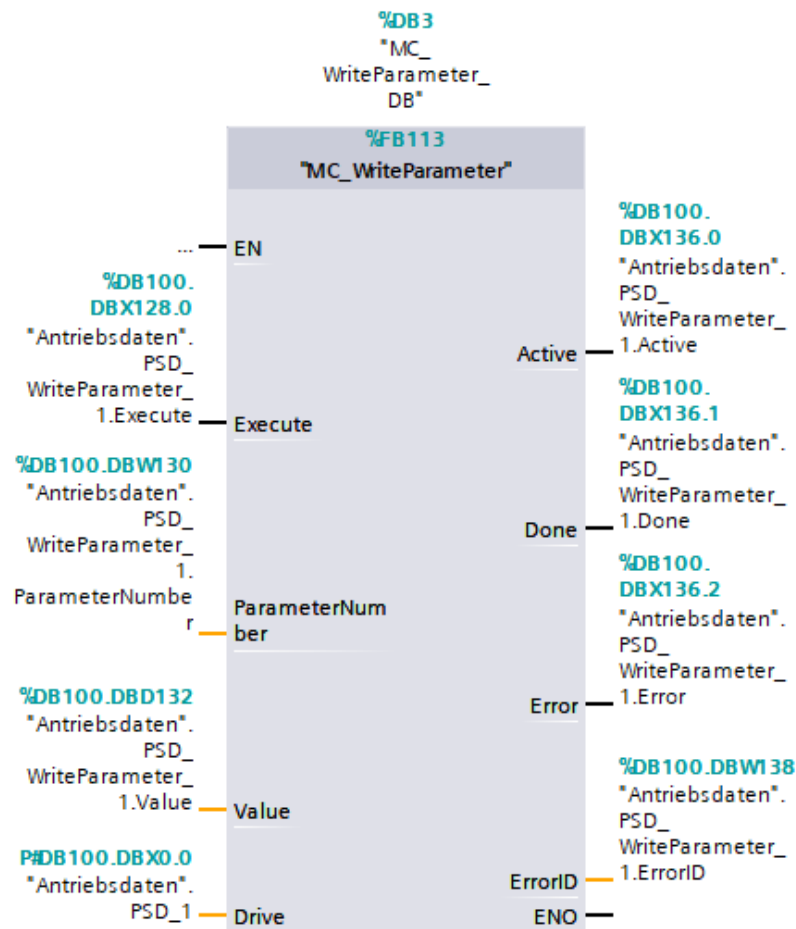
When an error occurred, the value will be 0.

5.13 MC_WriteParameter (FB113)

With this FB values of parameters can be written into the drive.



(View in Step 7 Classic V5.5)



(View in Step 7 TIA)

Execute

Start of a writing process

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

Description:

When issuing a rising edge, a writing process of the parameter which is specified by "ParameterNumber" with the value which is specified by the input "Value" is started. For a new writing process, a new rising edge has to be generated. When deasserting the bit, the outputs fall back to their specified default value.

ParameterNumber

Parameter number of the parameter to be written

- Type: INT
- Initial value: 0
- Direction: INPUT

Value

Value to be written to the parameter

- Type: DINT
- Initial value: 0

- Direction: INPUT

Active

Bit is set as long as the writing process is active

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

The bit is being deasserted as soon as the value has been written or an error occurred.

Done

Bit is set as soon as the parameter has been written successfully

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

The bit is being deasserted when starting a writing process.

Error

Bit is set if an error occurred during the execution of the FB

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

ErrorID

Error code (see table "ErrorID" in chapter 4)

- Type: WORD
- Default value: 0
- Direction: OUTPUT

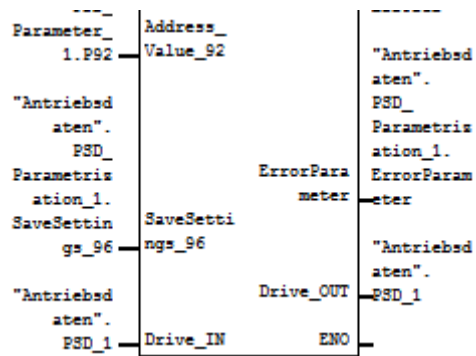
Drive errors are not considered when writing a parameter.

5.14 MC_Parametrization (FB114)

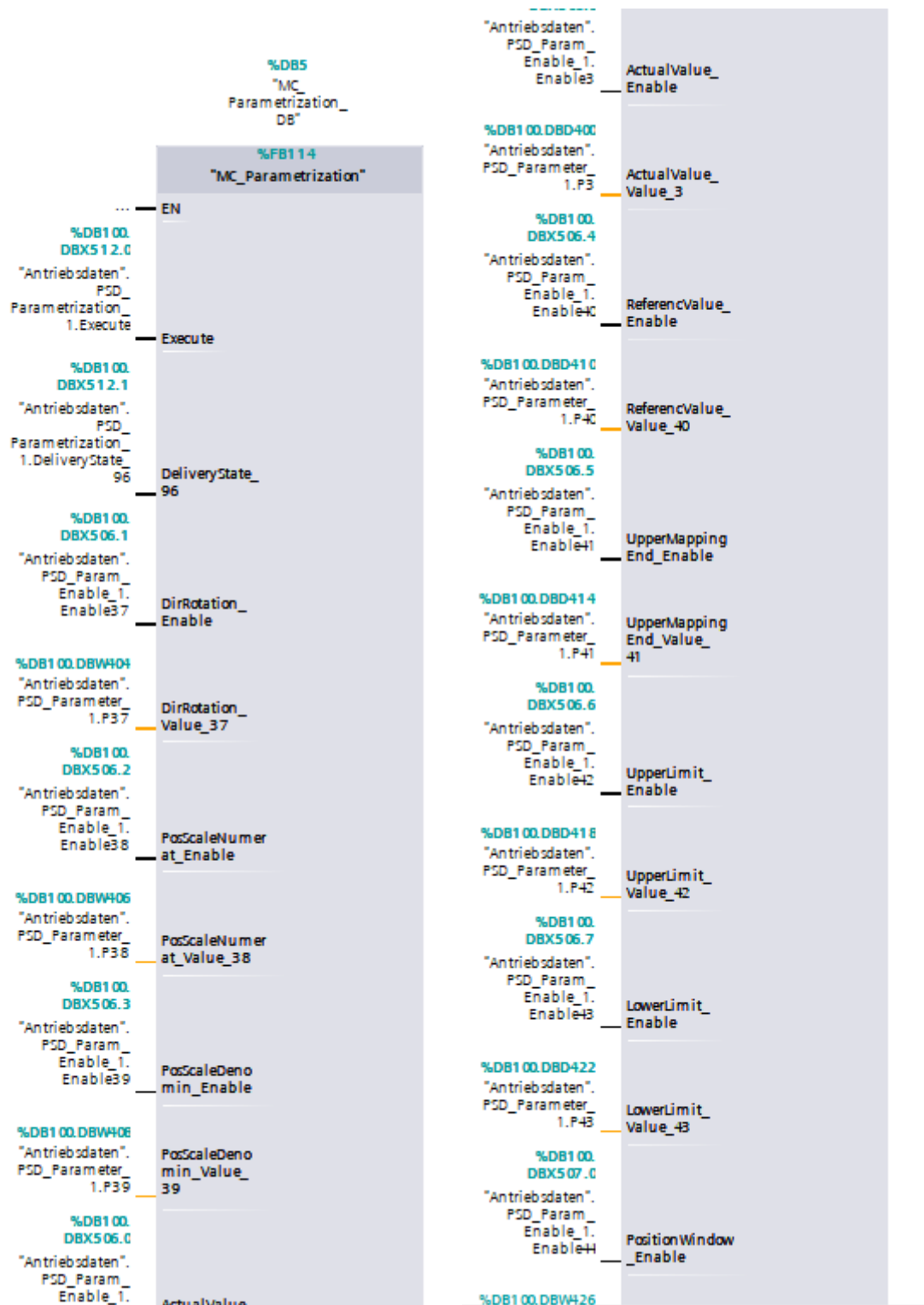
With this FB all parameters of the drive can be written.

	"Instance_ MC_ Parametris at"	"MC_ Parametrisation"	"Antriebsd aten". PSD_ Parameter_ 1.P40	ReferencV alue_ Value_40	"Antriebsd aten". PSD_Param_ Enable_1. Enable53	TargetSpe ed_Enable
...	EN					
"Antriebsd aten". PSD_ Parametris ation_1. Execute	Execute		"Antriebsd aten". PSD_Param_ Enable_1. Enable41	UpperMapp ingEnd_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P53	TargetSpe ed_Value_ 53
"Antriebsd aten". PSD_ Parametris ation_1. DeliverySt ate_96	DeliverySt ate_96		"Antriebsd aten". PSD_ Parameter_ 1.P41	UpperMapp ingEnd_ Value_41	"Antriebsd aten". PSD_Param_ Enable_1. Enable56	TargSpeed ManRun_ Enable
"Antriebsd aten". PSD_Param_ Enable_1. Enable37	DirRotati on_Enable		"Antriebsd aten". PSD_ Parameter_ 1.P42	UpperLimi t_Value_ 42	"Antriebsd aten". PSD_ Parameter_ 1.P56	TargSpeed ManRun_ Value_56
"Antriebsd aten". PSD_ Parameter_ 1.P37	DirRotati on_Value_ 37		"Antriebsd aten". PSD_Param_ Enable_1. Enable43	LowerLimi t_Enable	"Antriebsd aten". PSD_ Parameter_ 1.P57	SpeedLimi tAbort_ Value_57
"Antriebsd aten". PSD_Param_ Enable_1. Enable38	PosScaleN umerat_ Enable		"Antriebsd aten". PSD_ Parameter_ 1.P43	LowerLimi t_Value_ 43	"Antriebsd aten". PSD_Param_ Enable_1. Enable58	Accelerat ion_ Enable
"Antriebsd aten". PSD_ Parameter_ 1.P38	PosScaleN umerat_ Value_38		"Antriebsd aten". PSD_Param_ Enable_1. Enable44	PositionW indow_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P58	Accelerat ion_ Value_58
"Antriebsd aten". PSD_Param_ Enable_1. Enable39	PosScaled enomin_ Enable		"Antriebsd aten". PSD_ Parameter_ 1.P44	PositionW indow_ Value_44	"Antriebsd aten". PSD_Param_ Enable_1. Enable59	Decelerat ion_ Enable
"Antriebsd aten". PSD_ Parameter_ 1.P39	PosScaled enomin_ Value_39		"Antriebsd aten". PSD_Param_ Enable_1. Enable45	LoopLengh t_Enable	"Antriebsd aten". PSD_ Parameter_ 1.P59	Decelerat ion_ Value_59
"Antriebsd aten". PSD_Param_ Enable_1. Enable3	ActualVal ue_Enable		"Antriebsd aten". PSD_ Parameter_ 1.P45	LoopLengh t_Value_ 45	"Antriebsd aten". PSD_Param_ Enable_1. Enable63	MaxStartT orque_ Enable
"Antriebsd aten". PSD_ Parameter_ 1.P3	ActualVal ue_Value_ 3		"Antriebsd aten". PSD_Param_ Enable_1. Enable47	Readjustm ent_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P63	MaxStartT orque_ Value_63
"Antriebsd aten". PSD_Param_ Enable_1. Enable40	ReferencV alue_ Enable		"Antriebsd aten". PSD_ Parameter_ 1.P47	Readjustm ent_ Value_47	"Antriebsd aten". PSD_Param_ Enable_1. Enable64	MaxTorque _Enable

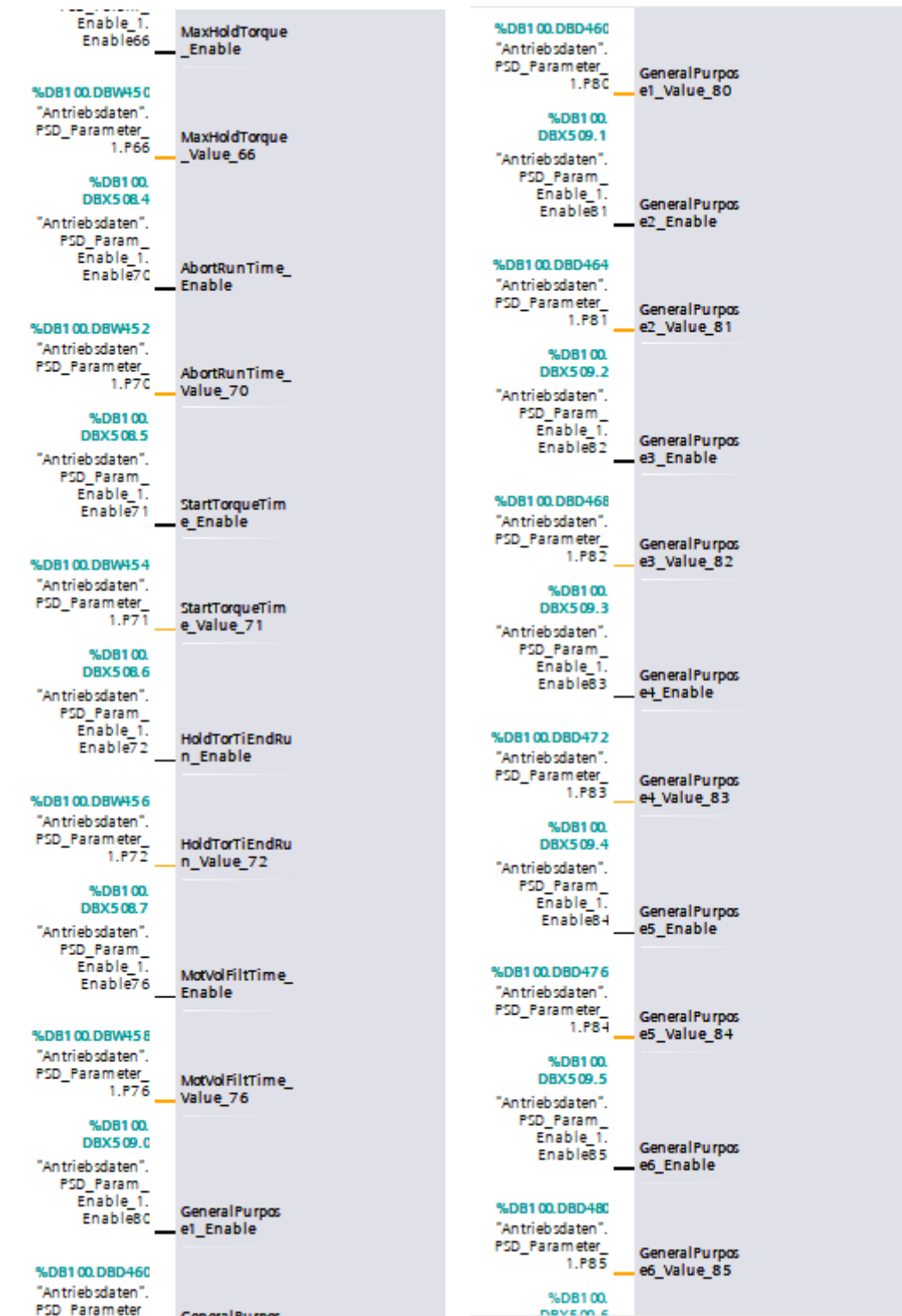
"Antriebsd aten". PSD_ Parameter_ 1.P64	MaxTorque _Value_64	"Antriebsd aten". PSD_Param_ Enable_1. Enable80	GeneralPu rpose1_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P86	GeneralPu rpose7_ Value_86	
"Antriebsd aten". PSD_Param_ Enable_1. Enable65	MaxHoldTor EndRun_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P80	GeneralPu rpose1_ Value_80	"Antriebsd aten". PSD_Param_ Enable_1. Enable87	GeneralPu rpose8_ Enable	
"Antriebsd aten". PSD_ Parameter_ 1.P65	MaxHoldTor EndRun_ Value_65	"Antriebsd aten". PSD_Param_ Enable_1. Enable81	GeneralPu rpose2_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P87	GeneralPu rpose8_ Value_87	
"Antriebsd aten". PSD_Param_ Enable_1. Enable66	MaxHoldTo rque_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P81	GeneralPu rpose2_ Value_81	"Antriebsd aten". PSD_Param_ Enable_1. Enable88	GeneralPu rpose9_ Enable	
"Antriebsd aten". PSD_ Parameter_ 1.P66	MaxHoldTo rque_ Value_66	"Antriebsd aten". PSD_Param_ Enable_1. Enable82	GeneralPu rpose3_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P88	GeneralPu rpose9_ Value_88	
"Antriebsd aten". PSD_Param_ Enable_1. Enable70	AbortRunT ime_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P82	GeneralPu rpose3_ Value_82	"Antriebsd aten". PSD_Param_ Enable_1. Enable89	GeneralPu rpose10_ Enable	
"Antriebsd aten". PSD_ Parameter_ 1.P70	AbortRunT ime_ Value_70	"Antriebsd aten". PSD_Param_ Enable_1. Enable83	GeneralPu rpose4_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P89	GeneralPur pose10_ Value_89	
"Antriebsd aten". PSD_Param_ Enable_1. Enable71	StartTorq ueTime_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P83	GeneralPu rpose4_ Value_83	"Antriebsd aten". PSD_Param_ Enable_1. Enable90	MinVolutag e_Enable	
"Antriebsd aten". PSD_ Parameter_ 1.P71	StartTorq ueTime_ Value_71	"Antriebsd aten". PSD_Param_ Enable_1. Enable84	GeneralPu rpose5_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P90	MinVolutag e_Value_ 90	"Antriebsd aten". PSD_ Parametris ation_1.
"Antriebsd aten". PSD_Param_ Enable_1. Enable72	HoldTorTi EndRun_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P84	GeneralPu rpose5_ Value_84	"Antriebsd aten". PSD_Param_ Enable_1. Enable91	MaxTemper ature_ Enable	Active Active
"Antriebsd aten". PSD_ Parameter_ 1.P72	HoldTorTi EndRun_ Value_72	"Antriebsd aten". PSD_Param_ Enable_1. Enable85	GeneralPu rpose6_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P91	MaxTemper ature_ Value_91	Done Done
"Antriebsd aten". PSD_Param_ Enable_1. Enable76	MotVolFil tTime_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P85	GeneralPu rpose6_ Value_85	"Antriebsd aten". PSD_Param_ Enable_1. Enable92	Address_ Enable	Error Error
"Antriebsd aten". PSD_ Parameter_ 1.P76	MotVolFil tTime_ Value_76	"Antriebsd aten". PSD_Param_ Enable_1. Enable86	GeneralPu rpose7_ Enable	"Antriebsd aten". PSD_ Parameter_ 1.P92	Address_ Value_92	ErrorID ErrorID
						"Antriebsd aten". PSD_ Parametris ation_1.

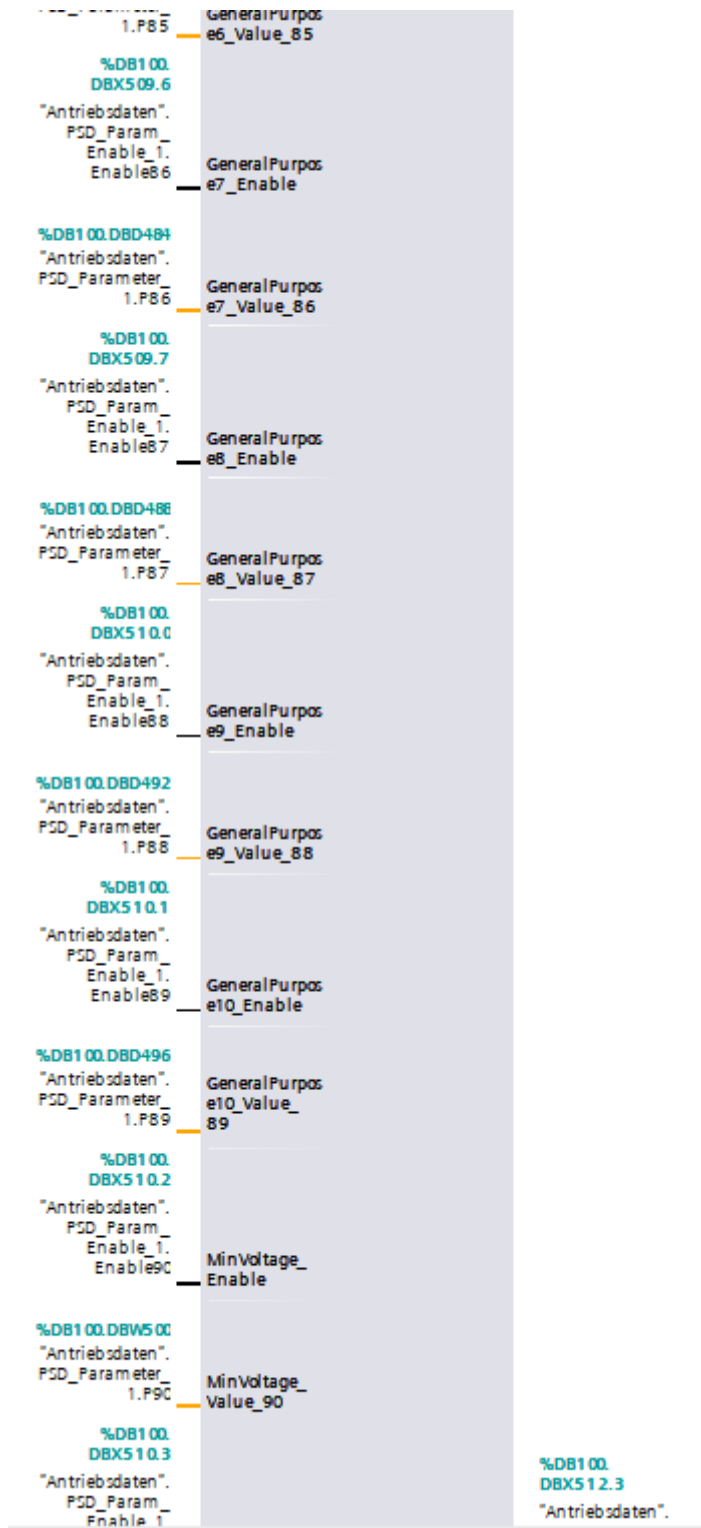


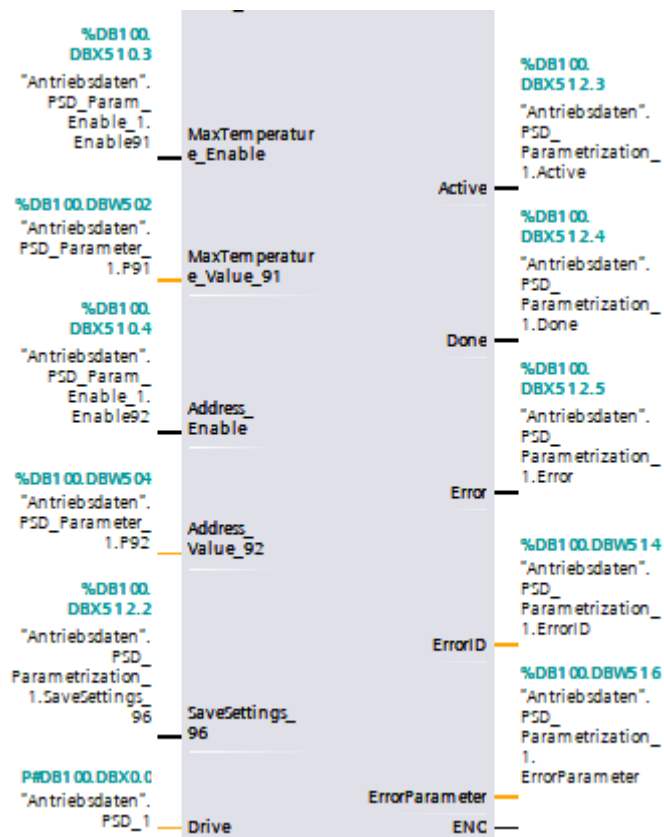
(View in Step 7 Classic V5.5)











(Ansicht in Step 7 TIA)

The following items have to be considered when using this FB:

- For each parameter value there's additionally an enable tag in order to determine whether the parameter shall be written or not.
Example: DirRotation_Enable = 1 → DirRotation_Value is written
- The order of the write accesses is like represented in the FB diagram ("DeliveryState" → "DirRotation" → ...).
- Optionally a delivery state might be commanded before setting a certain number of parameters. To do this, the input "DeliveryState_96" has to be set to TRUE before the execution of the FB. Thus the values of each parameter are set to the delivery state (initially without saving).
- Optionally at the end additionally the written values might be saved permanently. To do this, the input "SaveSettings_96" has to be set to TRUE before the execution of the FB.
- In case of an error while writing a parameter, the subsequent parameters are not written any more. Also no saving of the values is carried out, if the input "SaveSettings" is set.

Execute

Start of a parametrization process

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

Description:

When issuing a rising edge, a parametrization process with the given values is started. For a new parametrization process, a new rising edge has to be generated. When deasserting the bit, the outputs fall back to their specified default value.

DeliveryState

Loading of the delivery state (initially without saving)

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

However, device name and IP address stay unaffected.

x_Enable

If set, the corresp. parameter is written

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

x_Value

Desired value of the parameter

- Initial value: 0
- Direction: INPUT

The parameter number is given after the parameter name. The data type, a description as well as the value range can be extracted of the instruction manual of the PSD-4__-PN resp. PSC-4__-PN.

SaveSettings

Saving the settings permanently

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

Active

Bit is set as long as the parametrization process is active

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

The bit is being deasserted as soon as the parametrization has been finished successfully or an error occurred.

Done

Bit is set as soon as the parametrization has been finished successfully

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

The bit is being deasserted when starting a parametrization process.

Error

Bit is set if an error occurred during the execution of the FB

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

ErrorID

Error code (see table “ErrorID” in chapter 4)

- Type: WORD
- Default value: 0
- Direction: OUTPUT

Drive errors are not considered during a parametrization.

ErrorParameter

Parameter number that caused the error (in case of an error)

- Type: INT
- Default value: 0
- Direction: OUTPUT

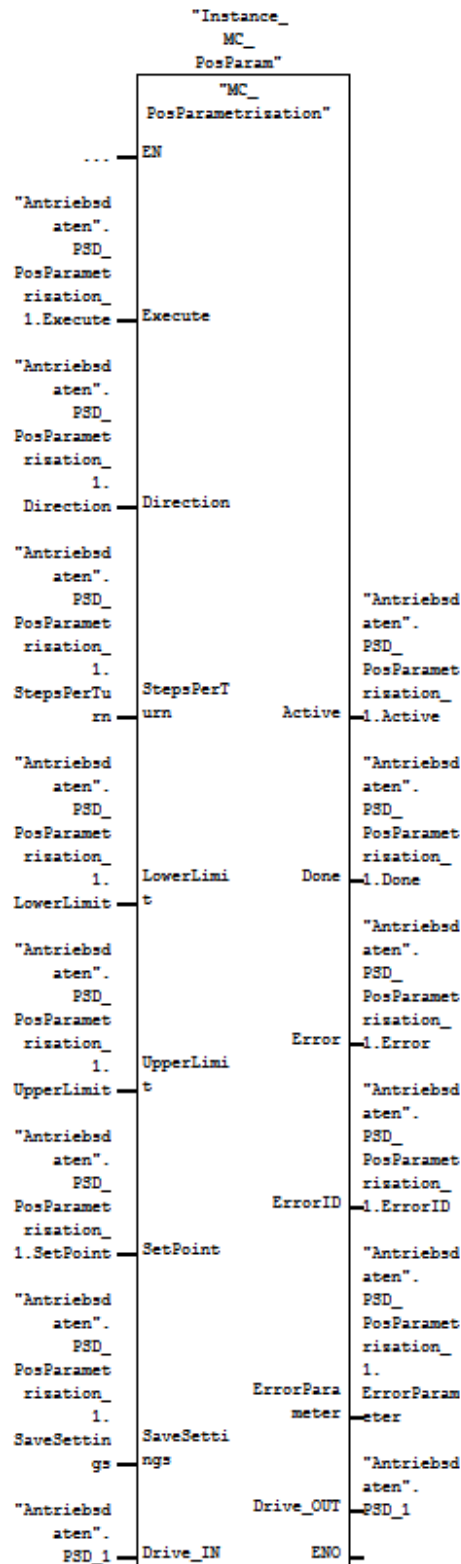
If no error occurred, this value is 0.

**INFORMATION!**

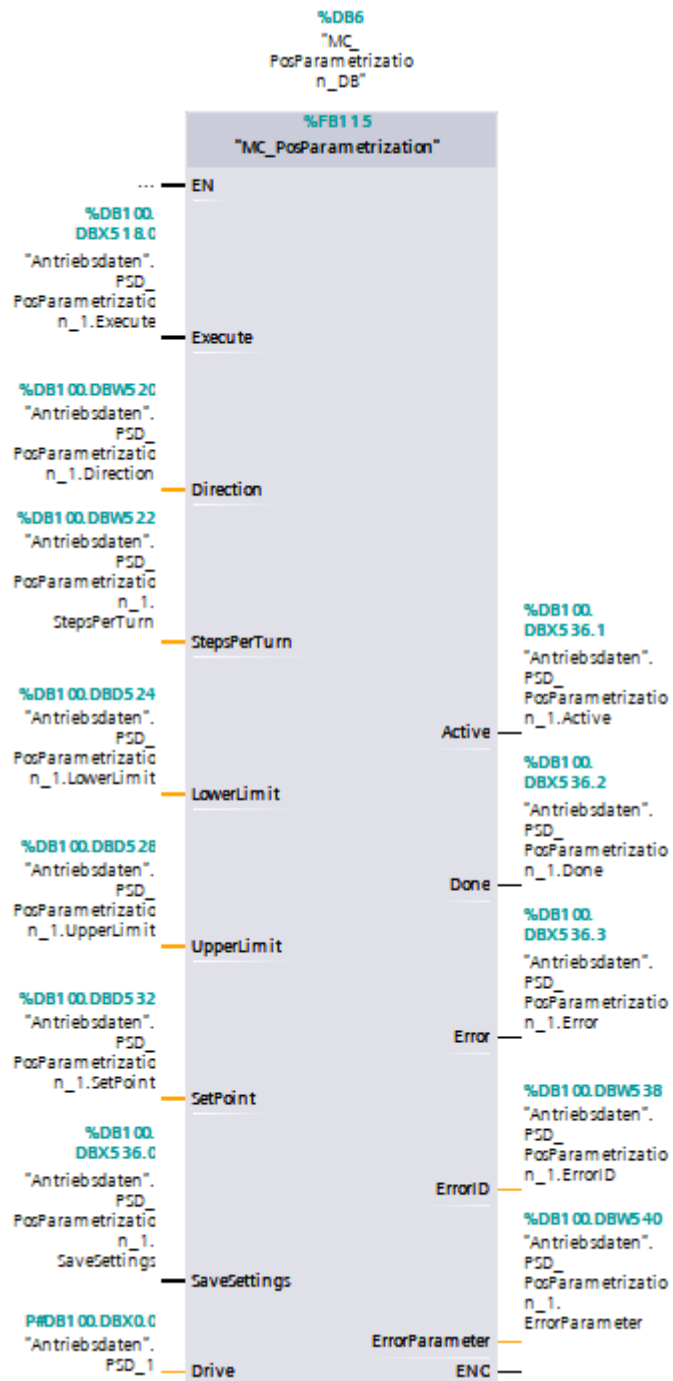
In case of the S7-300, for the function blocks “MC_Parametrization” and “MC_PosParametrization” the variable of the type “DRIVE_DATA” has to be connected to the input “DRIVE_IN” and to the output “DRIVE_OUT” (in case of the S7-1200 and S7-1500 in the TIA portal the variable is only connected one time to the Input/Output “Drive”).

5.15 MC_PosParametrization (FB115)

With this FB the parametrization of the position data can be carried out (parameters having an influence on the value of the displayed actual position).



(View in Step 7 Classic V5.5)



(View in Step 7 TIA)

The following items have to be considered when using this FB:

- Each value has to be written and the values have to have a reasonable relation to each other. Each value is processed, after that the following parameters are written in the order stated below:
 - Direction of rotation (Par. 37) = Direction
 - Position scaling, numerator (Par. 38) = 400
 - Position scaling, denominator (Par. 39) = StepsPerTurn
 - Actual value (Par. 3) = SetPoint

- If (SetPoint > UpperLimit):
Upper mapping end (Par. 41) = SetPoint + (3 x StepsPerTurn)
otherwise:
Upper mapping end (Par. 41) = UpperLimit + (3 x StepsPerTurn)
- Upper limit (Par. 42) = UpperLimit
- Lower limit (Par. 43) = LowerLimit
- The number of steps per revolution "StepsPerTurn" directly results in the value of the parameter "Position scaling, denominator" (Par. 39). Thereby it is assumed that the value of "Position scaling, numerator" (Par. 38) is in delivery state, thus 400.
- Before writing the parameters, the entered values are checked for validity.

Subsequently the conditions and error codes which are displayed if a condition is not satisfied.

Condition	ErrorID	ErrorParameter
StepsPerTurn \geq 1	16#6140	39
StepsPerTurn \leq 10000	16#6140	39
LowerLimit \leq UpperLimit	16#6140	42
(UpperLimit - LowerLimit) / StepsPerTurn \leq a	16#6140	43
Falls SetPoint < LowerLimit: (UpperLimit - SetPoint) / StepsPerTurn \leq a	16#6140	3
Falls SetPoint > UpperLimit: (SetPoint - LowerLimit) / StepsPerTurn \leq a	16#6140	3

Value „a“ depends on the gear variant in use:

Gear variant	PSD401/411, PSD422/432	PSD403/413	PSD426/436	PSD428/438
Gear variant	PSC401/411, PSC422/432	PSC402/412	PSC426/436	PSC428/438
Value a	4026	986	1274	977

- Optionally at the end additionally the written values might be saved permanently. To do this, the input "SaveSettings" has to be set to TRUE before the execution of the FB.
- In case of an error while writing a parameter, the subsequent parameters are not written any more. Also no saving of the values is carried out, if the input "SaveSettings" is set.

Execute

Start of a parametrization process

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

Description:

When issuing a rising edge, a parametrization process with the given values is started. For a new parametrization process, a new rising edge has to be generated. When deasserting the bit, the outputs fall back to their specified default value.

Direction

Direction in which the drive shall turn with larger values (if looking at the output shaft):

0 → CW, 1 → CCW

- Type: INT
- Initial value: 0
- Direction: INPUT

StepsPerTurn

Number of steps per revolution at the output shaft (resolution)

- Type: INT
- Initial value: 0
- Direction: INPUT

LowerLimit

Lower limit

- Type: DINT
- Initial value: 0
- Direction: INPUT

UpperLimit

Upper limit

- Type: DINT
- Initial value: 0
- Direction: INPUT

SetPoint

Value on which the measuring system is referenced (new actual value at the actual position)

- Type: DINT
- Initial value: 0
- Direction: INPUT

SaveSettings

Saving the settings permanently

- Type: BOOL
- Initial value: FALSE
- Direction: INPUT

Active

Bit is set as long as the parametrization process is active

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

The bit is being deasserted as soon as the parametrization has been finished successfully or an error occurred.

Done

Bit is set as soon as the parametrization has been finished successfully

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

The bit is being deasserted when starting a parametrization process.

Error

Bit is set if an error occurred during the execution of the FB

- Type: BOOL
- Default value: FALSE
- Direction: OUTPUT

ErrorID

Error code (see table "ErrorID" in chapter 4)

- Type: WORD
- Default value: 0
- Direction: OUTPUT

Drive errors are not considered during a parametrization.

ErrorParameter

Parameter number that caused the error (in case of an error)

- Type: INT
- Default value: 0
- Direction: OUTPUT

If no error occurred, this value is 0.



INFORMATION!

In case of the S7-300, for the function blocks "MC_Parametrization" and "MC_PosParametrization" the variable of the type "DRIVE_DATA" has to be connected to the input "DRIVE_IN" and to the output "DRIVE_OUT" (in case of the S7-1200 and S7-1500 in the TIA portal the variable is only connected one time to the Input/Output "Drive").