DATA PANEL
POWERED BY MURRELEKTRONIK

Quickstart Guide xBLOCK

Prerequisites:

- DATA PANEL xtremeBLOCK (DP-81000-1-200) module + accessories
- IFM control / display control
- CodeSYS with necessary packages
- DC 12V or 24 V supply

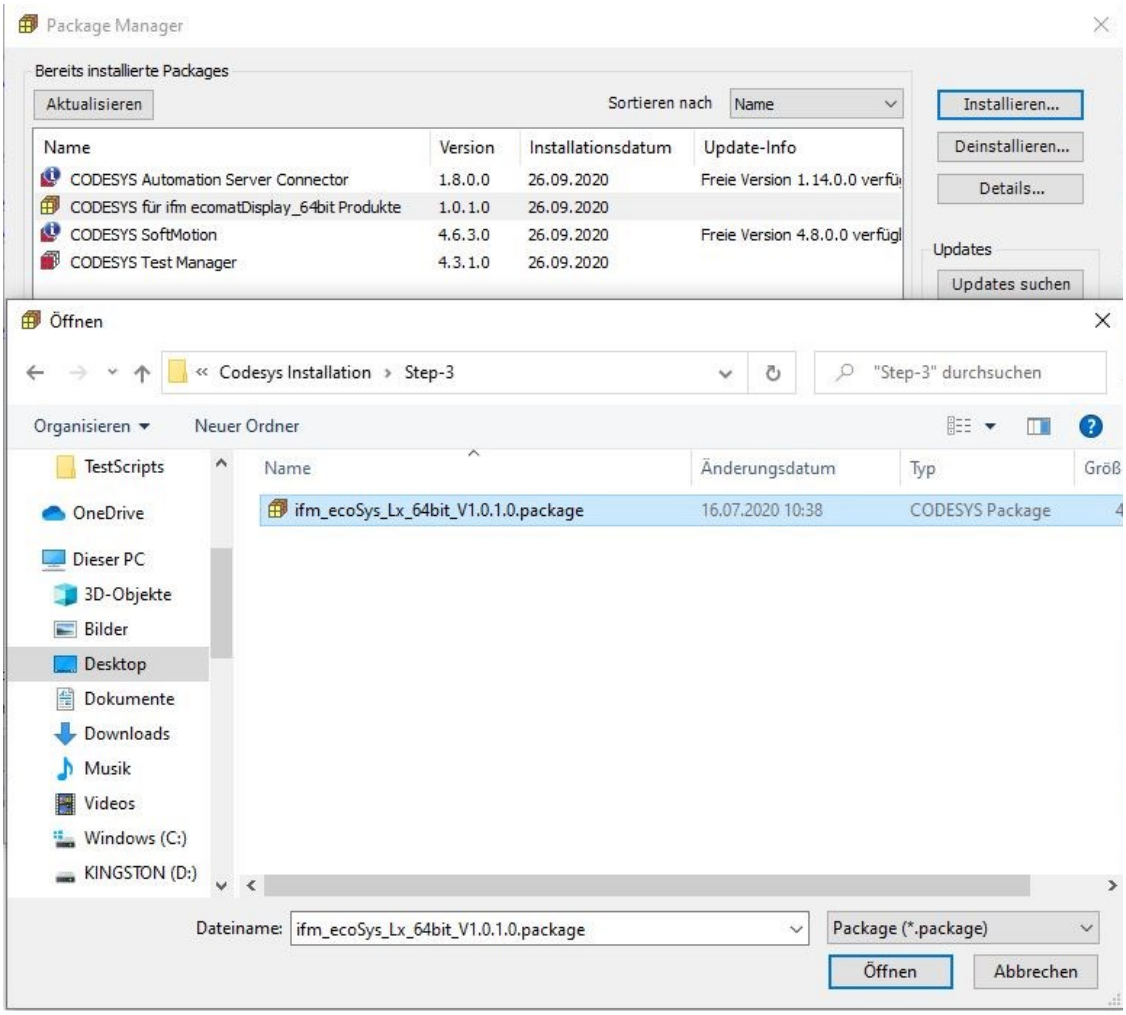
CODESYS PREPARATION

Prepare Codesys

Depending on the type of controller you are working with, the corresponding packages must be installed in Codesys.

The packages for the respective controller are supplied or you can obtain them on the manufacturer's website or in the Codesys Shop. It may be necessary to create an account for the corresponding pages.

- Open Codesys
- In the taskbar at the top via "**Tools -> Package Manager... "** open the package manager
- Right click on **"Install..."** and install the corresponding package
- In the following example the integration of an ifm display control is described



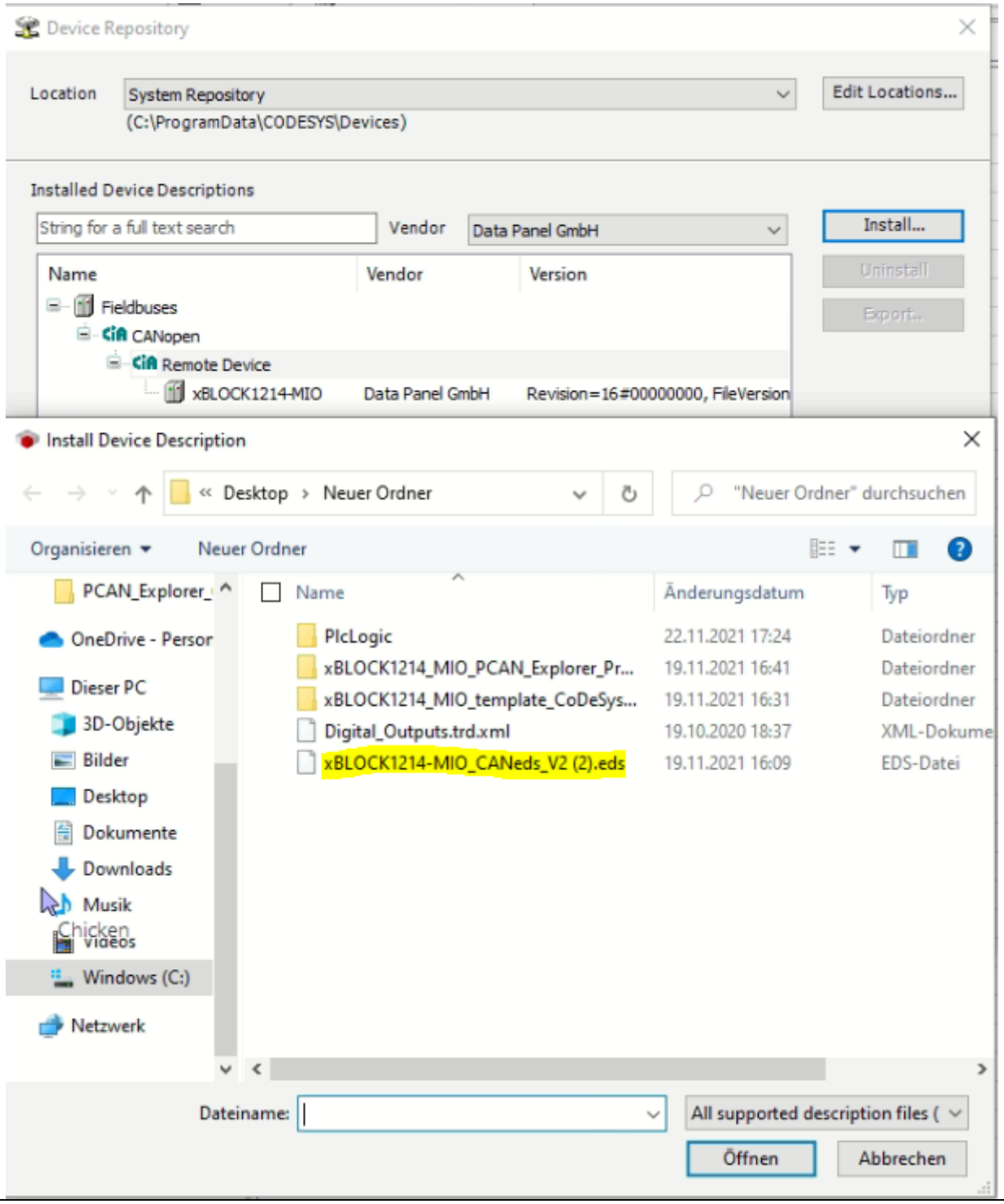
- Double-click or "Open" to install the package (this may take a moment)
- If the installation was successful, a corresponding message appears

CODESYS PREPARATION

Install EDS device file

- Open the device repository via **"Tools -> Device Repository" in the** taskbar at the top.
- Via the button **"Install..."** a new device can be installed
- **Double click** on the desired file, alternatively select and **open** the ***. eds file** manually **.** The device appears in the list of added devices

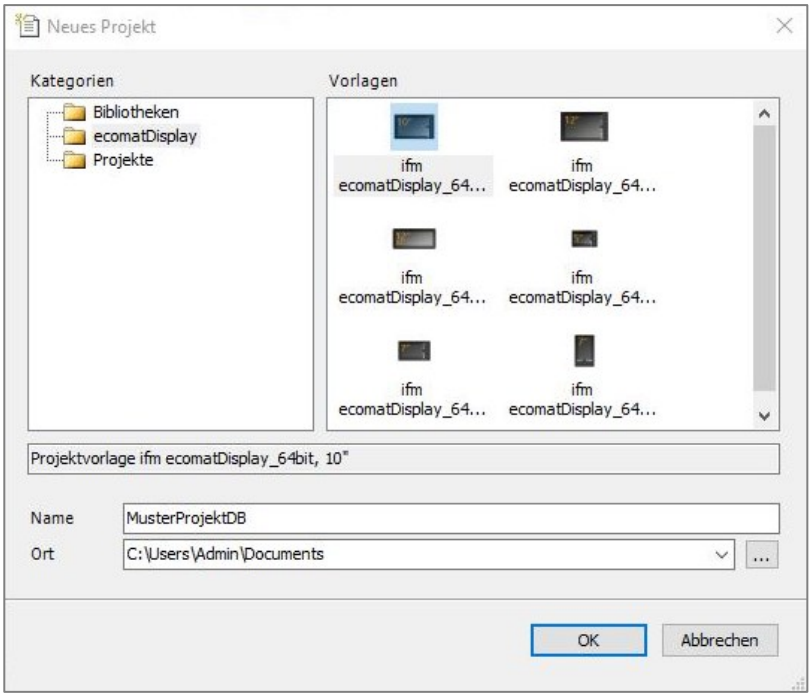
The link below for you to our latest *. eds file:
https://www.data-panel.eu/media/archive/DP-81000-1-200-EDS.zip



CODESYS PREPARATION

Codesys project

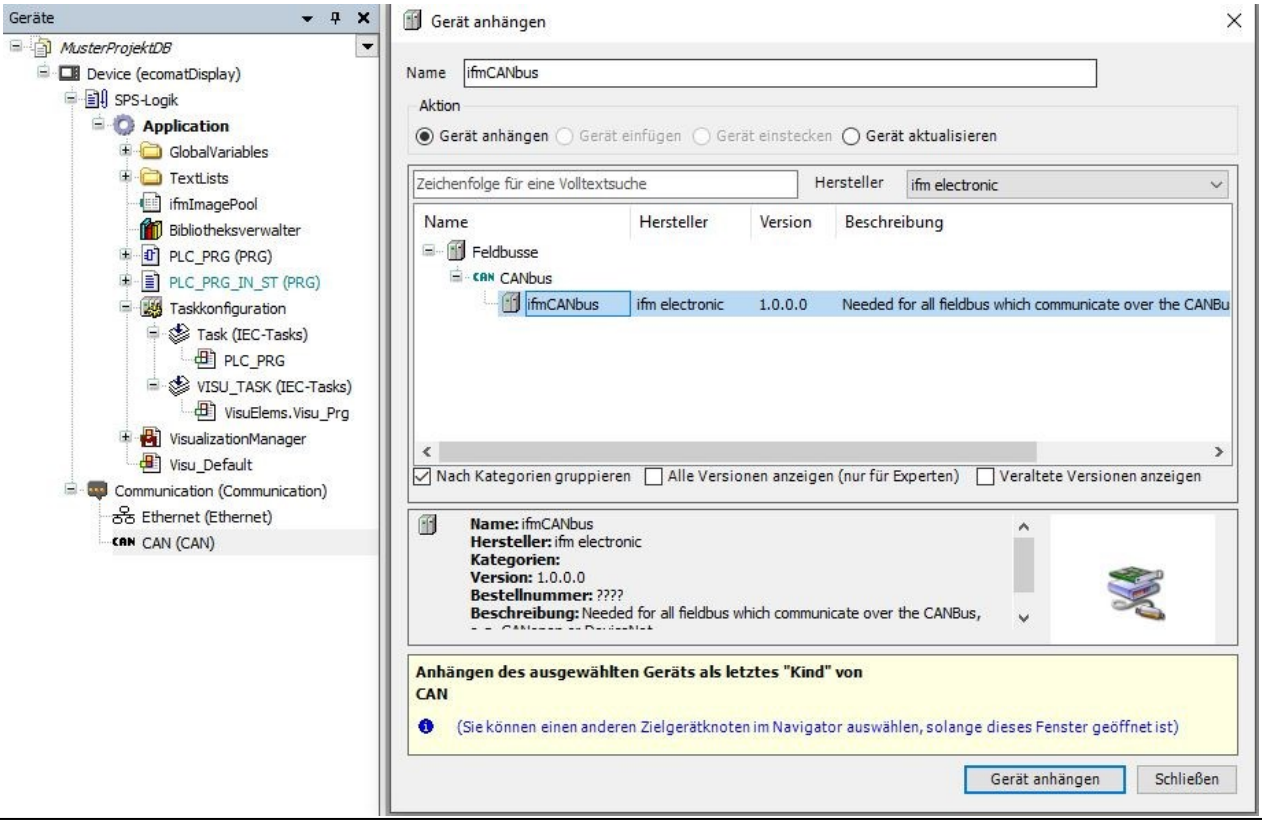
- Open Codesys
- Create a new project via **File -> New** project
- Select the connected controller via the library and confirm with **OK.** This action may take some time.



CAN communication

- Open the communication path and right click on "CAN -> Attach device".
- Select "ifm" under the manufacturer and append the "ifmCANbus
- Close window

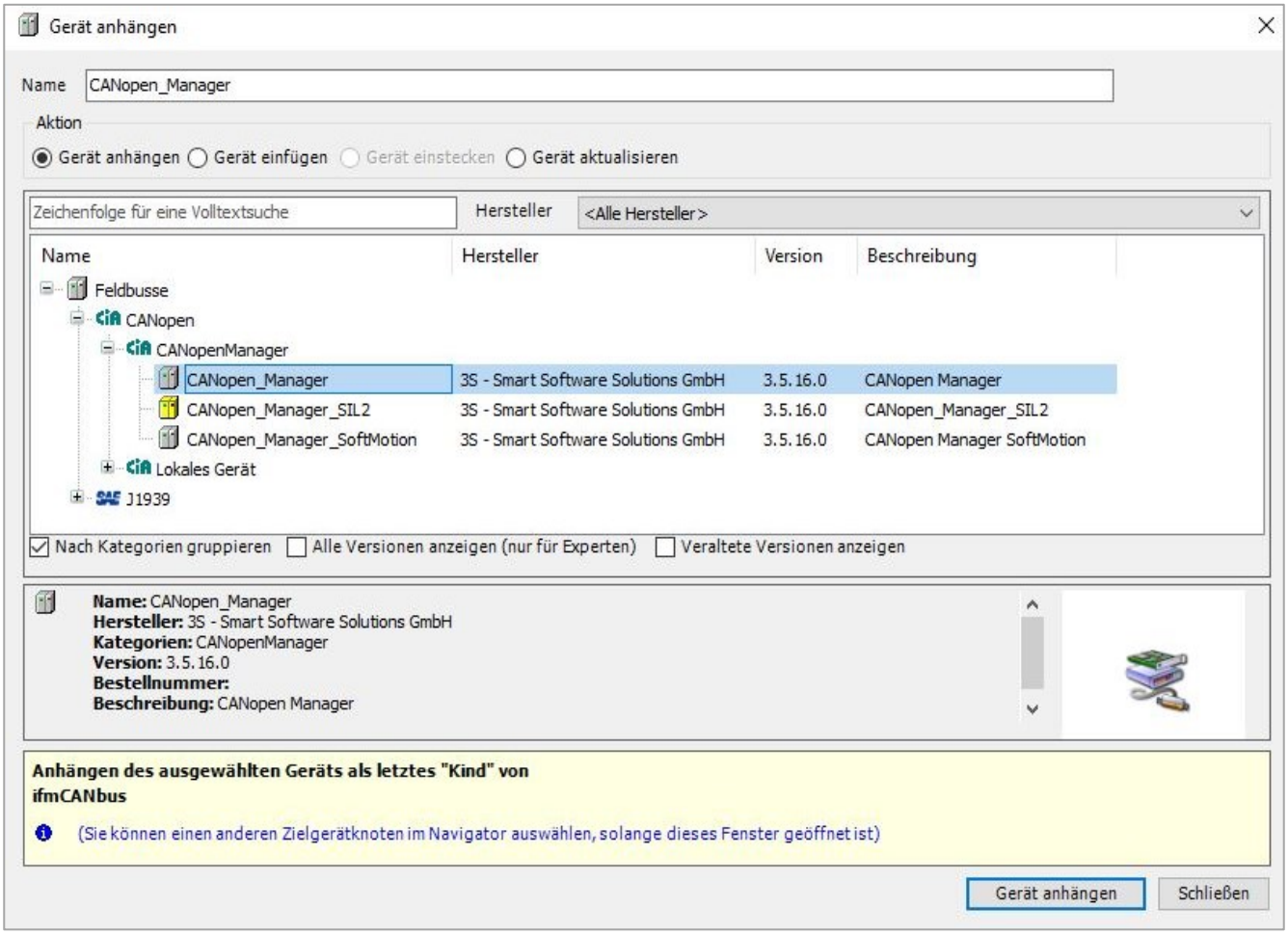
The controller must be CANopen capable. A J 1939 module variant is currently in preparation.



CODESYS PREPARATION

CANopen Manager

- Right click on the just inserted **"ifmCANbus -> Attach device".**
- Change manufacturer filter to **<all manufacturers>.**
- Select the device via **"CANopen -> CANopenManager -> CANopenManager"** and attach it.



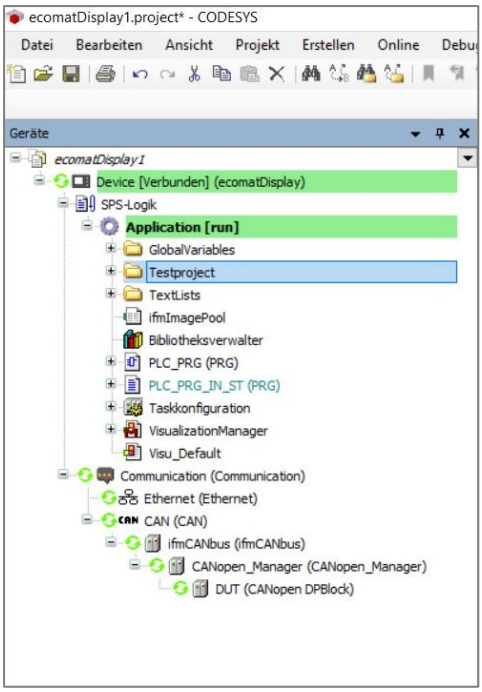
CODESYS PREPARATION

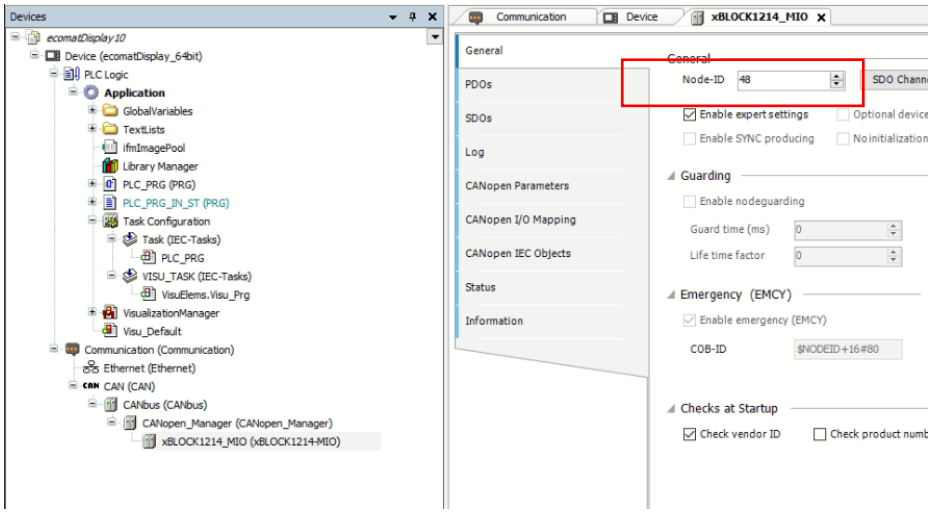
STEP 1

- Append a new device to the CANopen_Manager (right click -> Append device)
- Select the correct module based on the *. eds file and close it

STEP 2

- Open the configuration of the new device and set the node ID.
- Then go online, no errors should appear in the Codesys, the **COM LED** on the module should flash at short intervals



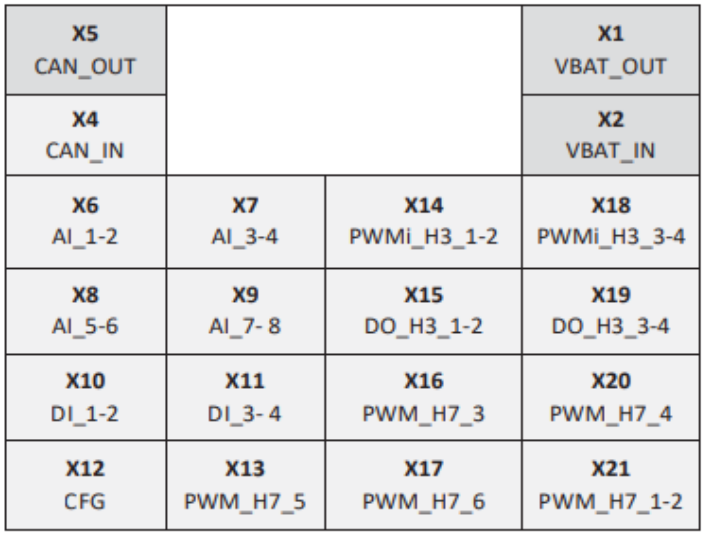


	Jumper Wire
Node ID	Offset	CFG (X12) PINs
48	0	-
49	1	2-3
50	2	1-2
51	3	3-4
52	4	2-3-4
53	5	1-2 & 3-4
54	6	1-4
55	7	2-3 & 1-4
56	8	1-2 & 1-4

ADDRESS

- The module has preset the base node ID 48 (0x30)
- By means of Node-ID connectors an offset can be set at the module. These can be found as accessories in the DP Shop.
- Alternatively, it is possible to set the node-ID via PDO configuration, see manual p. 46 - 47.

CONNECTION OVERVIEW



Attention, if the xtremeBlock is used in combination with the xtremeDB modules, CAN-L and CAN-H must be switched at the CAN IN port of the xtremeBlock.

If there are no further nodes, it is necessary to terminate the CAN line with a 120 Ohm resistor on port X5 CAN_OUT.

PORT no.	PORT Designation	Function
X2	VBAT_IN	Connection 2x supply voltage outputs and GND
X1	VBAT_OUT	Loop-on 1x supply voltage output and GND
X4	CAN_IN	Connection CAN and sensor voltage
X5	CAN_OUT	Loop on CAN and sensor voltage incl. GND
X6	AI_1-2	2x analog input 0 ... 10V / 0 ... 20 mA, alternatively DI
X7	AI_3-4	2x analog input 0 ... 10V / 0 ... 20 mA, alternatively DI
X8	AI_5-6	2x analog input 0 ... 10V / 0 ... 20 mA, alternatively DI
X9	AI_7-8	2x analog input 0 ... 10V / 0 ... 20 mA, alternatively DI
X10	DI_1-2	1x digital input PNP/NPN, frequency measurement 1x digital input PNP, frequency measurement
X11	DI_3-4	2x digital input PNP, frequency measurement
X12	CFG	2x configuration input Node-ID
X13	PWM_H7_5	High-side PWM output with current diagnostics
X14	PWMi_H3_1-2	High-side PWM output with accurate current diagnostics
X15	DO_H3_1-2	Digital output with current diagnostics
X16	PWM_H7_3	High-side PWM output with current diagnostics
X17	PWM_H7_6	High-side PWM output with current diagnostics
X18	PWMi_H3_3-4	High-side PWM output with accurate current diagnostics
X19	DO_H3_3-4	Digital output with current diagnostics
X20	PWM_H7_4	High-side PWM output with current diagnostics
X21	PWM_H_1-2	High-side PWM output with current diagnostics

DI / AI - PORT X6-X9

The module has 8 input pins that can be configured either as analog or digital input. To carry out the parameterization, the corresponding SDO of the individual pins must be configured.

Port / Pin Module	Codesys SDO	Index	Port /Pin	Codesys SDO	Index
X6_2	AI_1	2100	X8_2	AI_5	2104
X6_4	AI_2	2101	X8_4	AI_6	2105
X7_2	AI_3	2102	X9_2	AI_7	2106
X7_4	AI_4	2103	X9_4	AI_8	2107

SubIndex	Description
01	0 = Not used ; 1 = AI_VOLTAGE; 2 = AI_CURRENT; 3 = DI_PNP
32	SENSOR_SUPPLY
3D	FILTER_DEEP
3F	MIN_DEVIATION

- Via the **SubIndex 01 the** type of input is configured
- The sensor supply voltage can be activated via **subindex 32**
- The filter of the input can be configured via the **SubIndex 3D.** The function is only active if the pin is configured as an analog input. With analog input values, the measurement can be distorted by smoking. With the filter, the average of the last n measurements is formed and passed on.
- The measuring tolerance can be set via the **SubIndex 3F.** This function also affects the analog inputs. The measuring tolerance is used to set the change from which a measured value is updated.
  e.g. A voltage of 5 V is to be measured, the value fluctuates between 4.995 V and 5.005 V. This leads to a "noise" of the input value. If a deviation of 15 (mV) is set, the input will not change until it falls below or exceeds the measurement tolerance. The unit for voltage is given in mV and for current in µA.

EXAMPLE X6_1 = DI // X7_1 = AI_V // X7_2 =AI_A

To configure e.g. **pin X6_1** as digital input , **pin X7_1** as analog voltage input and **pin X7_2** as analog current input the following SDO are necessary:

	39	16#1800:16#01	Set and enable COB-ID	16#00000180	32
	40	16#1801:16#01	Disable PDO	16#80000280	32
X6_1	41	16#2100:16#01	AI_1_IO_TYPE (0 = Not Used, 1 = AI_VOLTAGE, 2 = AI_CURRENT, 3 = DI_PNP)	16#1	32
	42	16#2100:16#32	SENSOR_SUPPLY	16#1	16
X7_1	43	16#2102:16#01	AI_3_IO_TYPE (0 = Not Used, 1 = AI_VOLTAGE, 2 = AI_CURRENT, 3 = DI_PNP)	16#1	32
	44	16#2102:16#32	SENSOR_SUPPLY	16#1	16
	45	16#2102:16#3D	FILTER_DEEP	16#8	32
	46	16#2100:16#3F	MIN_DEVIATION	16#5	16
X7_2	47	16#2103:16#01	AI_4_IO_TYPE (0 = Not Used, 1 = AI_VOLTAGE, 2 = AI_CURRENT, 3 = DI_PNP)	16#2	32
	48	16#2103:16#32	SENSOR_SUPPLY	16#1	16
	49	16#2103:16#3D	FILTER_DEEP	16#8	32
	50	16#2103:16#3F	MIN_DEVIATION	16#10	16

Attention, the SDO pin configuration must be configured after the SDO "Disable PDO" (in this case SDO 1801), otherwise the module remains in the default configuration.

The variable **DIGITAL_INPUT_AI_1_AI_8** activates the configured digital inputs of the module. In this case **bit 0** which stands for **X6_1.**

DIGITAL_INPUT_AI_1_AI_8	%IB0	USINT	1
Bit0	%IX0.0	BOOL	TRUE
Bit1	%IX0.1	BOOL	FALSE
Bit2	%IX0.2	BOOL	FALSE
Bit3	%IX0.3	BOOL	FALSE
Bit4	%IX0.4	BOOL	FALSE
Bit5	%IX0.5	BOOL	FALSE
Bit6	%IX0.6	BOOL	FALSE
Bit7	%IX0.7	BOOL	FALSE
DIGITAL_INPUT_DI_P_1_4_PWMi_H3_1_4	%IB1	USINT	0
DIGITAL_INPUT_PWM_H7_1_6_DO_H3_1_2	%IB2	USINT	0
DIGITAL_INPUT_DO_H3_3_4	%IB3	USINT	0
Analog_INPUT_AI_1	%IW2	INT	0
Analog_INPUT_AI_2	%IW3	INT	59
Analog_INPUT_AI_3	%IW4	INT	167

The variable **Analog_INPUT_AI_3** activates the configured analog input. Which in this case applies to pin **X7_1.**

It is recommended to deactivate unused pins via the SDO configuration. This has a significant positive effect on the CAN bus load.

DI - PORT X10-X11

The module has slots that can be configured for **digital inputs.** In addition, these can also be configured as **frequency or encoder inputs.** To carry out the parameterization, the corresponding SDO of the individual pins should be configured.

Port / Pin Module	Codesys SDO	Index	Port /Pin	Codesys SDO	Index
X10_2	DI_P_1	2108	X11_2	DI_P_3	210A
X10_4	DI_P_2	2109	X11_4	DI_P_4	210B

SubIndex	Description
01	0 = Not used ; 3 = DI_PNP; 4 = FI_PNP; 1A = ENC_PNP
32	SENSOR_SUPPLY
3B	TIMEOUT_TIME
3E	GATE_TIME
44	RESOLUTION

X9_1 can additionally be configured as minus-switching NPN (digital or frequency) input

For the acquisition of encoder signals usually two inputs are necessary

- **SubIndex 01 is** used to configure the type of input. FI stands for a frequency-dependent input
- The sensor supply voltage can be activated via **subindex 32**
- The smallest detectable pulse duration can be determined via the **SubIndex 3F.** Default is 100ms. If no pulse is received for 100 ms, the I_DIRECTION bit = 0 This function applies to frequency and encoder inputs.
- The "gate time" can be set via the **SubIndex 3E.** This function concerns frequency measurement at digital inputs. The gate time is the period in which the pulses are counted. The values of the frequency and periodic time are determined via this procedure
- The resolution at the encoder input can be set via **subindex 44.** This function concerns the encoder input

EXAMPLE X10_2 NPN X10_4 PNP

To configure e.g. **pin X10_2** as digital input NPN and **pin X10_4** as digital input PNP the following SDO are necessary:

	69	16#1802:16#01	Set and enable COB-ID	16#00000380	32
	70	16#1803:16#01	Disable PDO	16#80000480	32
X10_2	71	16#2108:16#01	DI_P_1_IO_TYPE (0 = Not Used, 3 = DI_PNP, 4 = FI_PNP, 5 = DI_NPN, D = FI_N...	16#5	32
	72	16#2108:16#32	SENSOR_SUPPLY	16#1	16
X10_4	73	16#2109:16#01	DI_P_2_IO_TYPE (0 = Not Used, 3 = DI_PNP, 4 = FI_PNP, 1A = ENC_PNP)	16#3	32
	74	16#2109:16#32	SENSOR_SUPPLY	16#1	16

Attention, the SDO pin configuration must be configured mandatory after the SDO "Disable PDO" In this case SDO 1803, otherwise the module remains in the default configuration.

The variable **DIGITAL_INPUT_DI_P_1_4_PWMi_H3_1_4 Bit0** activates the configured digital inputs of the module. In this case the **Bit0 stands for the** configured pin **X10_2 NPN** and the **Bit1** stands for the configured pin **X10_4 PNP**

	DIGITAL_INPUT_DI_P_1_4_PWMi_H3_1_4	%IB1	USINT	3
X10_4	Bit0	%IX1.0	BOOL	TRUE
X10_2	Bit1	%IX1.1	BOOL	TRUE
X11_4	Bit2	%IX1.2	BOOL	FALSE
X11_2	Bit3	%IX1.3	BOOL	FALSE
	Bit4	%IX1.4	BOOL	FALSE
	Bit5	%IX1.5	BOOL	FALSE
	Bit6	%IX1.6	BOOL	FALSE
	Bit7	%IX1.7	BOOL	FALSE
	DIGITAL_INPUT_PWM_H7_1_6_DO_H3_1_2	%IB2	USINT	0

X10_4 was configured as NPN input, here the signal changes to FALSE if the connected sensor is attenuated

PWM / DO (DI) - PORT X13 & X16+X17 & X20+X21

The module has **five PWM pins** that can optionally be used as **digital inputs.** Alternatively, these PWM pins can also be configured as **digital outputs** with 3 or 7 amps current carrying capacity.

Port / Pin Module	Codesys SDO	Index	Port /Pin	Codesys SDO	Index
X13_2 & 4 (mirrored)	ANALOG_OUTPUT_U16_ PWM_H7_5	2114	X17_2 & 4 (mirrored)	ANALOG_OUTPUT_U16_ PWM_H7_6	2115
X16_2 & 4 (mirrored)	ANALOG_OUTPUT_U16_ PWM_H7_3	2112	X20_2 & 4 (mirrored)	ANALOG_OUTPUT_U16_ PWM_H7_4	2113
X21_2	ANALOG_OUTPUT_U16_ PWM_H7_1	2110	X21_4	ANALOG_OUTPUT_U16_ PWM_H7_2	2111

There are two PWM outputs on port X20

SubIndex	Description
01	PWM_H7_1_IO_TYPE (0 = Not Used, 3 = DI_PNP, 5 = DI_NPN, 6 = PWMO_HS3, 7 = DO_HS3, B = PWMO_HS7, C = DO_HS7)
1E	O_DIGITAL
1F	O_DUTYCYCLE
32	SENSOR_SUPPLY
33	PWM_FREQUENCY
34	DITHER_FREQUENCY
35	DITHER_AMPLITUDE
39	MAX_CURRENT
3A	OVERCURRENT_TIME
3D	FILTER_DEEP
3F	MIN_DEVIATION
40	MIN_CURRENT
41	OPENCIRCUIT_DETECTION

PWM / DO (DI) - PORT X13 & X16+X17 & X20+X21

- Via the **SubIndex 01 the** type of input is configured. Via the abbreviation HS the maximum current can be defined
- The output can be switched on via **SubIndex 1E**, provided that the pin is configured as a digital output. Has no meaning for other configurations
- **SubIndex 1F** can be used to set the keying behavior of the output, provided it has been configured as a digital output. Has no meaning for other configurations
- The sensor supply voltage can be activated via **subindex 32**
- Via **SubIndex 33 the** frequency of the PWM output can be set.
- The Dither_Frequency can be set via **SubIndex 34.** This value is superimposed on the piston of a hydraulic valve in order to keep it always minimally in motion at standstill. Without this value, the valve moves to a specified position. Here, the sliding friction changes to static friction, and when the valve is "restarted", stick-and-slip behavior occurs because the static friction is greater than the sliding friction. This behavior can be influenced by a dither frequency.
- The dither amplitude can be set via **subindex 35**
- Via **SubIndex 39 the** maximum current of the pin can be set
- The time after which the output switches off when an overcurrent is present can be set via the **SubIndex 3A.**
- The filter of the input can be configured via the **SubIndex 3D.** The function is only active if the pin is configured as an analog input. With analog values, the measurement can be distorted by noise. With the filter, the average of the last **n** measurements is formed and passed on
- The measuring tolerance can be set via the **SubIndex 3F.** This function also affects the analog inputs. The measuring tolerance is used to set the change from which a measured value is updated.
  e.g. A voltage of 5 V is to be measured, the value fluctuates between 4.995 V and 5.005 V. This leads to a "noise" of the input value. If a deviation of 15 (mV) is set, the input will not change until it falls below or exceeds the measurement tolerance. The unit for voltage is given in mV and for current in µA.
- The minimum current can be set via **subindex 40.** If less than the current stored here flows, a cable break is detected. Default 200 mA for H3 and 500 mA for H7
- The cable break detection can be activated or deactivated via the **SubIndex 41.** Additionally it can be configured whether the detection is only checked at boot time or cyclically during operation.

EXAMPLE X13_2 PWM / X16_2 PWM / X_21_2 DO

To configure e.g. **pin X13_2** as PWM 7A , **pin X16_2** as PWM 3A and **pin X21_2** as **digital output the** following SDO are necessary:

	15	16#1401:16#01	Set and enable COB-ID	16#00000330
	16	16#1402:16#01	Disable PDO	16#80000430
X21_2	17	16#2110:16#01	PWM_H7_1_IO_TYPE (0 = Not Used, 3 = DI_PNP, 5 = DI_NPN, 6 = PWMO_HS3, 7 = DO_HS3, B = P...	16#7
	18	16#2110:16#39	MAX_CURRENT	16#0
X16_2	19	16#2112:16#01	PWM_H7_3_IO_TYPE (0 = Not Used, 3 = DI_PNP, 5 = DI_NPN, 6 = PWMO_HS3, 7 = DO_HS3, B = P...	16#6
X13_2	20	16#2114:16#01	PWM_H7_5_IO_TYPE (0 = Not Used, 3 = DI_PNP, 5 = DI_NPN, 6 = PWMO_HS3, 7 = DO_HS3, B = P...	16#C

Attention, the SDO pin configuration must be configured after the SDO "Disable PDO" In this case SDO 1402, otherwise the module remains in the default configuration.

The variable **DIGITAL_OUTPUT_PWM_H3_1_4_PWM_H7_1_4 bit 4** activates the configured **pin X21_2**. If this bit is set to "**True**", the output at pin **X21_2 is** switched on.

The variable **ANALOG_OUTPUT_U16_PWM_H7_3** activates the configured pin **X16_2**, e.g. if the value "**500" is** stored here, the output is switched with a frequency of **500 Hz.**

The variable **ANALOG_OUTPUT_U16_PWM_H7_5** activates the configured pin **X13_2**, e.g. if the value "**1000" is** stored here, the output is switched with a frequency of **1,000 Hz.**

Variable	Mapping	Channel	Address	Type	Current Value
		DIGITAL_OUTPUT_PWMi_H3_1_4_PWM_H7_1_4	%QB0	USINT	16
		Bit0	%QX0.0	BOOL	FALSE
		Bit1	%QX0.1	BOOL	FALSE
		Bit2	%QX0.2	BOOL	FALSE
		Bit3	%QX0.3	BOOL	FALSE
X21_2		Bit4	%QX0.4	BOOL	TRUE
X21_4		Bit5	%QX0.5	BOOL	FALSE
X16_2		Bit6	%QX0.6	BOOL	FALSE
X20_2		Bit7	%QX0.7	BOOL	FALSE
		DIGITAL_OUTPUT_PWM_H7_5_6_DO_H3_1_4	%QB1	USINT	0
		Bit0	%QX1.0	BOOL	FALSE
		Bit1	%QX1.1	BOOL	FALSE
		Bit2	%QX1.2	BOOL	FALSE
		Bit3	%QX1.3	BOOL	FALSE
		Bit4	%QX1.4	BOOL	FALSE
		Bit5	%QX1.5	BOOL	FALSE
		Bit6	%QX1.6	BOOL	FALSE
		Bit7	%QX1.7	BOOL	FALSE
		ANALOG_OUTPUT_U16_PWMi_H3_1	%QW1	INT	0
		ANALOG_OUTPUT_U16_PWMi_H3_2	%QW2	INT	0
		ANALOG_OUTPUT_U16_PWMi_H3_3	%QW3	INT	0
		ANALOG_OUTPUT_U16_PWMi_H3_4	%QW4	INT	0
		ANALOG_OUTPUT_U16_PWM_H7_1	%QW5	INT	0
		ANALOG_OUTPUT_U16_PWM_H7_2	%QW6	INT	0
X16_2 & X16_4		ANALOG_OUTPUT_U16_PWM_H7_3	%QW7	INT	500
		ANALOG_OUTPUT_U16_PWM_H7_4	%QW8	INT	0
X13_2 & X13_4		ANALOG_OUTPUT_U16_PWM_H7_5	%QW9	INT	1000
		ANALOG_OUTPUT_U16_PWM_H7_6	%QW10	INT	0
		DIGITAL_INPUT_AI_1_AI_8	%IB0	USINT	0
		DIGITAL_INPUT_DI_P_1_4_PWMi_H3_1_4	%IB1	USINT	0

DO / PWM / (DI) - PORT X15 + X19

The module has **two PWM pins** that can optionally also be used as **digital outputs.** In addition, these pins can also be configured as **PWM outputs** with 3 amps or as **digital inputs.**

Port / Pin Module	Codesys SDO	Index	Port /Pin	Codesys SDO	Index
X15_2	DO_H3_1	2116	X19_2	DO_H3_3	2118
X15_4	DO_H3_2	2117	X19_4	DO_H3_4	2119

SubIndex	Description
01	PWM_H7_1_IO_TYPE (0 = Not Used, 3 = DI_PNP, 5 = DI_NPN, 7 = DO_HS3)
1E	O_DIGITAL
32	SENSOR_SUPPLY
39	MAX_CURRENT
3A	OVERCURRENT_TIME
3D	FILTER_DEEP
3F	MIN_DEVIATION
40	MIN_CURRENT
41	OPENCIRCUIT_DETECTION

- Via the **SubIndex 01 the** type of input is configured. Via the abbreviation HS the maximum current is specified
- The output can be switched on via **SubIndex 1E**, provided that the pin is configured as a digital output. Has no meaning for other configurations
- **SubIndex 1F** can be used to set the keying behavior of the output, provided it has been configured as a digital output. Has no meaning for other configurations
- The sensor supply voltage can be activated via **subindex 32**
- Via **SubIndex 33 the** frequency of the PWM output can be set.
- Via **SubIndex 39 the** maximum current of the pin can be set

DO / PWM / (DI) - PORT X15 + X19

- The time after which the output switches off when an overcurrent is present can be set via **SubIndex 3A.**
- The filter of the input can be configured via the **SubIndex 3D.** The function is only active if the pin is configured as an analog input. With analog values, the measurement can be distorted by noise. With the filter, the average of the last X measurements is formed and passed on.
- The measuring tolerance can be set via the **SubIndex 3F.** This function also affects the analog inputs. The measuring tolerance is used to set the change from which a measured value is updated.
  e.g. A voltage of 5 V is to be measured, the value fluctuates between 4.995 V and 5.005 V. This leads to a "noise" of the input value. If a deviation of 15 (mV) is set, the input will not change until it falls below or exceeds the measurement tolerance. The unit for voltage is given in mV and for current in µA.
- The minimum current can be set via **subindex 40.** If less than the current stored here flows, a cable break is detected. Default 200 mA for H3 and 500 mA for H7.
- The cable break detection can be activated or deactivated via the **SubIndex 41.** In addition, it can be configured whether the detection is only checked at startup or cyclically during operation.

EXAMPLE X15_2 DO

To configure e.g. **pin X15_2** as digital output with 3 A the following SDO are necessary:

	2	16#1400:16#01	Disable PDO	16#80000230
X15_2	3	16#2116:16#01	DO_H3_1_IO_TYPE (0 = Not Used, 3 = DI_PNP, 5 = DI_NPN, 7 = DO_HS3)	16#7

Attention, the SDO pin configuration must be configured after the SDO "Disable PDO" In this case SDO 1400, otherwise the module remains in the default configuration.

The variable **DIGITAL_OUTPUT_PWM_H7_1_6_DO_H3_1_4 bit 2** activates the configured **pin X15_2**. If this bit is set to "**True**", the output at pin **X15_2 is** switched on.

Variable	Mapping	Channel	Address	Type	Current Value
		DIGITAL_OUTPUT_PWMi_H3_1_4_PWM_H7_1_4	%QB0	USINT	0
		DIGITAL_OUTPUT_PWM_H7_5_6_DO_H3_1_4	%QB1	USINT	4
		Bit0	%QX1.0	BOOL	FALSE
		Bit1	%QX1.1	BOOL	FALSE
X15_2		Bit2	%QX1.2	BOOL	TRUE
X15_4		Bit3	%QX1.3	BOOL	FALSE
X19_2		Bit4	%QX1.4	BOOL	FALSE
X19_4		Bit5	%QX1.5	BOOL	FALSE
		Bit6	%QX1.6	BOOL	FALSE
		Bit7	%QX1.7	BOOL	FALSE

DO / PWMI / (DI) - PORT X14 + X18

The module has **four PWM pins** that can optionally also be used as **digital inputs.** In addition, these PWM pins can also be configured as **digital outputs** with 3 or 7 amps.

Port / Pin Module	Codesys SDO	Index	Port /Pin	Codesys SDO	Index
X14_2	PWMi_H3_1	210C	X18_2	PWMi_H3_3	210E
X14_4	PWMi_H3_2	210D	X18_4	PWMi_H3_4	210F

CPWMO corresponds to PWMi

SubIndex	Description
01	PWMi_H3_1_IO_TYPE (0 = Not Used, 3 = DI_PNP, 5 = DI_NPN, 6 = PWMO_HS3, 7 = DO_HS3, A = CPWMO_HS3)
1E	O_DIGITAL
1F	O_DUTYCYCLE
20	O_HCURRENT
32	SENSOR_SUPPLY
33	PWM_FREQUENCY
34	DITHER_FREQUENCY
35	DITHER_AMPLITUDE
36	CURRENT_CONTROL_P
37	CURRENT_CONTROL_I
38	CURRENT_CONTROL_E
39	MAX_CURRENT
3A	OVERCURRENT_TIME
3D	FILTER_DEEP
3F	MIN_DEVIATION
40	MIN_CURRENT
41	OPENCIRCUIT_DETECTION

DO / PWMi / (DI) - PORT X14 + X18

- Via the **SubIndex 01 the** type of input is configured. Via the abbreviation HS the maximum current can be defined. CPWMO stands for PWMi
- The output can be switched on via **SubIndex 1E**, provided that the pin is configured as a digital output. Has no meaning for other configurations
- **SubIndex 1F** can be used to set the keying behavior of the output, provided it has been configured as a digital output. Has no meaning for other configurations
- Via **SubIndex 20 the** maximum current for the PWMi can be set
- The sensor supply voltage can be activated via **subindex 32**
- Via **SubIndex 33 the** frequency of the PWM output can be set.
- The Dither_Frequency can be set via **SubIndex 34.**
- The dither amplitude can be set via **subindex 35**
- Via **subindex 36 the proportional part of** the current controlled PWMi pin can be set
- Via **subindex 37 the integral part of** the current controlled PWMi pin can be set
- Via **subindex 38 the differential part of** the current controlled PWMi pin can be set
- Via **SubIndex 39 the** maximum current of the pin can be set
- The time after which the output switches off when an overcurrent is present can be set via the **SubIndex 3A.**
- The filter of the input can be configured via the **SubIndex 3D.**
- The deviation can be set via the **SubIndex 3F.**
- The minimum current can be set via **subindex 40.** If less than the current stored here flows, a cable break is detected. Default 200 mA for H3 and 500 mA for H7
- Via **SubIndex 41 the** cable break detection can be activated or deactivated. In addition, it can be configured whether the detection is only checked at startup or cyclically during operation.

EXAMPLE X14_2 PWMI & X14_4 PWMI

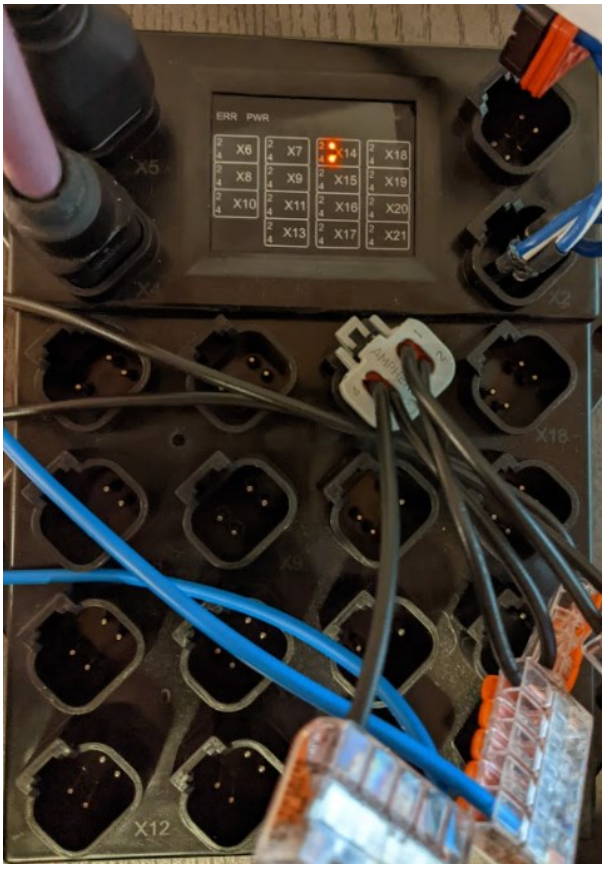
To configure e.g. the two pins **X14_2 & X14_4** as PWMi pin the following SDO are necessary:

	9	16#1401:16#01	Disable PDO	16#80000330
X14_2	10	16#210C:16#01	PWMi_H3_1_IO_TYPE (0 = Not Used, 3 = DI_PNP, 5 = DI_NPN, 6 = PWMO_HS3, 7 = DO_HS3, A = CPWMO_HS3)	16#A
X14_4	11	16#210D:16#01	PWMi_H3_2_IO_TYPE (0 = Not Used, 3 = DI_PNP, 5 = DI_NPN, 6 = PWMO_HS3, 7 = DO_HS3, A = CPWMO_HS3)	16#A

Attention, the SDO pin configuration must be configured after the SDO "Disable PDO" In this case SDO 1401, otherwise the module remains in the default configuration.

The variable **ANALOG_OUTPUT_U16_PWMi_H3_1 takes care of the** configured **pin X14_2**. The variable **ANALOG_OUTPUT_U16_PWMi_H3_2 takes** care of the **pin X14_4**

Variable	Mapping	Channel	Address	Type	Current Value
		DIGITAL_OUTPUT_PWMi_H3_1_4_PWM_H7_1_4	%QB0	USINT	0
		DIGITAL_OUTPUT_PWM_H7_5_6_DO_H3_1_4	%QB1	USINT	0
	X14_2	ANALOG_OUTPUT_U16_PWMi_H3_1	%QW1	INT	200
	X14_4	ANALOG_OUTPUT_U16_PWMi_H3_2	%QW2	INT	200
	X18_2	ANALOG_OUTPUT_U16_PWMi_H3_3	%QW3	INT	0
	X18_4	ANALOG_OUTPUT_U16_PWMi_H3_4	%QW4	INT	200
		ANALOG_OUTPUT_U16_PWM_H7_1	%QW5	INT	0
		ANALOG_OUTPUT_U16_PWM_H7_2	%QW6	INT	0
		ANALOG_OUTPUT_U16_PWM_H7_3	%QW7	INT	0
		ANALOG_OUTPUT_U16_PWM_H7_4	%QW8	INT	0



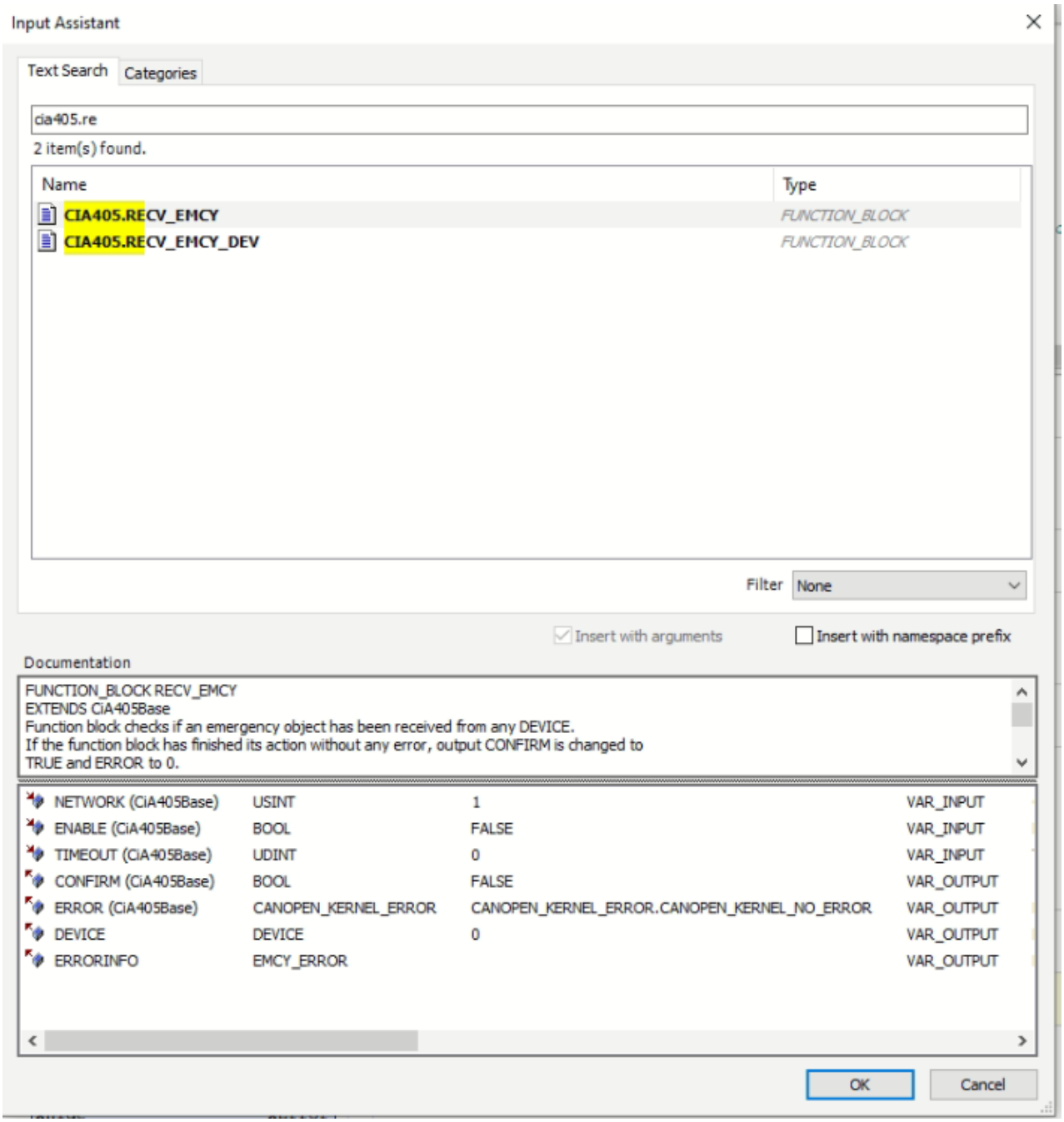
GENERAL DIAGNOSIS

The module brings a variety of general diagnostics that can be read back via the Emergency Messages reported back from the CANopen CIA405 specification. An overview of the individual messages is shown in the following table

Code	Description
0x0000	No error or error reset
0x1000	Generic error
0x2300	Total current is too high
0x3100	Voltage outside the required tolerance
0x4200 0x8110	Device temperature too high CAN data overrun (objects lost)
0x8130	Life Guard Error or Heartbeat Error
0x8140	Recoverd from Bus-Off
0x8210	Processing error due to incorrect length of the PDOs
0x8220	PDO length exceeded
0xff00 0xff01	Configuration error on the device IO pin Overvoltage
0xff02	IO pin overcurrent
0xff03	IO pin Supplyfault
0xff04 0xff05	Reserved IO pin open load
0xff06	IO pin timeout
0xff07	IO pin CC_Unlock

GENERAL DIAGNOSIS

The general diagnostics can be read out via the Emergency Messages. A standardized block of the CIA405 is used for this purpose. In this case the block CIA405.RECV_EMCY



GENERAL DIAGNOSIS

First it is necessary to declare the block:

```
VAR
  xEnable : BOOL; // Enables the block
  xEMCYPending : BOOL;
  EMCY : CIA405.EMCY_ERROR; //contains the CAN error information
  NodeID : CIA405.DEVICE; //Node ID of the CAN node
  RecvEMCY : CIA405.RECV_EMCY; // Returned error message
  EMCY_array_string : ARRAY[0..1000] OF STRING; //EMCY
  i : INT := 0; // Auxiliary variable for programming
END_VAR
```

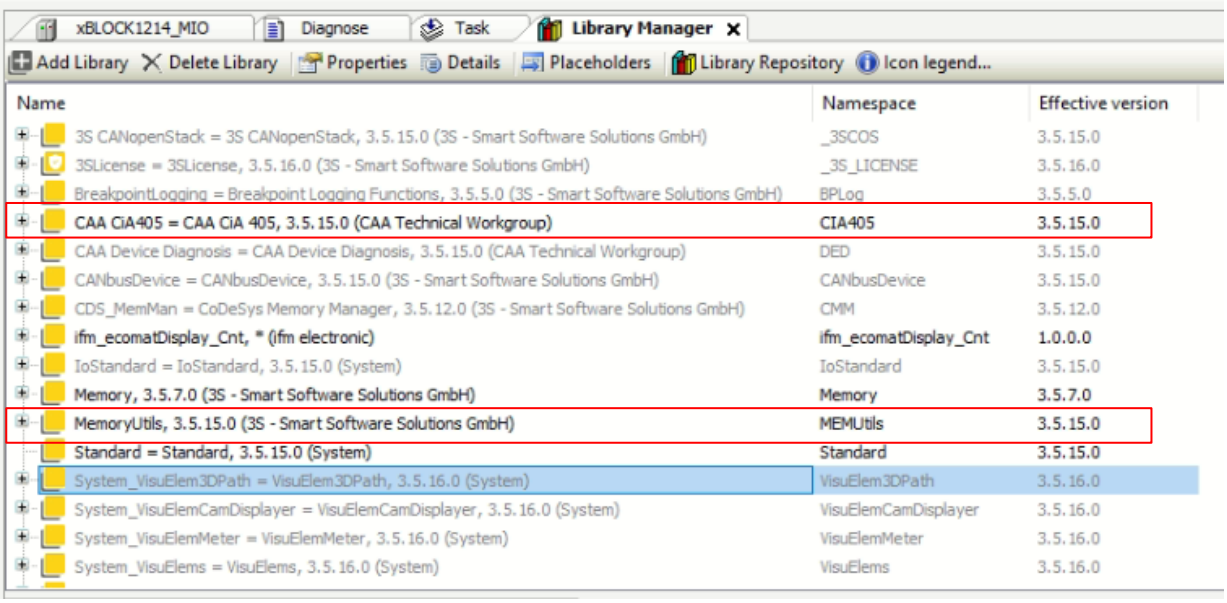
Afterwards a small logic application has to be programmed to read out the values

```
xEnable := TRUE; // Activates the block
IF RecvEMCY.CONFIRM = TRUE THEN // Error message is present
  IF RecvEMCY.DEVICE <> 0 THEN // Error message is present at device with NodeID
    xEMCYPending := TRUE; // An error message is present
    EMCY := RecvEMCY.ERRORINFO;
    NodeID := RecvEMCY.DEVICE;
    i :=i + 1;
  ELSE
    xEMCYPending := FALSE; // No error message is present
  END_IF
  CASE EMCY.EMCY_ERROR_CODE OF // Assign the error message
    16#0000: EMCY_array_string[i] := '0x0000 : No Error or Error Reset (WR)';
...  Insert error messages from the table
...
    16#FF07: EMCY_array_string[i] := '0xff07 : IO port CC_UNLOCK';
  END_CASE;
  xEnable := FALSE; // Block is activated
ELSIF RecvEMCY.ERROR <> CIA405.CANOPEN_KERNEL_ERROR.CANOPEN_KERNEL_NO_ERROR THEN
  // an error occurred while processing RecvEMCY, e.g. an incorrect NodeID or NetworkID.
  xEnable := FALSE;
END_IF
RecvEMCY( ENABLE := xEnable,
        NETWORK := CANbus.NetId + 1, //CODESYS NetId starts with 0 (the number entered in the CANbus configurator); CiA405 NETWORK with 1; That is the reason why we have to increment it here.
        TIMEOUT := 0 // no timeout required for RecvEMCY because it is processed locally.
);
IF i >= 999 THEN // Utility program for incrementing
  i:=0;
END_IF;
```

PORT-BASED DIAGNOSES

Additionally it is also possible to read out the status or e.g. current directly of a port. For this purpose, the data is read directly from an SDO. For this there is a standardized module of the **canOpen CIA405 specification (CAA CiA405 library)** additionally the read back value should be interpreted or converted. For this the Codesys library **MemoryUtils is** used. In the following example the current of a **PWMi port (X14 P4) is** read back.

The current values of each port are always returned in **subindex 13 in** this case **index 210D subindex 13.** In a first step the necessary variables and blocks must be declared.




```
PROGRAM Diagnosis
VAR
   xEnable : BOOL; // This variable activates the block, which is automatically set to FALSE again after the value has been read out
   dwValue : DWORD; //Read back value
   dwAbortCode : DWORD; //Error value if reading is not possible
   dwStatusValue : DWORD;
   dwStatusAbortCode : DWORD;
   SPWR_1_SDORead : CIA405.SDO_READ4; //Block for reading the SDO
END_VAR
```

PORT-BASED DIAGNOSES

If the variable **xEnable** is set to "TRUE", the following program code is executed. Via the block **SPWR_1_SDOREAD the** SDO with the **index 210D subindex 13 is** read. Then this value is "copied" and converted. Afterwards the result can be found in the variable **dwValue.**

```
IF SPWR_1_SDORead.CONFIRM THEN
   //Value read back without error, copy data from block
   MEMUtils.MemCpy(pbyDest := ADR(dwValue), pbySrc := ADR(SPWR_1_SDORead.DATA[1]), dwSize :=
SIZEOF(dwValue));
   //On big-endian systems: dwValue must be changed to CANopen byte order (= little endian).
   MEMUtils.SwapLocalToIntel(pAddress := ADR(dwValue), iSize := TO_INT(SIZEOF(dwValue)); //Function
swaps only on big-endian systems
   xEnable := FALSE;
ELSIF SPWR_1_SDORead.ERROR <> CiA405.CANOPEN_KERNEL_ERROR.CANOPEN_KERNEL_NO_ERROR THEN
   // Error during readout
   IF SPWR_1_SDORead.ERROR = CIA405.CANOPEN_KERNEL_ERROR.CANOPEN_KERNEL_OTHER_ERROR THEN
      MEMUtils.MemCpy(pbyDest := ADR(dwAbortCode), pbySrc := ADR(SPWR_1_SDORead.ERRORINFO), dwSize
:= SIZEOF(dwAbortCode));
      MEMUtils.SwapLocalToIntel(pAddress := ADR(dwAbortCode), iSize := TO_INT(SIZEOF(dwAbortCode));
   END_IF
   xEnable := FALSE;
END_IF
SPWR_1_SDORead(
   NETWORK:= CANbus.NetId + 1, //CODESYS NetId starts by 0 (the number entered into the CANbus
configurator); CiA405 NETWORK by 1; That is the reason why we have to increment it here.
   ENABLE:= xEnable,
   TIMEOUT:= 0, //Timeout for SDO read in ms. Here: 0 ==> no timeout
   DEVICE:= xBLOCK1214_MIO.NodeID, // NodeID of destination device
   CHANNEL:= 0, //SDO channel which should be used. 0 means auto channeling ==> the next free channel will
be used. It depends on the slave which settings are working here but 0 and channel 1 is always working.
   INDEX:= 16#210D, //Index to be read out
   SUBINDEX:= //subindex13 to be read out );
```

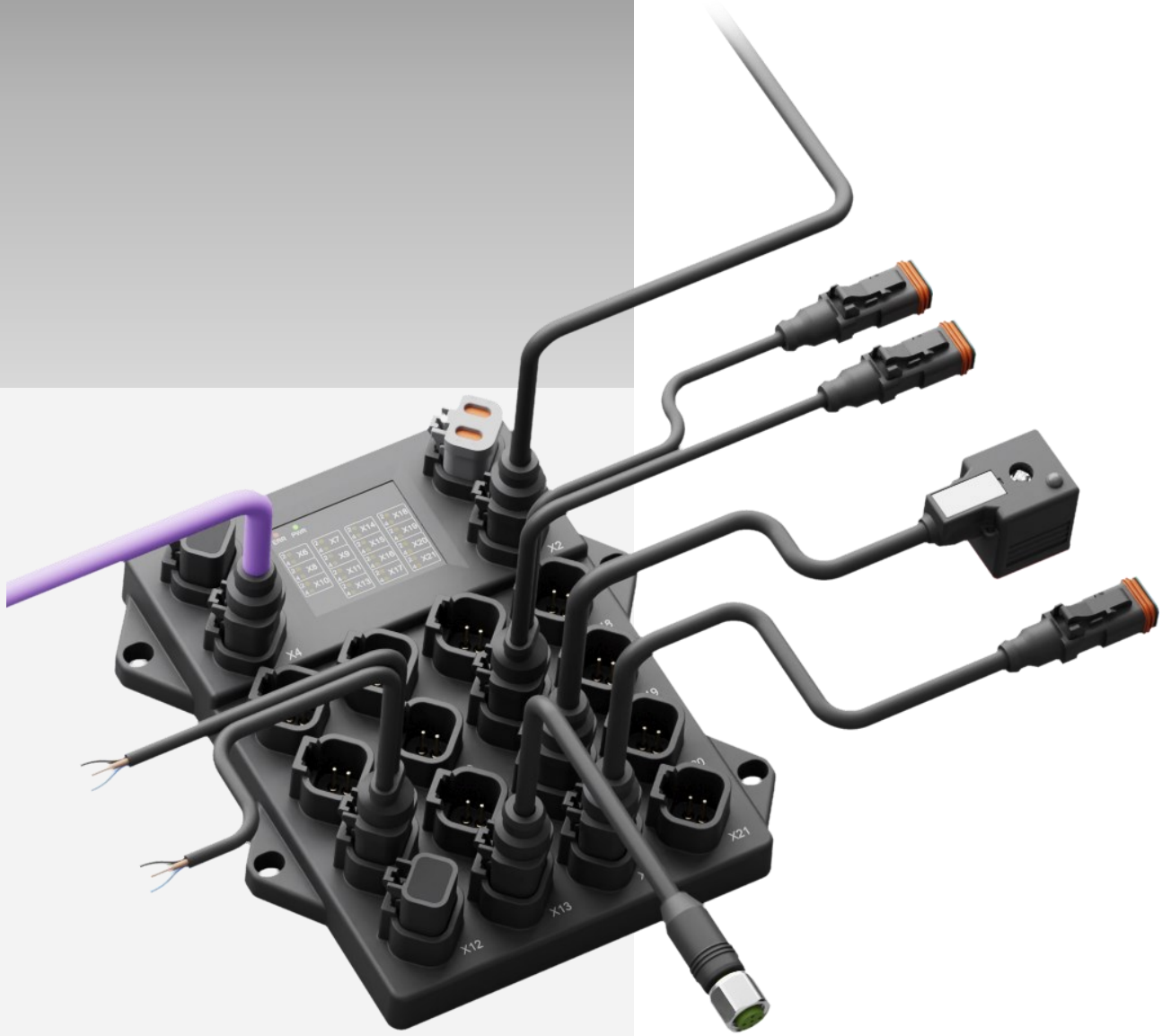
ANALOG_OUTPUT_U16_PWMi_H3_1	%QW1	INT	0	
ANALOG_OUTPUT_U16_PWMi_H3_2	%QW2	INT	1000	
ANALOG_OUTPUT_U16_PWMi_H3_3	%QW3	INT	0	
ANALOG_OUTPUT_U16_PWMi_H3_4	%QW4	INT	0	

xBLOCK1214_MIO | Diagnose | Task | Library Manager

Device.Application.Diagnose

Expression	Type	Value	Prepared value
xEnable	BOOL	FALSE	
dwValue	DWORD	999	
dwAbortCode	DWORD	0	
dwStatusValue	DWORD	0	
dwStatusAbortCode	DWORD	0	
SPWR_1_SDORead	CIA405.SDO_READ4		

DATA PANEL
POWERED BY MURRELEKTRONIK



Application solutions & products for simple, decentralized and high-quality machine installation

Mobile automation plugged in - what else!