X90BC124.32-00

1 General information

The modular X90 mobile control and I/O system opens up a wide range of possibilities in mobile automation. With the X90 mobile system, flexible automation concepts can be implemented based on a standardized complete system.

Bus controller module X90BC124.32-00 is a configurable CANopen slave module for sensor and actuator management. 32 multifunction inputs and outputs are available for a wide range of tasks.

The robust die-cast aluminum housing enables use in harsh ambient conditions.

- 32 multifunction inputs or outputs
- Sensor power supply
- CANopen connection (daisy chain)

The X90 mobile system has been designed for the following application areas:

- Agricultural and forestry machines
- Construction machinery
- Municipal utility vehicles
- Stationary outdoor applications

2 Order data

Model number	Short description	Figure
	X90 Bus controllers	
X90BC124.32-00	X90 mobile 120 bus controller, 1 CANopen to CMC header, 32 multifunction I/Os, die-cast aluminum housing	
	Optional accessories	
	BreakOut Box	
X90AC-BB.12-00	X90 mobile 120 bus controller test set, template and cable adapter for breakout box X90AC-BB.17-00 for testing X90BC124.32-00	
	CMC connector	
X90TB120.01-00	X90 mobile 120, connector for CMC header, with connector contacts and dummy plugs	
	Wire harness	
X90CA124.02-00	X90 mobile 120, wiring harness starter set for X90BC124, 2 m length, for CMC header	

Table 1: X90BC124.32-00 - Order data

Information:

When ordering, use order code X90B2100000000-000 instead of the order number.

Required accessories
For a general overview, see section "Accessories - Overview" of the X90 mobile system user's manual.

Information:

See the data sheet for the permissible temperature range for using cables.

Customized logos

The X90 mobile bus controller can be delivered with a customized logo instead of the B&R logo. For additional information about the exact ordering procedure, please contact the local sales office.

your logo could be here
X90B2100000000-000
X90BC124.32-00 C0
E24 10R-05 3105
KW1H1234567
B&R | 5142 Eggelsberg | AUSTRIA
Made in Austria
Do not disconnect unless power has been switched off!
Close connectors that are not being used with caps!
X1

3 Technical data

Model number	**X90BC124.32-00**
Short description	
Bus controller	CANopen
General information	
Insulation voltage for GND and housing	500 V_{eff}[1]
B&R ID code	0xF23F
Cooling	Fanless
Status indicators	No
Diagnostics	
Module run/error	Yes, using software status
Controller function	Yes, using software status
Short circuit	MF-AI, MF-DO and MF-PWM: Yes, from the application [2]
Load break	MF-DO and MF-PWMY: Yes, from the application
Analog inputs	Limit value overshoot
Overload	MF-DO and MF-PWMY: Yes, from the application
Overtemperature	Yes, from the application
Power consumption [3]	With active 5 V sensor power supply: 3 to 3.3 W With active 10 V sensor power supply: 6.1 to 9 W
Certifications	
CE	Yes
Input power supply	
Input voltage	9 to 32 VDC
Controller	
Real-time clock	No
FPU	Yes
Processor	
Type	ARM Cortex-M4
Integrated I/O processor	Processes I/O data points in the background
Option boards	0
Interfaces	
CAN	
Variant	Connection on CMC header X1
Transfer rate	Max. 1 Mbit/s
Max. distance	1000 m
Terminating resistor	External 120 Ω must be provided. [4]
Multi-function inputs	
Multifunction digital inputs (MF-DI)	
Quantity	4
Functions [5]	**Digital input** Sink/Source circuit, configurable software input filter, fixed or ratiometrically configurable threshold value, open-circuit and short-circuit detection. Counter function Input frequency max. 50 kHz, period duration or gate measurement, AB/ABR encoder, DF/edge counter, rising or falling edge **Analog input** Measurement range 0 to 32 V, configurable analog filter, configurable limit values, integrated input protection

Table 2: X90BC124.32-00 - Technical data

Model number	**X90BC124.32-00**
Multifunction analog inputs (MF-AI)	
Quantity	12
Functions [5]	**Analog input** MF-AI-T Measurement ranges 0 to 10 V / 0 to 32 V / 0 to 20 mA / 4 to 20 mA / 0 to 4 kΩ / -80 to 270°C, configurable analog filter, configurable limit values, integrated input protection MF-AI-1 Measurement ranges 0 to 10 V / 0 to 32 V / 0 to 20 mA / 4 to 20 mA, configurable analog filter, configurable limit values, integrated input protection MF-AI-2 Measurement ranges 0 to 10 V / 0 to 32 V, configurable analog filter, configurable limit values, integrated input protection **LED driver** Max. 20 mA **Digital input** Sink/Source circuit, configurable software input filter, fixed or ratiometrically configurable threshold value, open-circuit and short-circuit detection.
Multi-function outputs	
Multifunction digital outputs (MF-DO)	
Quantity	4
Functions [5]	**Digital output** 4 A nominal current, 2 channels source circuit, 2 channels sink circuit, integrated output protection, parallel connection of source channels possible. **Digital input** Sink circuit, configurable software input filter, fixed configurable threshold value, open-circuit and short-circuit detection. **Analog input** Measurement range 0 to 32 V, configurable analog filter, configurable limit values, integrated input protection
Multifunction PWM outputs (MF-PWM)	
Quantity	8
Functions [5]	**Digital output** 4 A nominal current, source circuit, integrated output protection, parallel connection possible. **PWM output** 4 A nominal current, PWM frequency 15 Hz to 1 kHz, current control function, integrated output protection, parallel connection possible, configurable switch-on/switch-off ramp, dither **Digital input** Sink circuit, configurable software input filter, fixed or ratiometrically configurable threshold value, open-circuit and short-circuit detection. **Analog input** Measurement range 0 to 32 V, configurable analog filter, configurable limit values, integrated input protection
Multifunction PVG outputs (MF-PVG)	
Quantity	4
Functions [5]	**Digital output** 10 mA nominal current at 24 V, sink/source circuit, integrated output protection. **PVG output** 10 mA nominal current at 24 V, PVG frequency 15 Hz to 1 kHz. **Digital input** Sink/Source circuit, configurable software input filter, fixed configurable threshold value, open-circuit and short-circuit detection. **Analog input** Measurement range 0 to 32 V, configurable analog filter, configurable limit values, integrated input protection
CPU/Controller power supply	
Voltage range	9 to 32 VDC
Input current	Max. 3 A
Integrated protection	No, required fuse max. 3 A, slow-blow
I/O power supply	
Voltage range	9 to 32 VDC
Input current	Max. 10 A per pin
Integrated protection	No, required fuse max. 10 A, slow-blow
Digital inputs	
Quantity	0 to 32, depends on the use of multifunction inputs/outputs
Nominal voltage	12 / 24 VDC
Input voltage	9 to 32 VDC
Input current at 24 VDC	MF-DI: 1.2 mA MF-AI: 1.2 mA MF-DO_plus: <1 mA MF-PWM: <1 mA MF-PVG: Typ. 10 mA
Input circuit	Sink/Source, configurable
Input filter	
Hardware	MF-DI: 3 µs if switching threshold = 50% supply voltage MF-AI: 600 µs if switching threshold = 50% supply voltage MF-DO: 300 µs MF-PWM: 150 µs
Software	Default 1 ms, configurable between 0 and 25 ms in 0.1 ms increments
Input resistance	MF-AI and MF-DI: Typ. 22 kΩ

Table 2: X90BC124.32-00 - Technical data

Model number	X90BC124.32-00
Additional functions	**Counter functions** (inputs 13 to 16) Edge detection: Max. 50 kHz Period duration / Gate measurement: Max. 3.5 kHz ABR, AB and DF counters
Input frequency	MF-DI: Max. 50 kHz
Switching threshold	MF-DI: 50% of supply voltage MF-AI: Switching threshold and hysteresis configurable with software MF-DO: 50% supply voltage MF-PWM: 50% supply voltage
Analog inputs	
Input	0 to 10 V / 0 to 32 V / 0 to 20 mA / 4 to 20 mA / 0 to 4 kΩ / -80 to 270°C
Digital converter resolution	12-bit
Conversion time	250 µs
Output format	
Data type	INT, UINT (resistance)
Voltage	Voltage 0 to 10 V: INT 0x0000 - 0x7FFF / 1 LSB = 0x0008 = 2.44 mV Voltage 0 to 32 V: INT 0x0000 - 0x7FFF / 1 LSB = 0x0008 = 7.81 mV
Current	INT 0x0000 - 0x7FFF / 1 LSB = 0x0008 = 4.9 µA
Resistance	0 to 4000, 1 Ω increments
Temperature input	-80 to 270, 0.1°C increments
Input impedance in signal range	
Voltage	>100 kΩ
Current	-
Measurement current resistance / Temperature input	<1.2 mA
Load	
Voltage	-
Current	<300 Ω
Open-circuit detection	From the application
Reverse polarity protection	Yes
Conversion procedure	SAR
Input filter	Low-pass filter / Cutoff frequency voltage input / Current input 200 Hz
Max. error	
Voltage	
Gain	<1% [6]
Offset	<1% [7]
Current	
Gain	<1% [6]
Offset	<1% [8]
Resistance	<1% [9]
Temperature input	<1% [10]
Max. gain drift	
Voltage	<0.03%/°C [6]
Current	<0.04%/°C [6]
Resistance	0.034% [6]
Temperature input	0.024% [6]
Max. offset drift	
Voltage	<0.006%/°C [7]
Current	<0.02%/°C [8]
Resistance	0.0018%/°C [9]
Temperature input	0.027%/°C [10]
Sensor power supply	
Voltage	5 / 10 V [11]
Current	Maximum 400 mA, accuracy: ±4%
Digital outputs	
Quantity	0 to 23, depends on the use of multifunction outputs
Variant	7x MF-AI, LED driver: Current-sinking 2x MF-DO_minus: Current-sinking 2x MF-DO_plus: Current-sourcing, channels can be connected in parallel (max. 6.5 A). 8x MF-PWM: Current-sourcing, channels can be connected in parallel. 4x MF-PVG: Current-sourcing/Current-sinking
Nominal voltage	12 / 24 VDC
Digital converter resolution	12-bit
Output format	MF-AI LED driver: INT 0x0000 to 0x7FFF / 1 LSB = 0x0008 = 4.9 µA MF-DO_plus: INT 0x0000 to 0x7FFF / 1 LSB = 0x0008 = 1.22 mA MF-DO_minus: INT 0x0000 to 0x7FFF / 1 LSB = 0x0008 = 2.21 mA MF-PWM: INT 0x0000 to 0x7FFF / 1 LSB = 0x0008 = 1.22 mA MF-PVG: INT 0x0000 to 0x7FFF / 1 LSB = 0x0008 = 7.81 mV [12]
Nominal output current	MF-AI LED driver: 20 mA MF-DO: 4 A MF-PWM: 4 A MF-PVG: 10 mA
Output protection	Shutdown in the event of overcurrent or short circuit, integrated protection for switching inductive loads
Diagnostic status	Overload

Table 2: X90BC124.32-00 - Technical data

Model number	X90BC124.32-00
Leakage current when the output is switched off	MF-AI LED driver: 10 µA MF-DO: 10 µA MF-PWM: 10 µA MF-PVG: 10 µA
$R_{DS(on)}$	MF-AI LED driver: 2 Ω MF-DO_minus: 70 mΩ MF-DO_plus: 50 mΩ MF-PWM: 50 mΩ MF-PVG: 2 Ω
Residual voltage	<1 V at 4 A nominal current
Peak short-circuit current	MF-AI LED driver: 20 mA MF-DO_minus: Max. 24 A MF-DO_plus: Max. 90 A MF-PWM: Max. 90 A MF-PVG: Max. 10 mA
Switching frequency	
Resistive load	MF-AI LED driver: Max. 100 Hz MF-DO: Max. 100 Hz MF-PWM: 15 Hz to 1 kHz MF-PVG: 15 Hz to 1 kHz
Max. error at 25°C	
Gain	MF-AI LED driver: <1% MF-DO: <12% MF-PWM: <0.2% MF-PVG: <5% [6]
Offset	MF-AI LED driver: <1% MF-DO: <1% MF-PWM: <0.1% MF-PVG: <1%
Max. gain drift	MF-DO: <0.2%/°C MF-PWM: <0.04%/°C [6]
Max. offset drift	MF-DO: <0.005%/°C MF-PWM: <0.005%/°C [13]
Braking voltage when switching off inductive loads	MF-DO: Typ. 64 VDC
Switching delay	MF-DO: Max. 2 ms
Output voltage	
Nominal	9 to 32 VDC
Current measurement	
Current measurement range	MF-AI LED driver: 0 to 20 mA MF-DO_plus: 0 to 20 A MF-DO_minus: 0 to 5 A MF-PWM: 0 to 5 A
Sampling frequency	MF-AI LED driver: 250 µs MF-DO: 250 µs MF-PWM: 50 µs MF-PVG: 250 µs
PWM output	
Quantity	0 to 8x 4 A Depends on the use of multifunction outputs
Nominal voltage	12 / 24 VDC
Supply voltage (permissible range)	9 to 32 VDC
Digital converter resolution	12-bit
Output format	MF-PWM: INT 0x0000 to 0x7FFF / 1 LSB = 0x0008 = 1.22 mA
PWM frequency	15 Hz to 1 kHz
Duty cycle	0 to 32767, equivalent 0 to 100%
Max. error at 25°C	
Gain	<0.2% [13]
Offset	<0.1% [6]
Max. gain drift	<0.04%/°C [6]
Max. offset drift	<0.005%/°C
Common mode error	0.015%/V
Output protection	Thermal shutdown in the event of overcurrent or short circuit, integrated protection for switching inductive loads
Variant	Current-sourcing FET; channels can be connected in parallel in pairs. [14]
Diagnostic status	Overload
Peak short-circuit current	90 A
Current measurement	
Current measurement range	0 to 5 A
Conversion time	250 µs
Electrical properties	
Summation current	
Complete system	Max. 25.5 A [15]
Operating conditions	
Mounting orientation	
Any	Yes
Degree of protection	IP69K

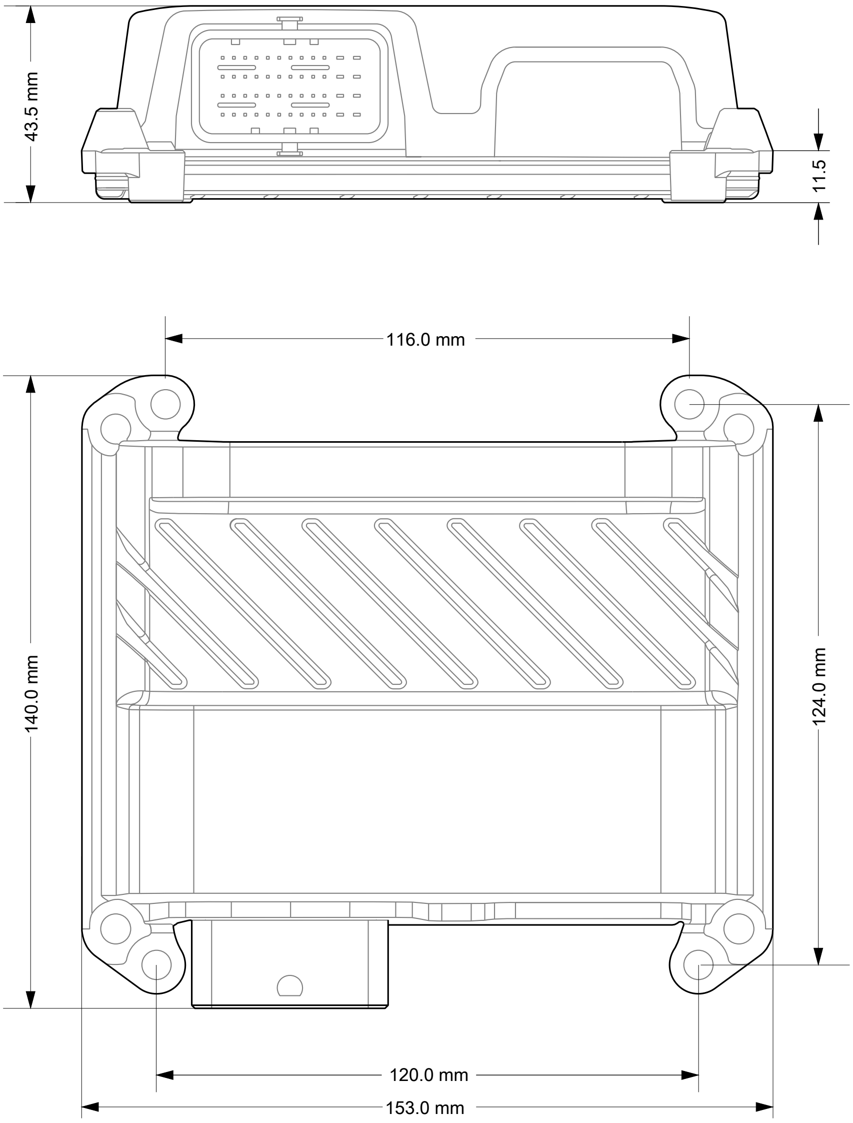
Table 2: X90BC124.32-00 - Technical data

Model number	X90BC124.32-00
Ambient conditions	
Temperature	
Operation	-40 to 85°C housing surface
Derating	-
Storage	-40 to 85°C
Transport	-40 to 85°C
Relative humidity	
Operation	5 to 100%, condensing
Storage	5 to 95%, non-condensing
Transport	5 to 95%, non-condensing
Mechanical properties	
Dimensions	
Width	153 mm
Length	140 mm
Height	44 mm
Weight	700 g

Table 2: X90BC124.32-00 - Technical data

1) Connected to capacitors and 60 V VDR protective element.
2) MF-AI short-circuit detection only for 0 to 20 mA, 4 to 20 mA and LED driver.
3) Power consumption of the controller and with maximum sensor power supply
4) Connection on the first and last CAN station according to the general CAN specification.
5) Open-circuit and short-circuit detection only when configured as a diagnostics-capable input and external circuit (NAMUR)
6) Based on the current measured value.
7) Based on the 10 V or 32 V measurement range.
8) Based on the 20 mA measurement range.
9) Based on the 4 kΩ measurement range.
10) Based on the maximum measurement range (-80 to 270°C).
11) At 10 V sensor power supply, the CPU power supply must be ≥12 VDC.
12) MF-PVG: Always 0 in tri-state mode
13) Based on the 25 A measurement range.
14) See section "Operating the power channels".
15) Total current of I/O power supply without load currents on MF-DO_minus

4 Dimensions



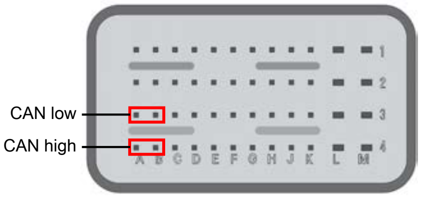
Use suitable M6 screws to lock the module into position.

Information:

M6 screws are not included in delivery.

5 Operating and connection elements

5.1 CAN bus interface



The CAN bus is connected on the CMC multi-header.

Pin	Channel
A3, B3	CAN_L
A4, B4	CAN_H

Table 3: CAN bus interface

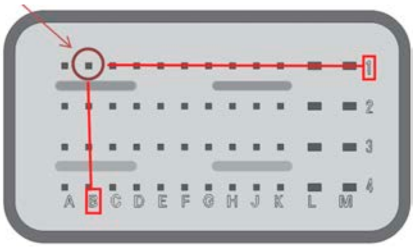
Information:

A separate CAN-GND connection is not available. Any GND connection can be used, e.g. M1.

5.2 Pinout

The pinout of the connection is encoded using letters and numbers.

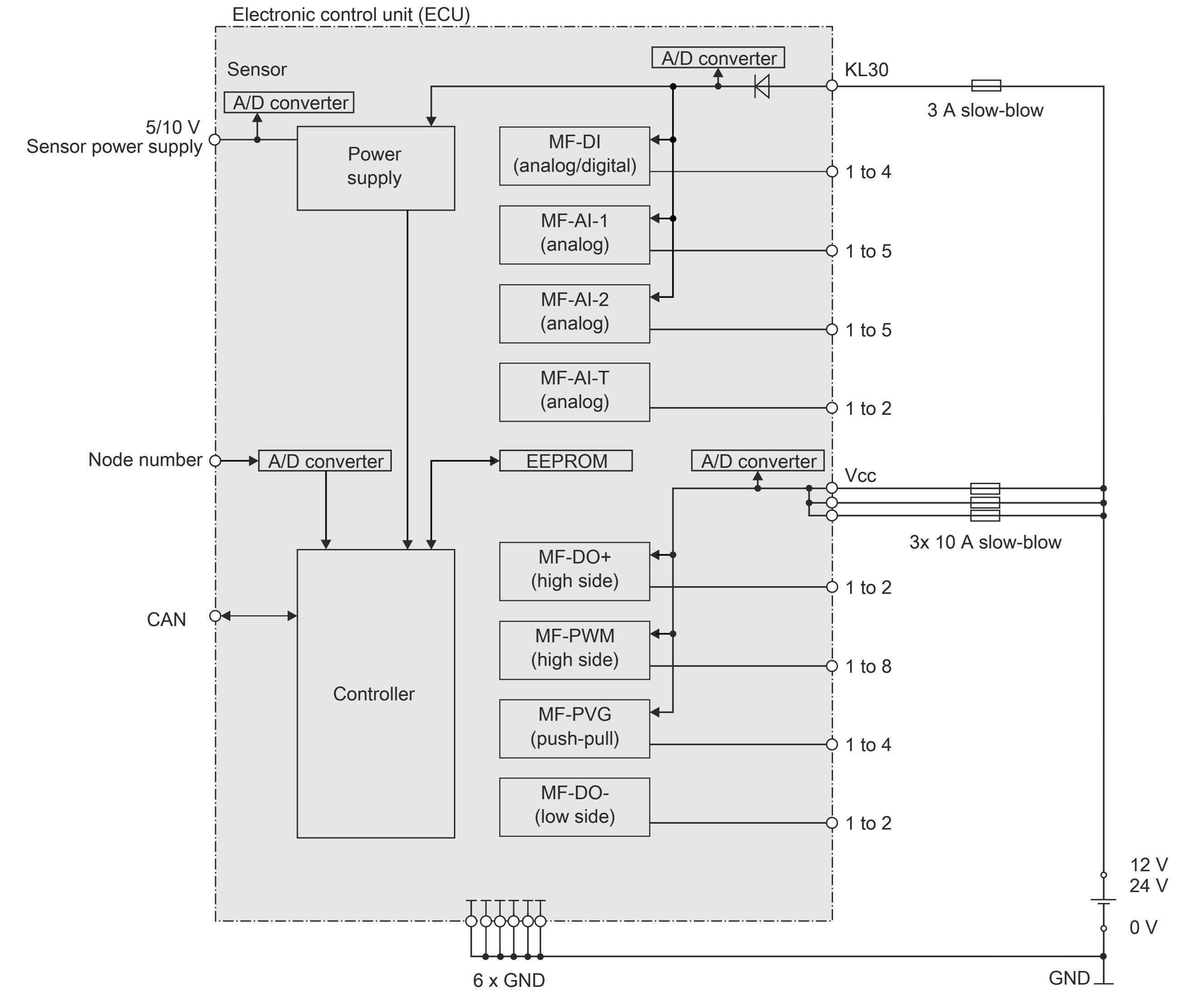
Example: Pin B1



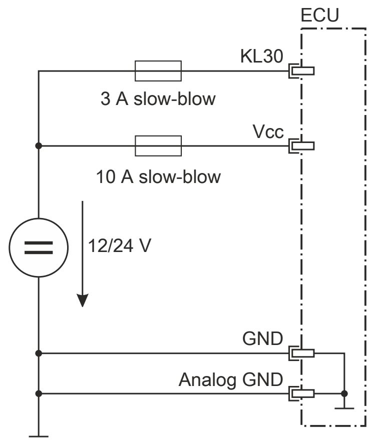
Pinout		
Pin	**Description**	**Channel**
A1	Sensor power supply 5 or 10 V	5 V / 10 V
B1	GND	
C1	Controller power supply	KL30
D1	MF-AI-2	12
E1	MF-DO+	31
F1	MF-DO+	32
G1	MF-DO-	30
H1	MF-PWM	24
J1	MF-PWM	17
K1	MF-PWM	18
L1	MF-PWM	19
M1	GND	
A2	Node number	
B2	MF-PVG	27
C2	MF-AI-1	7
D2	MF-AI-2	8
E2	MF-PVG	26
F2	MF-AI-T	1
G2	MF-AI-1	6
H2	MF-DO-	29
J2	MF-DI	13
K2	MF-PWM	22
L2	GND	
M2	I/O power supply	Vcc
A3	CAN-L (in)	CAN_L
B3	CAN-L (out)	CAN_L
C3	MF-AI-2	11
D3	MF-AI-2	10
E3	MF-AI-2	9
F3	MF-AI-1	4
G3	Analog GND	
H3	MF-DI	15
J3	MF-DI	16
K3	MF-PWM	23
L3	I/O power supply	Vcc
M3	GND	
A4	CAN-H (in)	CAN_H
B4	CAN-H (out)	CAN_H
C4	MF-PVG	25
D4	MF-PVG	28
E4	MF-AI-T	2
F4	MF-AI-1	3
G4	MF-AI-1	5
H4	MF-DI	14
J4	MF-PWM	21
K4	MF-PWM	20
L4	GND	
M4	I/O power supply	Vcc

For information about installing the connector, see section "Mechanical handling - Installing, removing and accessory installation - Accessory installation - Mating connector X1" of the X90 user's manual.

6 Block diagram

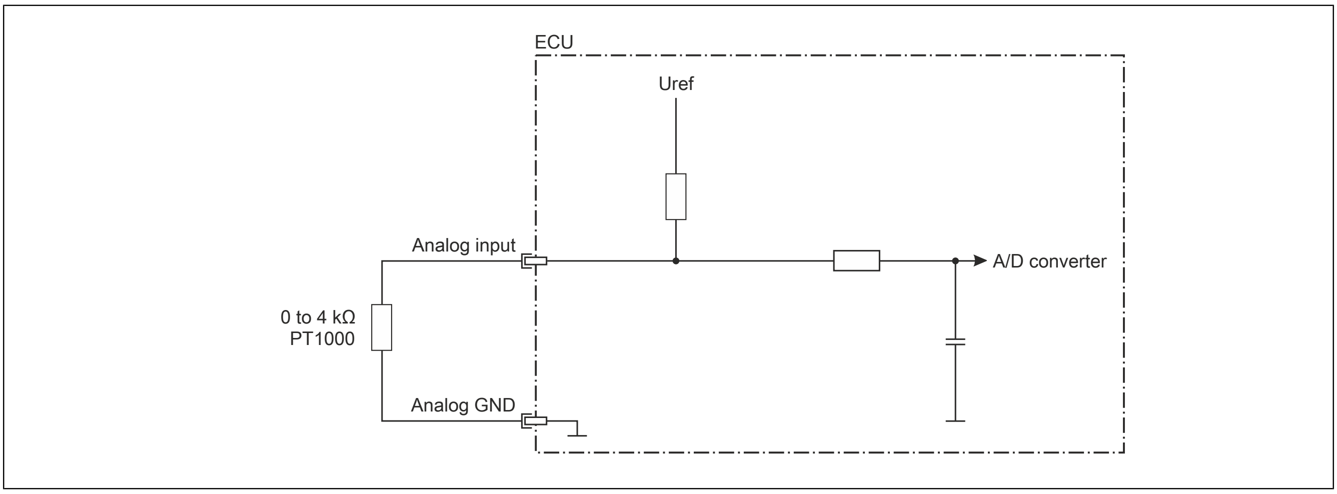


6.1 Power supply

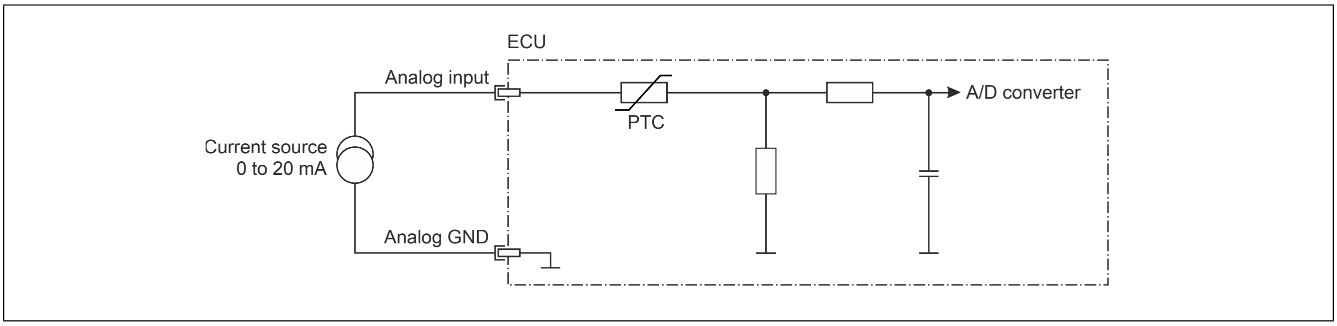


6.2 Input circuit diagram

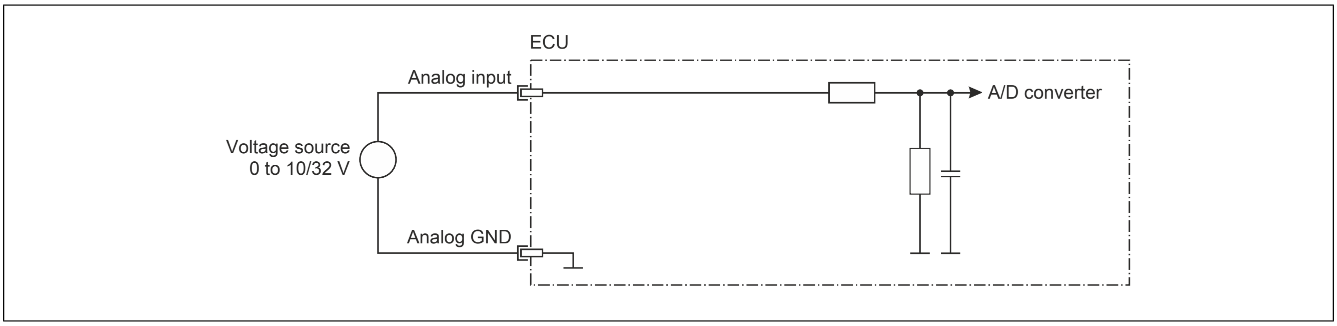
Multifunction analog input MF-AI-T resistance and PT1000 input



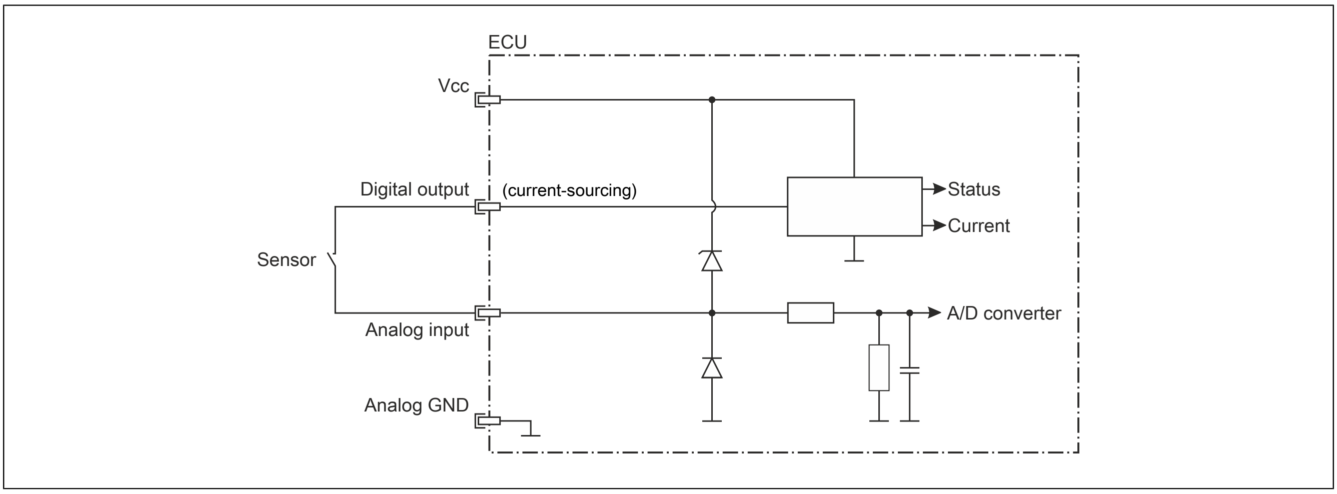
Multifunction analog input MF-AI-T / MF-AI-1 current input



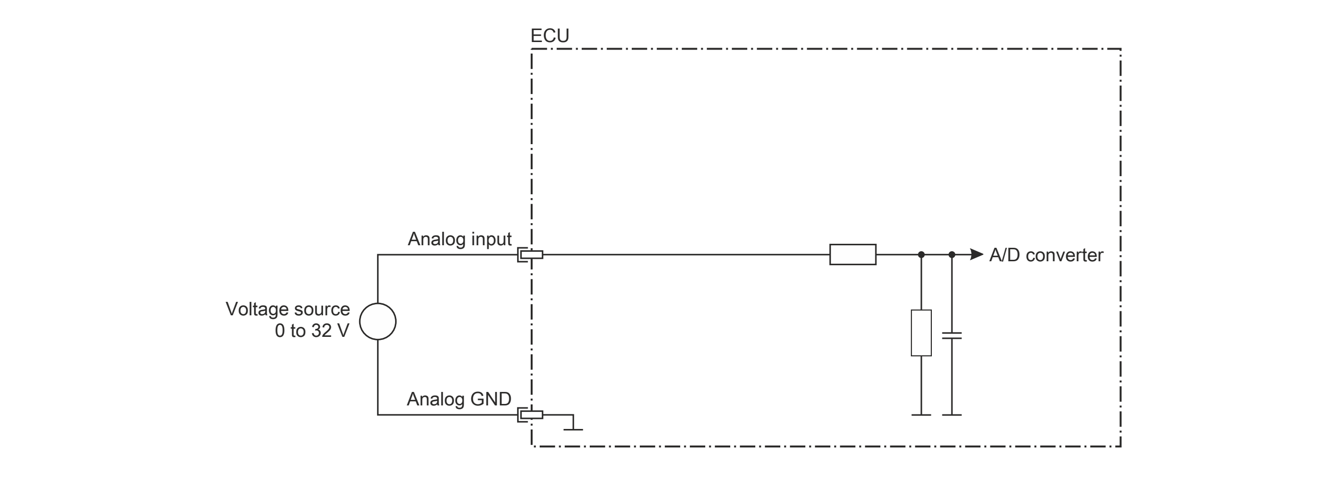
Multifunction analog input MF-AI-T / MF-AI-1 / MF-AI-2 voltage input



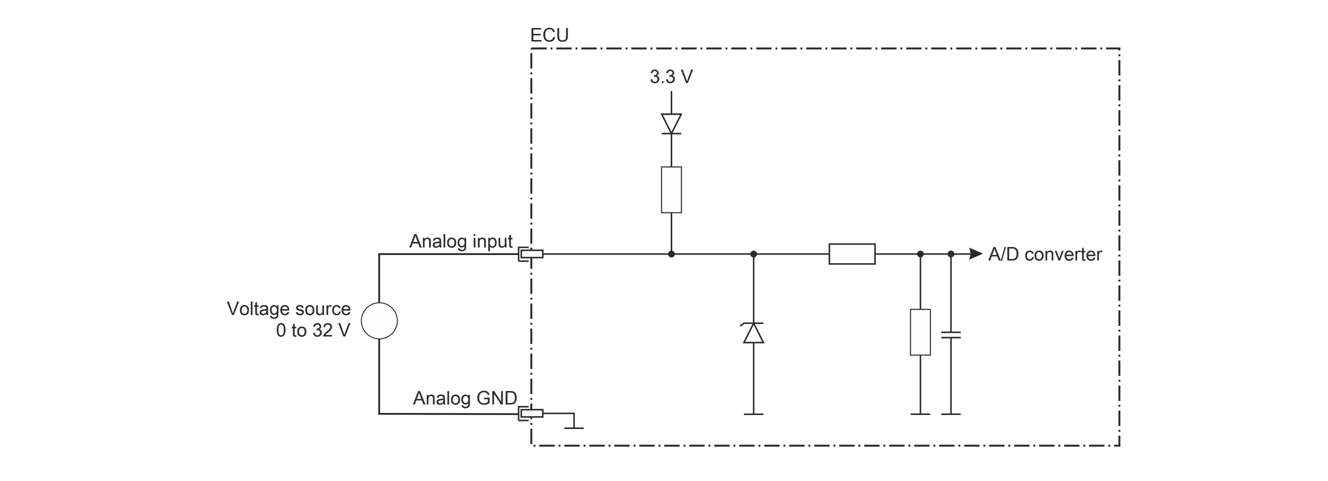
Multifunction analog input MF-PWM voltage input



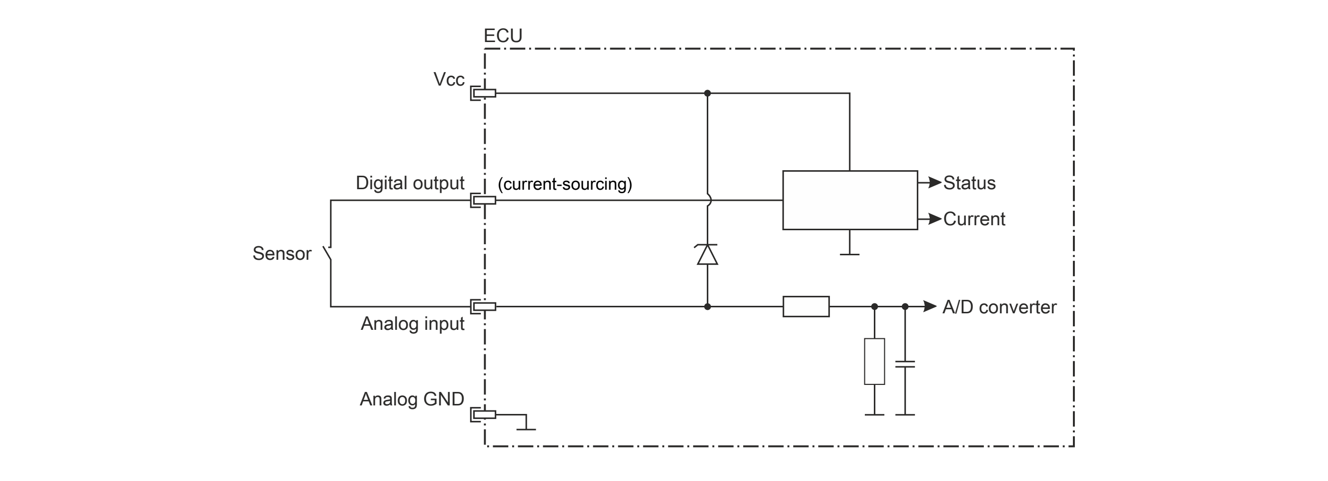
Multifunction analog input MF-PVG voltage input



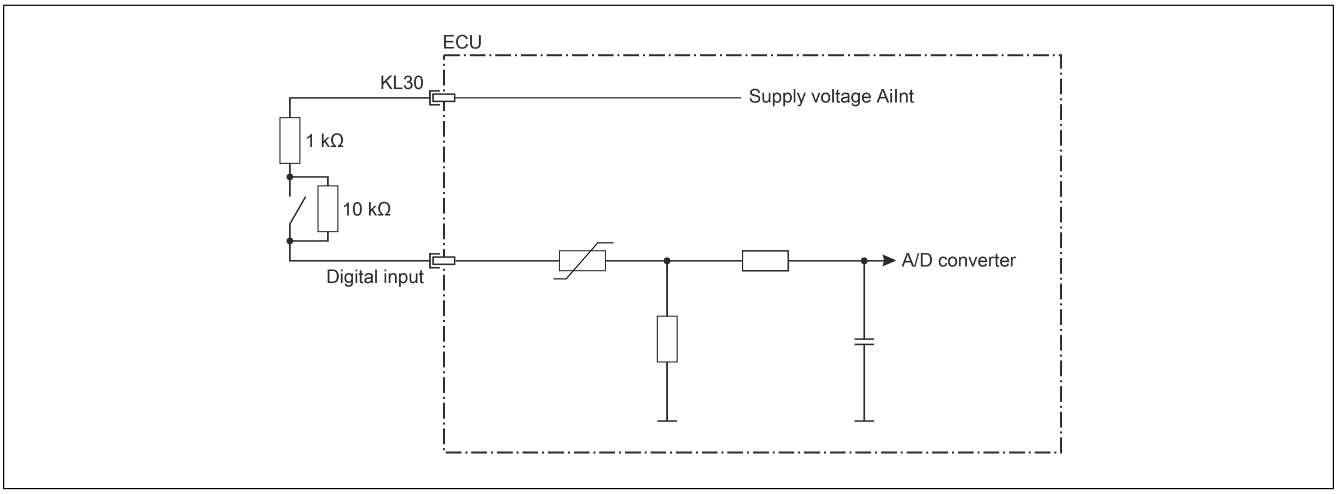
Multifunction analog input MF-DO_minus voltage input



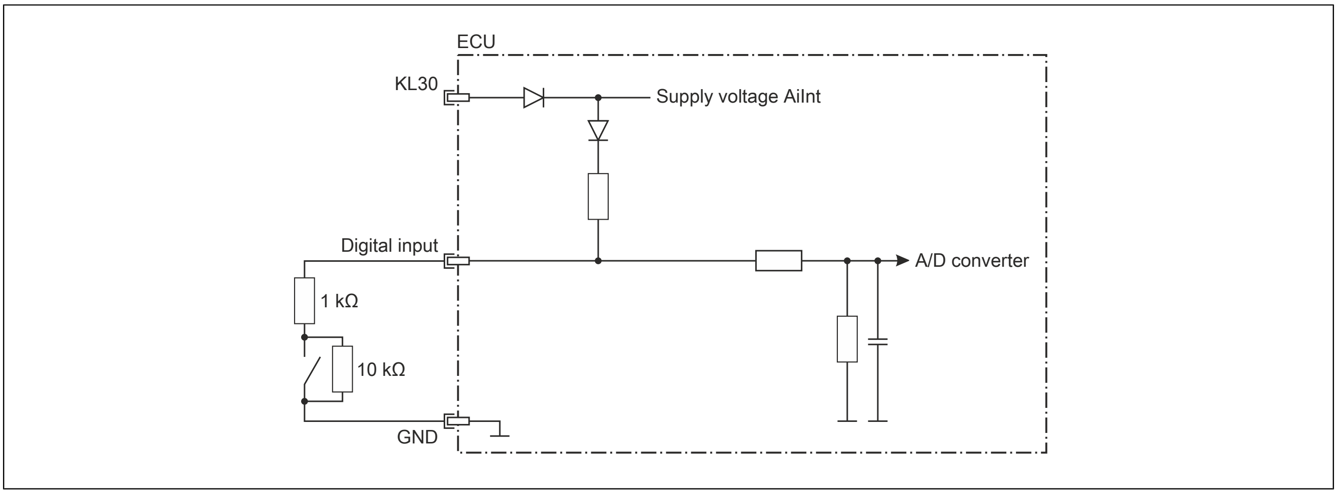
Multifunction analog input MF-DO_plus voltage input



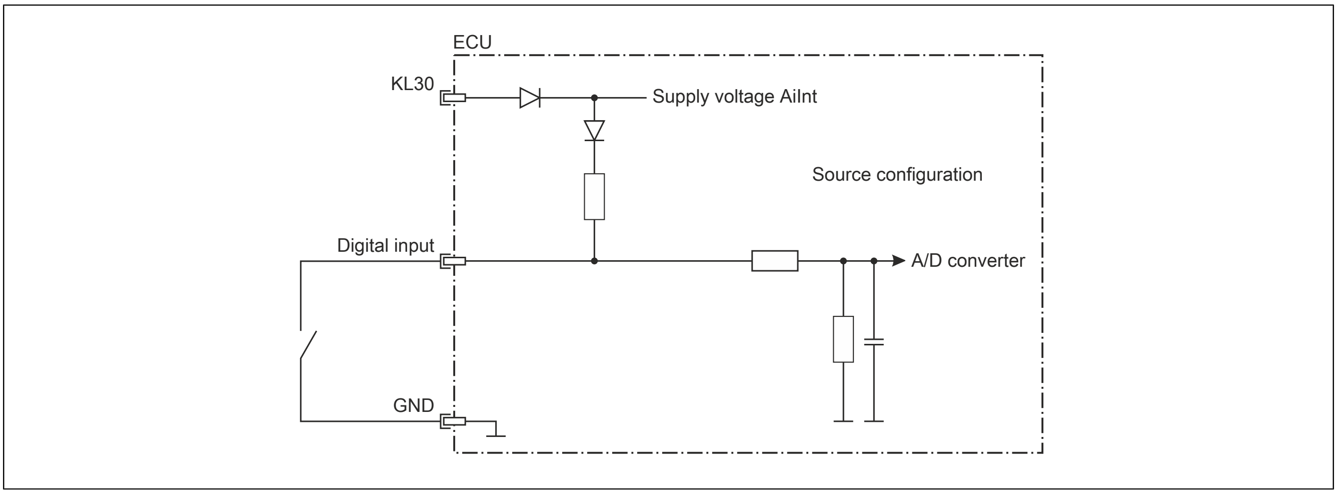
Multifunction analog input MF-AI-T / MF-AI-1 diagnostics-capable current input



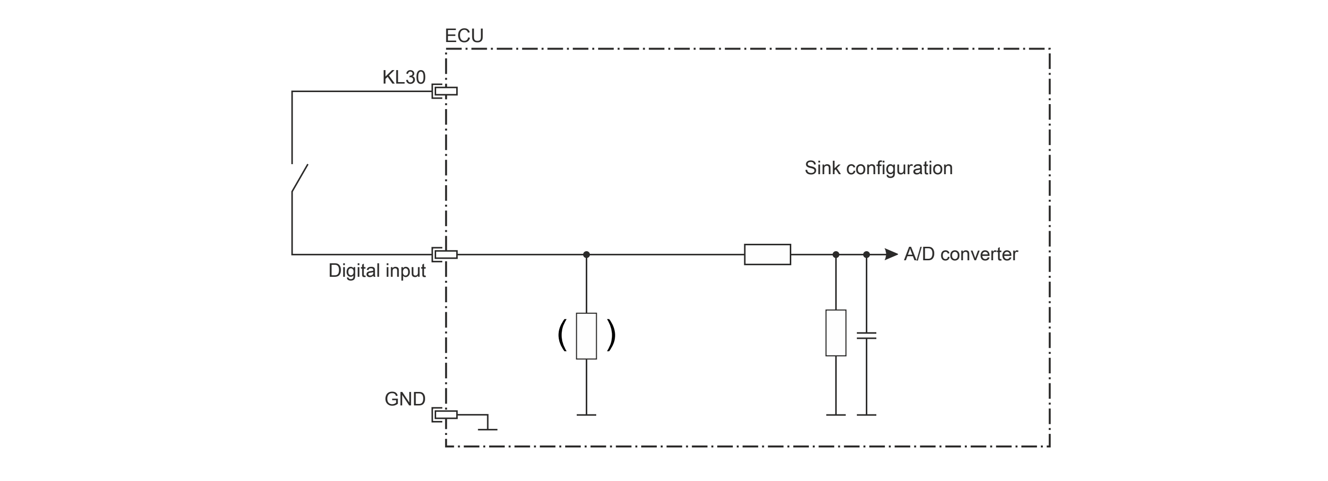
Multifunction digital input MF-AI-2 / MF-DI diagnostics-capable voltage input



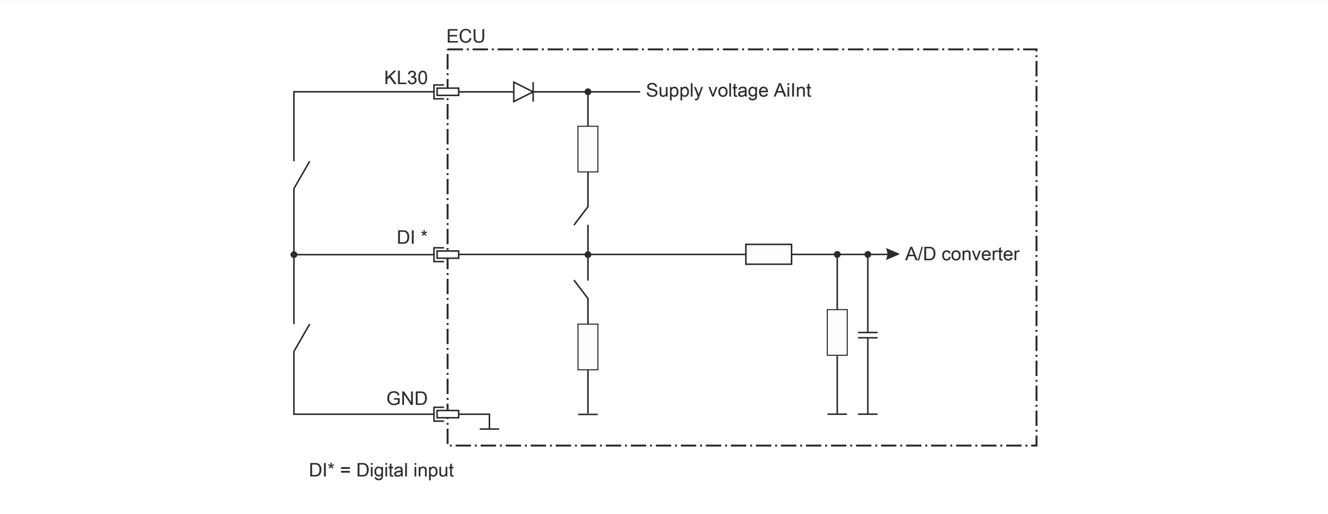
Multifunction digital input MF-AI-T / MF-AI-2 / MF-DI negative switching



Multifunction digital input MF-AI-1 / MF-AI-2 / MF-DI positive switching

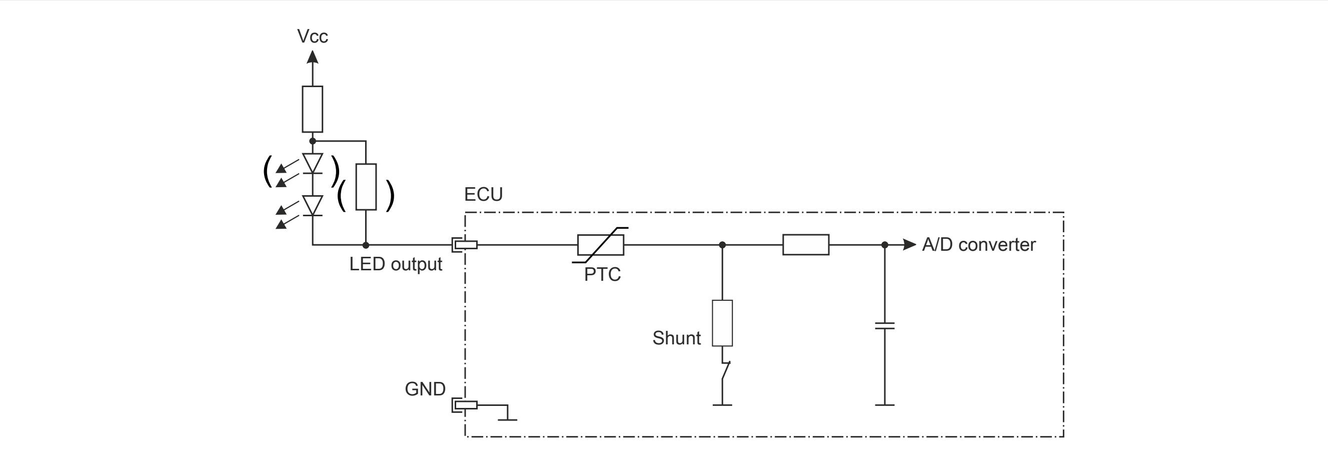


Multifunction digital input MF-DI counter, with switchable pull-up/pull-down resistors

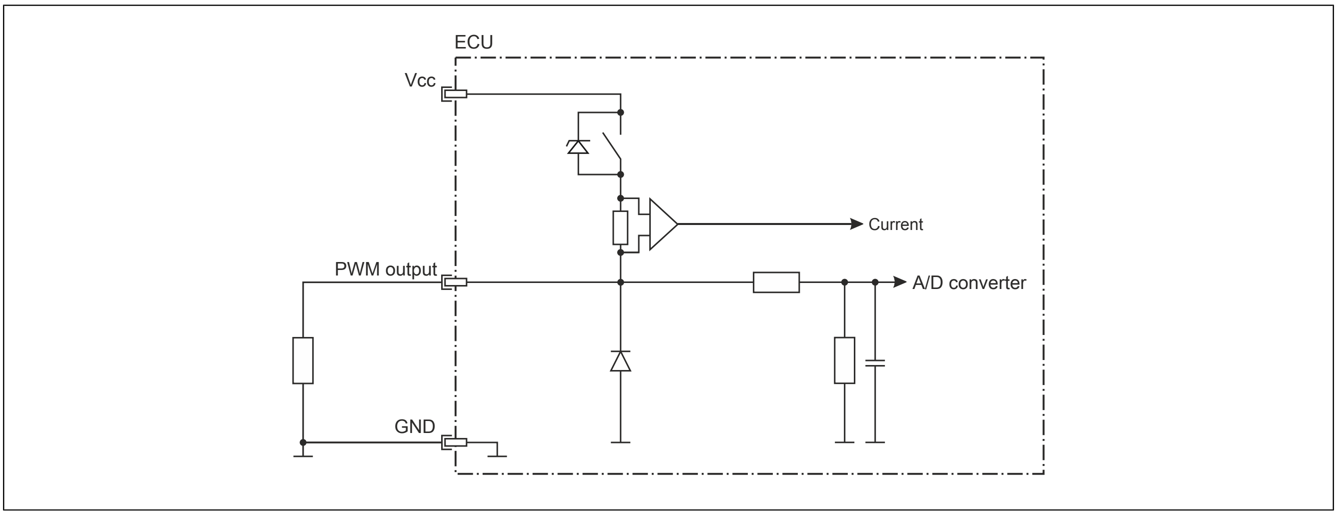


6.3 Output circuit diagram

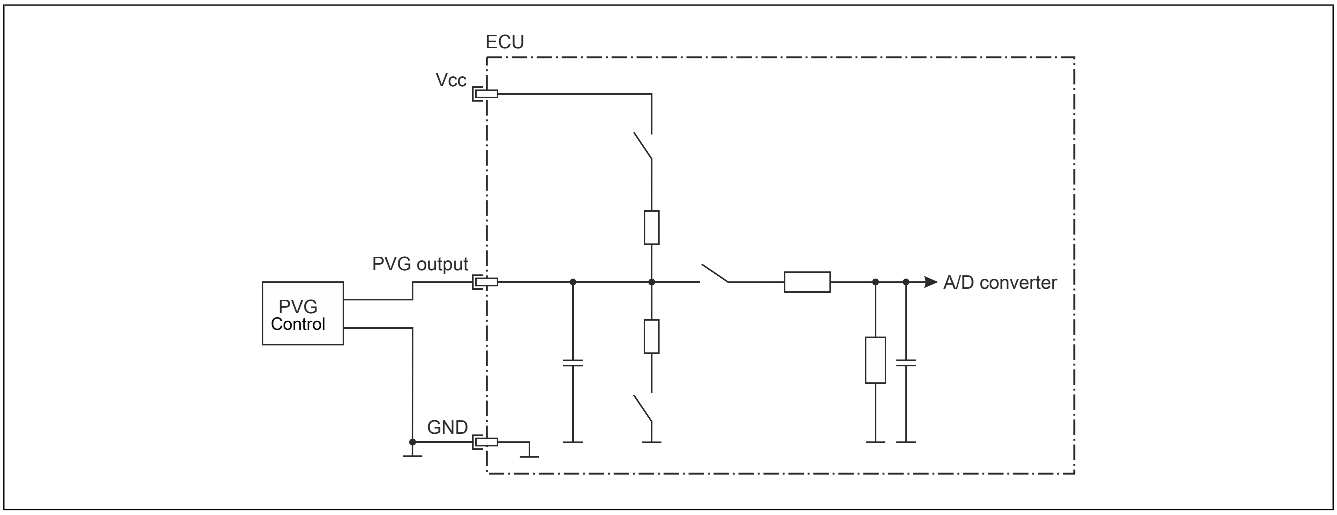
Multifunction output MF-AI-T / MF-AI-1 LED driver



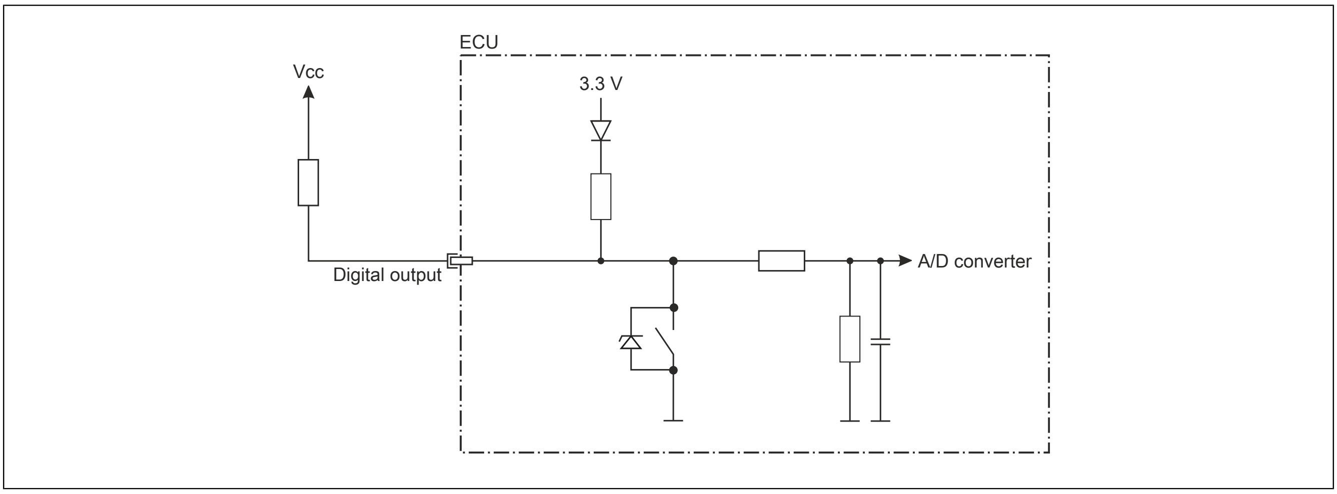
Multifunction output MF-PWM PWM output source circuit



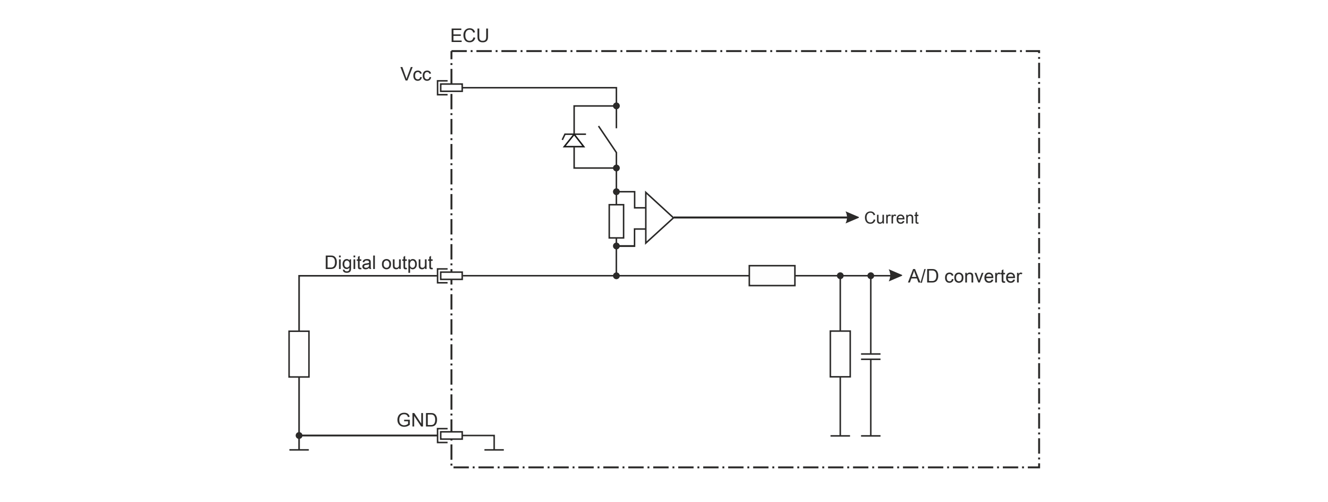
Multifunction output MF-PVG PVG output source circuit



Multifunction output MF-DO_minus



Multifunction output MF-DO_plus



6.4 External fuse protection

The power supply lines and the power outputs must be protected by suitable circuit breakers or fuses (line protection).

Pin	Description	Fuse protection
C1	Controller power supply	3 A slow-blow
M2	I/O power supply	10 A slow-blow
L3	I/O power supply	10 A slow-blow
M4	I/O power supply	10 A slow-blow

7 Node number and transfer rate

The node number can be defined via pin A2 on the CMC multi-header.

The following table shows how the node number is defined by using resistors and connecting to GND or Vcc:

Node number	Resistance	Connected to:
4	-	Not connected
17	<100 Ω	GND
30	1 kΩ	GND
56	4.7 kΩ	GND
82	10 kΩ	GND
108	22 kΩ	GND
124[1]	22 kΩ	Vcc
95	10 kΩ	Vcc
69	4.7 kΩ	Vcc
43	≤1 kΩ	Vcc

[1] Service node number with fixed 250 kbit transfer rate

Automatic transfer rate detection

After startup, the bus controller goes into "Listen only" mode. This means the bus controller behaves passively on the bus and only listens.

The bus controller attempts to receive valid objects. If receive errors occur, the controller switches to the next transfer rate in the lookup table.

If no objects are received, all transfer rates are tested cyclically. This procedure is repeated until valid objects are received.

Lookup table

The bus controller tests the transfer rate according to this table. Beginning with the starting transfer rate (1000 kbit/s), the controller switches to the next lower transfer rate. At the end of the table, the bus controller restarts the search from the beginning.

Transfer rate
1000 kbit/s
800 kbit/s
500 kbit/s
250 kbit/s
125 kbit/s
100 kbit/s
50 kbit/s
20 kbit/s
10 kbit/s

7.1 Allocated COB IDs

Depending on the number of RPDOs and TPDOs used, the bus controller allocates a different number of COB IDs. The following table shows the base values of the COB IDs for the corresponding objects. The value used is calculated as follows:

COB ID = Base value + Node number

Example

RPDO COB ID of 0x500 and node number 17: 1280 + 17 = 1297

TPDO COB ID of 0x380 and node number 69: 896 + 69 = 965

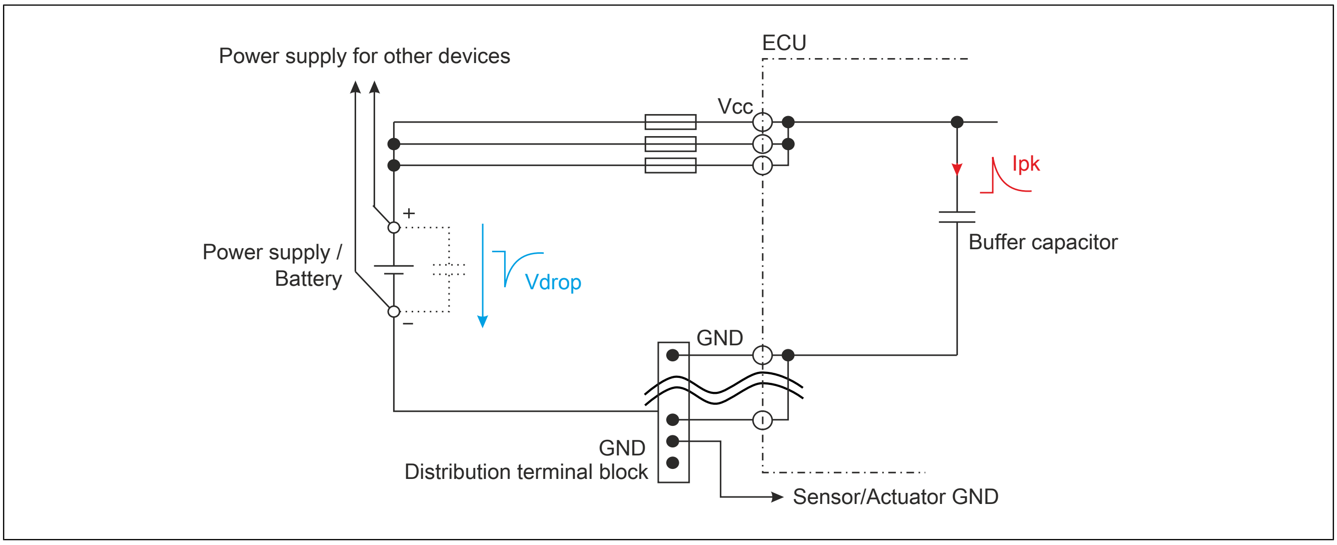
	Base value	
	Hex	Dec.
RPDO	0x200	512
	0x300	768
	0x400	1024
	0x500	1280
	0x240	576
TPDO	0x180	384
	0x280	640
	0x380	896
	0x480	1152
	0x1C0	448
	0x2C0	704
	0x3C0	960
	0x4C0	1216
	0x1A0	416
	0x2A0	672
	0x3A0	928
	0x4A0	1184
	0x1E0	480
	0x2E0	736
	0x3E0	992
	0x4E0	1248
	0x580	1408
	0x600	1536

8 Inrush current

If the X90 mobile system is supplied with power from a power supply instead of a battery, it is important to ensure sufficient support capacity of ≥ 4700 µF.

The reason for this is that a buffer capacitor is charged when the bus controller is switched on, which results in a voltage dip of <1 ms.

The X90 mobile system is not affected by this. This can have a negative influence on other components using the same power supply, however.



9 Current monitoring

Both the output current of individual power channels and the summation current of all MF-PWM, MF-DO_plus and MF-DO-Minus power outputs of the module are measured.

9.1 Output current monitoring

The output current of individual module outputs is measured. If the output current of the MF-PWM, MF-DO_plus and MF-DO_minus channels reaches or exceeds the limit value of 4.5 A, the module will perform the following actions:

- The respective Overload error bit is set.
- The affected output is cut off.

To switch the output on again, the error must be acknowledged with OverloadClear for the respective channel. A lock time does not have to be observed.

> **Information:**
>
> **If the output current is exceeded again after error acknowledgment, this results in another immediate cutoff.**

9.2 Monitoring the summation current

The maximum permissible summation current of all power outputs except for MF-DO_minus is monitored. If the maximum permissible summation current of 25.5 A is reached or exceeded, the module performs the following actions:

- Error bit Overcurrent is set.
- All power outputs are cut off.

To switch the output on again, the error must be acknowledged with OverloadShutdownClear. A lock time does not have to be observed.

> **Information:**
>
> **If the overcurrent occurs again after error acknowledgment, this results in another immediate cutoff.**

10 Overtemperature

Due to no air circulation, insufficient air circulation or heating resulting from heavy loads on the power outputs (depending on the number of simultaneously used outputs, current, PWM frequency, etc.), the X90 bus controller may heat up considerably. Direct sunlight or external heat sources (e.g. combustion engines) can also have a negative effect on the temperature.

If the internal temperature of the bus controller exceeds 95°C, all digital power outputs (channels 17 to 24 and 29 to 32) are cut off automatically. In addition, bit 7 of the system status is set and an error message is output.

If the internal temperature drops below 85°C, the digital outputs are automatically re-enabled and return to the switching state that existed before the cutoff.

> ## Danger!
>
> **After an error message has occurred, the application must ensure that no damage to property or personal injury occurs when the digital inputs are switched back on.**

11 Power failure

It is important to ensure that in the event power failure (VCC), external voltages on the multifunction analog inputs are also cut off. Otherwise, the module will be supplied via the integrated protective diodes of analog inputs MF-DO_plus and MF_PWM, which may damage the module.

For this reason, external sensors and actuators should only be supplied via the module's own digital outputs or the available sensor power supply!

12 Operating the power channels

To avoid damage to the module, it must be ensured that neither the maximum total current nor the permissible operating temperature is exceeded.

Maximum total current

The maximum total output current of a power channel pair is 6.5 A. Each power channel pair begins with the odd, lower channel number. These are channels 17 and 18 to channels 23 and 24 as well as channels 31 and 32.

Maximum operating temperature

> ## Notice!
>
> **The module does not have automatic shutdown in the event of overtemperature. It is therefore important to ensure in the application that the maximum operating temperature of 105°C is not exceeded.**

12.1 Parallel connection of channels

To operate higher loads (higher than 4.5 A), digital MF-DO_plus channels 31 and 32 as well as PWM outputs 17 to 24 can be connected in parallel. Parallel connection is only possible for power channel pairs (e.g. PWM channels 17 and 18, PWM channels 19 and 20). The main channels is the channel with the lower channel number. It is important to ensure the following regarding configuration:

- Configure the main channel as digital (MF-DO_plus) or PWM output (MF-PWM).
- Configure the following channel as "Output connected in parallel with previous channel".

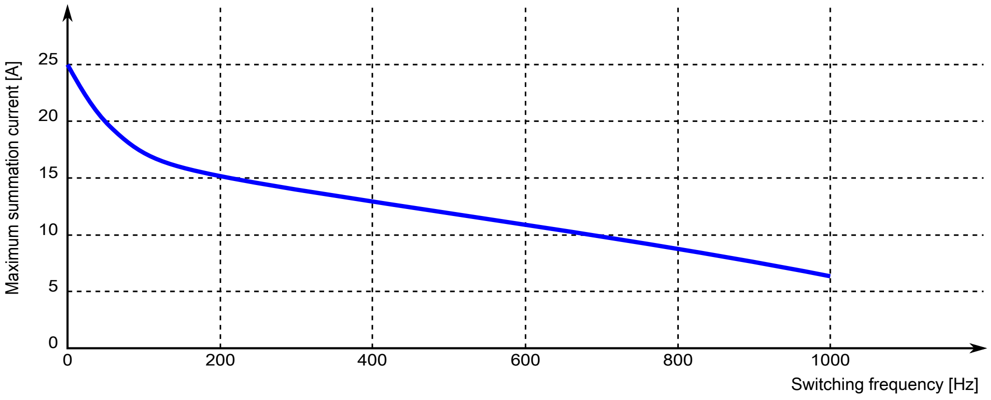
If the maximum total current of the outputs is exceeded, an overload error message occurs and both channels are cut off. After error acknowledgment of the main channel, both channels are switched on again simultaneously.

> ## Notice!
>
> **To avoid an uneven load distribution on the outputs, it is important to ensure that the connecting cables of the parallel-connected channels are of equal length and cross sections.**

12.2 Parallel switching of higher loads

If larger loads must be switched on several channels simultaneously, it is important to ensure that the maximum total current of 6.5 A of the respective power channel pairs is not exceeded (see above "Maximum total current"). If the maximum total current is exceeded, the load MUST be divided between multiple power channel pairs.

In addition, the maximum permissible summation current must be observed depending on the switching frequency.



Information:

For simultaneous PWM operation, the (different) frequencies and output loads must be selected such that the operating temperature of 105°C is not exceeded.

Examples

Correct distribution of the output power to the power channels

The total summation current of this example is 13.5 A. According to the summation current curve, the channels are only permitted to be operated with a maximum switching frequency of 300 Hz.

Channel 17	Channel 18	Channel 19	Channel 20	Channel 21
PWM-1 = 4 A	(not used)[1]	PWM 2 = 3.25 A	PWM 4 = 3.25 A	PWM 4 = 3 A

[1] Not used since the total current would be 4 A + 3.25 A = 7.25 A.

Incorrect distribution of the output power to the power channels

Channel 17	Channel 18	Channel 19	Channel 20	Channel 21
PWM 1 = 4 A[1]	PWM 1 = 4 A[1]	-	-	-

[1] **Notice!**
No error message is displayed by the module. Nevertheless, the module may be damaged.

13 Multifunction I/Os

The X90 CANopen bus controller is equipped with 32 multifunction I/Os:

- 2x MF-AI-T: Analog/Digital inputs
- 5x MF-AI-1: Analog/Digital inputs
- 5x MF-AI-2: Analog/Digital inputs
- 4x MF-DI: Counter/Analog/Digital inputs
- 8x MF-PWM: PWM/Digital outputs, analog/digital inputs
- 4x MF-PVG: PVG/Digital outputs, analog/digital inputs
- 2x MF-DO_minus: Digital sink outputs, analog/digital inputs
- 2x MF-DO_plus: Digital source outputs, analog/digital inputs

All multifunction I/Os can also be used as analog or digital inputs.

For the exact pinout of the channels on the CMC multi-header, see "Pinout" on page 9.

13.1 MF-AI-T: Analog/Digital inputs

Functions of channels 1 to 2:

- Analog input
  - Voltage measurement range from 0 to 10 V or 0 to 32 V
  - Current measurement range from 0 to 20 mA or 4 to 20 mA
  - Resistance measurement from 0 to 4 kΩ
  - Temperature measurement -80 to 270°C
- LED driver with max. 20 mA
- Digital input, configurable in 6 ways
  - Sink configuration
  - Diagnostic voltage/current measurement
  - Sink with threshold value (3 ways)

For details about the configuration, see "Channels 1 to 12 (analog/digital inputs)" on page 29.

13.2 MF-AI-1: Analog/Digital inputs

Functions of channels 3 to 7:

- Analog input
  - Voltage measurement range from 0 to 10 V or 0 to 32 V
  - Current measurement range from 0 to 20 mA or 4 to 20 mA
- LED driver with max. 20 mA
- Digital input, configurable in 5 ways
  - Sink configuration
  - Diagnostic current measurement
  - Sink with threshold value (3 ways)

For details about the configuration, see "Channels 1 to 12 (analog/digital inputs)" on page 29.

13.3 MF-AI-2: Analog/Digital inputs

Functions of channels 8 to 12:

- Analog input
  - Voltage measurement range from 0 to 10 V or 0 to 32 V
- Digital input, configurable in 8 ways
  - Sink/Source configuration
  - Diagnostic voltage measurement
  - Sink/Source with threshold value (5 ways)

For details about the configuration, see "Channels 1 to 12 (analog/digital inputs)" on page 29.

13.4 MF-DI: Counter/Analog/Digital inputs

Functions of channels 13 to 16:

- Counter input
  - Maximum input frequency 50 kHz
  - Measurement of positive and/or negative edges possible
  - ABR, AB and DF counters
  - Period duration and gate measurement
- Analog input
  - Voltage measurement range from 0 to 32 V
- Digital input, configurable in 7 ways
  - Sink/Source configuration
  - Diagnostic voltage measurement
  - Sink/Source with threshold value (4 ways)

For details about the configuration, see "Channels 13 to 16 (counter/analog/digital inputs)" on page 30.

13.5 MF-PWM: PWM/Digital outputs, analog/digital inputs

Functions of channels 17 to 24:

- PWM output
  - Maximum output current 4 A
  - Maximum frequency 1 kHz
  - Channels can be connected in parallel in pairs.
- Digital output
  - Channels can be connected in parallel in pairs.
- Analog input
  - Voltage measurement range from 0 to 32 V
- Digital input, configurable in 5 ways
  - Sink configuration
  - Diagnostic voltage measurement
  - Sink with threshold value (2 ways)

For details about the configuration, see "Channels 17 to 24 (PWM/digital outputs, analog/digital inputs)" on page 30.

13.6 MF-PVG: PVG/Digital outputs, analog/digital inputs

Functions of channels 25 to 28:

- PVG output
  - Proportional valve control
  - Maximum frequency 1 kHz
  - Maximum output current 10 mA at 24 V_{CC}
  - Supported types PVEA, H, S
- Digital output
- Analog input
  - Voltage measurement range from 0 to 32 V
- Digital input, configurable in 3 ways
  - Sink configuration
  - Diagnostic voltage measurement
  - Sink with threshold value

For details about the configuration, see "Channels 25 to 28 (PVG/digital outputs, analog/digital inputs)" on page 30.

13.7 MF-DO_minus: Digital sink outputs, analog/digital inputs

Functions of channels 29 to 30:

- Digital output
  - Maximum output current 4 A
  - Diagnostic function
- Analog input
  - Voltage measurement range from 0 to 32 V
- Digital input, configurable in 3 ways
  - Sink configuration
  - Diagnostic voltage measurement
  - Sink with threshold value

For details about the configuration, see "Channels 29 to 32 (digital outputs, analog/digital inputs)" on page 31.

13.8 MF-DO_plus: Digital source outputs, analog/digital inputs

Functions of channels 31 to 32:

- Digital output
  - Maximum output current 4 A
  - Channels can be connected in parallel.
  - Diagnostic function
- Analog input
  - Voltage measurement range from 0 to 32 V
- Digital input, configurable in 3 ways
  - Sink configuration
  - Diagnostic voltage measurement
  - Sink with threshold value

For details about the configuration, see "Channels 29 to 32 (digital outputs, analog/digital inputs)" on page 31.

14 Register description

14.1 Overview of registers

14.1.1 System requirements

The following minimum versions are recommended to generally be able to use all functions:

- Automation Studio 4.7
- Automation Runtime 4.7

14.1.2 Configuration - Overview of registers

CANopen object index xx01

The registers of the following table can be addressed by CANopen objects 0x3501 and 0x3601.

Index number **N** corresponds to the assigned channel on the CMC multi-header.

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
Configuration						
Channel mode						
513 + (N-1) * 64	CfgPinModeN (index N = 01 to 32)	USINT				●
Analog inputs						
515 + (N-1) * 64	CfgPinOptionAN (index N = 01 to 32) (analog filter)	USINT				●
522 + (N-1) * 64	CfgPinOptionDN (index N = 01 to 32) (upper limit value)	UINT				●
526 + (N-1) * 64	CfgPinOptionEN (index N = 01 to 32) (lower limit value)	UINT				●
Digital inputs						
515 + (N-1) * 64	CfgPinOptionAN (index N = 1 to 32) (digital filter)	USINT				●
522 + (N-1) * 64	CfgPinOptionDN (index N = 1 to 32) (threshold)	UINT				●
526 + (N-1) * 64	CfgPinOptionEN (index N = 1 to 32) (hysteresis)	UINT				●
Counter inputs						
1283 + (N-13) * 64	CfgPinOptionAN (index N = 13 to 16) (counter function mode)	USINT				●
1285 + (N-13) * 64	CfgPinOptionBN (index N = 13 to 16) (prescaler)	USINT				●
1287+ (N-13) * 64	CfgPinOptionCN (index N = 13 to 16) (average value)	USINT				●
1290 + (N-13) * 64	CfgPinOptionDN (index N = 13 to 16) (latch events)	UINT				●
1294 + (N-13) * 64	CfgPinOptionEN (index N = 13 to 16) (timeout)	UINT				●

CANopen object index xx02

The registers of the following table can be addressed by CANopen objects 0x3502 and 0x3602.

Index number **N** corresponds to the assigned channel on the CMC multi-header.

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
Configuration						
PWM outputs						
1539 + (N-17) * 64	CfgPinOptionAN (index N = 17 to 24) (measurement type)	USINT				•
1541 + (N-17) * 64	CfgPinOptionBN (index N = 17 to 24) (dither frequency)	USINT				•
1543 + (N-17) * 64	CfgPinOptionCN (index N = 17 to 24) (dither amplitude)	USINT				•
1550 + (N-17) * 64	CfgPinOptionEN (index N = 17 to 24) (measurement time)	UINT				•
1554 + (N-17) * 64	CfgPinOptionFN (index N = 17 to 24) (switch-on ramp)	UINT				•
1558 + (N-17) * 64	CfgPinOptionGN (index N = 17 to 24) (switch-off ramp)	UINT				•
1564 + (N-17) * 64	CfgPinOptionHN (index N = 17 to 24) (P value)	UDINT				•
1572 + (N-17) * 64	CfgPinOptionIN (index N = 17 to 24) (I value)	UDINT				•
PVG outputs						
2066 + (N-25) * 64	CfgPinOptionFN (index N = 25 to 28) (switch-on ramp)	UINT				•
2070 + (N-25) * 64	CfgPinOptionGN (index N = 25 to 28) (switch-off ramp)	UINT				•
Digital outputs						
1539 + (N-17) * 64	CfgPinOptionAN (index N = 17 to 24 and 29 to 32) (measurement type)	USINT				•
1550 + (N-17) * 64	CfgPinOptionEN (index N = 17 to 24 and 29 to 32) (measurement time)	UINT				•
System settings						
2562	CfgSysIo	UINT				•

14.1.3 Communication - Register overview

CANopen object index xx00

The registers of the following table can be addressed by CANopen objects 0x3200 to 0x3400.

Index number **N** or the number on the register name corresponds to the assigned channel on the CMC multi-header.

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
Digital inputs						
Input state						
1	Input state of digital inputs 01 to 08	USINT	•			
	DigitalInput01	Bit 0				
	...	...				
	DigitalInput08	Bit 7				
3	Input state of digital inputs 09 to 16	USINT	•			
	DigitalInput09	Bit 0				
	...	...				
	DigitalInput16	Bit 7				
5	Input state of digital inputs 17 to 24	USINT	•			
	DigitalInput17	Bit 0				
	...	...				
	DigitalInput24	Bit 7				
7	Input state of digital inputs 25 to 32	USINT	•			
	DigitalInput25	Bit 0				
	...	...				
	DigitalInput32	Bit 7				
Open circuit or error status						
9	Input status 1	USINT	•			
	WireBreak01	Bit 0				
	...	...				
	WireBreak08	Bit 7				
11	Input status 1	USINT	•			
	WireBreak09	Bit 0				
	...	...				
	WireBreak16	Bit 7				
13	Input status 1	USINT	•			
	WireBreak17	Bit 0				

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
	...	...				
	WireBreak24	Bit 7				
15	Input status 1	USINT	●			
	WireBreak25	Bit 0				
	...	...				
	WireBreak32	Bit 7				
Overload or short circuit status						
17	Input status 2	USINT	●			
	ShortCircuit01	Bit 0				
	...	...				
	ShortCircuit08	Bit 7				
19	Input status 2	USINT	●			
	ShortCircuit09	Bit 0				
	...	...				
	ShortCircuit16	Bit 7				
21	Input status 2	USINT	●			
	ShortCircuit17	Bit 0				
	...	...				
	ShortCircuit24	Bit 7				
23	Input status 2	USINT	●			
	ShortCircuit25	Bit 0				
	...	...				
	ShortCircuit32	Bit 7				
Analog inputs						
Input state						
50 + (N-1) * 4	AnalogInputN (index N = 01 to 32)	INT	●			
50 + (N-1) * 4	CurrentN (index N = 01 to 24)	INT	●			
146 + (N-25) * 4	VoltageN (index N = 25 to 28)	INT	●			
162 + (N-29) * 4	CurrentN (index N = 29 to 32)	INT	●			
Underflow status						
25	Status of underflow 01 to 08	USINT	●			
	UnderflowAnalogInput01	Bit 0				
	...	...				
	UnderflowAnalogInput08	Bit 7				
27	Status of underflow 09 to 16	USINT	●			
	UnderflowAnalogInput09	Bit 0				
	...	...				
	UnderflowAnalogInput16	Bit 7				
29	Status of underflow 17 to 24	USINT	●			
	UnderflowAnalogInput17	Bit 0				
	...	...				
	UnderflowAnalogInput24	Bit 7				
31	Status of underflow 25 to 32	USINT	●			
	UnderflowAnalogInput25	Bit 0				
	...	...				
	UnderflowAnalogInput32	Bit 7				
Overflow status						
33	Status of overflow 01 to 08	USINT	●			
	OverflowAnalogInput01	Bit 0				
	...	...				
	OverflowAnalogInput08	Bit 7				
35	Status of overflow 09 to 16	USINT	●			
	OverflowAnalogInput09	Bit 0				
	...	...				
	OverflowAnalogInput16	Bit 7				
37	Status of overflow 17 to 24	USINT	●			
	OverflowAnalogInput17	Bit 0				
	...	...				
	OverflowAnalogInput24	Bit 7				
39	Status of overflow 25 to 32	USINT	●			
	OverflowAnalogInput25	Bit 0				
	...	...				
	OverflowAnalogInput32	Bit 7				
Measurement range overshoot status						
41	Status of overshoot 01 to 08	USINT	●			
	OutOfRangeAnalogInput01	Bit 0				
	...	...				
	OutOfRangeAnalogInput08	Bit 7				
43	Status of overshoot 09 to 16	USINT	●			
	OutOfRangeAnalogInput09	Bit 0				
	...	...				
	OutOfRangeAnalogInput16	Bit 7				

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
45	Status of overshoot 17 to 24	USINT	●			
	OutOfRangeAnalogInput17	Bit 0				
	...	...				
	OutOfRangeAnalogInput24	Bit 7				
47	Status of overshoot 25 to 32	USINT	●			
	OutOfRangeAnalogInput25	Bit 0				
	...	...				
	OutOfRangeAnalogInput32	Bit 7				
Counter inputs						
178 + (N-13) * 4	CounterN (index N = 13 to 16)	INT	●			
178	Encoder13	INT	●			
182[1]	Encoder13	INT	●			
186	Encoder15	INT	●			
194 + (N-13) * 4	LatchCounterValueN (index N = 13 to 16)	INT	●			
198[1]	LatchCounterValue13	INT	●			
210 + (N-13) * 4	LatchCounterEventsN (index N = 13 to 16)	INT	●			
214	LatchCounterEvents13	INT	●			
228 + (N-13) * 8	GateTimeN (index N = 13 to 16)	UDINT	●			
228 + (N-13) * 8	PeriodN (index N = 13 to 16)	UDINT	●			
387	Switching the latch on/off	USINT			●	
	LatchEnable13	Bit 4				
	...	...				
	LatchEnable16	Bit 7				
391	Switching the latch on/off	USINT			●	
	LatchEnable13	Bit4				
395	Clear the counter value	USINT			●	
	CounterReset13	Bit 4				
	...	...				
	CounterReset16	Bit 7				
399	Clear the counter value	USINT			●	
	EncoderReset13	Bit 4				
	EncoderReset15	Bit 6				
Digital outputs						
Switching state						
385	State of LED/digital outputs 1 to 7	USINT			●	
	DigitalOutput01	Bit 0				
	...	...				
	DigitalOutput07	Bit 6				
389	State of digital outputs 17 to 24	USINT			●	
	DigitalOutput17	Bit 0				
	...	...				
	DigitalOutput24	Bit 7				
391	State of digital outputs 25 to 32	USINT			●	
	DigitalOutput25	Bit 0				
	...	...				
	DigitalOutput32	Bit 7				
Open circuit or error status						
9	Output status 1	USINT	●			
	ErrorDigitalOutput01	Bit 0				
	...	...				
	ErrorDigitalOutput07	Bit 6				
13	Output status 1	USINT	●			
	ErrorDigitalOutput17	Bit 0				
	...	...				
	ErrorDigitalOutput24	Bit 7				
15	Output status 1	USINT	●			
	ErrorDigitalOutput25	Bit 0				
	...	...				
	ErrorDigitalOutput32	Bit 7				
Overload or short circuit status						
21	Output status 2	USINT	●			
	Overload17	Bit 0				
	...	...				
	Overload24	Bit 7				
23	Output status 2	USINT	●			
	Overload25	Bit 0				
	...	...				
	Overload32	Bit 7				
Clear overload error						
397	Acknowledgment of overload shutdown	USINT			●	
	OverloadClear17	Bit 0				
	...	...				
	OverloadClear24	Bit 7				

Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
399	Acknowledgment of overload shutdown	UINT			●	
	OverloadClear29	Bit 4				
	...	...				
	OverloadClear32	Bit 7				
PWM + PVG						
402 + (N-17) * 4	PWMDutyN (index N = 17 to 24)	UINT			●	
402 + (N-17) * 4	PWMCurrentN (index N = 17 to 24)	UINT			●	
450	PWMPeriodGroup01	INT			●	
454	PWMPeriodGroup02	INT			●	
434 + (N-25) * 4	PVGDuty (index N = 25 to 28)	INT			●	
482	PVGPeriodGroup01	INT			●	
466	PVGPeriodGroup02	INT			●	
391	PVG enabling	USINT	●			
	OutputEnable25	Bit 0				
	...	...				
	OutputEnable28	Bit 3				
General system information						
258	SysStat	INT	●			
262	AiCurrentSum	INT	●			
266	AiVextSensor	INT	●			
270	AiInt	INT	●			
274	AiVDriveIn	INT	●			
278	AiVDriveOut	INT	●			
282	AiTemp	INT	●			
286	AiNode	INT	●			
497	CfgVDriveOut	USINT			●	
	OverloadShutdownClear	Bit 2				

1) Only with "DF" counter configuration

14.2 Multiconfiguration of I/O channels

This overview includes only the CfgPinMode and CfgPinOptionx registers, which are capable of multiconfiguration.

CfgPinOptionx registers containing only 1 configuration option are not listed in this overview.

14.2.1 Physical configuration

Name:
CfgPinMode01 to CfgPinMode12
CfgPinMode13 to CfgPinMode16
CfgPinMode17 to CfgPinMode24
CfgPinMode25 to CfgPinMode28
CfgPinMode29 to CfgPinMode28

These registers configure the function of the channels on the CMC multi-header.

- Channels 1 to 12 (analog/digital inputs)
- Channels 13 to 16 (counter/analog/digital inputs)
- Channels 17 to 24 (PWM/digital outputs, analog/digital inputs)
- Channels 25 to 28 (PVG/digital outputs, analog/digital inputs)
- Channels 29 to 32 (digital outputs, analog/digital inputs)

Channels 1 to 12 (analog/digital inputs)

Channels 1 to 12 can be configured as both digital as well as analog inputs.

Digital inputs can be configured as sink or source. In addition, the input can be used to control an LED.

Data type	Values	Information
USINT	0	Channel switched off
	1	LED driver
	32	Digital input source configuration
	42	Digital input sink configuration
	50	Digital input with voltage measurement diagnostics
	51	Digital input with current measurement diagnostics
	62	Digital input source with threshold value
	67	Digital input source with ratiometric threshold value
	72	Digital input sink with threshold value
	75	Digital input Sink with ratiometric threshold and external pull-up
	77	Digital input sink with ratiometric threshold value
	80	Analog input 0 to 10 V
	81	Analog input 0 to 32 V
	82	Analog input 0 to 20 mA
	83	Analog input 4 to 20 mA
	84	Resistance measurement
	85	Temperature measurement (PT1000)

Not every function can be assigned to every input. The following matrix provides an overview of the possible configurations:

Function	Channel											
	1	2	3	4	5	6	7	8	9	10	11	12
LED driver	X	X	X	X	X	X	X					
Digital input source configuration								X	X	X	X	X
Digital input sink configuration	X	X	X	X	X	X	X	X	X	X	X	X
Digital input with voltage measurement diagnostics	X	X						X	X	X	X	X
Digital input with current measurement diagnostics	X	X	X	X	X	X	X					
Digital input source with threshold value								X	X	X	X	X
Digital input source with ratiometric threshold value								X	X	X	X	X
Digital input sink with threshold value	X	X	X	X	X	X	X	X	X	X	X	X
Digital input sink with ratiometric threshold value	X	X	X	X	X	X	X	X	X	X	X	X
Digital input Sink with ratiometric threshold and external pull-up	X	X	X	X	X	X	X	X	X	X	X	X
Analog input 0 to 10 V	X	X	X	X	X	X	X	X	X	X	X	X
Analog input 0 to 32 V	X	X	X	X	X	X	X	X	X	X	X	X
Analog input 0 to 20 mA	X	X	X	X	X	X	X					
Analog input 4 to 20 mA	X	X	X	X	X	X	X					
Resistance measurement	X	X										
Temperature measurement (PT1000)	X	X										

Channels 13 to 16 (counter/analog/digital inputs)

Channels 13 to 16 can be configured as both digital as well as analog inputs.

Digital inputs can be configured as sink or source. In addition, the inputs can be used as counter or encoder inputs.

Data type	Values	Information
USINT	0	Channel switched off
	32	Digital input source configuration
	42	Digital input sink configuration
	50	Digital input with voltage measurement diagnostics
	62	Digital input source with threshold value
	67	Digital input source with ratiometric threshold value
	72	Digital input sink with threshold value
	75	Digital input Sink with ratiometric threshold and external pull-up
	81	Analog input 0 to 32 V
	90	Counter - No pull-up/pull-down[1]
	91	Counter - Source[1]
	92	Counter - Sink[1]

[1] The exact counter configuration must also be configured in register "CfgPinOptionA" on page 40.

Input and counter functions are available for all channels. The encoder function cannot be assigned to every input. The following matrix provides an overview of the possible configurations:

Function	Channel				
	13	14	15	16	
Event counters	X	X	X	X	
AB encoders	X		X		
ABR encoders	X				
DF encoders	X				

Channels 17 to 24 (PWM/digital outputs, analog/digital inputs)

Channels 17 to 24 can be configured as both digital or analog inputs as well as digital outputs. The outputs support PWM.

A parallel connection is possible when configured as a digital output or PWM. A parallel connection is only possible with outputs that follow each other in pairs (channels 1 and 2, 3 and 4, etc.)

The main output is always the channel with the lower channel number. The summation current of the channels connected in parallel is obtained by adding the two channel values.

Data type	Values	Information
USINT	0	Channel switched off
	1	Digital output
	10	PWM
	11	PWM with current control
	20	Output connected in parallel with previous channel
	42	Digital input sink configuration
	50	Digital input with voltage measurement diagnostics
	72	Digital input sink with threshold value
	75	Digital input Sink with ratiometric threshold and external pull-up
	81	Analog input 0 to 32 V

Channels 25 to 28 (PVG/digital outputs, analog/digital inputs)

Channels 25 to 28 can be configured as both digital or analog inputs as well as digital outputs. The outputs support PVG.

Data type	Values	Information
USINT	0	Channel switched off
	1	Digital output
	10	PVG
	42	Digital input sink configuration
	50	Digital input with voltage measurement diagnostics
	72	Digital input sink with threshold value
	81	Analog input 0 to 32 V

Channels 29 to 32 (digital outputs, analog/digital inputs)

Channels 29 to 31 can be configured as both digital or analog inputs as well as digital outputs. A parallel connection is only possible for channels 31 and 32.

Data type	Values	Information
USINT	0	Channel switched off
	1	Digital output
	20	Output connected in parallel with previous channel[1]
	42	Digital input sink configuration
	50	Digital input with voltage measurement diagnostics
	72	Digital input sink with threshold value
	81	Analog input 0 to 32 V

[1] Only channels 31 and 32

14.2.2 CfgPinOptionA

Depending on the set operating mode, different configurations for the respective channel can be made using this register.

Settings can be made for the following operating modes:

● Analog input:	Analog input filter
● Digital input:	Digital input filter
● Digital output:	Current measurement - Measurement type
● Counter function:	Counter or encoder function mode
● PWM mode:	Current measurement - Measurement type

14.2.3 CfgPinOptionB

Depending on the set operating mode, different configurations for the respective channel can be made using this register.

Settings can be made for the following operating modes:

● Counter function:	Prescaler configuration
● PWM mode:	Dither frequency

14.2.4 CfgPinOptionC

Depending on the set operating mode, different configurations for the respective channel can be made using this register.

Settings can be made for the following operating modes:

● Counter function:	Average value configuration
● PWM mode:	Dither amplitude

14.2.5 CfgPinOptionD

Depending on the set operating mode, different configurations for the respective channel can be made using this register.

Settings can be made for the following operating modes:

● Analog input:	Upper limit value
● Digital input:	Threshold
● Counter function:	Latch event configuration

14.2.6 CfgPinOptionE

Depending on the set operating mode, different configurations for the respective channel can be made using this register.

Settings can be made for the following operating modes:

● Analog input:	Lower limit value
● Digital input:	Hysteresis
● Digital output:	Current measurement - Measurement period
● Counter function:	Timeout
● PWM mode:	Current measurement - Number of PWM periods

14.3 Digital inputs

The module is equipped with 32 digital inputs for 1-wire connections. The inputs of the module are designed for sink and/or source circuits.

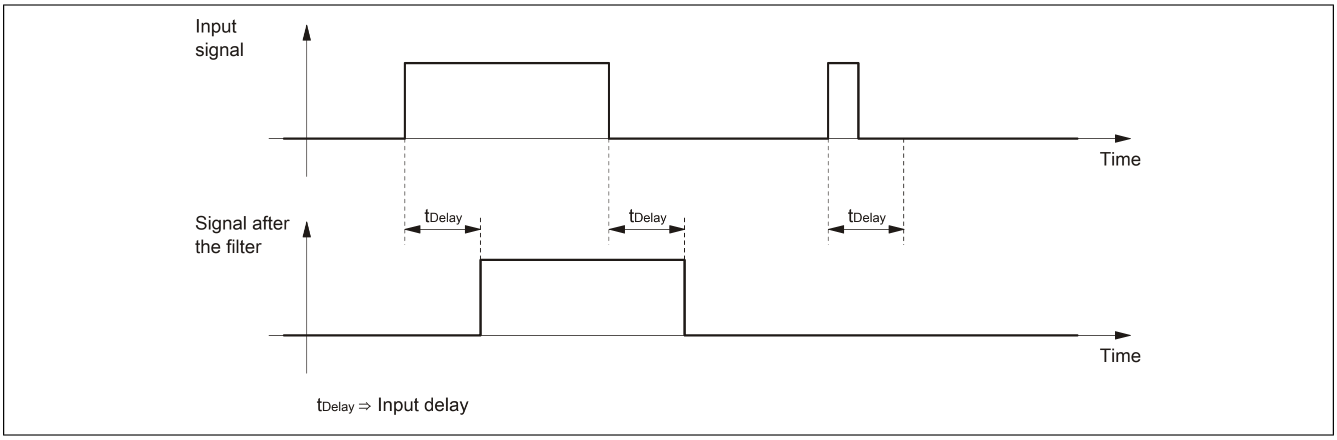
Input impedance is tightly defined by the physical configuration.

14.3.1 Digital input filter

Name:
CfgPinOptionA01 to CfgPinOptionA32

The filter value for all digital inputs is configured in 100-μs steps in this register. The filter is implemented as a ramp filter.

Data type	Values	Information
USINT	0	No software filter
	1	0.1 ms
	...	
	10	1 ms (default)
	...	
	250	25 ms - Higher values are limited to this value.



14.3.2 Configurable switching threshold

Name:
CfgPinOptionD01 to CfgPinOptionD32
CfgPinOptionE01 to CfgPinOptionE32

A switching threshold with associated hysteresis can be configured for channels 9 to 24.

Registers CfgPinOptionD01 to CfgPinOptionD32:

Threshold configuration

Data type	Values	Information
INT	0 to 32,000	Corresponds to 0 to 32 V if an absolute switching threshold is configured.
	0 to 1000	Corresponds to 0 to 100% if a ratiometric switching threshold is configured.

If the pin is configured as a digital input, the switching threshold [mV] can be set with this register. When taking into account the configured hysteresis, a voltage level under the threshold value results in "0" on the corresponding bit; a voltage level above the threshold value results in "1".

Example

Desired level: 16 V configuration value: 16000

Registers CfgPinOptionE01 to CfgPinOptionE32:

Hysteresis configuration

Data type	Values	Information
INT	0 to 32,000	Corresponds to 0 to 15 V if an absolute switching threshold is configured.
	0 to 1000	Corresponds to 0 to 100% if a ratiometric switching threshold is configured.

If the pin is configured as a digital input, the hysteresis [mV] can be set with this register in order to avoid frequent state changes in the measurement range near the threshold value. When taking into account the configured threshold value, a voltage level under threshold value "Threshold - Hysteresis" results in "0" on the corresponding bit; a voltage level above threshold value "Threshold + Hysteresis" results in "1".

Example

Desired hysteresis level: ±5 V configuration value: 5000

Notice!

The sum of hysteresis CfgPinOptionExx and threshold CfgPinOptionDxx is not permitted to exceed the limit of >32 V or >100%.

The difference between hysteresis CfgPinOptionExx and threshold CfgPinOptionDxx is not permitted to be negative.

14.3.3 Input state of the digital inputs

Name:
DigitalInput01 to DigitalInput08
DigitalInput09 to DigitalInput16
DigitalInput17 to DigitalInput24
DigitalInput25 to DigitalInput32

This register contains the input state of digital inputs 1 to 32.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	DigitalInputxx[1]	0 or 1	Input state of digital input x
...	...	...	...
7	DigitalInputxx + 7	0 or 1	Input state of digital input x + 7

[1] For xx, see the name of the register.

14.3.4 Status of the digital inputs

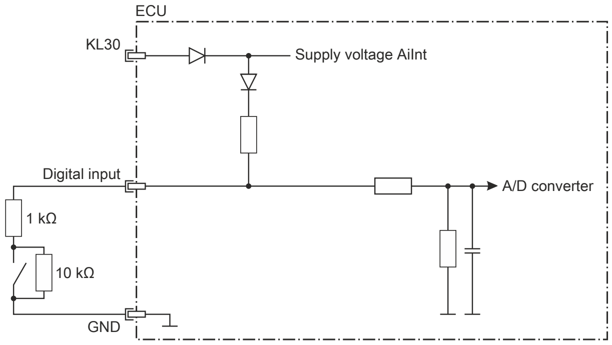
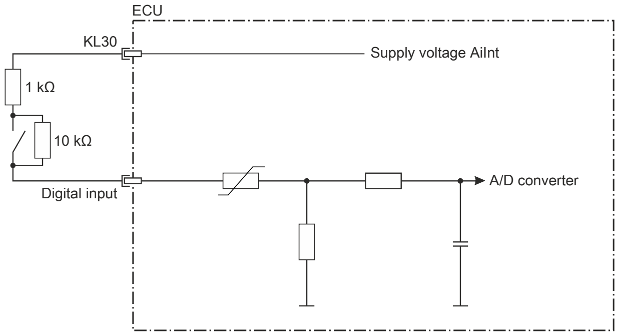
Name:
ShortCircuit01 to ShortCircuit08
ShortCircuit09 to ShortCircuit16
ShortCircuit17 to ShortCircuit24
ShortCircuit25 to ShortCircuit32
WireBreak01 to WireBreak08
WireBreak09 to WireBreak16
WireBreak17 to WireBreak24
WireBreak25 to WireBreak32

The module is equipped with open circuit and short circuit detection. For this, the sensor must be connected to resistors accordingly.

Diagnostic values are only valid in configuration "Digital input with voltage measurement diagnostics" or "Digital input with current measurement diagnostics".

The resistors are connected in series or parallel to the sensor. The following values are defined for the resistances:

Resistance	Value
Serial (R_S)	1 kΩ
Parallel (R_P)	10 kΩ

<table>
<tr><th>Sensor circuit</th><th>Description</th><th>Detection</th></tr>
<tr><td>
</td><td>Series and parallel resistance in mode "Diagnostics-capable voltage input"</td><td>Open circuit and short circuit</td></tr>
<tr><td>
</td><td>Series and parallel resistance in mode "Diagnostics-capable current input"</td><td>Open circuit and short circuit</td></tr>
</table>

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	ShortCircuitxx[1]	0	No short circuit
		1	Short circuit
...	...	...	...
7	ShortCircuitxx + 7	0	No short circuit
		1	Short circuit

[1] For xx, see the name of the register.

Bit	Description	Value	Information
0	WireBreakxx[1]	0	No error
		1	Error occurred
...	...	...	...
7	WireBreakxx + 7	0	No error
		1	Error occurred

[1] For xx, see the name of the register.

14.4 Analog inputs

The module is equipped with 32 analog inputs for 1-wire connections.

14.4.1 Analog input filter

Name:
CfgPinOptionA01 to CfgPinOptionA32

A filter can be defined to prevent large input jumps. This filter is used to bring the input value closer to the actual analog value over a period of several system cycles. Filtering takes place after any input ramp limiting has been carried out.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 to 3	Filter level	0	Disabled (default)
		1	Filter level 2
		2	Filter level 4
		3	Filter level 8
		4	Filter level 16
		5	Filter level 32
		6	Filter level 64
		7	Filter level 128
4 to 7	Input ramp limiting	0	Disabled (default)
		1	Limit value = 16383
		2	Limit value = 8191
		3	Limit value = 4095
		4	Limit value = 2047
		5	Limit value = 1023
		6	Limit value = 511
		7	Limit value = 255

Input ramp limiting

The difference of the input value change is checked for exceeding the specified limit. In the event of overshoot, the tracked input value is equal to the old value ± the limit value.

Example 1

The input value jumps from 8000 to 17000. The diagram shows the tracked input value with the following settings:

Input ramp limiting = 4 = 0x07FF = 2047

Filter level = 2

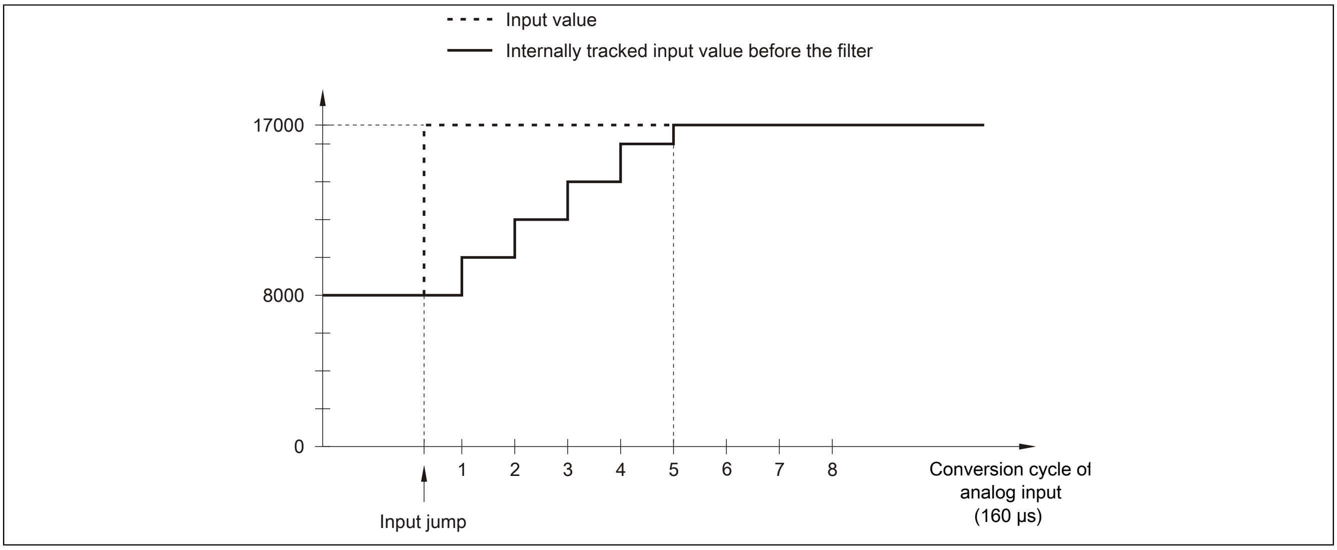


Figure 1: Tracked input value for input jump

Example 2

A disturbance interferes with the input value. The diagram shows the tracked input value with the following settings:

Input ramp limiting = 4 = 0x07FF = 2047

Filter level = 2

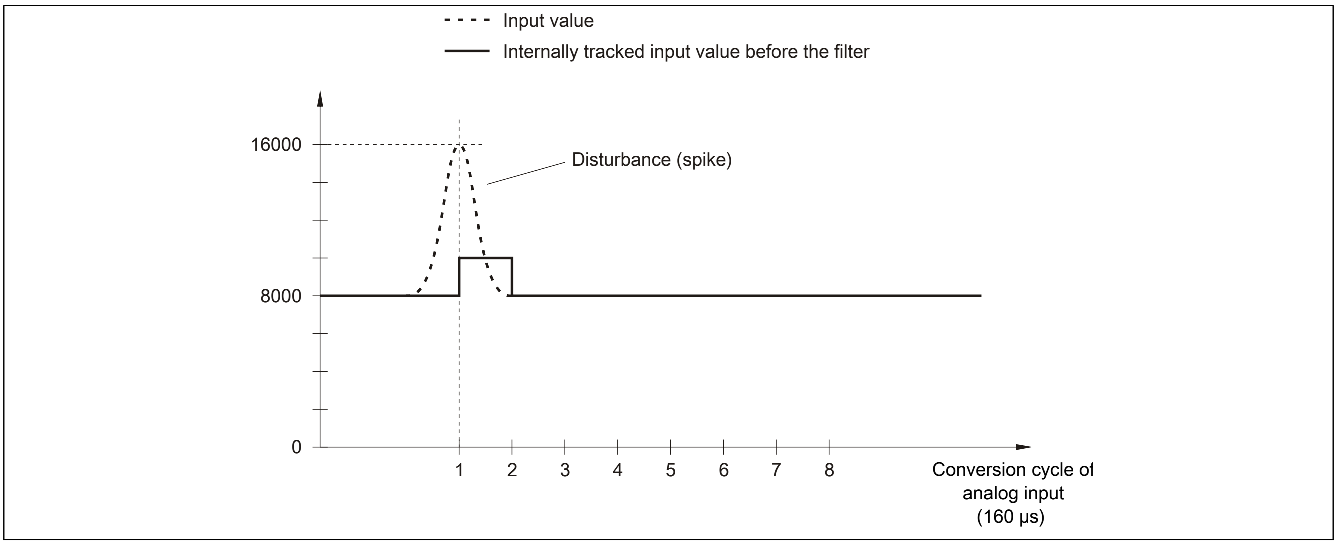


Figure 2: Adjusted input value for disturbance

Filter level

The input value is evaluated more or less strongly depending on the filter level. Input ramp limiting can then be applied based on this evaluation.

Formula for evaluating the input value:

$$Value_{New} = Value_{Old} - \frac{Value_{Old}}{Filter\ level} + \frac{Input\ value}{Filter\ level}$$

Example 1

The input value jumps from 8000 to 16000. The diagram shows the calculated value with the following settings:

Input ramp limiting = 0

Filter level = 2 or 4

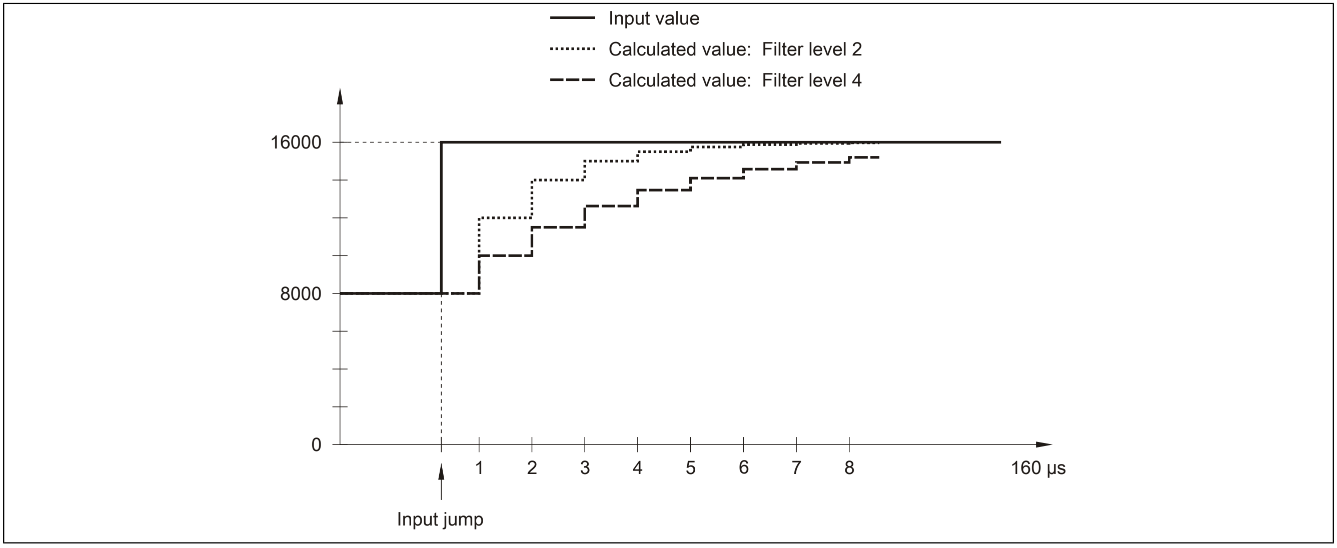


Figure 3: Calculated value during input jump

Example 2

A disturbance interferes with the input value. The diagram shows the calculated value with the following settings:

Input ramp limiting = 0

Filter level = 2 or 4

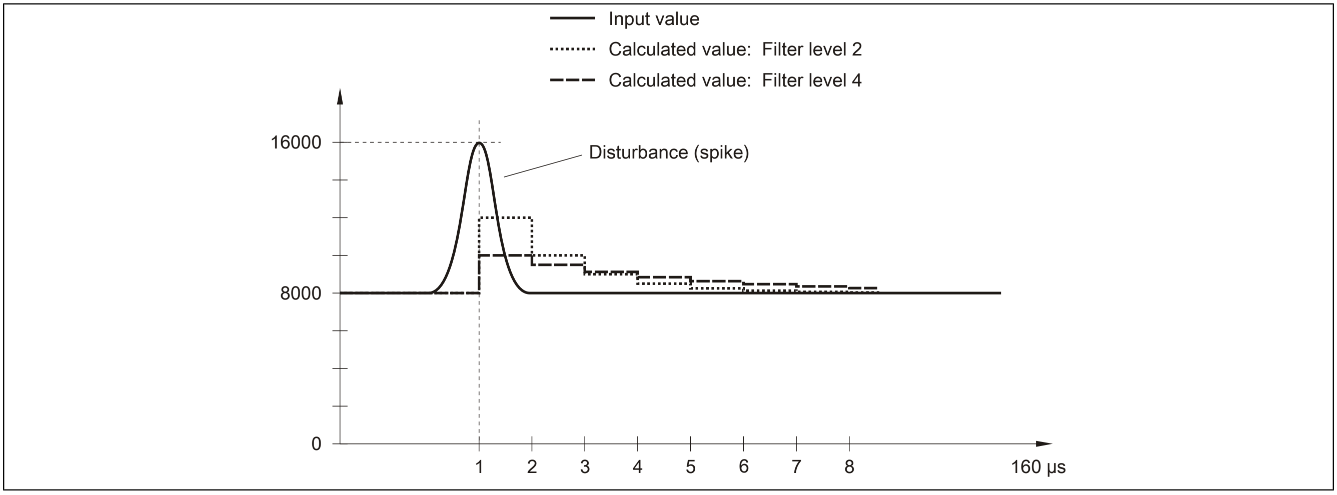


Figure 4: Calculated value during disturbance

14.4.2 Upper and lower limit value

Name:
CfgPinOptionD01 to CfgPinOptionD32 (upper limit value)
CfgPinOptionE01 to CfgPinOptionE32 (lower limit value)

The lower/upper limit value of the analog value is set in these registers. If the analog value goes above or below the respective limit value, it is frozen at this value and the corresponding error status bits are set.

Data type	Lower limit value	Upper limit value	Information
INT	0	32767	Analog input 0 to 10 VDC
			Analog input 0 to 32 VDC
			Analog input 0 to 20 mA
	0	4000	Resistance measurement 1 to 4000 Ω
	-8192	32767	Analog input 4 to 20 mA
	-800	2700	Temperature measurement -80 to 270°C

14.4.3 Input values of analog inputs

Name:
AnalogInput01 to AnalogInput32
Current01 to Current24
Current29 to Current32
Voltage25 to Voltage28

These registers contain the analog input value depending on the configured operating mode.

Data type	Values	Information
INT	0 to 32767	0 to 10 VDC
		0 to 32 VDC
		0 to 20 mA
	-8192 to 32767	4 to 20 mA (-8192 = 0 mA, 0 = 4 mA, 32767 = 20 mA)
	-800 to 2700	PT1000 temperature measurement (-80 to 270°C)
	0 to 4000	Resistance measurement 0 to 4000 Ω

14.4.4 Status of the analog inputs

Name:
OutOfRangeAnalogInput01 to OutOfRangeAnalogInput08
OutOfRangeAnalogInput09 to OutOfRangeAnalogInput16
OutOfRangeAnalogInput17 to OutOfRangeAnalogInput24
OutOfRangeAnalogInput25 to OutOfRangeAnalogInput32
OverflowAnalogInput01 to OverflowAnalogInput08
OverflowAnalogInput09 to OverflowAnalogInput16
OverflowAnalogInput17 to OverflowAnalogInput24
OverflowAnalogInput25 to OverflowAnalogInput32
UnderflowAnalogInput01 to UnderflowAnalogInput08
UnderflowAnalogInput09 to UnderflowAnalogInput16
UnderflowAnalogInput17 to UnderflowAnalogInput24
UnderflowAnalogInput25 to UnderflowAnalogInput32

The state of the analog inputs is stored in these registers. The following states are monitored:

- Overflow
- Underflow
- Measurement range overshoot

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	OutOfRangeAnalogInputxx[1]	0	Measured value within limit values
		1	Measured value outside limit values
...	...	...	...
7	OutOfRangeAnalogInputxx + 7	0	Measured value within limit values
		1	Measured value outside limit values

[1] For xx, see the name of the register.

Bit structure:

Bit	Description	Value	Information
0	OverflowAnalogInputxx UnderflowAnalogInputxx[1]	0	No limit value underflow or overflow
		1	Limit value underflow or overflow
...	...	...	...
7	OverflowAnalogInputxx + 7 UnderflowAnalogInputxx + 7	0	No limit value underflow or overflow
		1	Limit value underflow or overflow

[1] For xx, see the name of the register.

Limiting the analog value

In addition to the status information, the analog value is set to the values listed below by default when an error occurs. The analog value is limited to the new values if the limit values were changed.

Error state	Digital value on error (default values)		
	4 to 20 mA	Resistance	Temperature
Upper limit value overshot	32767	4000	2700
Lower limit value undershot	-8192	0	-800

14.5 Counter functions

High-speed digital inputs 13 to 16 can be used for counter functions. The following functions are available.

- Event counters
- AB incremental counter
- DF counter function
- ABR counter function

A latch function is also available for the counters.

14.5.1 Counter or encoder function mode

Name:
CfgPinOptionA13 to CfgPinOptionA16

If the channel is configured as a counter or encoder input, the mode of the function is set in this register:

Data type	Values
USINT	See the bit structure.

Bit structure:

Values	Information
0	Disabled
1	Edge counter (both edges)
2	Period measurement
3	Gate measurement
4	AB encoders
5	DF counter
6	ABR encoders
17	Edge counter (falling edge)
19	Gate time measurement low-active
21	DF counter (falling edge)
33	Edge counter (rising edge)
35	Gate time measurement high-active
37	DF counter (rising edge)

Information:

An AB counter can only be configured on channel 13 or channel 15; an ABR or DF counter can only be configured on channel 13.

Counter	Channel			
	13	14	15	16
Event counters	X	X	X	X
AB incremental counter	A	B	A	B
ABR counter function	A	B	R[3]	E[4]
DF counter function	D[1]	F[2]	R[3]	E[4]

[1] Direction
[2] Frequency
[3] Reference pulse
[4] Reference enable (Enable)

14.5.2 Prescaler configuration

Name:
CfgPinOptionB13 to CfgPinOptionB16

The prescaler of the counter can be configured with these registers.

Example:

Set prescaler:	500 kHz
Displayed counter value:	2500
Frequency:	500 kHz / 2500 = 200 Hz
Period duration:	1 / 200 Hz = 5 ms

Data type	Values	Information
INT	0	1 Mhz
	1	500 kHz
	2	250 kHz
	3	125 kHz
	4	62.5 kHz
	5	31.25 kHz
	6	15.625 kHz
	7	7812.5 Hz
	8	3906.25 Hz
	9	1953.13 Hz
	10	976.56 Hz
	11	488.28 Hz
	12	244.14 Hz
	13	122.07 Hz
	14	61.04 Hz
	15	30.52 Hz
	16	15.26 Hz

14.5.3 Average value configuration

Name:
CfgPinOptionC13 to CfgPinOptionC16

Contains the number of measured periods that are used to calculate the average for gate measurement and period measurement.

Data type	Values	Information
UINT	0	No averaging
	1 to 65535	Number of measured cycles

14.5.4 Latch event configuration

Name:
CfgPinOptionD13 to CfgPinOptionD16

These registers determine at which states of A, B, R or D, F the counter values of the counter are applied to the associated latch registers. When applied, the associated latch event counter is incremented.

Data type	Values
INT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	Input 13 high level	0	Disabled
		1	Enabled
...	...		
3	Input 16 high level	0	Disabled
		1	Enabled
4	Input 13 low level	0	Disabled
		1	Enabled
...	...		
7	Input 16 low level	0	Disabled
		1	Enabled
8 to 15	Latch mode of the counter	0	Single shot
		1	Continuous
		2 to 254	Reserved
		255	Disabled

14.5.5 Timeout

Name:
CfgPinOptionE13 to CfgPinOptionE16

A timeout for period measurement is set in these registers. If an edge is not detected in the specified time frame, the corresponding counter is set to 0.

Data type	Values	Information
UINT	10 to 50,000	Timeout of counter x (1 ms to 5 s) (1 LSB = 100 µs)

14.5.6 Counter value

Name:
Counter13 to Counter16
Encoder13 and Encoder15

The current counter values or encoder values are saved in these registers.

Data type	Values	Information
INT	-32768 to 32767	Current counter value

14.5.7 Switching the latch on/off

Name:
LatchEnable13 to LatchEnable16

These registers start the latch procedure with the corresponding bit.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 3	Reserved		
4	LatchEnable13	0	Disabled
		1	Enabled
...	...	...	
7	LatchEnable16	0	Disabled
		1	Enabled

14.5.8 Clear the counter value

Name:
CounterReset13 to CounterReset16
EncoderReset13 and EncoderReset15

These registers reset the counter or encoder value with the corresponding bit.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 3	Reserved		
4	CounterReset13	0	Do not clear the counter value
		1	Clear the counter value
...	...	...	
7	CounterReset16	0	Do not clear the counter value
		1	Clear the counter value

Bit structure:

Bit	Description	Value	Information
0 - 3	Reserved		
4	EncoderReset13	0	Do not clear the encoder value
		1	Clear the encoder value
5	Reserved	...	
6	EncoderReset15	0	Do not clear the encoder value
		1	Clear the encoder value
7	Reserved		

14.5.9 Latched counter value

Name:
LatchCounterValue13 to LatchCounterValue16

As soon as the latch conditions have been met, the contents of the respective counter are copied to these registers.

Data type	Values	Information
INT	-32768 to 32767	Latched counter value

14.5.10 Counter value of latch events

Name:
LatchCounterEvents13 to LatchCounterEvents16

These registers hold the counter values of latch events that have occurred. This allows detection of whether a new latched counter value has been saved.

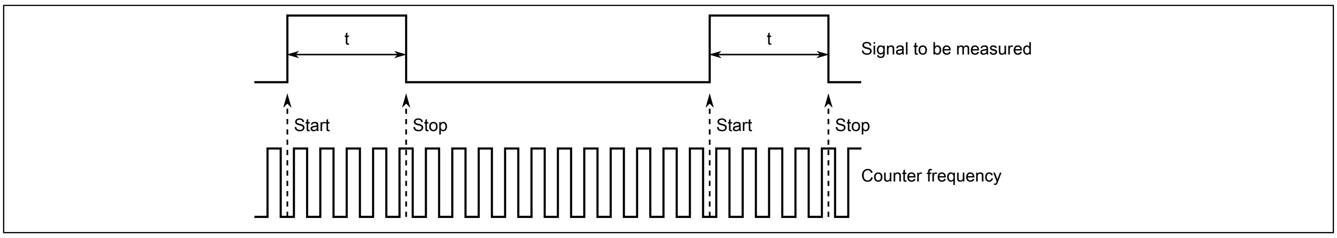
Data type	Values	Information
INT	-32768 to 32767	Latched counter value

14.5.11 Period time measurement

Name:
GateTime13 to GateTime16

The average times over the number of measured periods are indicated in these registers.

Data type	Values	Information
INT	-32768 to 32767	



The measurement can begin at the decreasing or increasing edge depending on the configuration register. Measurement always occurs up to the next edge. The counter frequency can be set in 2^n steps (1 MHz to 15.2 Hz).

The measured counter value is a 32-bit value and displayed in counters 13 to 16.

A measured value acquisition every 250 μs is possible. Faster signals are rejected. The measured value equals 0 on 4 consecutive errors. If the time or measured value is exceeded, the measured value is set to 0xFFFFFFFF. The measured value is averaged based on the set sampling periods.

14.6 Digital outputs

The module is equipped with 23 digital outputs for 1-wire connections.

14.6.1 Output status of the digital outputs

Name:
DigitalOutput01 to DigitalOutput07
DigitalOutput17 to DigitalOutput24
DigitalOutput25 to DigitalOutput32

These registers contain the initial state of digital LED states 1 to 7 and digital outputs 17 to 32.

> ## Information:
>
> **Only source outputs can be connected in parallel to the corresponding main channels.**

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	DigitalOutputxx[1]	0 or 1	Output state of digital output xx
...	...	...	...
7	DigitalOutputxx + 7	0 or 1	Output state of digital output xx + 7

[1] For xx, see the name of the register.

14.6.2 Status of the digital outputs

Name:
ErrorDigitalOutput01 to ErrorDigitalOutput07
ErrorDigitalOutput17 to ErrorDigitalOutput24
ErrorDigitalOutput25 to ErrorDigitalOutput32
Overload17 to Overload24
Overload25 to Overload32

These registers contain the state of LED outputs 1 to 7 and digital outputs 17 to 32.

Output error

If the switched output level of a digital output does not correspond to the read back value, then the corresponding bit is set. As soon as the switched output level again corresponds to the read back value, the corresponding bit will be reset again.

If a pin is configured as PWM or DI, then the corresponding bit is not maintained.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	ErrorDigitalOutputxx[1]	0	No error
		1	Error occurred
...	...	...	...
7	ErrorDigitalOutputxx + 7	0	No error
		1	Error occurred

[1] For xx, see the name of the register.

Overload shutdown

If the output is cut off due to an overload, then the corresponding bit in the register is set and the channel is cut off. To switch the output on again, the error must be acknowledged with OverloadClear for the respective channel. A lock time does not have to be observed.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	Overloadxx[1]	0	Not cut off
		1	Cutoff due to overload
...	...	...	
7	Overloadxx + 7	0	Not cut off
		1	Cutoff due to overload

[1] For xx, see the name of the register.

14.6.3 Acknowledgment of overload shutdown

Name:
OverloadClear17 to OverloadClear24
OverloadClear25 to OverloadClear32

The Overloadxx status bits of register Status of the digital outputs can be reset with these registers. The respective output is also re-enabled with the reset.

The reset always refers only to the currently applied overload. If the error situation persists, the error bit is therefore always set anew.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	OverloadClearxx[1]	0	Do not reset
		1	Reset Overloadxx status bit
...	...	...	...
7	OverloadClearxx + 7	0	Do not reset
		1	Reset Overloadxx status bit

[1] For xx, see the name of the register.

14.7 Pulse width modulation (PWM) and PVG

Digital inputs 17 to 24 can be configured as PWM outputs.

Digital inputs 25 to 28 can be configured as PVG outputs. PVG is a variant of pulse width modulation for special electrohydraulic valves.

2 data points are available per channel for controlling the PWM or PVG signal.

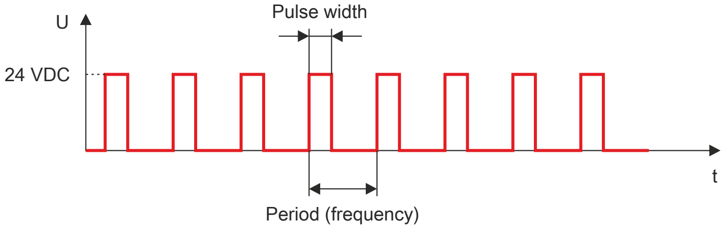


Figure 5: The PWM signal is controlled by setting the pulse width and period duration.

14.7.1 Dither frequency

Name:
CfgPinOptionB17 to CfgPinOptionB25

The dither frequency for the PWM outputs is configured in these registers.

These registers are not used by the PVG outputs.

Data type	Values	Information
USINT	0	Dither switched off
	1 to 255	Dither frequency in Hz

14.7.2 Dither amplitude

Name:
CfgPinOptionC17 to CfgPinOptionC24

The dither amplitude for the PWM outputs is configured in these registers.

These registers are not used by the PVG outputs.

Data type	Values	Information
USINT	0	Dither switched off
	1 to 25	Corresponds to 1 to 25% of the PWM/PVG period duration, dither amplitude of PWM/PVG pin "x"

14.7.3 Ramp function

Name:
CfgPinOptionF17 to CfgPinOptionF24
CfgPinOptionG17 to CfgPinOptionG24

The switch-on ramp or switch-off ramp of the PWM signal is set in these registers. This sets the time during which the module changes the existing period duration value to a new value. The set ramp function time refers to the maximum change from 0 to 100%, however, and must be converted to the actual change time required.

Example

Current set period duration: 60% of the switch-on time
New desired period duration: 20% of the switch-on time
Ramp time: 60 = 6 s

Calculation

$$\text{Time} = \frac{\text{Ramp time}}{100} \times \left(\text{Value}_{\text{New}} - \text{Value}_{\text{Old}}\right) = \frac{6\ s}{100} \times \left(20\% - 60\%\right) = -24\ s$$

The new period duration value is reached by means of an ascending ramp after 2.4 s. A positive result indicates an ascending ramp; a negative result indicates a descending ramp.

In the event of short circuit, the output is not switched off with the configured ramp, but immediately.

Data type	Values	Information
UINT	0	Ramp switched off
	1 to 65,535	Ramp time in 0.1 s

14.7.4 PI current controller

Name:
CfgPinOptionH17 to CfgPinOptionH24
CfgPinOptionI17 to CfgPinOptionI24

The values for PI current control of the PWM outputs can be configured in these registers. The following applies:

- CfgPinOptionH: Corresponds to the P value
- CfgPinOptionI: Corresponds to the I value

For a meaningful configuration, a multiple of 250 µs should be selected for the period duration of the output.

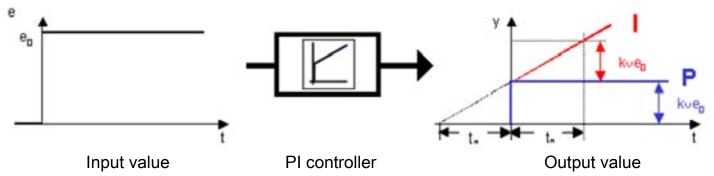
> ## Information:
>
> **The set period length is not permitted to be less than 4 ms (250 Hz).**

PI controller

With the PI controller, manipulated variable Y corresponds to an addition of the output variables of a P and an I controller. The manipulated variable is internally limited to values between 1 and 32767. "Anti-windup" is present.

The manipulated variable is initially changed in the same way as for P control. Subsequently, a further change of the manipulated variable takes place that corresponds to the time integral of the control difference as with the I controller.

The PI controller combines the advantages of both controllers: It is called in a 1-ms cycle and thus reacts quickly to control deviations (P component) and compensates these completely (I component).



Data type	Values	Information
UDINT	0 to 4,294,967,295	Unit 1/10000, e.g. value 2500 corresponds to 0.25

14.7.5 Period duration of the PWM/PVG outputs

Name:
PWMPeriodGroup01 to PWMPeriodGroup02
PVGPeriodGroup01 to PVGPeriodGroup01

These registers define the period duration, i.e. the time base for the respective PWM/PVG output. This time represents the 100% value, which can be incremented through the duty cycle.

Data type	Values	Information
UINT	1000 to 65535	Period duration in µs

14.7.6 Duty cycle of the PWM/PVG outputs

Name:
PWMCurrent17 to PWMCurrent24
PWMDuty17 to PWMDuty24
PVGDuty25 to PVGDuty28

The ratio of the duty cycle of the respective PWM or PVG output in relation to the period duration is set in these registers.

If the outputs are used as current controllers, the current setpoint is specified in this register.

Data type	Values	Information
UINT	0 to 32767	Duty cycle of the output in 0 to 100%
	0 to 32767	Current setpoint of the output from 0 to 5000 mA [1]

[1] Values > 29500 (approx. 4500 mA) should not be used since an overcurrent shutdown can occur at any time. See section "Output current monitoring" on page 18.

Example

Period duration T = 4000 [µs] with a duty cycle of 25% equals a switch-on time t_1 of 1000 [µs].

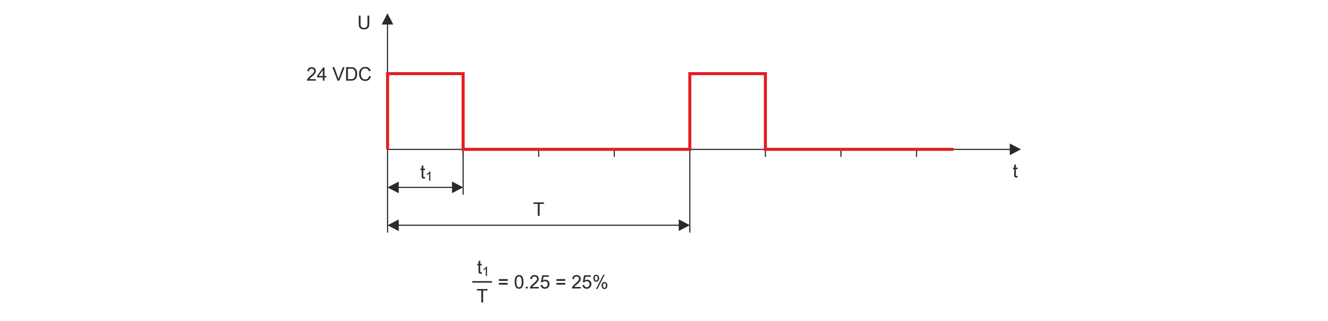


Figure 6: Switch-on time depending on the period duration and duty cycle

14.7.7 Enabling the PVM/PVG output

Name:
OutputEnable17 to OutputEnable24
OutputEnable25 to OutputEnable28

These registers can be used to switch the PWM/PVG outputs to the idle state (high-impedance state).

Data type	Values
UINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	OutputEnable17	0	PWM output high-impedance
		1	PWM output enabled
...	...		
7	OutputEnable24	0	PWM output high-impedance
		1	PWM output enabled

Bit	Description	Value	Information
0	OutputEnable25	0	PVG output high-impedance
		1	PVG output enabled
...	...		
3	OutputEnable28	0	PVG output high-impedance
		1	PVG output enabled
4 - 7	Reserved		

14.7.8 Dither

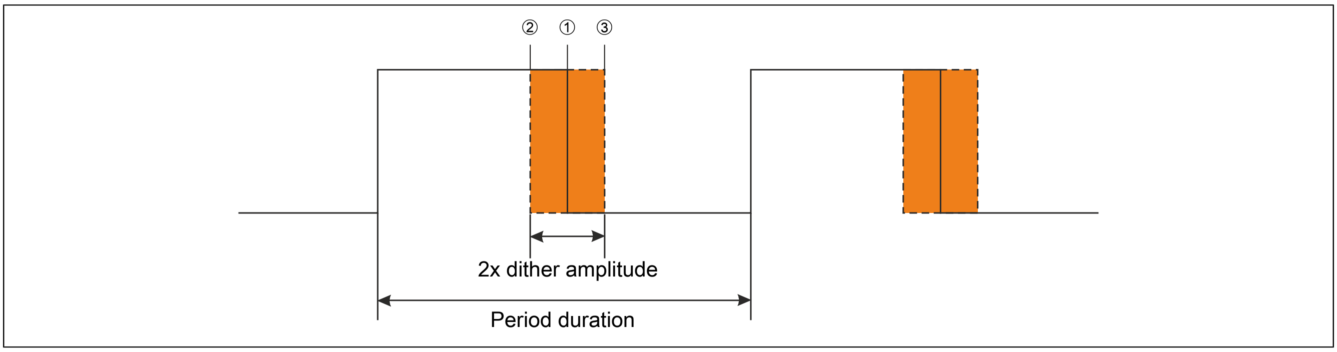
Using two configuration data points, a dither can be configured with an amplitude and an frequency. The amplitude is a relative value to the PWM period length. The dither is applied in the form of a ramp.

No dither is used for the PVG outputs.

Example

Period duration: 2 ms = 500 Hz
Duty cycle: 50% = 1 ms
Dither amplitude: 10% of period duration = ±0.2 ms
Dither frequency: 20 Hz = 50 ms

At the set period duration of 2 ms, the duty cycle changes every 50 ms, i.e. 20 times per second, by ±0.2 ms between 0.8 and 1.2 ms.



Legend
① Set duty cycle = 1 ms
② Shortest duty cycle = 0.8 ms
③ Longest duty cycle = 1.2 ms

14.8 Current measurement

The module has the ability to measure the currents supplied by the outputs in 3 different ways.

- Current value
- Root mean square (RMS)
- Arithmetic mean value

These currents can be measured when wired as a digital output as well as in configuration "PWM".

Current value

For each system tick, the current measured value for the current provided by the hardware is published as the input value.

If a level changeover has been performed in the current hardware conversion cycle, this measured current value is published as the input value.

Root mean square (RMS) / Arithmetic mean value

For current measurement, the measured value is stored every 50 μs after the rising edge of the control signal. A mean value is calculated from the last 5 measured values per multiplex cycle.

A mean value is calculated over 1 to 32 PWM periods in PWM mode using parameter CfgPinOptionS. When operated as a digital output, the time is specified in μs.

The total measurement duration is not permitted to exceed 63750 μs.

14.8.1 Current measurement configuration

Name:
CfgPinOptionA01 to CfgPinOptionA32 (measurement type)
CfgPinOptionE01 to CfgPinOptionE32 (measured period)

Configuration of the measurement type

Data type	Values	Information
USINT	0	Instantaneous value
	1	Root mean square (RMS)
	2	Arithmetic mean value
	3 to 7	Reserved
	8	Inductive load: Instantaneous value [1]
	9	Inductive load: Root mean square (RMS) [1]
	10	Inductive load: Arithmetic mean value [1]
	11 to 255	Reserved

[1] These modes are used for current measurement with inductive load. They are only useful in PWM mode, however, where the cyclic output of the signal results in the average current. For cyclic pulse operation as digital output, these modes are not applicable.

Configuration of the measured period

The configured value specifies how far back the last current value taken into account for the calculation goes. With microsecond measurement, the result is rounded to 250 µs units.

Data type	Values	Information
UINT	1000 to 63,000	Measurement range in microseconds
	0 to 31	Number of periods measured in PWM mode Number of periods = Values + 1, i.e. value 0 = 1 period

14.8.2 Measured current

Name:
Current29 to Current32

These registers contain the analog current measured value of the digital power outputs.

1 unit = 610 µA.

Data type	Values	Information
INT	0 to 32767	Measured current = Value * Unit

Name:
Current17 to Current24

These registers contain the analog current measured value of the PWM power outputs.

1 unit = 152 µA.

Data type	Values	Information
INT	0 to 32767	Measured current = Value * Unit

14.9 Operating management

14.9.1 Voltage selection of sensor power supply

Name:
CfgSysIo

The sensor power supply and I/O enabling are configured in this register.

Data type	Values
UINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0	Voltage selection of sensor power supply	0	5 V sensor power supply
		1	10 V sensor power supply
1 to 6	Reserved	-	
7	Enable I/O system after configuration	0	Do not enable I/O system
		1	Enable I/O system
8 to 15	Reserved	-	

14.9.2 Current average value of the I/O power supply

Name:
AiCurrentSum

This data point provides the sum of the positive currents of the digital and PWM outputs. The returned value is averaged over the last 500 ms.

Data type	Values	Measured current
INT	0 to 32767	0 to 25 A

14.9.3 Displaying the node number

Name:
AiNode

Contains the currently used node number

For possible numbers, see "Node number and transfer rate" on page 16.

Data type	Values	Information
INT	4 to 120	Current node number

14.9.4 Measuring operating temperature

Name:
AiTemp

This data point returns the measured operating temperature.

> **Notice!**
>
> **The module does not have automatic shutdown in the event of overtemperature. It is therefore important to ensure in the application that the maximum operating temperature of 105°C is not exceeded.**

Data type	Values	Information
INT	-800 to 2700	Temperature measurement, resolution 0.1°C

14.9.5 Measurement of the I/O power supply at the supply

Name:
AiVDriveIn

This data point returns the measured I/O supply voltage directly at the supply.

Data type	Values	Measured voltage
INT	0 to 32767	Corresponds to 0 to 40 V

14.9.6 Measurement of the I/O power supply at the circuit breakers

Name:
AiVDriveOut

This data point returns the internal supply voltage that is available for the PWM and DO_plus outputs.

Data type	Values	Measured voltage
INT	0 to 32767	Corresponds to 0 to 40 V

14.9.7 Measuring sensor voltage

Name:
AiVextSensor

This data point returns the measured sensor voltage. This value is used as a reference value for the internal measuring circuit.

Data type	Values	Measured voltage
INT	0 to 32767	Corresponds to 0 to 11 V

14.9.8 Measurement of the controller supply voltage

Name:
AiInt

This data point returns the measured supply voltage for the controller.

Data type	Values	Measured voltage
INT	0 to 32767	Corresponds to 0 to 40 V

14.9.9 System status

Name:
SysStat

These registers contain general system status information.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 1	Reserved		
2	Overcurrent	0	No error
		1	Overcurrent occurred
3	Sensor power supply	0	Error in sensor power supply
		1	Sensor power supply OK
4 - 5	Reserved		
6	Power supply	0	Error in power supply
		1	Voltage OK
7	Overtemperature	0	No error
		1	Overtemperature occurred

Overcurrent

The common summation current of the outputs is monitored and is not permitted to permanently exceed 25 A. A temporary overshoot of 2% is possible.

The power dissipation is proportional to the quadratic current. The mean value of the quadratic current should not significantly exceed 25 A. An integral is formed over the quadratic current for the last ~2 s, and as soon as (I_{Nom} x 1.02)² x t A²s is exceeded, the status bit is set and all active digital outputs are disabled.

I_{Nom} = 25 A

t = 2 s

ð (25 A * 1.02)² x 2 s = 1300.5 A²s

14.9.10 Resetting the overcurrent bit

Name:
CfgVDriveOut

The overcurrent bit of register "SysStat" on page 52 can be reset with these registers. All drivers active before the cutoff are re-enabled.

The reset always refers only to the currently applied overcurrent. If the error situation persists, the error bit is therefore always set anew.

Data type	Values
USINT	See the bit structure.

Bit structure:

Bit	Description	Value	Information
0 - 1	Reserved		
2	OverloadShutdownClear	0	Do not reset
		1	Reset overcurrent bit
3 - 7	Reserved		

15 X90 bus controller

15.1 Startup

Using an EDS/DCF file

Since the bus controller does not have an automatic configuration, PDO mapping is only possible after the EDS or DCF file has been transferred to the bus controller or a manual configuration has been performed. This is done by calling objects 0x380x and 0x390x. See "Manual configuration example" on page 73.

Several automatic restarts of the bus controller occur during a new configuration process or when transferring a significantly modified configuration. The error message about non-existent objects returned during this time is system-related and can be ignored.

Using a BIN file

When using a BIN file, the startup behavior depends on the PDO mapping range used.

- **Default I/O range 0x6000 to 0x6FFF (6000-6FFF, DS401-compliant)**
  The BIN file is only transferred once, and the bus controller is not additionally restarted.
- **I/O range 0x3200 to 0x37FF**
  The BIN file must be transferred twice, and the bus controller may be automatically restarted several times.

15.1.1 Reducing the startup time

When using a device description file, the startup time is extended for the X90 bus controller. Depending on the setting, this can take several seconds and is made up of the startup and configuration time of the bus controller.

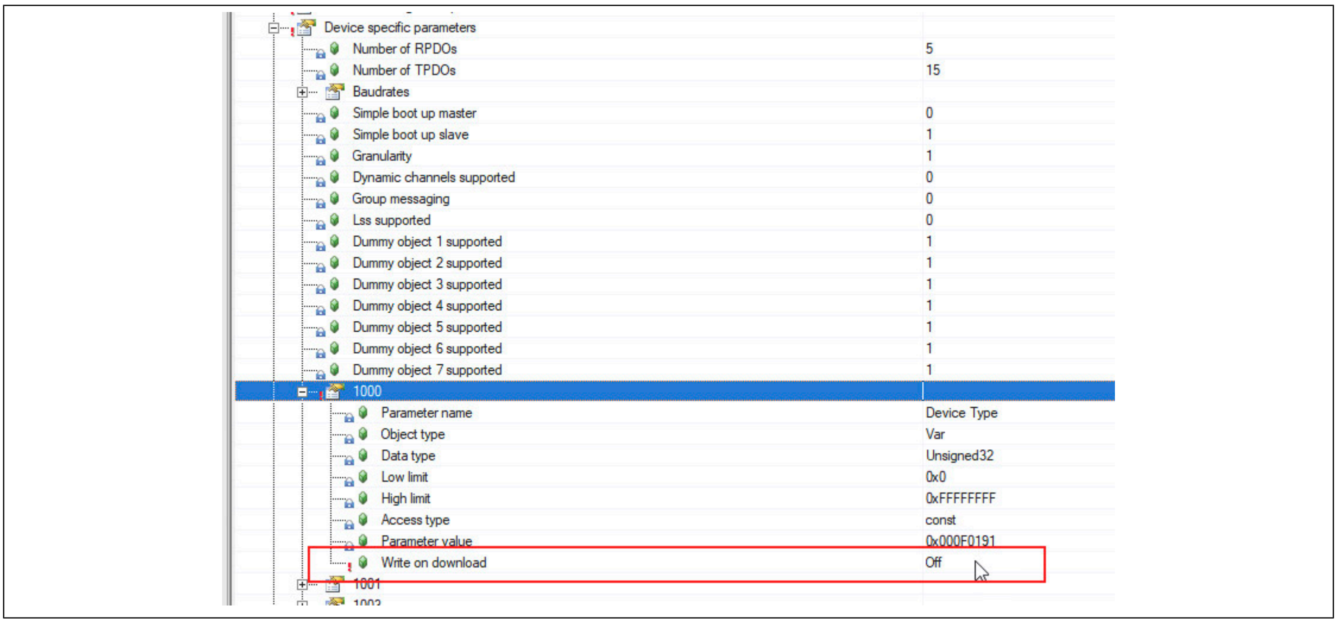
The bus controller configuration is transferred during each startup. The first time the configuration is transferred to the bus controller, an additional cycle is required and delays startup by several seconds, depending on the setting and configuration scope.

Several options are available to prevent the transfer of the configuration during each startup and thus reduce the startup time:

- Disabling objects for configuration
- Disabling the entire configuration
- Configuring the startup delay on the master
- Configuration at runtime
    - Manual configuration
    - Configuration using a BIN file

Disabling objects for configuration

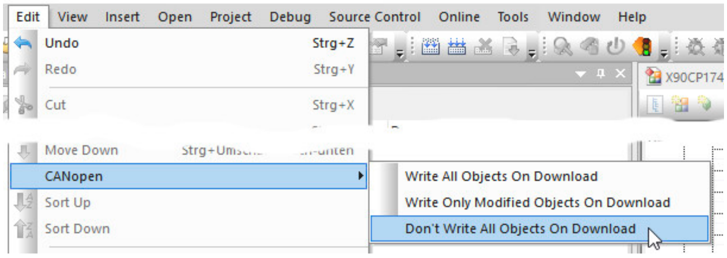
Individual objects can be disabled with "Write on download = Off" under "Device-specific parameters" in the configuration interface of the X90BC.



Device specific parameters	
Number of RPDOs	5
Number of TPDOs	15
Baudrates	
Simple boot up master	0
Simple boot up slave	1
Granularity	1
Dynamic channels supported	0
Group messaging	0
Lss supported	0
Dummy object 1 supported	1
Dummy object 2 supported	1
Dummy object 3 supported	1
Dummy object 4 supported	1
Dummy object 5 supported	1
Dummy object 6 supported	1
Dummy object 7 supported	1
1000	
Parameter name	Device Type
Object type	Var
Data type	Unsigned32
Low limit	0x0
High limit	0xFFFFFFFF
Access type	const
Parameter value	0x000F0191
Write on download	Off
1001	
1002	

Disabling the entire configuration

It is possible to disable the entire configuration by clicking in the configuration interface of the X90 bus controller. This displays the additional menu option "CANopen" under "Edit". Selecting "Don't write all objects on download" disables the configuration transfer.



Information:

It is important to ensure that a valid configuration already exists on the bus controller or that it is transferred in another way.

Configuring the startup delay on the master

Settings can also be made on the CANopen master to minimize the startup time.

This includes, for example, the baud rate or the "CAN driver cycle time" on a B&R CANopen master. The priority of the CAN driver can also be adjusted.

Information:

Selecting incorrect parameters can result in a greatly increased CAN bus load.

15.2 Firmware update

New functions and improved versions of the bus controller can be implemented by updating the firmware. Firmware files can be obtained from the B&R support team.

Procedure for a firmware update:

1) Manually set the bus controller to mode PREOPERATIONAL. This can be done, for example, using function block "CANopenNMT" of library AsCANopen. See Automation Help → Programming → Libraries → Communication → AsCANopen.
   When mode PREOPERATIONAL is called manually, all values in the bus controller are retained; in contrast, all values are set to 0 when called automatically. "Automatic" calls occur in the event of error or connection loss, for example.
2) Transfer the new firmware to object 0x1F50 Sub 1.
3) Restart the bus controller.

15.3 Restart

If the bus controller must be restarted, this can be done in the following ways:

- Restarting manually
- Using function block "CANopenNMT" of library AsCANopen. See Automation Help → Programming → Libraries → Communication → AsCANopen.
- Writing value 2 to object 0x1F51 Sub 1

15.4 Error corrections

Since the X90 CANopen bus controller does not have an LED status indicator, the current status of the bus controller cannot be seen from outside. If the bus controller can no longer be addressed, this may be caused by various error sources.

Possible error sources for an unresponsive bus controller:

Cause of error	Workaround
The bus controller is not supplied with power.	Check and restore the power supply.
The used node number or baud rate does not match.	Resetting the bus controller
An incorrect node number is used.	Check the node number contacts and wiring.
Other errors	Check for damaged hardware • Defective CAN cable? • Missing terminating resistor? • Etc.
The bus controller is defective.	Please contact the B&R service center.

Resetting the bus controller

Restoring the factory settings via the hardware is not possible due to missing node number switches; however, it can be done using service node number 124 and a fixed baud rate of 250 kbit/s.

- Set node number 124. The bus controller should now be accessible with a baud rate of 250 kbit/s.
- Delete the configuration by accessing object 0x1011, subIndex 0x1.
- Restart the bus controller.
- Set the desired node number. On the next startup, the baud rate is automatically detected and set again.

16 CANopen

The X90 CANopen bus controller can be operated on any CANopen master. Since the configuration of the bus controller is transferred to the bus controller via the master, there are various options available for transferring the configuration data to the master:

- Manual configuration in the application
- Transfer using a BIN file
- Using a device description file (EDS or DCF file)

Using a B&R master

When using a B&R CANopen master, the hardware upgrade file in Automation Studio cannot be used for configuration. Instead, a device description file must be transferred as with any other CANopen master. The hardware upgrade itself is only used to create the device description file. This is created in Automation Studio using a virtual CANopen CPU. For details, see "Creating device description files" on page 79.

16.1 I/O configuration

With the I/O configuration, B&R Automation Studio V4.7 or later provides selection menus and wizards for determining which function model will be used, which cyclic input and output data will be registered and which values for the module configuration will be written to the I/O module by the bus controller when the module is started.

Automation Studio generates a DCF or EDS file suitable for all of the hardware nodes (module IDs, configuration values, etc.). For CANopen environments that do not support DCF or EDS imports, an HTML file is also generated that contains the mapping, configuration values, etc. This enables the user to look up the necessary SDO instructions and to implement them in their own CANopen environment.

Changes can still be made to the module parameters during operation by using CANopen objects "I/O objects" on page 65.

Transferring the configuration

When transferring a new configuration, an existing configuration is overwritten but not deleted. If the new configuration is therefore shorter than the existing one, old entries will remain at the end.

> ## Information:
>
> **Transferring a new configuration does not delete an existing one. A manual deletion procedure should therefore always be carried out beforehand.**

A manual deletion procedure can be carried out by calling object "0x1011 - Restore default parameters" on page 58 and then restarting the bus controller.

16.2 CANopen communication

16.2.1 Device profiles

With CANopen, device properties are described in device profiles. Based on the device type, certain data and parameters (called "objects" in CANopen) are permanently defined. The CAN in Automation (CiA) organization describes the device profiles in various standards. "Draft standard 401", for example, deals with digital and analog I/O devices.

16.2.2 Object dictionary

The "object dictionary" contains an overview of all data and parameters (objects) for a CANopen device. The data reflects the process image, whereas the parameters can be used to influence the functionality of a CANopen device. The objects are indexed so that they can be clearly identified and addressed. This index can also be divided into several sub-indexes. The structure of the object dictionary, the assignment of index numbers and several mandatory entries are specified in the device profiles.

The object dictionary is saved for the user as an EDS file. The EDS file contains all objects and their properties (index, sub-index, name, data type, default value, access options, etc.). In this way, the entire functionality of the CANopen device is described in the EDS file.

16.2.3 Service and process data objects

Data in a CANopen network is exchanged in the form of telegrams with which the payload data is transferred. A distinction is made between service data objects (SDOs) and process data objects (PDOs). All entries made in the object dictionary can be accessed using the SDOs. They are usually only used for initialization during the boot procedure, however. PDOs bundle all objects (variables and parameters) from the object data dictionary. A PDO (max. 8 bytes each) can consist of various objects.

PDO (process data objects)	SDO (service data objects)
- Transfer real-time data	- Transfer system parameters
- No response to telegram (high-speed data transfer)	- Response to telegram (slow data transfer)
- High-priority identifiers	- Low-priority identifiers
- Max. 8 bytes/telegram	- Distribution of data across multiple telegrams
- Defined data format	- Index-addressable data

16.3 The object dictionary

This CANopen bus controller was designed in accordance with CiA standards DS-301 and DS-401 and adheres to the majority of the specifications they contain.

16.3.1 Supported objects from CiA standard DS-301

The CANopen bus controller communicates using the mechanisms specified in CiA standard DS-301. For information about data types, access types, default values, etc., see CiA standard DS-301.

The following objects from CiA standard DS-301 are supported:

Index	Description	Description
0x1000	Device type	Description of device type: For the CANopen bus controller this is always 0x000F0191. This indicates: **Device profile number** 0x0191 (=401 dec.) The bus controller supports CiA standard DS-401 **I/O functions** 0x000F Bits 16-19 are enabled. Support for digital inputs (bit 16), digital outputs (bit 17), analog inputs (bit 18), analog outputs (bit 19)
0x1001	Error register	Displays the general error state (part of every emergency telegram) Error register assignments: **Bit 0** General error bit **Bit 1** Not used **Bit 2** Voltage error: A module connected to the bus controller is registering a supply voltage error. **Bits 3-6** Not used **Bit 7** Vendor-specific error or data present (always set)
0x1003	Pre-defined error field	Error history of bus controller: The last 32 error messages are placed in this field. The subindex 0 contains the number of errors currently present. The most recent error is in subindex 1. Every new error is entered in subindex 1, and the previous entries are shifted back until they drop out of the history. Writing the value 0 to subindex 0 will delete the error history.
0x1005	COB ID SYNC	Sets the COB ID of the synchronization message
0x1008	Manufacturer device name	Product name as plain text (ASCII string, segmented SDO upload protocol)
0x1009	Manufacturer hardware version	Hardware revision of the bus controller as plain text in the format Vxxxx.xxxx (ASCII character string, segmented SDO upload protocol).
0x100A	Manufacturer software version	Software (firmware) version of the bus controller in Vxxxx.xxxx format (ASCII character string, segmented SDO upload protocol).
0x100C	Guard time	Sets the guard time (ms): This is used when the node guarding protocol is used for failure monitoring.
0x100D	Life time factor	The life time factor is a multiplier for the guard time.
0x1010	Store parameters	Stores the defined parameters in the bus controller's internal flash memory - First applied after restarting (power off/on or software reset - object 0x1F51 sub1) The bus controller parameters can be divided into three groups: **Communication parameters** Object index range 0x1000 - 0x1FFF **Manufacturer-specific parameters** Object index range 0x2000 - 0x5FFF **Application parameters** Object index range 0x6000 - 0x7FFF The bus controller supports subindexes 1 to 4 according to the predefined specification. To save the respective parameters, "save" or 0x65766173 (the hexadecimal value for the word "evas") must be written to the corresponding subindex. **Subindex 1** Saves all parameters **Subindex 2** Saves the communication parameters **Subindex 3** Saves the application parameters **Subindex 4** Saves the vendor-specific parameters The value read back from the individual subindexes is 1.
0x1011	Restore default parameters	Reset to factory setting: For a breakdown of subindexes, see register 0x1010 ("store parameters"). To restore the factory settings, write parameter "load" or 0x64616F6C (the hexadecimal value of the word "daol") to the corresponding subindex. **Subindex 1** Deletes all parameters (factory setting) **Subindex 2** Deletes the communication parameters **Subindex 3** Deletes the application parameters **Subindex 4** Deletes the vendor-specific parameters The value read back from the individual subindexes is 1.
0x1014	COB ID EMCY	Sets the COB ID of the emergency telegrams (default: 0x80 + node ID)
0x1015	Inhibit time EMCY	Specifies the minimum time between 2 error messages: This is useful if the master stores the emergency telegrams in a logbook (longer saving procedure), for example, and any errors registered in the meantime would otherwise be lost. The resolution of the inhibit time is 100 µs.
0x1016	Consumer heartbeat time	Sets the "consumer heartbeat time" and "consumer heartbeat COB ID": **Bits 0-15** "consumer heartbeat time" **Bits 16-23** "Node ID" The resolution of the time is 1 ms.
0x1017	Producer heartbeat time	Sets the "producer heartbeat time": The resolution of the time is 1 ms.

Index	Description	Description
0x1018	Identity object	Description of the bus controller in hexadecimal format **Subindex 1** **Vendor ID** CANopen vendor ID of the bus controller **Subindex 2** **Product code** Product code of the bus controller **Subindex 3** **Revision number** Revision number of the bus controller, equal to 0x100A ("manufacturer software version") **Subindex 4** **Serial number** Serial number of the bus controller
0x1020	Verify configuration	Comparison of stored configuration vs current configuration **Subindex 1** **Configuration date** Date configuration was created (specified in days since January 1, 1984) **Subindex 2** **Configuration time** Time configuration was created (specified in ms since midnight) See section "Application example - Verify configuration" on page 60.
0x1029	Error behavior	Bus controller behavior in the event of an error (communication error): Valid for node guarding errors, consumer heartbeat errors and internal CAN chip problems (subsequently: "Bus off"). **Subindex 1** **Communication error** 0 Change to PREOPERATIONAL **(default)** 1 No state change 2 Change to Stop
0x1200	1st SDO server parameter	Sets the COB-IDs for the first (default) SDO connection **Subindex 1** **COB ID client-to-server** 0x600 + Node ID (default) **Subindex 2** **COB ID server-to-client** 0x580 + Node ID (default) **IMPORTANT!** **During a segmented transfer, the object dictionary is locked andcommunication is not possible even via another SDO channel.**
0x1201	2nd SDO server parameter	Sets the COB IDs for the second SDO connection (optional) **Subindex 1** **COB ID client-to-server** 0x80000000 (disabled) **Subindex 2** **COB ID server-to-client** 0x80000000 (disabled) **Subindex 3** **Node ID of the SDO client** 0x00 (informative, no effect on application) **IMPORTANT!** **During a segmented transfer, the object dictionary is locked andcommunication is not possible even via another SDO channel.**
0x1400 - 0x141F	RPDO communication parameter	Set the properties of RPDOs **Subindex 1** **COB ID used by RPDO** COB ID **Subindex 2** **Transmission type** Transmission type for the RPDOs. 0x00 to 0xF0 and 0xFF are supported. 0x00, 0x01 Synchronous: The data (e.g. digital outputs) is refreshed after every SYNC telegram. 0x02 - 0xF0 Synchronous: The data is refreshed after every nth SYNC telegram. For example, value 8 → After the 8th SYNC telegram, the data is taken over by the RPDO and written to the outputs. 0xFF Event-based: The data from the received RPDO will be applied immediately **Subindex 3** **Inhibit time** Not used **Subindex 4** **Compatibility entry** Not used **Subindex 5** **Event timer** Not used Subindex 6 for the "SYNC start value" is not supported.
0x1600 - 0x161F	RPDO mapping parameter	Sets the RPDO mappings: 64 mapping entries are supported in order to allow bit mapping to its fullest extent.
0x1800 - 0x181F	TPDO communication parameter	Set the properties of TPDOs **Subindex 1** **COB ID used by TPDO** COB ID **Subindex 2** **Transmission type** Transfer method of the RPDO. 0x00 to 0xF0, 0xFC, 0xFD and 0xFF are supported. 0x00 Synchronous (acyclic) 0x01 Synchronous (cyclic with each SYNC telegram) 0x02 Synchronous (cyclic with each 2nd SYNC telegram) ... ... 0xF0 Synchronous (cyclic with each 240th SYNC telegram) ... ... 0xFC Only RTR (synchronous) 0xFD Only RTR (event-controlled) ... ... 0xFF Event-controlled **Subindex 3** **Inhibit time** Smallest interval between two TPDOs in 0.1 ms resolution **Subindex 4** **Compatibility entry** Not used **Subindex 5** **Event timer** Minimum transmission interval for this TPDO in ms. "Inhibit time" is higher priority.
0x1A00 - 0x1A1F	TPDO mapping parameter	Sets the TPDO mappings: 64 mapping entries are supported in order to allow bit mapping to its fullest extent.

16.3.1.1 Application example - Verify configuration

- The master has saved the configuration data for the individual nodes.
- At the beginning of the startup procedure, the bus controller configuration saved on the master is compared with the current configuration. Comparison of date saved.
- If configurations do not match:
  - The configuration saved on the master is transferred to the bus controller.
  - The corresponding date and time is set.
  - Data is saved to the bus controller's flash memory.
- Startup ended

The configuration data will only be transferred after a bus controller has been replaced or if the master is provided with a more recent configuration. This considerably shortens the startup phase.

The resolution of the subindex should conform to CiA standard DS-301.

Writing subindex 1 with a UNIX timestamp (seconds since January 1, 1970) or saving a CRC32 in subindex 2 is not prohibited by the bus controller.

16.3.1.2 Application example - Error behavior

Subindex 1

In the event the node necessary for transferring the consumer heartbeat (consumer heartbeat enabled) failed, it is possible to switch to state PREOPERATIONAL. This also involves performing the error response measures specified in CiA standard DS-401, such as resetting and setting outputs and writing a certain value to analog outputs.

16.3.2 Supported objects from CiA standard DS-302

For more detailed information about data types, access types, default values, etc., please refer to CiA standard DS-302.

The following objects from CiA standard DS-302 are supported:

Index	Description	Description
0x1F50	Program data	Replaces the bus controller's firmware or loads the object dictionary configuration in one block[1] **Subindex 1** **Firmware for BC** Bus controller firmware **Subindex 2** **Configuration for BC** Bus controller configuration This data is transferred via "segmented transfer" (CiA standard).
0x1F51	Program control	Bus controller reset triggered externally[1] **Subindex 1** **Firmware for BC** Writing value 2 triggers a restart of the bus controller and enables new firmware. **Subindex 2** **Configuration for BC** Writing value 2 triggers a restart of the bus controller. For the two subindexes, value 1 is returned when reading (program/configuration active). The reset is only permitted in operating state PREOPERATIONAL.
0x1F52	Verify application software	Identification of the firmware based on the timestamp: This object **CANNOT** be written to and is directly related to the firmware version (version assignment by the manufacturer). **Subindex 1** **Firmware (bus controller)** Date **Subindex 2** **Firmware (bus controller)** Time
0x1F56	Application software identification	Identification of the stored firmware **Subindex 1** **Firmware for BC** The current firmware version can be displayed by reading this entry. Same as object 0x1018 ("identify object"), subindex 3 ("revision number").
0x1F57	Flash status indication	Displays the flash status - Subindex 1 (firmware download): **Error code 0** Download successfully completed **Error code 1** Unable to boot firmware

[1] See "Firmware update" on page 54.

16.3.3 Supported objects from CiA standard DS-401

For more detailed information about data types, access types, default values, etc., please refer to CiA standard DS-401.

The following objects from CiA standard DS-401 are supported:

Index	Description	Description
0x6000	Read input 8-bit	Reads the digital inputs as bytes **Subindex 0** **Number of inputs 8-bit** Number of digital input bytes **Subindex 1 - 254** **Read input n - (n + 7)** Value of the digital input byte
0x6005	Global interrupt enable digital 8-bit	Turning the global digital IRQs on/off: Assuming that the PDOs are transmitted according to "Transmission type" when a value changes, the TPDOs will not be transferred when a value changes on a digital input (global digital IRQ disabled). **Value 0** Disabled **Value 1** Enabled
0x6006	Interrupt mask any change 8-bit	Defines whether an IRQ should be generated for the respective TPDO when there is a value change - Sets a bit for each digital input **Subindex 1** Digital inputs 1 to 8 **Subindex 2** Digital inputs 9 to 16 **Subindex ...** ... **Subindex 254** Digital inputs 2025 to 2032
0x6007	Interrupt mask low-to-high 8-bit	Defines whether a digital input should generate an IRQ for the respective TPDO on a positive edge. **Subindex 1** Digital inputs 1 to 8 **Subindex 2** Digital inputs 9 to 16 **Subindex ...** ... **Subindex 254** Digital inputs 2025 to 2032
0x6008	Interrupt mask high-to-low 8-bit	Defines whether a digital input should generate an IRQ for the respective TPDO on a negative edge. **Subindex 1** Digital inputs 1 to 8 **Subindex 2** Digital inputs 9 to 16 **Subindex ...** ... **Subindex 254** Digital inputs 2025 to 2032
0x6020 - 0x6027	Read input bit 1 to 1024	Digital inputs: The first 1024 digital inputs are set up as individual bits. **Subindex 0** **Number of inputs 1-bit** Number of digital inputs in this object (up to 0x80) **Subindex 1 - 254** **Read single input n** Value of the digital input (0 or 1).
0x6200	Write output 8-bit	Writes the digital outputs as bytes **Subindex 0** **Number of outputs 8-bit** Number of digital output bytes **Subindex 1 - 254** **Write output n - (n + 7)** Value of the digital output byte The value of the outputs can be read:
0x6206	Error mode output 8-bit	Definition specifying whether an error value is provided for a digital output: This value is used whenever an error occurs. **Subindex 0** **Number of outputs 8-bit** Number of digital output bytes **Subindex 1 - 254** **Error mode output n - (n + 7)** Error mode for digital outputs. One bit is available for each output. Default: 0xFF, value 0 → Disabled, value 1 → Enabled
0x6207	Error value output 8-bit	Determines the output value in case of error. **Subindex 0** **Number of outputs 8-bit** Number of digital output bytes **Subindex 1 - 254** **Error value output n - (n + 7)** Error value for digital outputs. One bit is available for each output.
0x6220 - 0x6227	Write output bit 1 to 1024	Digital outputs: The outputs 1 to 1024 are set up as individual bits. **Subindex 0** **Number of outputs 1-bit** Number of outputs in this object (up to 0x80) **Subindex 1 - 254** **Write output n** Value of the digital output (0 or 1).
0x6400	Read analog input 8-bit	Reads the analog inputs - scaled to 8-bit **Subindex 0** **Number of analog inputs 8-bit** Number of analog inputs **Subindex 1 - 254** **Analog input n** Value of the analog input scaled to 8 bits
0x6401	Read analog input 16-bit	Reads the analog inputs - scaled to 16-bit **Subindex 0** **Number of analog inputs 16-bit** Number of analog inputs **Subindex 1 - 254** **Analog input n** Value of the analog input scaled to 16 bits
0x6402	Read analog input 32-bit	Reads the analog inputs - scaled to 32-bit **Subindex 0** **Number of analog inputs 32-bit** Number of analog inputs **Subindex 1 - 254** **Analog input n** Value of the analog input scaled to 32 bits
0x6410	Write analog output 8-bit	Sets the analog outputs - scaled to 8-bit **Subindex 0** **Number of analog outputs 8-bit** Number of analog outputs **Subindex 1 - 254** **Analog output n** Value of the analog output scaled to 8 bits
0x6411	Write analog output 16-bit	Sets the analog outputs - scaled to 16-bit **Subindex 0** **Number of analog outputs 16-bit** Number of analog outputs **Subindex 1 - 254** **Analog output n** Value of the analog output scaled to 16 bits
0x6412	Write analog output 32-bit	Sets the analog outputs - scaled to 32-bit **Subindex 0** **Number of analog outputs 32-bit** Number of analog outputs **Subindex 1 - 254** **Analog output n** Value of the analog output scaled to 32 bits

Index	Description	Description
0x6421	Analog input trigger selection	Trigger conditions for the analog inputs: The trigger conditions are represented by objects 0x6424, 0x6425 and 0x6426. The trigger conditions 0x6427 (positive delta) and 0x6428 (negative delta) are not supported. Default value: 0x07 **Bit 0** **Upper limit value** **Bit 1** **Lower limit value** **Bit 2** **Analog input change** Analog input changes more than the delta value Default value (subindex 1 to 254 - analog input 1 to 254): 0x07 → All analog inputs react to the upper/lower limit value as well as overshoot of the delta value
0x6423	Analog input global interrupt enable	Global enable/disable of analog IRQs **Value 0** **Global interrupt** Disabled (default) **Value 1** **Global interrupt** Enabled
0x6424	Analog input interrupt upper limit integer	Determining the upper limit for an analog IRQ: An analog IRQ is triggered when the analog value exceeds the limit (≥), or any time the value changes above the limit, unless prevented by other conditions. **Subindex 0** **Number of analog inputs** Number of analog inputs **Subindex 1 - 254** **Analog input n** Threshold value for the respective analog input
0x6425	Analog input interrupt lower limit integer	Determining the lower limit for an analog IRQ: An analog IRQ is triggered when the analog value goes below the limit (<), or any time the value changes below the limit, unless prevented by other conditions. **Subindex 0** **Number of analog inputs** Number of analog inputs **Subindex 1 - 254** **Analog input n** Threshold value for the respective analog input
0x6426	Analog input interrupt delta unsigned	Definition of the minimum absolute value change (abs(new value - old value) > Δ): Prerequisite for triggering a new analog IRQ. The value change always refers to the last value sent. **Subindex 0** **Number of analog inputs** Number of analog inputs **Subindex 1 - 254** **Analog input n** Minimum delta
0x6443	Analog output error mode	Definition specifying whether an error value is provided for an analog output: This value is used whenever an error occurs. **Subindex 0** **Number of analog outputs** Number of analog outputs **Subindex 1 - 254** **Error mode analog output n** Error mode of the analog output
0x6444	Analog output error value integer	Definition of the analog output in case of error **Subindex 0** **Number of analog outputs** Number of analog outputs **Subindex 1 - 254** **Analog output n** Value of the analog output in case of error

Since the maximum data width of analog values is 32 bits, currently only the 32-bit values for the configurations 0x6424, 0x6425, 0x6426 and 0x6444 are supported. At lower analog values the low-order bytes lose their meaning. It therefore does not make sense to assign a 16-bit analog value an error value of 0x0000DC67.

16.3.4 Vendor-specific area

16.3.4.1 Bus controller objects

The following objects are available:

Index	Description
0x2041	Bus controller settings
0x3000	Enable hidden objects
0x3001	Configuration of output behavior
0x3011	Statistical error values for diagnostic purposes
0x3FFD	Reboot to factory settings
0x3FFE	Reboot to factory settings with communication parameters
0x3FFF	Reboot with all settings
0x9FFF	Reboot with all settings (not readable)

16.3.4.1.1 Bus controller settings

Object 0x2041 - "Bus controller settings"

This object is used to make settings on the bus controller.

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Ro	No	UNSIGNED8	4	-
0x1 - 0x3	-	-	-	-	Reserved
0x4	Rw	No	BOOL	1	Bus error handling **0** Bus controller does not automatically reset the CAN controller **1** Bus controller automatically resets the CAN controller If the bus controller automatically resets the CAN controller, then it is able to recover from a Bus-Off error without having to restart.
0x05	Rw	No	BOOL	0	Handling emergency telegrams in mode STOP **0** Emergency telegrams are being transmitted **1** Emergency telegrams are stored on the bus controller[1] and transmitted after exiting mode STOP. 1) Max. 63 telegrams

16.3.4.1.2 Enable hidden objects

Object 0x3000 configuration

Hidden objects can be enabled with this object.

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Ro	No	UNSIGNED8	15	-
0x10	Rw	No	BOOL	FALSE	Specifies whether hidden object entries from the EPLV2 profile should be enabled. (See objects with an "h" in the access type, e.g. object 0x3100 sub0x64.)

16.3.4.1.3 Configuration of output behavior

Object 0x3001 - Configuration of output behavior

This object can be used to set the output behavior of the bus controller.

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x3	Rw	No	UNSIGNED32	0	Output diagnostic time: This time counts down until it reaches 0. Incoming PDOs are not received while the time is running.

16.3.4.1.4 Statistical error values for diagnostic purposes

Object 0x3011 - Statistic values representing actual errors for diagnostic purposes

This object contains statistics counters for diagnostic purposes.

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Ro	No	UNSIGNED8	32	-
0x1	Rw	No	UNSIGNED32	-	Totalizer: When reading, the counter returns the sum of all following statistics counters.
0x2	Ro	No	UNSIGNED8	-	DIA_CanArbitrationLost_U8: Counter for shifted transmissions on the bus due to higher-priority objects
0x3	Ro	No	UNSIGNED8	-	DIA_CanErrorCode_U8: Error code of the last error that occurred on the CAN bus. Possible values: 0 No error 1 Stuff error 2 Form error 3 Confirmation error 4 Bit recessive error 5 Bit dominant error 6 CRC error 7 Set by software
0x4	Ro	No	UNSIGNED8	96	DIA_CanErrorWarningLimit_U8 If the error counter exceeds the set value, the CAN controller changes to error state "Error active".
0x5	Ro	No	UNSIGNED8	-	CAN Rx error: Register of the CAN connection[1]
0x6	Ro	No	UNSIGNED8	-	CAN Tx error: Register of the CAN connection[1]
0x7 - 0x2F	-	-	-	-	Reserved
Parameter entries					
0x30	Ro	No	UNSIGNED16	0	Index of the first faulty entry
0x31	Ro	No	UNSIGNED8	0	Subindex of the first faulty entry
0x32	Ro	No	UNSIGNED32	0	Error code of the first faulty entry
0x33	Ro	No	UNSIGNED32	0	Number of remaining entries

1) If object 0x2041/0x04 is set to 1 (bus automatically resets the CAN controller), this counter is reset to 0 in the case of a "Bus off" error.

Parameter entries

Statistics counters 0x30 to 0x33 refer to the configuration file transferred to the bus controller using 0x1F50, subindex 2. If faulty entries are present, the error information is output via counters 0x30 to 0x33.
The configuration file can only be checked for errors after the complete transfer.

16.3.4.1.5 Reboot to factory settings

Object 0x3FFD - Save_and_Reboot_Manufacturer

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Rw	No	UNSIGNED32	-	If the signature "save" or 0x65766173 (the hexadecimal value of the word "evas") is written to the object, the "Manufacturer" section of the object dictionary will be checked. **All parameters that were not written since the last reboot are set to their factory defaults!** A comparison is then made with the flash memory to determine if the parameters must be saved. If so, the parameters are saved to flash memory and a reboot is triggered. When read, the object returns 1 (same meaning as for object 0x1010).

16.3.4.1.6 Reboot to factory settings with communication parameters

Object 0x3FFE Save_and_Reboot_Manufacturer_Communication

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Rw	No	UNSIGNED32	-	Same function as 0x3FFD except that communication parameters will also be saved (the current communication parameter values are stored; there is no check for changes since the restart).

16.3.4.1.7 Reboot with all settings

Object 0x3FFF - Save_and_Reboot_All

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Rw	No	UNSIGNED32	-	Same function as 0x3FFE except that application parameters will also be saved (the current application parameter values are stored; there is no check for changes since the last restart).

16.3.4.1.8 Reboot with all settings (not readable)

Object 0x9FFF - Save_and_Reboot_All

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Wo	No	UNSIGNED32	-	Same function as 0x3FFF. The object is **NOT** readable, however.

16.3.4.2 I/O objects

Most of the objects (especially those used for configuration) are not accessible by default and must first be enabled by setting object 0x3000 / 0x10. The enabling itself is not stored and may have to be set after each startup.
In this document, objects that are hidden by default additionally have the identifier "h" for "hidden" in column "Access" (e.g. Roh, Rwh, Woh, etc.).

Below is a list of all the objects available in the bus controller:

Index	Description
0x310x	Configuring multifunction I/Os
0x320x	Byte access to input registers
0x330x	Word access to input registers
0x340x	Double word access to input registers
0x350x	Byte access to output registers
0x360x	Word access to output registers
0x380x	Module configuration registers
0x390x	Module configuration value

16.3.4.2.1 Configuring multifunction I/Os

Objects 0x3100 to 0x3102: Module configuration

This object is used to configure the multifunction I/Os. Due to the large amount of data, these are grouped into several objects, each of which is addressed by its own index.

<table>
<tr><th>Subindex</th><th>Access</th><th>PDO mapping</th><th>Range of values</th><th>Default value</th><th>Description</th></tr>
<tr><td>0x0</td><td>Ro</td><td>No</td><td>UNSIGNED8</td><td>254</td><td>-</td></tr>
<tr><td>0x1 - 0x10</td><td>-</td><td>-</td><td>-</td><td>-</td><td>Reserved</td></tr>
<tr><td>0x11</td><td>Ro</td><td>No</td><td>UNSIGNED16</td><td>-</td><td>Firmware version</td></tr>
<tr><td>0x12</td><td>Ro</td><td>No</td><td>UNSIGNED16</td><td>-</td><td>Hardware version</td></tr>
<tr><td>0x13</td><td>Ro</td><td>No</td><td>UNSIGNED32</td><td>-</td><td>Serial number</td></tr>
<tr><td>0x18 - 0x20</td><td>-</td><td>-</td><td>-</td><td>-</td><td>Reserved</td></tr>
<tr><td>0x21</td><td>Rw</td><td>No</td><td>UNSIGNED64</td><td>0</td><td>Acyclic read access for register:
Writing to this object triggers a read access. During the next read access, the value is returned. Format - see subindex 0x64.</td></tr>
<tr><td>0x22</td><td>Wo</td><td>No</td><td>UNSIGNED64</td><td>0</td><td>Acyclic write access for register:
Format - see subindex 0x64.</td></tr>
<tr><td>0x24</td><td>Rw</td><td>No</td><td>UNSIGNED32</td><td>-</td><td>Acyclic read access to a register:
Writing to this object causes the register to be read. The read value is stored in subindex 0x25.
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0-15</td><td>Register number. The higher-value byte contains the register bank.</td></tr>
<tr><td>16-31</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>0x25</td><td>Ro</td><td>No</td><td>UNSIGNED32</td><td>-</td><td>Acyclic read access to a register:
Value of the last register read acyclically.</td></tr>
<tr><td>0x26</td><td>Rw</td><td>No</td><td>UNSIGNED32</td><td>-</td><td>Acyclic write access to a register
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0-15</td><td>Register number. The higher-value byte contains the register bank.</td></tr>
<tr><td>16-31</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>0x27</td><td>Wo</td><td>No</td><td>UNSIGNED32</td><td>-</td><td>Acyclic write access to a register:
Writing to this object triggers write access to the register specified at subindex 0x26.</td></tr>
<tr><td>0x5B</td><td>Rw</td><td>No</td><td>UNSIGNED8</td><td>0</td><td>Number of valid configuration entries (0x64 - 0xFE or objects 0x38xx and 0x39xx)</td></tr>
<tr><td>0x64 - 0xFE</td><td>Rwh</td><td>No</td><td>UNSIGNED64</td><td>0</td><td>Configuration entry (0xDDDDDDDDTTSSNNNN), see table below</td></tr>
</table>

Configuration entry in subindex 0x64 - 0xFE

<table>
<tr><th>Bit</th><th>Configuration range</th><th>Explanation</th><th>Description</th></tr>
<tr><td>0 - 15</td><td>0xNNNN</td><td>Number</td><td>Register number. The higher-value byte contains the register bank.</td></tr>
<tr><td>16 - 23</td><td>0xSS</td><td>Size</td><td>Size in bytes.
<b>0</b> Entry not used</td></tr>
<tr><td>24 - 31</td><td>0xTT</td><td>Type</td><td><table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0-3</td><td>0000 - <b>0</b>: Dynamic cyclic input register
0001 - <b>1</b>: Dynamic cyclic output register
0010 - <b>2</b>: Fixed cyclic input register
0011 - <b>3</b>: Fixed cyclic output register
0100 - <b>4</b>: Acyclic input register
0101 - <b>5</b>: Acyclic output register
0110 - <b>6</b>: Reserved
0111 - <b>7</b>: Set parameters</td></tr>
<tr><td>4</td><td>Reserved</td></tr>
<tr><td>5</td><td>Hide register</td></tr>
<tr><td>6</td><td>Analog register</td></tr>
<tr><td>7</td><td>Mask register</td></tr>
</table></td></tr>
<tr><td>32 - 63</td><td>0xDDDDDDDD</td><td>Value</td><td><b>Type = 0</b> Default value for input data until the module is enabled for the first time
<b>Type = 1</b> Default value for output data if no other data is available
<b>Type = 2</b> Default value for input data until the module is enabled for the first time
<b>Type = 3</b> Default value for output data if no other data is available
<b>Type = 5</b> Initial value written before the module is enabled
<b>Type = 7</b> Parameter set before the module is enabled</td></tr>
<tr><td colspan="4">A bus controller can store up to 2024 configuration entries of this type for all modules.</td></tr>
</table>

Acyclic reading and writing

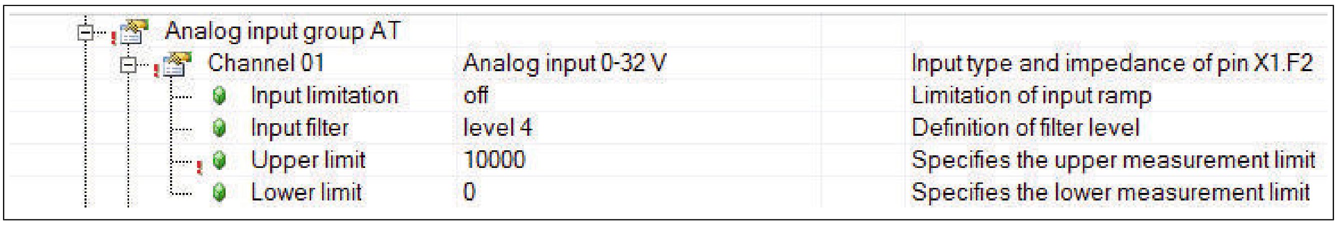
Registers can be read or written acyclically via objects 0x31xx - sub 0x21 to 0x27. The number of object calls depends on the master system used:

- 32-bit system: 2 object calls 0x31xx - sub 0x24 to 0x27
- 64-bit system: 1 object call 0x31xx - sub 0x21 and 0x22

Information:

Register values changed in this way are only retained until the bus controller is restarted. For permanent changes, the corresponding configuration entries must be adjusted.

Example



Analog input group AT		
Channel 01	Analog input 0-32 V	Input type and impedance of pin X1.F2
Input limitation	off	Limitation of input ramp
Input filter	level 4	Definition of filter level
Upper limit	10000	Specifies the upper measurement limit
Lower limit	0	Specifies the lower measurement limit

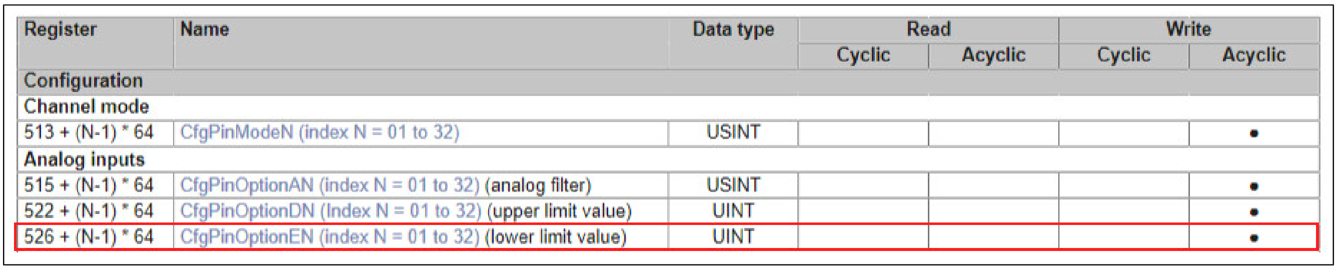
For this example, the first channel was set as an analog input with 0 to 32 V and a lower limit value of 0. For application-related reasons, this limit value should be temporarily raised to 100.

Procedure

1. Determine register address

First, the address for the desired register (lower limit value for channel 01) is selected from the register description.

For this example the register address is 526 → Hex: 0x020E



Register	Name	Data type	Read		Write	
			Cyclic	Acyclic	Cyclic	Acyclic
Configuration						
Channel mode						
513 + (N-1) * 64	CfgPinModeN (index N = 01 to 32)	USINT				•
Analog inputs						
515 + (N-1) * 64	CfgPinOptionAN (index N = 01 to 32) (analog filter)	USINT				•
522 + (N-1) * 64	CfgPinOptionDN (Index N = 01 to 32) (upper limit value)	UINT				•
526 + (N-1) * 64	CfgPinOptionEN (index N = 01 to 32) (lower limit value)	UINT				•

2. Determine data type size

The data type is UINT according to the register description. The size is specified in bytes.

Size = 2

3. Determine register type

According to the register description, the register type is acyclic write. The corresponding value can be read from section Configuration entry in subindex 0x64 to 0xFE.

Type = 0x05

Bit	Configuration range	Explanation	Description	
			Bit	**Description**
24 - 31	0xTT	Type	0-3	0000 - **0**: Dynamic cyclic input register 0001 - **1**: Dynamic cyclic output register 0010 - **2**: Fixed cyclic input register 0011 - **3**: Fixed cyclic output register 0100 - **4**: Acyclic input register 0101 - **5**: Acyclic output register 0110 - **6**: Reserved 0111 - **7**: Set parameters
			4	Reserved
			5	Hide register
			6	Analog register
			7	Mask register

4. Read register value for checking

For this example, the change should only be made if the lower limit value is still 0. The value of the register is therefore read for checking purposes. The desired register address is first determined with write access; therefore, 2 object accesses are always necessary for a read operation:

	32-bit			64-bit		
	Object	Subindex	Value	Object	Subindex	Value
Select register address (Write access)	0x3101	0x24	0x0502020E →Register: 0x020E →Size: 02 (bytes) →Type: 05	0x3101	0x21	0x000000000502020E
Read value (Read access)	0x3101	0x25	Value of register 0x020E	0x3101	0x21	Value of register 0x020E

5. Write register value

New desired value is 100 → Hex: 0x0064

Depending on the bit size, the register value is written with either 1 or 2 object calls:

	32-bit			64-bit		
	Object	Subindex	Value	Object	Subindex	Value
Select register address (Write access)	0x3101	0x26	0x0502020E →Register: 0x020E →Size: 02 (bytes) →Type: 05	0x3101	0x22	0x000000640502020E
Write value (Write access)	0x3101	0x27	0x00000064			

The lower limit value has now been changed to 100.

16.3.4.2.2 Byte access to input registers

Objects 0x3200 to 0x3202: Byte access to all input registers

These objects can be used to perform byte access to the bus controller registers.

Use of object numbers 00 to 02:

- 00: Object accesses to registers in the table CANopen object index xx00
- 01: Object accesses to registers in the table CANopen object index xx01
- 02: Object accesses to registers in the table CANopen object index xx02

The subindex is not the register number, but rather the n-th register in the register list.
The cyclic data points are lined up depending on the module configuration entry (registration of cyclic registers / data points - object 0x31xx or 0x38xx and 0x39xx).

> **Information:**
>
> **Only cyclic registers can be mapped in a PDO.**

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Roh	No	UNSIGNED8	254	-
0x1 - 0xFE	Rwrh	Yes	UNSIGNED8	-	Register

16.3.4.2.3 Word access to input registers

Objects 0x3300 to 0x3302: Word access to all input registers

These objects can be used to perform word access to the bus controller registers.

Use of object numbers 00 to 02:

- 00: Object accesses to registers in the table CANopen object index xx00
- 01: Object accesses to registers in the table CANopen object index xx01
- 02: Object accesses to registers in the table CANopen object index xx02

The subindex is not the register number, but rather the n-th register in the register list.
The cyclic data points are lined up depending on the module configuration entry (registration of cyclic registers / data points - object 0x31xx or 0x38xx and 0x39xx).

> **Information:**
>
> **Only cyclic registers can be mapped in a PDO.**

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Roh	No	UNSIGNED8	254	-
0x1 - 0xFE	Rwrh	Yes	UNSIGNED16	-	Register

16.3.4.2.4 Double word access to input registers

Objects 0x3400 to 0x3402: Long access to all input registers

These objects can be used to perform double word access to the bus controller registers.

Use of object numbers 00 to 02:

- 00: Object accesses to registers in the table CANopen object index xx00
- 01: Object accesses to registers in the table CANopen object index xx01
- 02: Object accesses to registers in the table CANopen object index xx02

The subindex is not the register number, but rather the n-th register in the register list.
The cyclic data points are lined up depending on the module configuration entry (registration of cyclic registers / data points - object 0x31xx or 0x38xx and 0x39xx).

> **Information:**
>
> **Only cyclic registers can be mapped in a PDO.**

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Roh	No	UNSIGNED8	254	-
0x1 - 0xFE	Rwrh	Yes	UNSIGNED32	-	Register

16.3.4.2.5 Byte access to output registers

Objects 0x3500 to 0x3502: Byte access to all output registers

These objects can be used to perform byte access to the bus controller registers.

Use of object numbers 00 to 02:

- 00: Object accesses to registers in the table CANopen object index xx00
- 01: Object accesses to registers in the table CANopen object index xx01
- 02: Object accesses to registers in the table CANopen object index xx02

The subindex is not the register number, but rather the n-th register in the register list.
The cyclic data points are lined up depending on the module configuration entry (registration of cyclic registers / data points - object 0x31xx or 0x38xx and 0x39xx).

> ## Information:
>
> **Only cyclic registers can be mapped in a PDO.**

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Roh	No	UNSIGNED8	254	-
0x1 - 0xFE	Rwwh	Yes	UNSIGNED8	-	Register

16.3.4.2.6 Word access to output registers

Objects 0x3600 to 0x3602: Word access to all output registers

These objects can be used to perform word access to the bus controller registers.

Use of object numbers 00 to 02:

- 00: Object accesses to registers in the table CANopen object index xx00
- 01: Object accesses to registers in the table CANopen object index xx01
- 02: Object accesses to registers in the table CANopen object index xx02

The subindex is not the register number, but rather the n-th register in the register list.
The cyclic data points are lined up depending on the module configuration entry (registration of cyclic registers / data points - object 0x31xx or 0x38xx and 0x39xx).

> ## Information:
>
> **Only cyclic registers can be mapped in a PDO.**

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Roh	No	UNSIGNED8	254	-
0x1 - 0xFE	Rwwh	Yes	UNSIGNED16	-	Register

16.3.4.2.7 Module configuration registers

Objects 0x3800 to 0x3802: Module configuration register

This object is used to configure the integrated modules (maximum 3), each of which is accessed using its own index. The lower value byte of the index indicates the position of the module described.

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Roh	No	UNSIGNED8	Number of configuration registers for this module (31XX / 5B)	-
0x1 - 0x9B	Rwh	No	UNSIGNED32	-	Configuration entry for register/type, see table below

Configuration entry in subindex 0x10 - 0x9B

<table>
<tr><th>Bit</th><th>Configuration range</th><th>Explanation</th><th>Description</th></tr>
<tr><td>0-15</td><td>0xNNNN</td><td>Number</td><td>Register number. The higher-value byte contains the register bank.</td></tr>
<tr><td>16-23</td><td>0xSS</td><td>Size</td><td>Size in bytes.
0 Entry not used</td></tr>
<tr><td>24-31</td><td>0xTT</td><td>Type</td><td>
<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0-3</td><td>0000 - **0**: Dynamic cyclic input register
0001 - **1**: Dynamic cyclic output register
0010 - **2**: Fixed cyclic input register
0011 - **3**: Fixed cyclic output register
0100 - **4**: Acyclic input register
0101 - **5**: Acyclic output register
0110 - **6**: Reserved
0111 - **7**: Set parameters</td></tr>
<tr><td>4</td><td>Reserved</td></tr>
<tr><td>5</td><td>Hide register</td></tr>
<tr><td>6</td><td>Analog register</td></tr>
<tr><td>7</td><td>Mask register</td></tr>
</table>
</td></tr>
</table>

16.3.4.2.8 Module configuration value

Objects 0x3900 to 0x3902: Module configuration value

This object is used to configure the integrated modules (maximum 3), each of which is accessed using its own index. The lower value byte of the index indicates the position of the module described (starts at position 0).

Subindex	Access	PDO mapping	Range of values	Default value	Description
0x0	Roh	No	UNSIGNED8	Number of configuration registers for this module (31XX / 5B)	-
0x1 - 0x9B	Rwh	No	UNSIGNED32	-	Configuration entry - Register value: This object can be used to include initial values for the registers configured in 38XX. This makes it possible, for example, to set sensor types for temperature modules or filter registers for input modules.

16.4 Emergency objects - Error messages

The bus controller supports the emergency protocol and an error history of 32 entries (object 0x1003). Sent error messages are automatically added to the error history. The error register (object 0x1001) is also set automatically. Bits 0, 2 and 7 of the error register are supported.

Bit	Description
0	General error
1	Current error
2	Voltage error
3	Temperature error
4	Communication error
5	Specific to device profile
6	Reserved (0)
7	Manufacturer-specific

The bus controller sends the following error messages:

Error	Byte				
	0 - 1	**2**	**3 - 4**	**5 - 6**	**7**
	Error code	**ErrReg[1]**	**Manufacturer-specific**		
Error free	0x0000	0x00	0x0000	0	0
Analog IRQ not active	0x0080	0x81	0		
Faulty power supply	0x3010	0x84	0x40	ChNo[2]	1
Overtemperature	0x4200	0x88	0x45	ChNo	1
Upper measurement limit exceeded	0x5000	0x81	0x31	ChNo	1
Lower measurement limit exceeded	0x5000	0x81	0x32	ChNo	1
Sensor fault	0x5000	0x81	0x33	ChNo	1
Input error	0x5000	0x81	0x41	ChNo	1
Output error	0x5000	0x81	0x42	ChNo	1
Configuration data error	0x6100	0x81	Index	Subindex	0
CAN overflow	0x8110	0x81	0x8110	0	0
CAN passive	0x8120	0x81	0x8120	0	0
Heartbeat node guarding	0x8130	0x81	0x8130	0	0
CAN recover	0x8140	0x81	0x8140	0	0
RxPDO too short	0x8210	0x81	Set length	COB ID	Actual length
RxPDO too long	0x8220	0x81	Set length	COB ID	Actual length

[1] **ErrReg:** Error register. Possible error codes:
0x81 = General error
0x84 = Voltage error
0x85 = General and voltage errors occurring together.

[2] **KnlNr:** Channel number on the module (starting with 1)

The PDO length error and configuration data error are not acknowledged. All other errors are acknowledged by the bus controller when they are no longer active. When there are no more errors on the bus controller, an emergency message is sent with all data set to 0 (according to CiA standard DS-301).

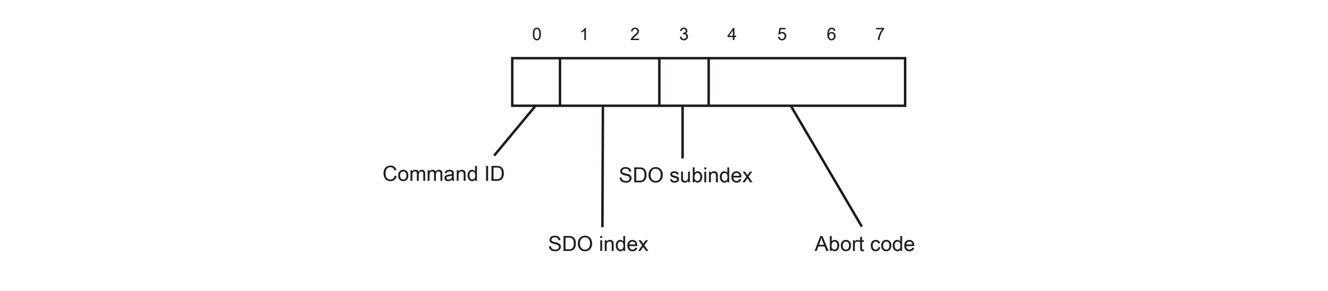
Example:

The power outputs of the module are cut off due to overtemperature.

Error message	0x0042884500210001
Error corrected	0x0000004500210001

16.5 SDO abort error messages

The following diagram illustrates the protocol structure for sending an error code.



The error codes listed in the table below are UNSIGNED32 values.

Abort code	Description
0x0503 0000	No change of state in the toggle bit
0x0504 0000	Timeout in SDO protocol
0x0504 0001	Client/Server "command specifier" invalid or unknown
0x0504 0002	Invalid block size (only if block mode enabled)
0x0504 0003	Invalid sequence number (only if block mode enabled)
0x0504 0004	CRC error (only if block mode enabled)
0x0504 0005	Outside the valid memory range
0x0601 0000	Access to object not supported
0x0601 0001	Attempt to read a "write-only" object
0x0601 0002	Attempt to write to a "read-only" object
0x0602 0000	Object not in object dictionary
0x0604 0041	Object can not be mapped to a PDO
0x0604 0042	Number and length of objects to be mapped would exceed the PDO length
0x0604 0043	General parameter incompatibility
0x0604 0047	General internal incompatibility in the device
0x0606 0000	Access failed due to a hardware error
0x0607 0010	Invalid data type, invalid length of service parameter
0x0607 0012	Invalid data type, value outside of permissible length for service parameter (too long)
0x0607 0013	Invalid data type, value outside of permissible length for service parameter (too short)
0x0609 0011	Subindex does not exist
0x0609 0030	Invalid parameter value (download only)
0x0609 0031	Value of the parameter to be written is too high (download only)
0x0609 0032	Value of the parameter to be written is too low (download only)
0x0609 0036	Maximum value is less than minimum value
0x060A 0023	Resource not available: SDO connection
0x0800 0000	General error
0x0800 0020	Data cannot be transferred or stored by the application.
0x0800 0021	Data cannot be transferred or stored by the application because of the local controller.
0x0800 0022	Data cannot be transferred or stored by the application because of the present device status.
0x0800 0023	Dynamically generated object dictionary is invalid or there is no object dictionary (e.g. object dictionary was generated from the file, with the generation failing due to a file error)
0x0800 0024	No data available

16.6 Manual configuration example

If an EDS or DCF file is transferred to the bus controller, then the entire configuration procedure takes place automatically.

If the master environment does not support the import of a device description file (EDS or DCF), then the configuration entries must be transferred manually.

This example illustrates the steps necessary to determine the configuration being used and to then transfer it.

16.6.1 Configuration procedure

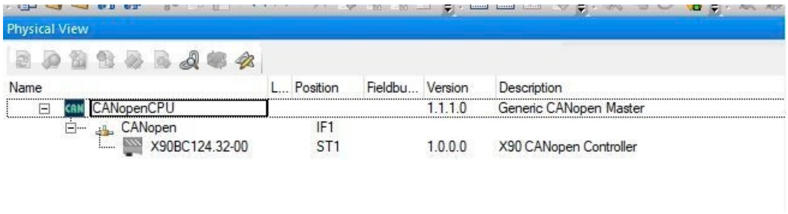
The following table shows the procedure for manually transferring the configuration to the bus controller.

	Object	Value	Information	
1. Prepare transfer				
	0x3000 - Sub0x10	True	Enables hidden EPLV2 entries	
2. Write module values for each module (loop 1)				
	0x31**xx** - Sub0x4	1	Configuration mode; see "Configuring multifunction I/Os" on page 65	
	0x31**xx** - Sub0x5B	Number of registers	Writes the number of registers to be transferred for the module in slot xx	
3. Write register values for each module (loop 2)				
	0x38**xx** - Sub0x01 + yy[1]	Register information	Contains the register number, size and type	32-bit transfer
	0x39**xx** - Sub0x01 + yy[1]	Register value	Writes the value to be transferred	
			or	
	0x31**xx** - Sub0x64 + yy[1]	Record value	Contains the 32-bit register information and register value	64-bit transfer
4. Saving a configuration				
	0x3FFF	0x65766173 ("save")	Saves the configuration on the bus controller	

[1] yy = Corresponds to object number "MOD_CfgEntry_yy_U64" on page 73

16.6.2 Determining the configuration being used

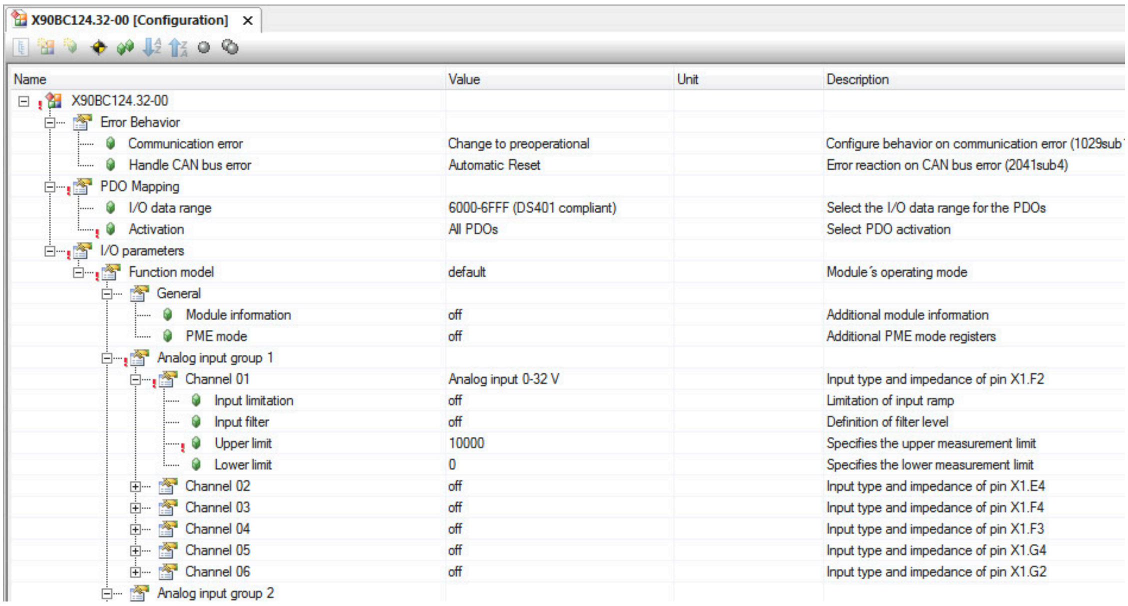
First, create a new project in Automation Studio (V4.7 or later) and add the CANopen bus controller X90BC124.32-00.



Physical View

Name	L...	Position	Fieldbu...	Version	Description
CANopenCPU				1.1.1.0	Generic CANopen Master
CANopen		IF1			
X90BC124.32-00		ST1		1.0.0.0	X90 CANopen Controller

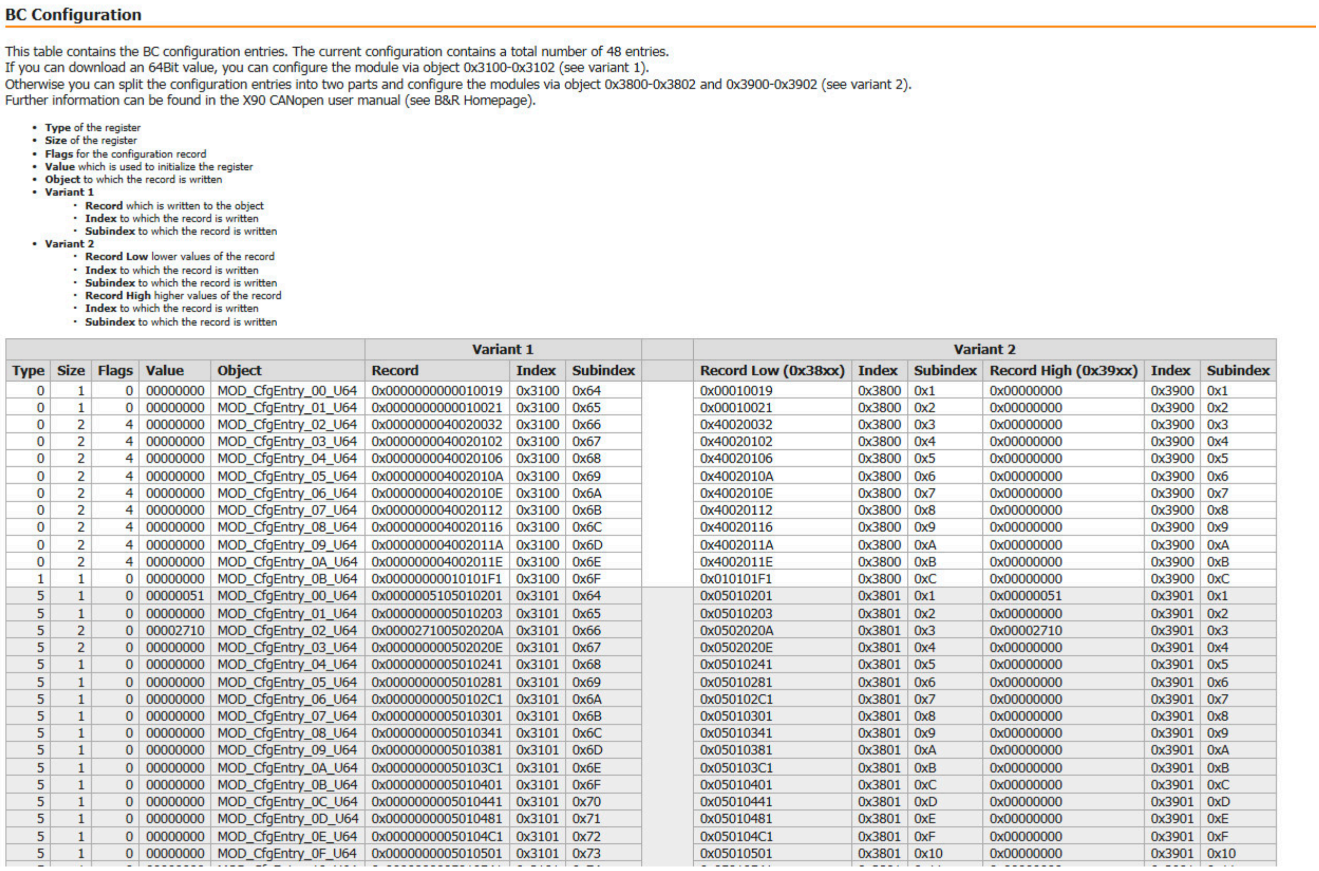
The bus controller is configured in the Configuration View.



X90BC124.32-00 [Configuration]

Name	Value	Unit	Description
X90BC124.32-00			
Error Behavior			
Communication error	Change to preoperational		Configure behavior on communication error (1029sub
Handle CAN bus error	Automatic Reset		Error reaction on CAN bus error (2041sub4)
PDO Mapping			
I/O data range	6000-6FFF (DS401 compliant)		Select the I/O data range for the PDOs
Activation	All PDOs		Select PDO activation
I/O parameters			
Function model	default		Module´s operating mode
General			
Module information	off		Additional module information
PME mode	off		Additional PME mode registers
Analog input group 1			
Channel 01	Analog input 0-32 V		Input type and impedance of pin X1.F2
Input limitation	off		Limitation of input ramp
Input filter	off		Definition of filter level
Upper limit	10000		Specifies the upper measurement limit
Lower limit	0		Specifies the lower measurement limit
Channel 02	off		Input type and impedance of pin X1.E4
Channel 03	off		Input type and impedance of pin X1.F4
Channel 04	off		Input type and impedance of pin X1.F3
Channel 05	off		Input type and impedance of pin X1.G4
Channel 06	off		Input type and impedance of pin X1.G2
Analog input group 2			

After the configuration is completed, 4 files are generated in the project's output folder. One of these is an HTML file containing a table with the values to be transferred listed under "BC configuration".



BC Configuration

This table contains the BC configuration entries. The current configuration contains a total number of 48 entries.
If you can download an 64Bit value, you can configure the module via object 0x3100-0x3102 (see variant 1).
Otherwise you can split the configuration entries into two parts and configure the modules via object 0x3800-0x3802 and 0x3900-0x3902 (see variant 2).
Further information can be found in the X90 CANopen user manual (see B&R Homepage).

- **Type** of the register
- **Size** of the register
- **Flags** for the configuration record
- **Value** which is used to initialize the register
- **Object** to which the record is written
- **Variant 1**
  - **Record** which is written to the object
  - **Index** to which the record is written
  - **Subindex** to which the record is written
- **Variant 2**
  - **Record Low** lower values of the record
  - **Index** to which the record is written
  - **Subindex** to which the record is written
  - **Record High** higher values of the record
  - **Index** to which the record is written
  - **Subindex** to which the record is written

					Variant 1				Variant 2					
Type	Size	Flags	Value	Object	Record	Index	Subindex		Record Low (0x38xx)	Index	Subindex	Record High (0x39xx)	Index	Subindex
0	1	0	00000000	MOD_CfgEntry_00_U64	0x0000000000010019	0x3100	0x64		0x00010019	0x3800	0x1	0x00000000	0x3900	0x1
0	1	0	00000000	MOD_CfgEntry_01_U64	0x0000000000010021	0x3100	0x65		0x00010021	0x3800	0x2	0x00000000	0x3900	0x2
0	2	4	00000000	MOD_CfgEntry_02_U64	0x0000000040020032	0x3100	0x66		0x40020032	0x3800	0x3	0x00000000	0x3900	0x3
0	2	4	00000000	MOD_CfgEntry_03_U64	0x0000000040020102	0x3100	0x67		0x40020102	0x3800	0x4	0x00000000	0x3900	0x4
0	2	4	00000000	MOD_CfgEntry_04_U64	0x0000000040020106	0x3100	0x68		0x40020106	0x3800	0x5	0x00000000	0x3900	0x5
0	2	4	00000000	MOD_CfgEntry_05_U64	0x000000004002010A	0x3100	0x69		0x4002010A	0x3800	0x6	0x00000000	0x3900	0x6
0	2	4	00000000	MOD_CfgEntry_06_U64	0x000000004002010E	0x3100	0x6A		0x4002010E	0x3800	0x7	0x00000000	0x3900	0x7
0	2	4	00000000	MOD_CfgEntry_07_U64	0x0000000040020112	0x3100	0x6B		0x40020112	0x3800	0x8	0x00000000	0x3900	0x8
0	2	4	00000000	MOD_CfgEntry_08_U64	0x0000000040020116	0x3100	0x6C		0x40020116	0x3800	0x9	0x00000000	0x3900	0x9
0	2	4	00000000	MOD_CfgEntry_09_U64	0x000000004002011A	0x3100	0x6D		0x4002011A	0x3800	0xA	0x00000000	0x3900	0xA
0	2	4	00000000	MOD_CfgEntry_0A_U64	0x000000004002011E	0x3100	0x6E		0x4002011E	0x3800	0xB	0x00000000	0x3900	0xB
1	1	0	00000000	MOD_CfgEntry_0B_U64	0x00000000010101F1	0x3100	0x6F		0x010101F1	0x3800	0xC	0x00000000	0x3900	0xC
5	1	0	00000051	MOD_CfgEntry_00_U64	0x0000005105010201	0x3101	0x64		0x05010201	0x3801	0x1	0x00000051	0x3901	0x1
5	1	0	00000000	MOD_CfgEntry_01_U64	0x0000000005010203	0x3101	0x65		0x05010203	0x3801	0x2	0x00000000	0x3901	0x2
5	2	0	00002710	MOD_CfgEntry_02_U64	0x000027100502020A	0x3101	0x66		0x0502020A	0x3801	0x3	0x00002710	0x3901	0x3
5	2	0	00000000	MOD_CfgEntry_03_U64	0x000000000502020E	0x3101	0x67		0x0502020E	0x3801	0x4	0x00000000	0x3901	0x4
5	1	0	00000000	MOD_CfgEntry_04_U64	0x0000000005010241	0x3101	0x68		0x05010241	0x3801	0x5	0x00000000	0x3901	0x5
5	1	0	00000000	MOD_CfgEntry_05_U64	0x0000000005010281	0x3101	0x69		0x05010281	0x3801	0x6	0x00000000	0x3901	0x6
5	1	0	00000000	MOD_CfgEntry_06_U64	0x00000000050102C1	0x3101	0x6A		0x050102C1	0x3801	0x7	0x00000000	0x3901	0x7
5	1	0	00000000	MOD_CfgEntry_07_U64	0x0000000005010301	0x3101	0x6B		0x05010301	0x3801	0x8	0x00000000	0x3901	0x8
5	1	0	00000000	MOD_CfgEntry_08_U64	0x0000000005010341	0x3101	0x6C		0x05010341	0x3801	0x9	0x00000000	0x3901	0x9
5	1	0	00000000	MOD_CfgEntry_09_U64	0x0000000005010381	0x3101	0x6D		0x05010381	0x3801	0xA	0x00000000	0x3901	0xA
5	1	0	00000000	MOD_CfgEntry_0A_U64	0x00000000050103C1	0x3101	0x6E		0x050103C1	0x3801	0xB	0x00000000	0x3901	0xB
5	1	0	00000000	MOD_CfgEntry_0B_U64	0x0000000005010401	0x3101	0x6F		0x05010401	0x3801	0xC	0x00000000	0x3901	0xC
5	1	0	00000000	MOD_CfgEntry_0C_U64	0x0000000005010441	0x3101	0x70		0x05010441	0x3801	0xD	0x00000000	0x3901	0xD
5	1	0	00000000	MOD_CfgEntry_0D_U64	0x0000000005010481	0x3101	0x71		0x05010481	0x3801	0xE	0x00000000	0x3901	0xE
5	1	0	00000000	MOD_CfgEntry_0E_U64	0x00000000050104C1	0x3101	0x72		0x050104C1	0x3801	0xF	0x00000000	0x3901	0xF
5	1	0	00000000	MOD_CfgEntry_0F_U64	0x0000000005010501	0x3101	0x73		0x05010501	0x3801	0x10	0x00000000	0x3901	0x10

The example configuration shown here contains 48 register entries. These must be transferred to the bus controller in order to operate it as desired.

16.6.3 Transferring the configuration

Depending on which master system is used, the configuration entries can be transferred using 32-bit or 64-bit values.

32-bit master system

Each configuration entry is transferred by 2 object calls:

- 0x38xx: Module configuration registers
- 0x39xx: Module configuration value

64-bit master system

Each configuration entry is transferred by 1 object call:

- 0x31xx: Configuring multifunction I/Os

Due to the large amount of data, these are grouped into several objects, each of which is addressed by its own index. For detailed information about the indexes used for the register calls, see the generated HTML file.

The following table lists all configuration objects for both systems that are transmitted to the bus controller.

After transferring the configuration, the bus controller must be manually set to mode OPERATIONAL. This can be done, for example, using function block "CANopenNMT" of library AsCANopen. See Automation Help → Programming → Libraries → Communication → AsCANopen.

Record	32-bit master system		64-bit master system	
	Value	**Object number**	**Object number**	**Value**
0x0000000000010019	0x00010019 →Register: 0x0019 →Size: 01 (bytes) →Type: 00[2]	0x3800 0x1	0x3100 - Sub0x64	0x0000000000010019
	0x00000000[1]	0x3900 0x1		
0x0000000000010021	0x00010021 →Register: 0x0021 →Size: 01 (bytes) →Type: 00[2]	0x3800 0x2	0x3100 - Sub0x65	0x0000000000010021
	0x00000000[1]	0x3900 0x2		
0x0000000040020032	0x40020032 →Register: 0x0032 →Size: 02 (bytes) →Type: 40[2]	0x3800 0x3	0x3100 - Sub0x66	0x0000000040020032
	0x00000000[1]	0x3900 0x3		
0x0000000040020102	0x40020102 →Register: 0x0102 →Size: 02 (bytes) →Type: 40[2]	0x3800 0x4	0x3100 - Sub0x67	0x0000000040020102
	0x00000000[1]	0x3900 0x4		
0x0000000040020106	0x40020106 →Register: 0x0106 →Size: 02 (bytes) →Type: 40[2]	0x3800 0x5	0x3100 - Sub0x68	0x0000000040020106
	0x00000000[1]	0x3900 0x5		
0x000000004002010A	0x4002010A →Register: 0x010A →Size: 02 (bytes) →Type: 40[2]	0x3800 0x6	0x3100 - Sub0x69	0x000000004002010A
	0x00000000[1]	0x3900 0x6		
0x000000004002010E	0x4002010E →Register: 0x010E →Size: 02 (bytes) →Type: 40[2]	0x3800 0x7	0x3100 - Sub0x6A	0x000000004002010E
	0x00000000[1]	0x3900 0x7		
0x0000000040020112	0x40020112 →Register: 0x0112 →Size: 02 (bytes) →Type: 40[2]	0x3800 0x8	0x3100 - Sub0x6B	0x0000000040020112
	0x00000000[1]	0x3900 0x8		
0x0000000040020116	0x40020116 →Register: 0x0116 →Size: 02 (bytes) →Type: 40[2]	0x3800 0x9	0x3100 - Sub0x6C	0x0000000040020116
	0x00000000[1]	0x3900 0x9		
0x000000004002011A	0x4002011A →Register: 0x011A →Size: 02 (bytes) →Type: 40[2]	0x3800 0xA	0x3100 - Sub0x6D	0x000000004002011A
	0x00000000[1]	0x3900 0xA		

Record	32-bit master system		64-bit master system	
	Value	Object number	Object number	Value
0x000000004002011E	0x4002011E →Register: 0x011E →Size: 02 (bytes) →Type: 40[2]	0x3800 0xB	0x3100 - Sub0x6E	0x000000004002011E
	0x00000000[1]	0x3900 0xB		
0x00000000010101F1	0x010101F1 →Register: 0x01F1 →Size: 01 (bytes) →Type: 01[2]	0x3800 0xC	0x3100 - Sub0x6F	0x00000000010101F1
	0x00000000[1]	0x3900 0xC		
0x0000005105010201	0x05010201 →Register: 0x0201 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x1	0x3101 - Sub0x64	0x0000005105010201
	0x00000051	0x3901 0x1		
0x0000000005010203	0x05010203 →Register: 0x0203 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x2	0x3101 - Sub0x65	0x0000000005010203
	0x00000000	0x3901 0x2		
0x000027100502020A	0x0502020A →Register: 0x020A →Size: 02 (bytes) →Type: 05[2]	0x3801 0x3	0x3101 - Sub0x66	0x000027100502020A
	0x00002710	0x3901 0x3		
0x000000000502020E	0x0502020E →Register: 0x020E →Size: 02 (bytes) →Type: 05[2]	0x3801 0x4	0x3101 - Sub0x67	0x000000000502020E
	0x00000000	0x3901 0x4		
0x0000000005010241	0x05010241 →Register: 0x0241 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x5	0x3101 - Sub0x68	0x0000000005010241
	0x00000000	0x3901 0x5		
0x0000000005010281	0x05010281 →Register: 0x0281 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x6	0x3101 - Sub0x69	0x0000000005010281
	0x00000000	0x3901 0x6		
0x00000000050102C1	0x050102C1 →Register: 0x02C1 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x7	0x3101 - Sub0x6A	0x00000000050102C1
	0x00000000	0x3901 0x7		
0x0000000005010301	0x05010301 →Register: 0x0301 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x8	0x3101 - Sub0x6B	0x0000000005010301
	0x00000000	0x3901 0x8		
0x0000000005010341	0x05010341 →Register: 0x0341 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x9	0x3101 - Sub0x6C	0x0000000005010341
	0x00000000	0x3901 0x9		
0x0000000005010381	0x05010381 →Register: 0x0381 →Size: 01 (bytes) →Type: 05[2]	0x3801 0xA	0x3101 - Sub0x6D	0x0000000005010381
	0x00000000	0x3901 0xA		
0x00000000050103C1	0x050103C1 →Register: 0x03C1 →Size: 01 (bytes) →Type: 05[2]	0x3801 0xB	0x3101 - Sub0x6E	0x00000000050103C1
	0x00000000	0x3901 0xB		
0x0000000005010401	0x05010401 →Register: 0x0401 →Size: 01 (bytes) →Type: 05[2]	0x3801 0xC	0x3101 - Sub0x6F	0x0000000005010401
	0x00000000	0x3901 0xC		
0x0000000005010441	0x05010441 →Register: 0x0441 →Size: 01 (bytes) →Type: 05[2]	0x3801 0xD	0x3101 - Sub0x70	0x0000000005010441
	0x00000000	0x3901 0xD		
0x0000000005010481	0x05010481 →Register: 0x0481 →Size: 01 (bytes) →Type: 05[2]	0x3801 0xE	0x3101 - Sub0x71	0x0000000005010481
	0x00000000	0x3901 0xE		

Record	32-bit master system		64-bit master system	
	Value	Object number	Object number	Value
0x00000000050104C1	0x050104C1 →Register: 0x04C1 →Size: 01 (bytes) →Type: 05[2]	0x3801 0xF	0x3101 - Sub0x72	0x00000000050104C1
	0x00000000	0x3901 0xF		
0x0000000005010501	0x05010501 →Register: 0x0501 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x10	0x3101 - Sub0x73	0x0000000005010501
	0x00000000	0x3901 0x10		
0x0000000005010541	0x05010541 →Register: 0x0541 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x11	0x3101 - Sub0x74	0x0000000005010541
	0x00000000	0x3901 0x11		
0x0000000005010581	0x05010581 →Register: 0x0581 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x12	0x3101 - Sub0x75	0x0000000005010581
	0x00000000	0x3901 0x12		
0x00000000050105C1	0x050105C1 →Register: 0x05C1 →Size: 01 (bytes) →Type: 05[2]	0x3801 0x13	0x3101 - Sub0x76	0x00000000050105C1
	0x00000000	0x3901 0x13		
0x0000000005010601	0x05010601 →Register: 0x0601 →Size: 01 (bytes) →Type: 05[2]	0x3802 0x1	0x3102 - Sub0x64	0x0000000005010601
	0x00000000	0x3902 0x1		
0x0000000005010641	0x05010641 →Register: 0x0641 →Size: 01 (bytes) →Type: 05[2]	0x3802 0x2	0x3102 - Sub0x65	0x0000000005010641
	0x00000000	0x3902 0x2		
0x0000000005010681	0x05010681 →Register: 0x0681 →Size: 01 (bytes) →Type: 05[2]	0x3802 0x3	0x3102 - Sub0x66	0x0000000005010681
	0x00000000	0x3902 0x3		
0x00000000050106C1	0x050106C1 →Register: 0x06C1 →Size: 01 (bytes) →Type: 05[2]	0x3802 0x4	0x3102 - Sub0x67	0x00000000050106C1
	0x00000000	0x3902 0x4		
0x0000000005010701	0x05010701 →Register: 0x0701 →Size: 01 (bytes) →Type: 05[2]	0x3802 0x5	0x3102 - Sub0x68	0x0000000005010701
	0x00000000	0x3902 0x5		
0x0000000005010741	0x05010741 →Register: 0x0741 →Size: 01 (bytes) →Type: 05[2]	0x3802 0x6	0x3102 - Sub0x69	0x0000000005010741
	0x00000000	0x3902 0x6		
0x0000000005010781	0x05010781 →Register: 0x0781 →Size: 01 (bytes) →Type: 05[2]	0x3802 0x7	0x3102 - Sub0x6A	0x0000000005010781
	0x00000000	0x3902 0x7		
0x00000000050107C1	0x050107C1 →Register: 0x07C1 →Size: 01 (bytes) →Type: 05[2]	0x3802 0x8	0x3102 - Sub0x6B	0x00000000050107C1
	0x00000000	0x3902 0x8		
0x0000000005010801	0x05010801 →Register: 0x0801 →Size: 01 (bytes) →Type: 05[2]	0x3802 0x9	0x3102 - Sub0x6C	0x0000000005010801
	0x00000000	0x3902 0x9		
0x0000000005010841	0x05010841 →Register: 0x0841 →Size: 01 (bytes) →Type: 05[2]	0x3802 0xA	0x3102 - Sub0x6D	0x0000000005010841
	0x00000000	0x3902 0xA		
0x0000000005010881	0x05010881 →Register: 0x0881 →Size: 01 (bytes) →Type: 05[2]	0x3802 0xB	0x3102 - Sub0x6E	0x0000000005010881
	0x00000000	0x3902 0xB		

Record	32-bit master system		64-bit master system	
	Value	Object number	Object number	Value
0x00000000050108C1	0x050108C1 →Register: 0x08C1 →Size: 01 (bytes) →Type: 05[2]	0x3802 0xC	0x3102 - Sub0x6F	0x00000000050108C1
	0x00000000	0x3902 0xC		
0x0000000005010901	0x05010901 →Register: 0x0901 →Size: 01 (bytes) →Type: 05[2]	0x3802 0xD	0x3102 - Sub0x70	0x0000000005010901
	0x00000000	0x3902 0xD		
0x0000000005010941	0x05010941 →Register: 0x0941 →Size: 01 (bytes) →Type: 05[2]	0x3802 0xE	0x3102 - Sub0x71	0x0000000005010941
	0x00000000	0x3902 0xE		
0x0000000005010981	0x05010981 →Register: 0x0981 →Size: 01 (bytes) →Type: 05[2]	0x3802 0xF	0x3102 - Sub0x72	0x0000000005010981
	0x00000000	0x3902 0xF		
0x00000000050109C1	0x050109C1 →Register: 0x09C1 →Size: 01 (bytes) →Type: 05[2]	0x3802 0x10	0x3102 - Sub0x73	0x00000000050109C1
	0x00000000	0x3902 0x10		
0x0000008005020A02	0x05020A02 →Register: 0x0A02 →Size: 02 (bytes) →Type: 05[2]	0x3802 0x11	0x3102 - Sub0x74	0x0000008005020A02
	0x00000080	0x3902 0x11		

[1] Configuration value 0x00 is mapped when using cyclic registers.

[2]
- Type 00: Dynamic cyclic input register
- Type 01: Dynamic cyclic output register
- Type 05: Acyclic output register
- Type 40: Analog input register

16.7 Example for using configuration files

The bus controller and all integrated I/O channels can be configured using Automation Studio V4.7 or later.

Automation Studio can be downloaded at no cost from the B&R website (www.br-automation.com). The evaluation license is permitted to be used to create complete configurations for fieldbus bus controllers at no cost.

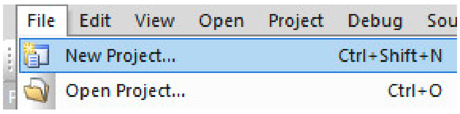
All integrated I/O channels can be easily configured via selection menus. Variables can be defined in the I/O mapping as usual.
When a project is compiled, configuration files are created that can be either integrated directly in another service provider's development environment or manually transferred to the bus controller.

16.7.1 Creating device description files

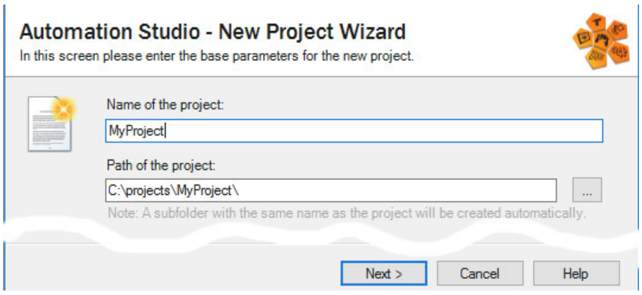
Device description files contain all relevant information and defined configurations of the X90 bus controller. Since the setting options are very extensive, Automation Studio should ideally be used to configure and create the required device description file. Depending on which device description file the master requires, the corresponding file generated by Automation Studio is given to the master. This allows the user to configure the X90 CANopen bus controller according to the settings.

16.7.1.1 Creating an Automation Studio project

- Create a new Automation Studio project by selecting "New project".



- Assign a project name and set up the project path.



Automation Studio - New Project Wizard

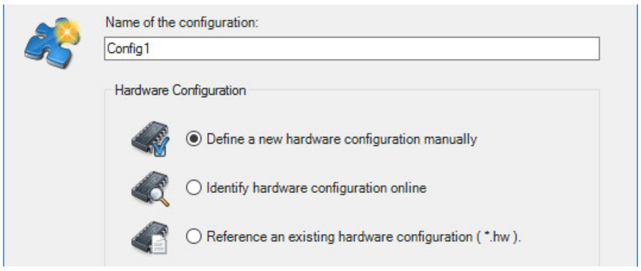
In this screen please enter the base parameters for the new project.

Name of the project: MyProject

Path of the project: C:\projects\MyProject\

Note: A subfolder with the same name as the project will be created automatically.

- Assign the hardware configuration type and configuration name.



Name of the configuration: Config1

Hardware Configuration

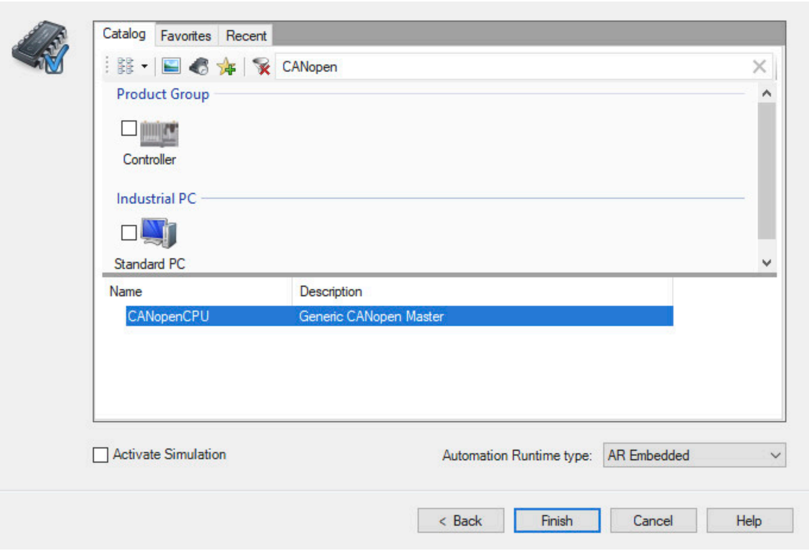
● Define a new hardware configuration manually

○ Identify hardware configuration online

○ Reference an existing hardware configuration (*.hw).

- The next step is to select the hardware. In order to simplify the search, different filters can be set in the Hardware Catalog. Lastly, highlight the required hardware and create the Automation Studio project by clicking on "Finish".

In this example, a virtual CANopen master specifically designed for creating device description files is used as the CPU for creating device description files.



Name	Description
CANopenCPU	Generic CANopen Master

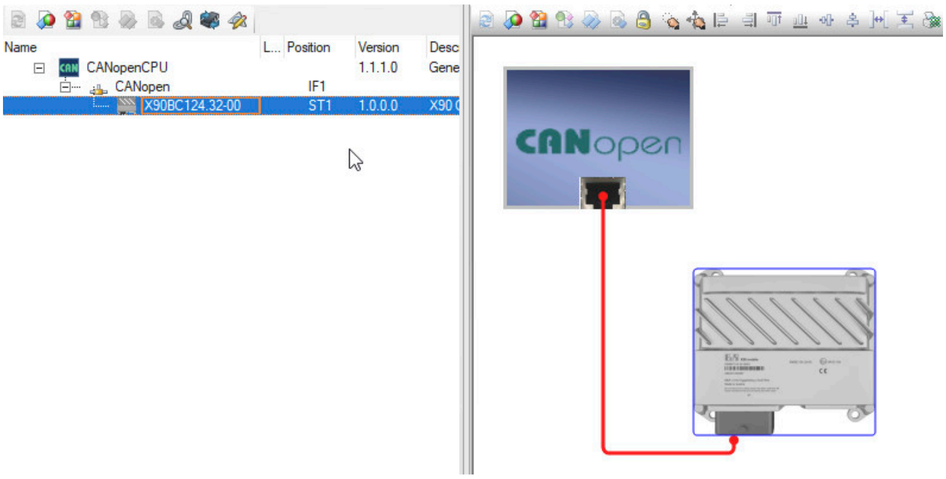
□ Activate Simulation

Automation Runtime type: AR Embedded

- To connect the X90 bus controller to the CANopen CPU, the hardware upgrade must be downloaded from the B&R website. The hardware upgrade is located in the download area of bus controller X90BC124-32-00.

- The hardware upgrade is imported to Automation Studio via Tools → Upgrades.

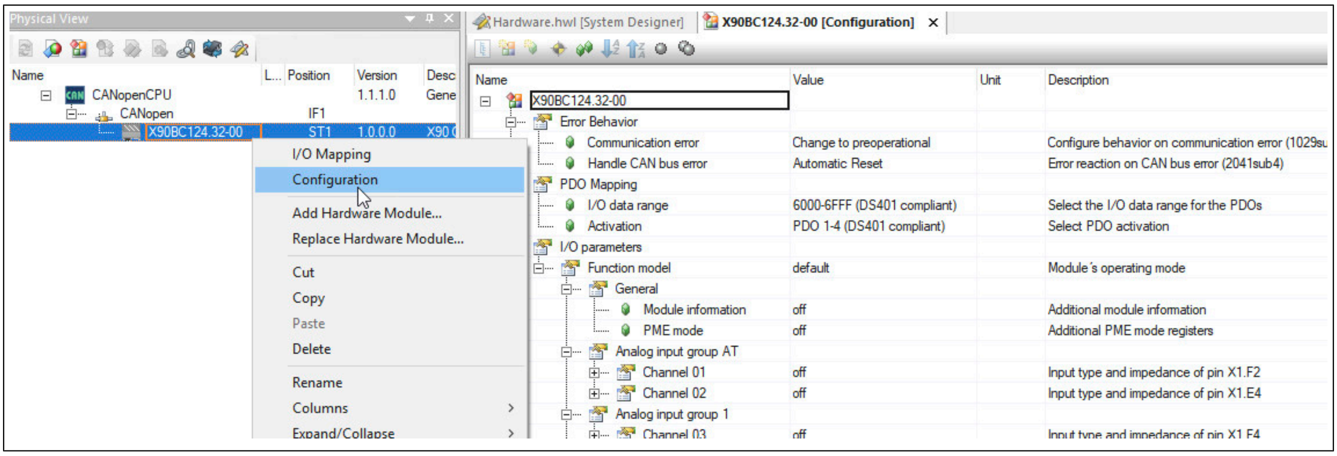
- After selecting the CANopen interface of the CANopen CPU, the X90 bus controller can be selected in the Hardware Catalog and is attached to the CANopen master by double-clicking or drag-and-drop.



Name	L...	Position	Version	Desc
CANopenCPU			1.1.1.0	Gene
CANopen		IF1		
X90BC124.32-00		ST1	1.0.0.0	X90 (

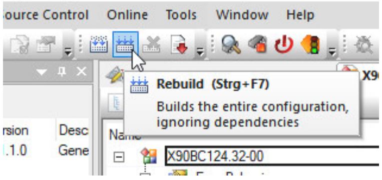
16.7.1.2 Configuring the bus controller

Now the X90 bus controller can be configured. To do this, right-click on the X90 bus controller and select "Configuration" to open the configuration interface and make the desired settings.



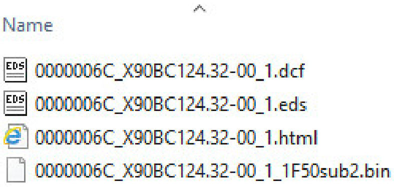
Name	Value	Unit	Description
X90BC124.32-00			
Error Behavior			
Communication error	Change to preoperational		Configure behavior on communication error (1029su
Handle CAN bus error	Automatic Reset		Error reaction on CAN bus error (2041sub4)
PDO Mapping			
I/O data range	6000-6FFF (DS401 compliant)		Select the I/O data range for the PDOs
Activation	PDO 1-4 (DS401 compliant)		Select PDO activation
I/O parameters			
Function model	default		Module´s operating mode
General			
Module information	off		Additional module information
PME mode	off		Additional PME mode registers
Analog input group AT			
Channel 01	off		Input type and impedance of pin X1.F2
Channel 02	off		Input type and impedance of pin X1.E4
Analog input group 1			
Channel 03	off		Input type and impedance of pin X1.F4

- After the bus controller has been configured, the device description files are created by compiling the project.



Rebuild (Strg+F7)
Builds the entire configuration, ignoring dependencies

If compilation takes place without errors, the resulting files are stored in the project folder under "Temp → Objects →*Name of configuration*→ CANopen_CPU → AsFDOutput". Depending on the master used, one of the generated files (EDS, DCF, BIN) is used to configure the bus controller.



Name

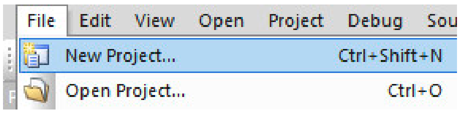
0000006C_X90BC124.32-00_1.dcf
0000006C_X90BC124.32-00_1.eds
0000006C_X90BC124.32-00_1.html
0000006C_X90BC124.32-00_1_1F50sub2.bin

16.7.2 Using the EDS or DCF file

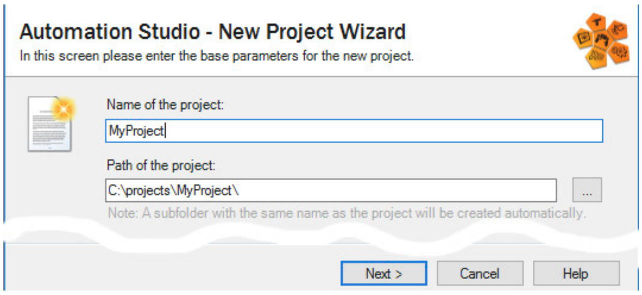
In this example, the CAN interface of an X90CPU is configured as the CANopen master, and the X90 bus controller is configured using an EDS device description file.

16.7.2.1 Creating an Automation Studio project

- Create a new Automation Studio project by selecting "New project".



- Assign a project name and set up the project path.



Automation Studio - New Project Wizard

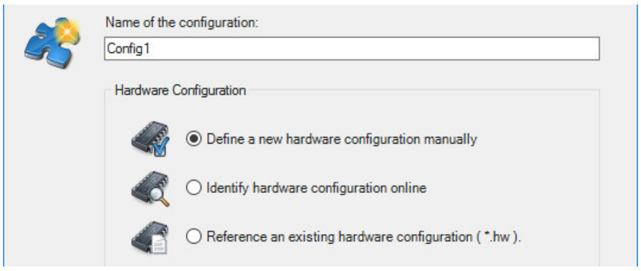
In this screen please enter the base parameters for the new project.

Name of the project: MyProject

Path of the project: C:\projects\MyProject\

Note: A subfolder with the same name as the project will be created automatically.

- Assign the hardware configuration type and configuration name.



Name of the configuration: Config1

Hardware Configuration

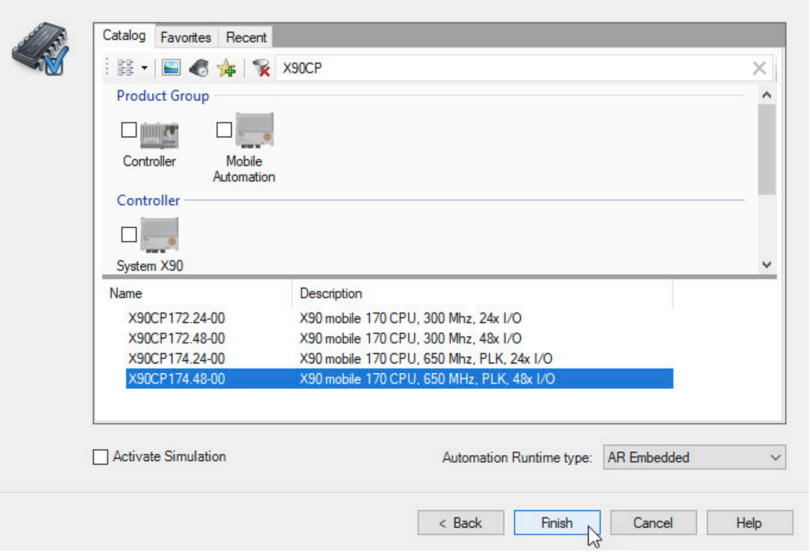
● Define a new hardware configuration manually

○ Identify hardware configuration online

○ Reference an existing hardware configuration (*.hw).

- The next step is to select the hardware. In order to simplify the search, different filters can be set in the Hardware Catalog. Lastly, highlight the required hardware and create the Automation Studio project by clicking on "Finish".

In this example, the CANopen master is the CAN interface of an X90CPU (X90CP174.48-00) that is configured as a CANopen master.



Name	Description
X90CP172.24-00	X90 mobile 170 CPU, 300 Mhz, 24x I/O
X90CP172.48-00	X90 mobile 170 CPU, 300 Mhz, 48x I/O
X90CP174.24-00	X90 mobile 170 CPU, 650 Mhz, PLK, 24x I/O
X90CP174.48-00	X90 mobile 170 CPU, 650 MHz, PLK, 48x I/O

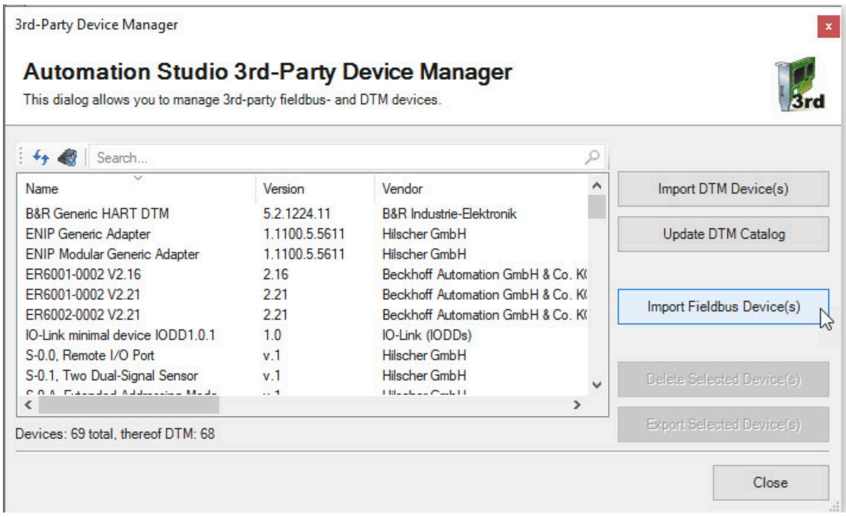
□ Activate Simulation

Automation Runtime type: AR Embedded

16.7.2.2 Adding the device description file and configuring the CAN master

To add and use a device description file in Automation Studio, perform the following steps:

- Open the dialog box in Automation Studio under "Tools - Manage 3rd-party devices" and select "Import fieldbus device(s)".



3rd-Party Device Manager

Automation Studio 3rd-Party Device Manager

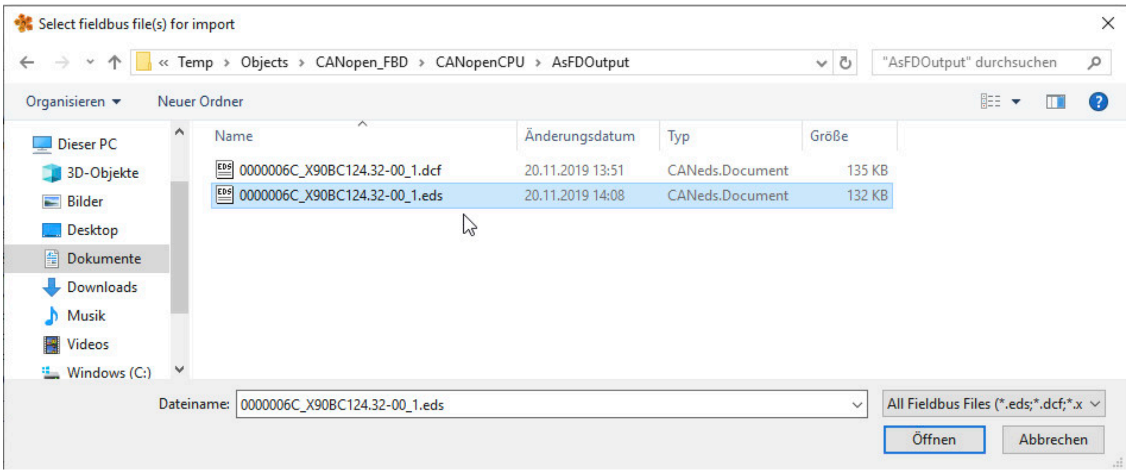
This dialog allows you to manage 3rd-party fieldbus- and DTM devices.

Name	Version	Vendor
B&R Generic HART DTM	5.2.1224.11	B&R Industrie-Elektronik
ENIP Generic Adapter	1.1100.5.5611	Hilscher GmbH
ENIP Modular Generic Adapter	1.1100.5.5611	Hilscher GmbH
ER6001-0002 V2.16	2.16	Beckhoff Automation GmbH & Co. K(
ER6001-0002 V2.21	2.21	Beckhoff Automation GmbH & Co. K(
ER6002-0002 V2.21	2.21	Beckhoff Automation GmbH & Co. K(
IO-Link minimal device IODD1.0.1	1.0	IO-Link (IODDs)
S-0.0, Remote I/O Port	v.1	Hilscher GmbH
S-0.1, Two Dual-Signal Sensor	v.1	Hilscher GmbH

Devices: 69 total, thereof DTM: 68

Search...
Import DTM Device(s)
Update DTM Catalog
Import Fieldbus Device(s)
Delete Selected Device(s)
Export Selected Device(s)
Close
3rd

- Select the EDS file to be imported and confirm with OK. The EDS file is imported into Automation Studio.



Select fieldbus file(s) for import

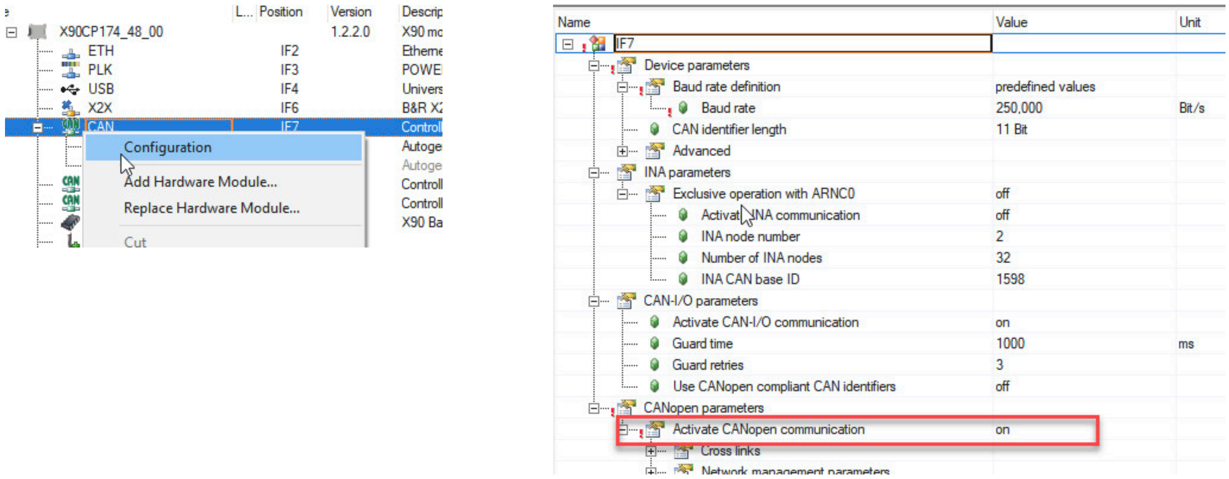
« Temp › Objects › CANopen_FBD › CANopenCPU › AsFDOutput

Name	Änderungsdatum	Typ	Größe
0000006C_X90BC124.32-00_1.dcf	20.11.2019 13:51	CANeds.Document	135 KB
0000006C_X90BC124.32-00_1.eds	20.11.2019 14:08	CANeds.Document	132 KB

Dateiname: 0000006C_X90BC124.32-00_1.eds

All Fieldbus Files (*.eds;*.dcf;*.x

- Next, the CAN interface of the X90CPU must be set as the CANopen master. This is done by opening the configuration interface of the interface and enabling CANopen communication by selecting "ON".



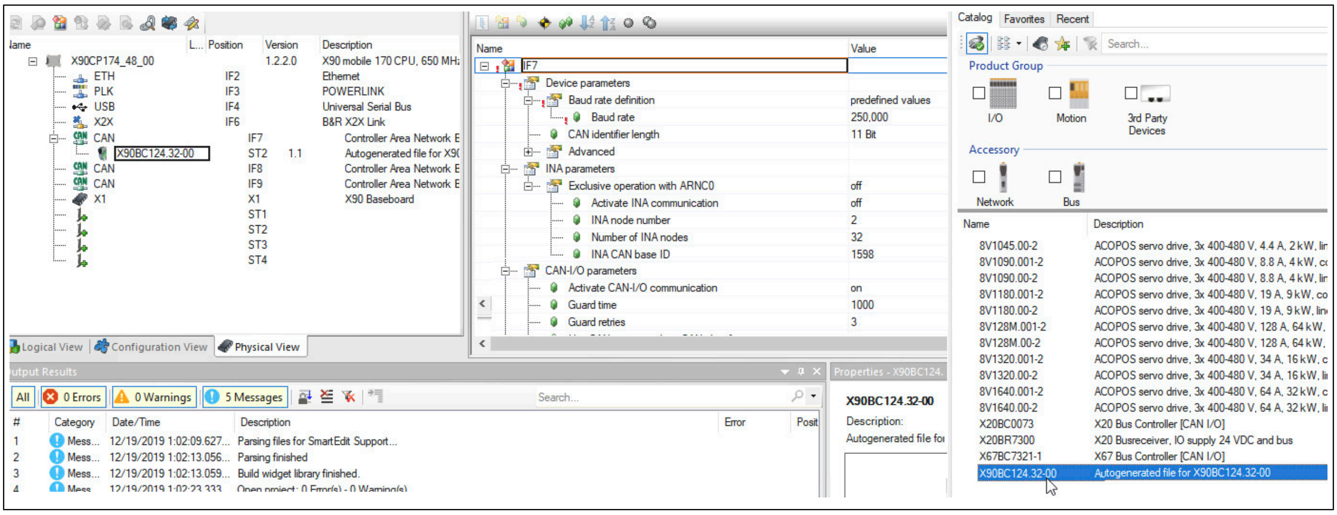
	L...	Position	Version	Descrip
X90CP174_48_00			1.2.2.0	X90 mo
ETH		IF2		Etherne
PLK		IF3		POWE
USB		IF4		Univers
X2X		IF6		B&R X2
CAN		IF7		Control

Name	Value	Unit
IF7		
Device parameters		
Baud rate definition	predefined values	
Baud rate	250,000	Bit/s
CAN identifier length	11 Bit	
Advanced		
INA parameters		
Exclusive operation with ARNC0	off	
Activate INA communication	off	
INA node number	2	
Number of INA nodes	32	
INA CAN base ID	1598	
CAN-I/O parameters		
Activate CAN-I/O communication	on	
Guard time	1000	ms
Guard retries	3	
Use CANopen compliant CAN identifiers	off	
CANopen parameters		
Activate CANopen communication	on	
Cross links		
Network management parameters		

In addition, the baud rate, SYNC, etc. can also be set on the master.

16.7.2.3 Configuring the device description file

- Select the CAN interface in the Physical View in Automation Studio and append it to the CAN interface from the Hardware Catalog using drag-and-drop.



Name	L...	Position	Version	Description
X90CP174_48_00			1.2.2.0	X90 mobile 170 CPU, 650 MHz
ETH		IF2		Ethernet
PLK		IF3		POWERLINK
USB		IF4		Universal Serial Bus
X2X		IF6		B&R X2X Link
CAN		IF7		Controller Area Network E
X90BC124.32-00		ST2	1.1	Autogenerated file for X9(
CAN		IF8		Controller Area Network E
CAN		IF9		Controller Area Network E
X1		X1		X90 Baseboard
		ST1		
		ST2		
		ST3		
		ST4		

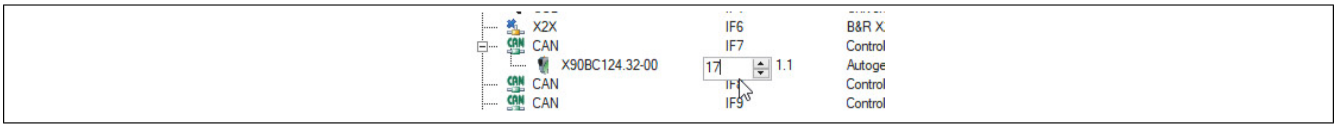
Name	Value
IF7	
Device parameters	
Baud rate definition	predefined values
Baud rate	250,000
CAN identifier length	11 Bit
Advanced	
INA parameters	
Exclusive operation with ARNC0	off
Activate INA communication	off
INA node number	2
Number of INA nodes	32
INA CAN base ID	1598
CAN-I/O parameters	
Activate CAN-I/O communication	on
Guard time	1000
Guard retries	3

Name	Description
8V1045.00-2	ACOPOS servo drive, 3x 400-480 V, 4.4 A, 2 kW, lir
8V1090.001-2	ACOPOS servo drive, 3x 400-480 V, 8.8 A, 4 kW, co
8V1090.00-2	ACOPOS servo drive, 3x 400-480 V, 8.8 A, 4 kW, lir
8V1180.001-2	ACOPOS servo drive, 3x 400-480 V, 19 A, 9 kW, co
8V1180.00-2	ACOPOS servo drive, 3x 400-480 V, 19 A, 9 kW, lin
8V128M.001-2	ACOPOS servo drive, 3x 400-480 V, 128 A, 64 kW,
8V128M.00-2	ACOPOS servo drive, 3x 400-480 V, 128 A, 64 kW,
8V1320.001-2	ACOPOS servo drive, 3x 400-480 V, 34 A, 16 kW, c
8V1320.00-2	ACOPOS servo drive, 3x 400-480 V, 34 A, 16 kW, li
8V1640.001-2	ACOPOS servo drive, 3x 400-480 V, 64 A, 32 kW, c
8V1640.00-2	ACOPOS servo drive, 3x 400-480 V, 64 A, 32 kW, li
X20BC0073	X20 Bus Controller [CAN I/O]
X20BR7300	X20 Busreceiver, IO supply 24 VDC and bus
X67BC7321-1	X67 Bus Controller [CAN I/O]
X90BC124.32-00	Autogenerated file for X90BC124.32-00

Properties - X90BC124.
X90BC124.32-00
Description:
Autogenerated file for

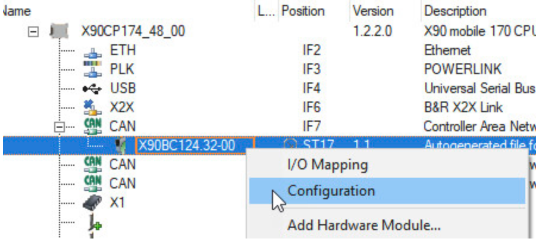
#	Category	Date/Time	Description
1	Mess...	12/19/2019 1:02:09.627...	Parsing files for SmartEdit Support...
2	Mess...	12/19/2019 1:02:13.056...	Parsing finished
3	Mess...	12/19/2019 1:02:13.059...	Build widget library finished.
4	Mess...	12/19/2019 1:02:23.333	Open project: 0 Error(s) - 0 Warning(s)

- Specify the node number the CANopen master is searching for in the network to the device description file. If this does not match the node number set on the slave, no connection is established. The node ID of the slave can be changed by right-clicking on the device description file and selecting menu option "Node number → Change node number" or by double-clicking on the position of the device description file. For details about setting the node number on the slave, see "Node number and transfer rate" on page 16.

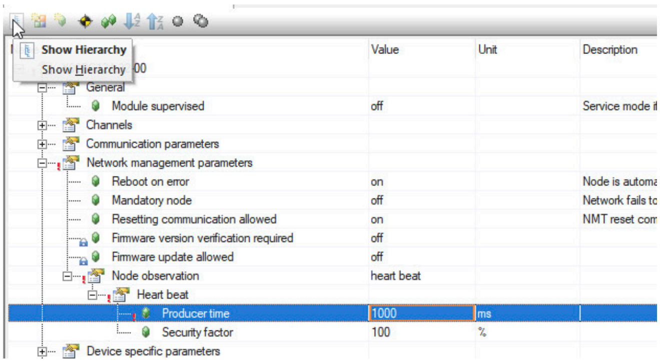


X2X	IF6		B&R X
CAN	IF7		Control
X90BC124.32-00	17	1.1	Autoge
CAN	IF		Control
CAN	IF9		Control

- In order to compile the project, the heartbeat must be set in the device description file. "Node observation" is already set to "Heartbeat" by default. "Producer time" must still be set, however.



Name	L...	Position	Version	Description
X90CP174_48_00			1.2.2.0	X90 mobile 170 CPU
ETH		IF2		Ethernet
PLK		IF3		POWERLINK
USB		IF4		Universal Serial Bus
X2X		IF6		B&R X2X Link
CAN		IF7		Controller Area Netw
X90BC124.32-00		ST17	1.1	Autogenerated file f
CAN				
CAN				
X1				



Name	Value	Unit	Description
General			
Module supervised	off		Service mode i
Channels			
Communication parameters			
Network management parameters			
Reboot on error	on		Node is automa
Mandatory node	off		Network fails to
Resetting communication allowed	on		NMT reset com
Firmware version verification required	off		
Firmware update allowed	off		
Node observation	heart beat		
Heart beat			
Producer time	1000	ms	
Security factor	100	%	
Device specific parameters			

All other settings are optional, e.g. "Transmission type" of the communication parameters (PDOs).

- Finally, the project is compiled and transferred to the CPU. If the X90 bus controller has been correctly attached to the CAN interface of the CPU, a connection between master and slave is established. This can be checked by opening the I/O mapping of the device description file and reading the "ModulOK" value in monitor mode. If "ModulOK" is TRUE, then a connection has been established and data can be exchanged.

16.7.3 Using the BIN file

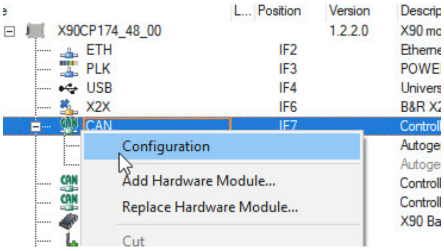
This example explains how a BIN file can be transferred via an application using a B&R X90 CPU with a CAN interface.

16.7.3.1 Creating an Automation Studio project

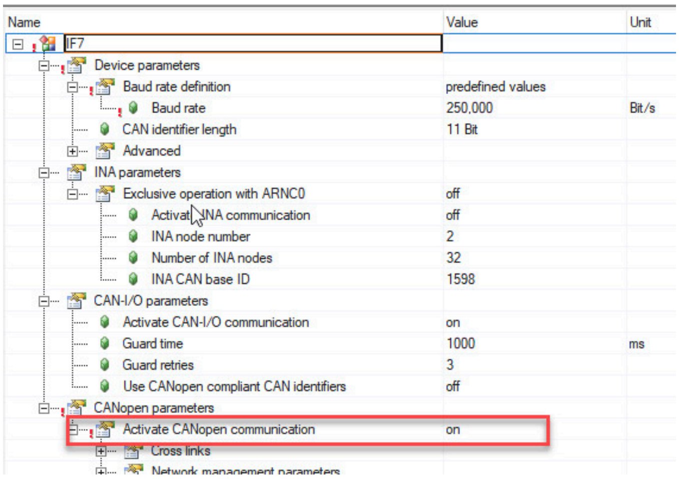
- For detailed information about creating an Automation Studio project, see "Creating an Automation Studio project" on page 82.

16.7.3.2 Configuring the CAN master

- The CAN interface of the X90CPU must be set as the CANopen master. This is done by opening the configuration interface of the interface, and the CANopen communication is enabled by selecting "ON".



	L...	Position	Version	Descrip
X90CP174_48_00			1.2.2.0	X90 mc
ETH		IF2		Etherne
PLK		IF3		POWE
USB		IF4		Univers
X2X		IF6		B&R X
CAN		IF7		Control
				Autoge
				Autoge
				Controll
				Controll
				X90 Ba



Name	Value	Unit
IF7		
Device parameters		
Baud rate definition	predefined values	
Baud rate	250,000	Bit/s
CAN identifier length	11 Bit	
Advanced		
INA parameters		
Exclusive operation with ARNC0	off	
Activate INA communication	off	
INA node number	2	
Number of INA nodes	32	
INA CAN base ID	1598	
CAN-I/O parameters		
Activate CAN-I/O communication	on	
Guard time	1000	ms
Guard retries	3	
Use CANopen compliant CAN identifiers	off	
CANopen parameters		
Activate CANopen communication	on	
Cross links		
Network management parameters		

In addition, the baud rate, SYNC, etc. can also be set on the master.

> ## Information:
>
> **Since no device description file was used for the configuration, no device is connected to the CANopen master interface in the Physical View in Automation Studio.**
>
> **For this reason, monitor mode cannot be used for the mapping in Automation Studio, and process data is only read or written via the application, e.g. using function blocks "CANopenPDORead8()" and "CANopenPDOWrite8()".**

16.7.3.3 Transferring the BIN file

- The created configuration can now be transferred to the bus controller via an application. To do this, the following steps are necessary:

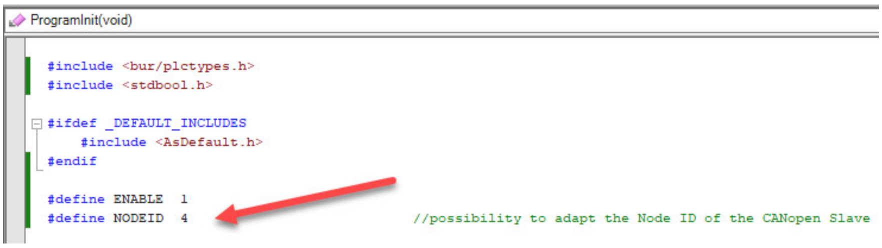
1) Load the BIN file into the application. This can be done in Automation Studio using library *FileIO*.
2) Set the X90 bus controller to mode PREOPERATIONAL, e.g. using function block "CANopenNMT()".
3) Delete the configuration by writing value "load" to object 0x1011 Sub 0x1. This can be done using function block "CANopenSDOWrite8()".
4) Restart the bus controller.
5) Transfer the BIN file on the bus controller to object 0x1F50 Sub0x2. This can be done with function block "CANopenSDOWriteData()".
6) If the I/O range 0x3200 to 0x37FF is used for mapping, transfer the BIN file again.
7) Set the X90 bus controller to mode OPERATIONAL again, e.g. using function block "CANopenNMT()".

16.7.3.4 Sample project

For a complete sample project that transfers the configuration via a BIN file, see Automation Help → Programming → Libraries → Communication → AsCANopen → Samples.

This sample project is only suitable if the default I/O object range 0x6000 to 0x6FFF (6000-6FFF DS401 compliant) is mapped in the PDO mapping. If the object range 0x3000 to 0x37FF should be mapped, the BIN file must be transferred twice.

The set node number of the CANopen slave can be adjusted at a central location.

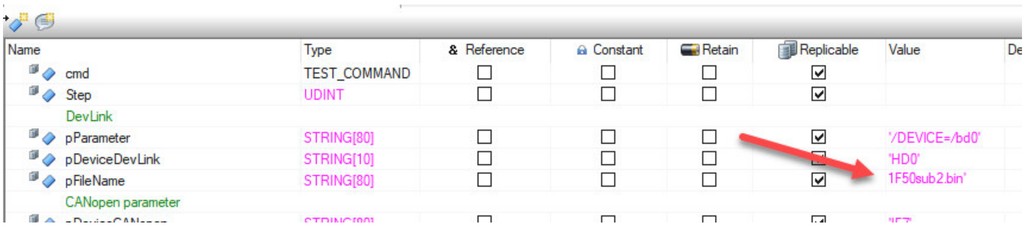


```
#include <bur/plctypes.h>
#include <stdbool.h>

#ifdef _DEFAULT_INCLUDES
    #include <AsDefault.h>
#endif

#define ENABLE  1
#define NODEID  4                    //possibility to adapt the Node ID of the CANopen Slave
```

The sample project reads the BIN file from a USB flash drive that must be plugged into the X90CPU. The name of the BIN file is "0000006C_X90BC124.32-00_1_1F50sub2.bin" and can be adjusted at the following location.



Name	Type	Reference	Constant	Retain	Replicable	Value	De
cmd	TEST_COMMAND	□	□	□	☑		
Step	UDINT	□	□	□	☑		
DevLink							
pParameter	STRING[80]	□	□	□	☑	'/DEVICE=/bd0'	
pDeviceDevLink	STRING[10]	□	□	□	☑	'HD0'	
pFileName	STRING[80]	□	□	□	☑	1F50sub2.bin'	
CANopen parameter							