Original Programming Manual
ecomatController/98-1

CR720S

Operating System V2.5.0.n
CODESYS® V3.5 SP11

English

7391124_02_UK 2017-12-19

Contents

1 About this manual 6

1.1 Legal and copyright information ... 6
1.2 Purpose of the document ... 7
1.3 Symbols and formats used ... 7
1.4 Overview: documentation modules for CR720S ... 8
1.5 Overview: documentation for CODESYS 3.n ... 8
1.6 How is this documentation structured? ... 9
1.7 History of the document CR0720 ... 9

2 Functions and features 10

3 Safety instructions 11

3.1 Please note! ... 11
3.2 What previous knowledge is required? ... 12
3.3 Important standards ... 13
3.4 Organise the creation of safe machinery with the V model ... 14
3.5 Start-up behaviour of the controller ... 16
3.6 Warnings used ... 16
3.7 Notes: serial number ... 17

4 Installation 18

4.1 System requirements ... 18
4.1.1 Hardware ... 18
4.1.2 Software ... 18
4.1.3 Licensing ... 18
4.2 Carry out installation ... 19
4.2.1 CODESYS programming software ... 19
4.2.2 Complete package for ecomatController CR720S ... 20
4.2.3 Check the operating system version of the device ... 22
4.2.4 Update the operating system of the device ... 24

5 System description 26

5.1 Hardware description ... 26
5.1.1 Hardware structure ... 26
5.1.2 Device supply (technology) ... 33
5.1.3 Monitoring concept ... 36
5.1.4 Inputs (technology) ... 38
5.1.5 Output types ... 44
5.1.6 Feedback in case of externally supplied outputs ... 51
5.2 Interfaces ... 52
5.2.1 Serial interface ... 52
5.2.2 Ethernet interface ... 52
5.2.3 CAN: Interfaces and protocols ... 53
5.3 Software description ... 54
5.3.1 Overview: Software ... 54
5.3.2 Software module for the device ... 56

6 Getting started 58

6.1 Start CODESYS ... 58
6.2 Create CODESYS project ... 58
6.2.1 Create new project with CR720S ... 59

6.2.2 Overview: Project structure with CR720S ... 60
6.3 Use CODESYS user manual ... 61
6.4 Configure programming interface ... 62
6.4.1 Set communication path of PLC ... 62
6.5 Add ifm function libraries to the application ... 63
6.6 Activate the access protection for a project ... 64

7 System configuration 65

7.1 Configure PLC ... 65
7.1.1 Allocate memory partition ... 65
7.1.2 Allocate inputs/outputs ... 66
7.1.3 Manage files ... 67
7.1.4 Manage users and groups ... 67
7.2 Configure inputs and outputs ... 68
7.2.1 via system configuration ... 68
7.2.2 via function block ... 68
7.3 Configure interfaces ... 69
7.3.1 Configure serial interface ... 69
7.3.2 Configure Ethernet interface ... 69
7.3.3 Configure CAN interfaces ... 71
7.3.4 Interface configuration file comconf.cfg ... 77

8 Programming 78

8.1 Objects of a PLC application ... 78
8.2 Create PLC application ... 79
8.2.1 Supported programming languages ... 79
8.2.2 Supported variable types ... 80
8.2.3 Options to access input and output data ... 81
8.2.4 Configure task processing ... 81
8.3 Use ifm function libraries ... 82
8.3.1 Access to inputs ... 82
8.3.2 Access to outputs ... 83
8.3.3 Control device ... 83
8.3.4 Read device information ... 83
8.4 Use IO mapping ... 84
8.4.1 Access inputs ... 85
8.4.2 Access outputs ... 86
8.4.3 Read diagnostic data of the device ... 87
8.5 Use RawCAN (CAN Layer 2) ... 88
8.5.1 RawCAN: Control CAN network nodes ... 88
8.5.2 RawCAN: Send and receive CAN messages ... 88
8.5.3 RawCAN: Request and send remote CAN messages ... 89
8.6 Use CANopen ... 90
8.6.1 CANopen: Send and receive SDO ... 90
8.6.2 CANopen: Network Management (NMT) ... 90
8.7 Use SAE J1939 ... 91

9 Operation 92

9.1 Transfer CODESYS project to device ... 92
9.1.1 Load the application to the device ... 92
9.1.2 Delete application from CR720S ... 93
9.2 Operating states ... 94
9.3 Status LEDs ... 95
9.3.1 Status LED: system ifm operating system (SYS0+SYS1) ... 95
9.3.2 Status LED: system PLC (SYS0, SYS1) ... 96
9.3.3 Status LED: System bootloader (SYS0) ... 96
9.3.4 Status LED: Ethernet interfaces (ETH0, ETH1) ... 97
9.3.5 Controlling LEDs in the applications ... 97

9.4 Reset 98
9.4.1 Supported reset variants 98
9.4.2 Reset application (warm) 98
9.4.3 Reset application (cold) 99
9.4.4 Reset application (origin) 99
9.5 Data transmission for series production 100
9.5.1 Transmission of the files with CODESYS 100
9.5.2 Data transmission with TFTP 101
9.5.3 Files for series production 101
9.6 Display system information 102

10 ifm function libraries 103

10.1 General 103
10.2 Library ifmCANopenManager.library 103
10.2.1 COP_GetNodeState 104
10.2.2 COP_SDOread 106
10.2.3 COP_SDOwrite 108
10.2.4 COP_SendNMT 110
10.2.5 NMT_SERVICE (ENUM) 112
10.2.6 NMT_STATES (ENUM) 112
10.3 Library ifmDeviceCR0721.library 113
10.3.1 CAN_BAUDRATE (ENUM) 114
10.3.2 CAN_CHANNEL (ENUM) 114
10.3.3 CANconstants (GVL) 114
10.3.4 SysInfo (GVL) 114
10.3.5 SysInfoStruct (STRUCT) 115
10.3.6 LED_COLOUR (ENUM) 115
10.3.7 LED_FLASH_FREQ (ENUM) 115
10.4 ifmFastInput.library 116
10.4.1 FastCount 117
10.4.2 IncEncoder 119
10.4.3 Period 121
10.4.4 COUNT_DIRECTION (ENUM) 123
10.4.5 ENCODER_RESOLUTION (ENUM) 123
10.4.6 FREQ_SENSE_PERIODS (ENUM) 123
10.4.7 MODE_FAST_COUNT (ENUM) 123
10.4.8 MODE_INC_ENCODER (ENUM) 124
10.4.9 MODE_PERIOD (ENUM) 124
10.5 Library ifmIOcommon.library 125
10.5.1 Input 126
10.5.2 Output 129
10.5.3 SetLED 132
10.5.4 SupplySwitch 134
10.5.5 SystemSupply 136
10.5.6 Temperature 138
10.5.7 FILTER_INPUT (ENUM) 140
10.5.8 FILTER_OUTPUT (ENUM) 140
10.5.9 MODE_INPUT (ENUM) 141
10.5.10 MODE_OUTPUT (ENUM) 142
10.5.11 SYS_VOLTAGE_CHANNEL (ENUM) 142
10.6 Library ifmOutGroup 143
10.6.1 OutputGroup 144
10.6.2 FILTER_OUTPUT_GROUP (ENUM) 147
10.6.3 MODE_OUTPUT_GROUP (ENUM) 147
10.7 Library ifmOutHBridge 148
10.7.1 HBridge 149
10.7.2 MODE_BRAKE (ENUM) 152
10.8 Library ifmOutPWM 153
10.8.1 CurrentControl 154
10.8.2 PWM1000 157
10.8.3 MODE_CURRENT_CONTROL (ENUM) 161
10.8.4 MODE_PWM (ENUM) 161

10.9 Library ifmRawCAN.library....162
10.9.1 CAN_Enable....163
10.9.2 CAN_Recover....165
10.9.3 CAN_RemoteRequest....167
10.9.4 CAN_RemoteResponse....169
10.9.5 CAN_Rx....171
10.9.6 CAN_RxMask....173
10.9.7 CAN_RxRange....175
10.9.8 CAN_Tx....177
10.9.9 CAN_Info (GVL)....179
10.9.10 CAN_BUS_STATE (STRUCT)....179

11 Troubleshooting 180
11.1 Error classes....180
11.2 Error messages....181
11.3 Messages / diagnostic codes of the function blocks....181

12 Appendix 182
12.1 Directory structure and file overview....182
12.2 ifm behaviour models for function blocks....183
12.2.1 General....183
12.2.2 Behaviour model ENABLE....183
12.2.3 Behaviour model EXECUTE....184

13 Glossary of Terms 185

14 Index 199

15 Notizen • Notes • Notes 203

16 ifm weltweit • ifm worldwide • ifm à l'échelle internationale 207

1 About this manual

Contents

Legal and copyright information 6
Purpose of the document 7
Symbols and formats used 7
Overview: documentation modules for CR720S 8
Overview: documentation for CODESYS 3.n 8
How is this documentation structured? 9
History of the document CR0720 9

202

1.1 Legal and copyright information

6088

All product names, pictures, companies or other brands used on our pages are the property of the respective rights owners:

- AS-i is the property of the AS-International Association, (→ www.as-interface.net)
- CAN is the property of the CiA (CAN in Automation e.V.), Germany (→ www.can-cia.org)
- CODESYS™ is the property of the 3S – Smart Software Solutions GmbH, Germany (→ www.codesys.com)
- DeviceNet™ is the property of the ODVA™ (Open DeviceNet Vendor Association), USA (→ www.odva.org)
- EtherNet/IP® is the property of the →ODVA™
- EtherCAT® is a registered trade mark and patented technology, licensed by Beckhoff Automation GmbH, Germany
- IO-Link® (→ www.io-link.com) is the property of the →PROFIBUS Nutzerorganisation e.V., Germany
- ISOBUS is the property of the AEF – Agricultural Industry Electronics Foundation e.V., Deutschland (→ www.aef-online.org)
- Microsoft® is the property of the Microsoft Corporation, USA (→ www.microsoft.com)
- PROFIBUS® is the property of the PROFIBUS Nutzerorganisation e.V., Germany (→ www.profibus.com)
- PROFINET® is the property of the →PROFIBUS Nutzerorganisation e.V., Germany
- Windows® is the property of the →Microsoft Corporation, USA

1.2 Purpose of the document

22852

This manual describes of the **ecomat*mobile*** family for mobile machines of **ifm electronic gmbh**:

- ecomatController <eco100-Bez> (Art.-Nr.: CR720S) firmware version V2.5.0.n and higher

The CODESYS programming system is required to program this device:
version 3.5 SP11 or higher

These instructions describe the following topics:

- Configuration of the device in the setup mode
- Firmware update of the device in the recovery mode
- Configuration of the device using CODESYS
- Programming of the device-internal PLC of the CR720S using the CODESYS programming system.
- Description of the device-specific CODESYS function libraries

1.3 Symbols and formats used

15989

⚠ WARNING

Death or serious irreversible injuries may result.

⚠ CAUTION

Slight reversible injuries may result.

NOTICE

Property damage is to be expected or may result.

	Important note Non-compliance can result in malfunction or interference
	Information Supplementary note
► ...	Request for action
> ...	Reaction, result
→ ...	"see"
abc	Cross-reference
123 0x123 0b010	Decimal number Hexadecimal number Binary number
[...]	Designation of pushbuttons, buttons or indications

1.4 Overview: documentation modules for CR720S

22853

The documentation for this devices consists of the following modules:
(Downloads from ifm's website → **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ p. 207))

Document	Contents / Description
Data sheet	Technical data in a table
Installation instructions (are supplied with the device)	• Instructions for installation, electrical installation, and commissioning • Technical data
Programming manual	• Functions of the setup menu of the device • Creation of a CODESYS project with this device • Target settings with CODESYS • Programming of the device-internal PLC with CODESYS • Description of the device-specific CODESYS function libraries
System manual "Know-How ecomatmobile"	Know-how about the following topics (examples): • Overview Templates and demo programs • CAN, CANopen • Control outputs • Visualisations • Overview of the files and libraries

1.5 Overview: documentation for CODESYS 3.n

22856

The following user documentation is provided by 3S GmbH for programming the CR720S with CODESYS:

Document	Content / Description
Online help	• Context-sensitive help • Description of the CODESYS programming system After the installation of the programming system store and accessible on the hard disk of the PC/laptop: `...\Programme (x86)\3S CODESYS\CODESYS\Online Help`
CODESYS installation and first steps	• Remarks about the installing of the programming system CODESYS • First steps for handling the programming system CODESYS After the installation of the programming system store and accessible on the hard disk of the PC/laptop: `...\Programme (x86)\3S CODESYS\CODESYS\Documentation`
CODESYS user manual Safety SIL2	• [H2] CODESYS Safety SIL2 - IEC Programming Guidelines.pdf This document is for programmers who program safety-related controllers. Download at: → www.codesys.com

1.6 How is this documentation structured?

204
1508

This documentation is a combination of different types of manuals. It is for beginners and also a reference for advanced users. This document is addressed to the programmers of the applications.

How to use this manual:

- Refer to the table of contents to select a specific subject.
- Using the index you can also quickly find a term you are looking for.
- At the beginning of a chapter we will give you a brief overview of its contents.
- Abbreviations and technical terms → Appendix.

In case of malfunctions or uncertainties please contact the manufacturer at:
Contact → **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ p. 207)

We want to become even better! Each separate section has an identification number in the top right corner. If you want to inform us about any inconsistencies, indicate this number with the title and the language of this documentation. Thank you very much for your support!

We reserve the right to make alterations which can result in a change of contents of the documentation. You can find the current version on **ifm's** website:
→ **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ p. 207)

1.7 History of the document CR0720

22863

What has been changed in this manual? An overview:

Date	State	Change
2017-06-13	01	new document
2017-12-19	02	Article number, operating system version, CODESYS version

2 Functions and features

23289

This device is used to control processes in applications.
For this, device 2 contains 2 PLCs that can be programmed independently of each other.
In the CODESYS software platform, these PLCs are called:
- standard PLC
- safety PLC

23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

3 Safety instructions

Contents

Please note! ... 11
What previous knowledge is required? ... 12
Important standards ... 13
Organise the creation of safe machinery with the V model ... 14
Start-up behaviour of the controller ... 16
Warnings used ... 16
Notes: serial number ... 17

213

3.1 Please note!

22910
11212

No characteristics are warranted on the basis of the information, notes and examples provided in this manual. The drawings, representations and examples imply no responsibility for the system and no application-specific particularities.

- ► The manufacturer of the machine/equipment is responsible for ensuring the safety of the machine/equipment.
- ► Follow the national and international regulations of the country in which the machine/installation is to be placed on the market!

⚠ WARNING

Non-observance of these instructions can lead to property damage or bodily injury!
ifm electronic gmbh does not assume any liability in this regard.

- ► The acting person must have read and understood the safety instructions and the corresponding chapters in this manual before working on and with this device.
- ► The acting person must be authorised to work on the machine/equipment.
- ► The acting person must have the qualifications and training required to perform this work.
- ► Adhere to the technical data of the devices!
  You can find the current data sheet on **ifm's** homepage.
- ► Observe the installation and wiring information as well as the functions and features of the devices!
  → supplied installation instructions or on **ifm's** homepage
- ► Please note the corrections and notes in the release notes for the existing hardware, software and documentation, available on the **ifm** website

Website → **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ p. 207)

8340

⚠ WARNING

The user is responsible for the reliable function of the application programs he designed. If necessary, he must additionally carry out an approval test by corresponding supervisory and test organisations according to the national regulations.

23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

3.2 What previous knowledge is required?

215

This document is intended for people with knowledge of control technology and PLC programming with IEC 61131-3.

To program the PLC, the people should also be familiar with the CODESYS software.

The document is intended for specialists. These specialists are people who are qualified by their training and their experience to see risks and to avoid possible hazards that may be caused during operation or maintenance of a product. The document contains information about the correct handling of the product.

Read this document before use to familiarise yourself with operating conditions, installation and operation. Keep the document during the entire duration of use of the device.

Adhere to the safety instructions.

3.3 Important standards

22977
23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

Among other things, the programmer of safety-related controllers should also know and observe the content of the following standards:

Standard	Title, content
IEC 61508	Standard: Functional safety of electrical/electronic/programmable electronic safety-related systems
ISO 13849	Standard: Safety of machinery, safety-related parts of control systems • Part 1: General design principles • Part 2: Validation here: The device can be used up to PL d
IEC 62061	Standard: Machine safety – functional safety of electric, electronic and programmable machine controllers • Specification of the functional requirements • Specification of the safety requirements here: The device can be used up to SIL CL 2

3.4 Organise the creation of safe machinery with the V model

13264

Summary

- ► Define, observe, check and document the workflow steps in the V model
- ► Define responsibilities for tasks
- ► Define a person responsible for safety technology (Functional Safety Manager)

Every machine manufacturer should define the responsibilities for the respective tasks within his company through organisational measures. This is independent of the work steps ...

- for machine design
- for the creation of the application program for the SafetyController

A staff member responsible for the safety technology (FSM = Functional Safety Manager) should also be appointed. In larger companies this role is assigned to staff members who have the primary responsibility.

► Tasks:
- prepare specifications
- prepare safety concept and machine specifications
- prepare functional specifications
- determine or calculate the reliability of the safety function
- document all work steps
- archive the documentation and keep it during the life cycle of the machine

► The execution and verification of a task must not be performed by the same staff member!
- create and verify specifications
- create and verify machine specifications
- create and verify the different safety-related function units
- verify the interaction of the function units

The 4-eye principle is typically applied here.

To illustrate this structure in graphic form the V model can be applied. The machine manufacturer adds the corresponding details and responsibilities to each work step of a particular application. The machine manufacturer could also – depending on the work packages – create several organisational units based on the V model.

It is important that this organisational structure is documented and archived.

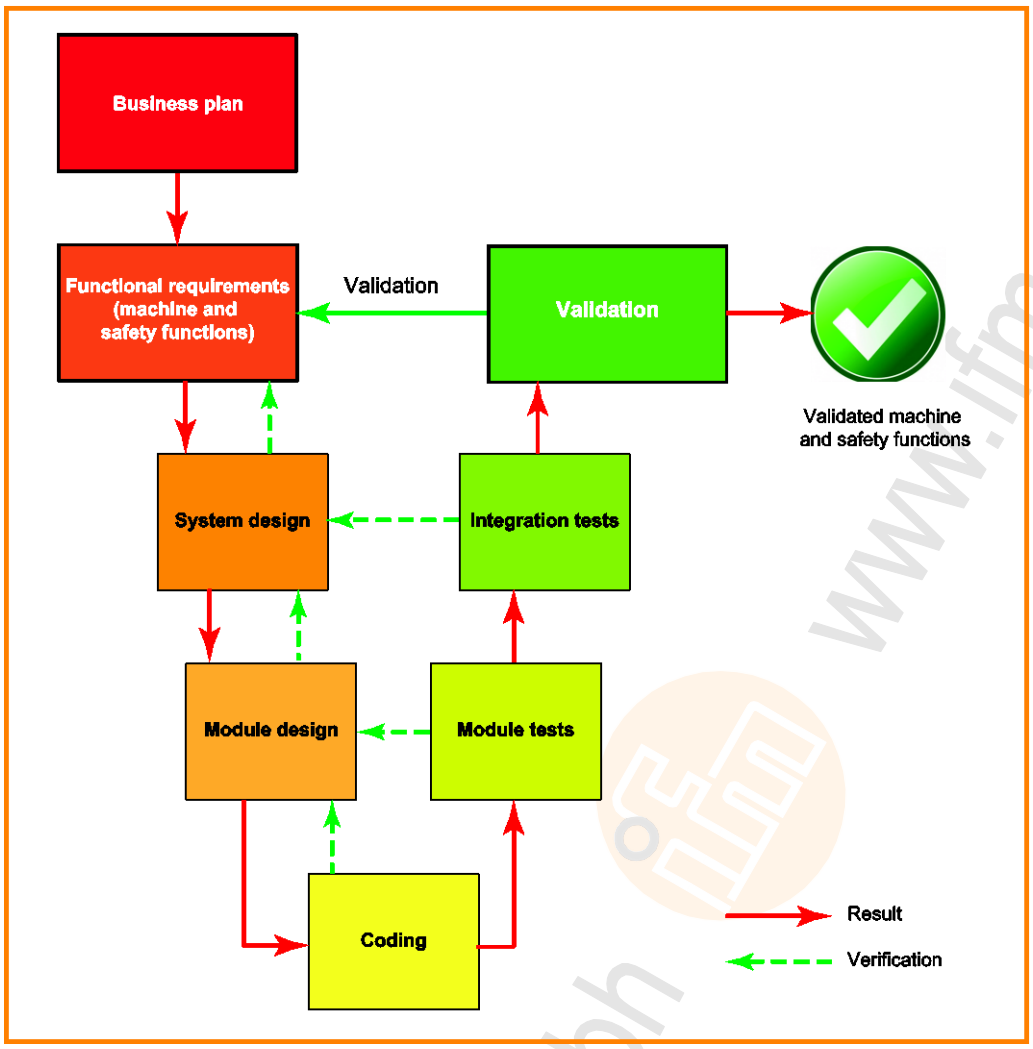


Figure: V model showing the individual work steps

- ► In accordance with the applicable directives and standards:
  - adhere to the mechanical and electrical layout of the safety functions when designing the mobile machine
  - document the respective steps of the specification and implementation process in a clearly structured way
  - make a statement regarding their reliability and the likelihood of a dangerous error
- ► Archive these documents during the entire life cycle of the mobile machine or machine series!

3.5 Start-up behaviour of the controller

15233
11575

⚠ WARNING

Danger due to unintentional and dangerous start of machine or plant sections!

► When creating the program, the programmer must ensure that no unintentional and dangerous start of machines or plant sections after a fault (e.g. e-stop) and the following fault elimination can occur!
⇨ Realise restart inhibit.

► In case of an error, set the outputs concerned to FALSE in the program!

A restart can, for example, be caused by:
- Voltage restoration after power failure
- Reset after the watchdog responded because the cycle time was too long
- Error elimination after an E-stop

To ensure safe controller behaviour:

► monitor the voltage supply in the application program.

► In case of an error switch off all relevant outputs in the application program.

► Additionally monitor actuators which can cause hazardous movements in the application program (feedback).

3.6 Warnings used

13685

⚠ WARNING

Death or serious irreversible injuries may result.

⚠ CAUTION

Slight reversible injuries may result.

NOTICE

Property damage is to be expected or may result.

Important note
Non-compliance may result in malfunction or interference.

Information
Supplementary note.

3.7 Notes: serial number

20780

- In the user's production facility, draw a diagram of the controller network in the machine. Enter the serial number of each controller installed into the network diagram.
- Before downloading a software component, read out this serial number and check the network diagram to make sure that you are accessing the right controller.

4 Installation

Contents

System requirements 18
Carry out installation 19

23075

This chapter describes the installation of the software components that are necessary to program the CR720S.

4.1 System requirements

Contents

Hardware 18
Software 18
Licensing 18

22911

Under which conditions can and may this device be programmed and operated?

4.1.1 Hardware

22912

- Device from the ifm product family ecomatController CR720S
- PC/laptop for CODESYS programming system
  (→ Chapter Software > System requirements CODESYS Development System V3.5)
- Ethernet connection between CODESYS PC/laptop and Ethernet interface of the CR720S
  (→ installation instructions)

4.1.2 Software

22913

To program the device-internal PLC of the CR720S, the following software components are required:

Component	of	Description	version
CODESYS Development System	3S	Programming software CODESYS for PLC programming complying with the standard IEC 61131-3	3.5 SP11
[CR720S]_V2.5.0.n.zip	ifm	Complete package for ecomatController CR720S, consisting of: → **Components of the complete package** (→ p. 20)	V2.5.0.n

The features and functions warranted in this manual can only be obtained by using the software components in the versions that are mentioned here.

On their website, ifm electronic provide the software components for download:
→ **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ p. 207)

4.1.3 Licensing

22914

By buying a controller CR720S, the buyer also purchases a licence that is valid for the use of the CODESYS 3.5 programming system.

4.2 Carry out installation

Contents

CODESYS programming software 19
Complete package for ecomatController CR720S 20
Check the operating system version of the device 22
Update the operating system of the device 24

23077

4.2.1 CODESYS programming software

7282

The CODESYS Development System (short: CODESYS) serves as platform for the creation of PLC applications according to the standard IEC 61131-3.

Install CODESYS Development System

23508

To install the software "CODESYS Development System":

For installation on the PC/laptop, administrator rights are required.

► Install the programming system CODESYS 3.5 SP11.
→ CODESYS installation and first steps

> CODESYS 3.5 SP11 is installed on the PC/laptop.

4.2.2 Complete package for ecomatController CR720S

23372

Components of the complete package

23393

To program the device-internal PLC, ifm provides a complete package "CODESYS for ifm R360III Products". The overall package is structured as follows and includes the following components:

Data name / path	Description
"CODESYS for ifm R360III Products".zip	Complete package
+ CODESYS_Package	Folder
+ PlugIn	Folder
+ CODESYS Safety SIL2 xyz.package	Package "CODESYS Safety SIL2 Plugin"
+ ifm Safety SIL2 Extensions Vn.n.n.n.package	Package "ifm Safety SIL2 Plugin"
+ ifm_ecomatController_Vn.n.n.n.package	Package "ecomatController (device description, libraries, etc.)"
+ Device	Folder
+ boot	Folder
+ boot.ifm	Bootloader
+ os	Folder
+ ifmOS.ifm	Runtime system

The following components must be installed:

- CODESYS Safety SIL2 Plugin
- ifm Safety SIL2 Plugin
- ecomatController package (device description, libraries, etc.)

Install package (PC/laptop)

23369

To install a package

Requirements

> CODESYS 3.5 SP11 is installed on the PC/laptop.
> ifm package "CODESYS for ifm R360III Products" is stored on the PC/laptop.

1 Start CODESYS

► Start CODESYS as administrator.
> CODESYS user interface appears.

2 Start Package Manager

► Select [Tools] > [Package Manager] to start the Package Manager.
> Package manager appears.
> Window shows installed packages.

3 Install package

► Click on [Install...].
> The file explorer appears.
► select the required file`*.package` and carry out a full installation.
> The [Package Manager] window shows the installed package.
► Click on [Close] to quit the Package Manager.
► Save the project.
► Close CoDeSys
► Start CODESYS
> The installed package is now available.

To install another package, proceed again as described.

Update package (PC/laptop)

23370

To update a package:

1 Uninstall the old version of the package

► **Uninstall package (PC/laptop)** (→ p. 22)

2 Install the new version of the package

► **Install package (PC/laptop)** (→ p. 21)

3 Update device

► In the device tree: Mark node [Device (CR720S)].
► Select [Project] > [Update Device...].
> Dialogue window appears.
► Click on [Update Device] to start the update process.
> CODESYS loads new device libraries.
> Device tree view is updated.
► Click on [Close] to quit the Package Manager.
► Save the project.

Uninstall package (PC/laptop)

23371

To uninstall a package:

1 Start package manager

► Select [Tools] > [Package Manager] to start the Package Manager.

> Window [Package Manager] shows installed packages.

2 Uninstall package

► Activate checkbox [Display versions].

> The window shows the version numbers of the installed packages.

► Select the package version to be uninstalled and uninstall it with [Uninstall...].

> Selected package version is uninstalled.

► Click on [Close] to quit the Package Manager.

4.2.3 Check the operating system version of the device

23560

Check the operating system version of the device

23590

To check the operating system version of the device:

► Connect to PLC (→ **Set communication path of PLC** (→ p. 62))

► Copy the file `\info\swinfo.txt` by clicking on [<<] to a local PC drive (→ **Manage files** (→ p. 67)).

► Open `swinfo.txt` in an editor, e.g. Notepad

> Content of the opened file (example):

```
[ifmOS]
Version=V1.4.0.3
BuildDate=21.10.2016 15:11:14

[Bootloader]
Version=1.2.3.4
BuildDate=01.01.2017 23:10:15
```

► Check the information in the [ifmOS] area behind [Version=]

> If the version deviates from the required version, the operating system must be updated.

Check the hardware version of the device

23587

To check the hardware version of the device:

- ► Connect to PLC (→ **Set communication path of PLC** (→ p. 62))
- ► Copy the fifle `\info\devinfo.txt` by clicking on [<<] to a local PC drive (→ **Manage files** (→ p. 67)).
- ► Open and check `devinfo.txt` in an editor, e.g. notepad

> `devinfo.txt` contains the following information:

Section	Key	Description
Hardware	Number	Hardware article number
HARDWARE	Name	Hardware article designation
MANUFACTURE	LPKnum	Production order number
MANUFACTURE	LPKerp	ERP material number
MANUFACTURE	Fabrnum	Production order number of the final unit

4.2.4 Update the operating system of the device

Contents

Update the operating system of the device with the batch file 24

23501

ATTENTION

Important: During the update process, the electric voltage supply of the device must be assured.

The device may only be disconnected from the voltage supply after the data transfer/update process is finished.

An incomplete operating system update may destroy the device!

23374

Important: Field test units cannot be updated!

► Before the update process, please ensure:
- Device is connected to the voltage supply and switched on
- The Ethernet interface is connected to the same network as the PC

Update the operating system of the device with the batch file

23825

ATTENTION

► Strictly follow the instructions on the screen during the entire update process!
> **Otherwise the controller may be destroyed!**

To update the operating system of the device with the batch file `update.bat`:

► Unpack the ZIP file with the batch file `update.bat` and the corresponding files in a local memory location.

Important: The data path in which the batch file is unpacked may not contain any blanks!

► Execute the batch file `update.bat`
► Follow the instructions on the screen.
> The [ecomatController Update Start Menu] appears.

If the device is not set to the standard IP address 192.168.82.247:

► Use the [i] key to select the menu item [Set device ip-address].
► Enter the IP address set in the device and confirm with [RETURN].
> The IP address setting in the batch program has been changed.

► Use the [P] key to call up the menu item [Ping device]

> The ping command is executed. The device must answer to the ping request to enable an update process.

If ping is successful:

► Use the [0] key to call up the menu item [Continue update process].

> Order number, software and hardware version are read from the device.

If an update is possible using the read data:

> The [ecomatController Update Menu] appears.

> Otherwise: Error message and return to the [ecomatController Update Start Menu].

► Follow the instructions on the screen.

> The updated process is executed.

ATTENTION

► After loading the first file `cmd.ifm` and after the instruction on the screen, execute a power-on reset of the controller.

> **Otherwise the controller may be destroyed!**

► After the power-on reset, continue the update process following the instructions on the screen.

> The user is informed about the success of the update process.

If the update process is finished successfully:

► Disconnect the voltage supply.

► Re-connect the voltage supply after the waiting time.

> The PLC boots with a new operating system.

5 System description

Contents

Hardware description ... 26
Interfaces ... 52
Software description ... 54

975

5.1 Hardware description

Contents

Hardware structure ... 26
Device supply (technology) ... 33
Monitoring concept ... 36
Inputs (technology) ... 38
Output types ... 44
List of outputs ... 49
Feedback in case of externally supplied outputs ... 51

14081

5.1.1 Hardware structure

Contents

Overview: Hardware ... 27
Note on wiring ... 29
Standard PLC and safety PLC ... 30
Available memory ... 31

15332

Overview: Hardware

22921

System context of the controller

22922
23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

All devices of this controller family can execute both security levels simultaneously:
• PLC for safety-related application (in the figure: yellow areas)
• PLC for standard applications (in the figure: blue areas)
Elements that can belong to both security levels are grey.

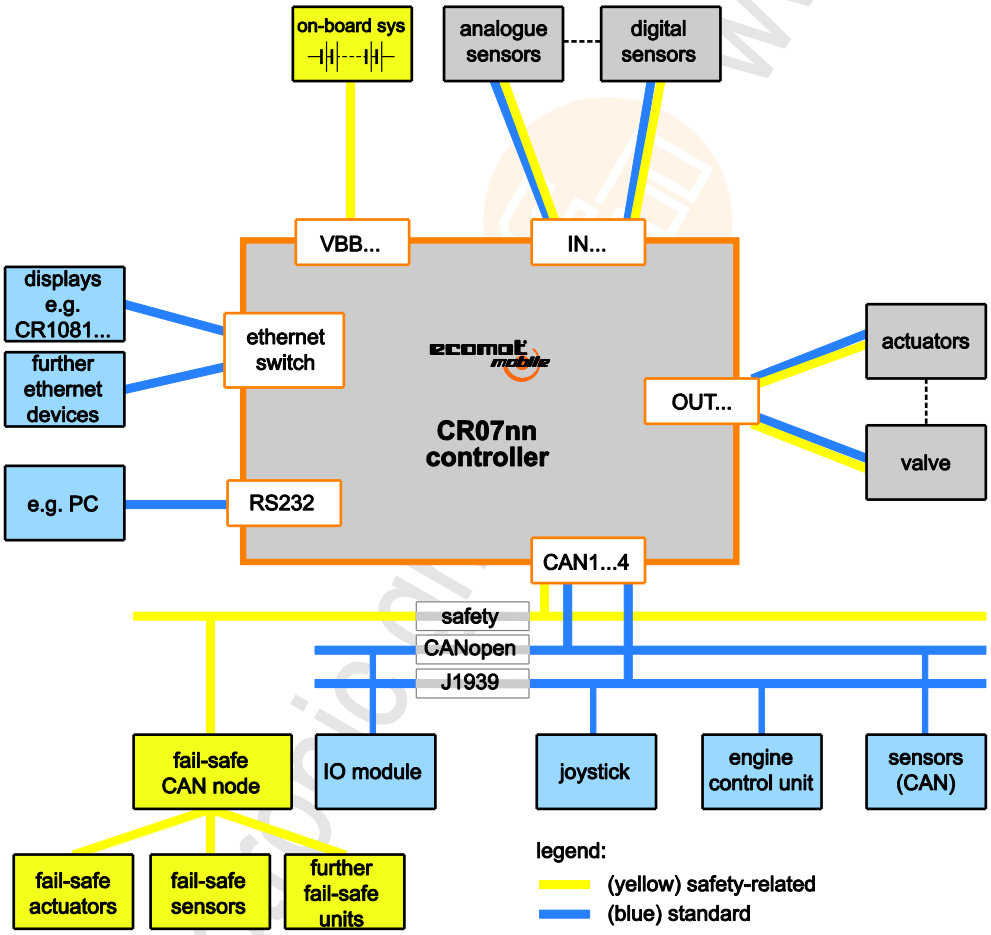
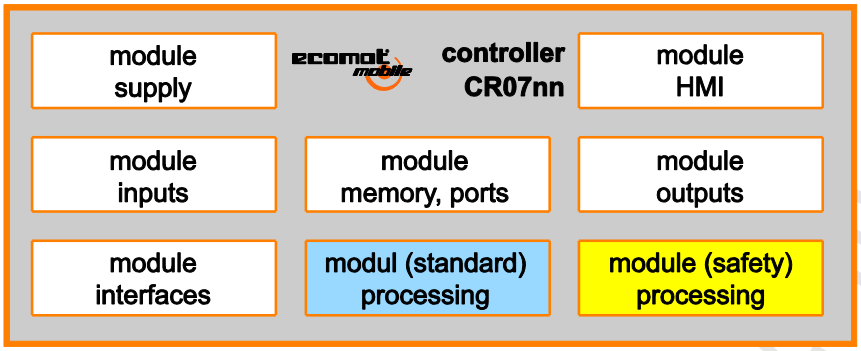


Figure: System context of the controller

System overview

23405



Overview of the system modules

Details:

Module	see
Power supply	**Device supply (technology)** (→ p. 33)
Inputs	**Inputs (technology)** (→ p. 38)
Interfaces	**Interfaces**
Memory, ports	**Available memory** (→ p. 31)
Processing	**Standard PLC and safety PLC**
HMI	**Status LEDs**
Outputs	**Outputs (technology)**

Block diagram of the supply and of the output deactivation

22936

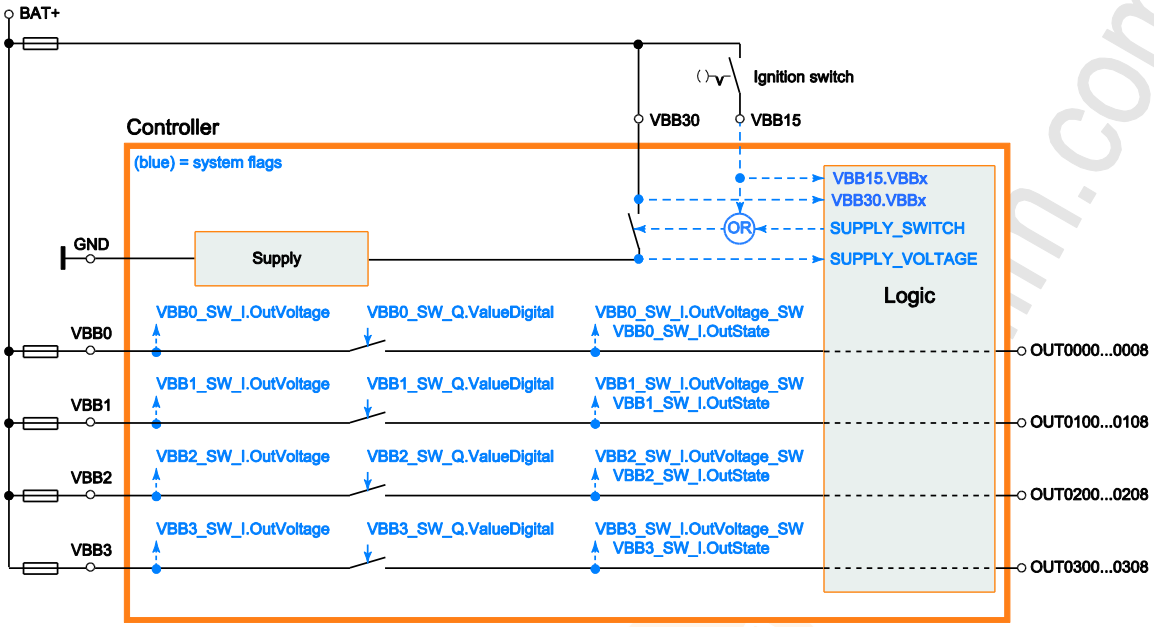


Figure: Block diagram of the supply

Note on wiring

23102

Group designations

23103

Inputs and outputs are assigned in groups.
Input groups are required for 2-channel safety-related inputs.

The identifier of an input or output results from the following principle:

1. Type (IN, OUT)
2. Group number (00...18)
3. Channel number (00...08)

Examples:

- IN0002 = input | group 00 | channel 02 in this group
- OUT0507 = output | group 05 | channel 07 in this group

In case of ready-to-use multipoint connectors, both the corresponding identifier and the corresponding pin number is indicated on each individual wire.

Standard PLC and safety PLC

22916
23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

The device features separate controllers:
- for standard functions
- for safety-related functions (safety)

Before the programming of the application may even begin:

► distribute the resources to both PLCs (Chapter **Configure PLC** (→ p. 65)).
(system, inputs, outputs, user LEDs)

Available memory

13736

Memory allocation

22928

IEC61131-1 divides the memory for the storage of the user data into:

- memory of the applications (parts are configurable)
  - IEC code non-safe
  - IEC code safe
- Application data memory (IEC data):
  - for volatile data (IEC RAM)
  - for non-volatile data (memory-remanent in case of voltage failure)

The device has the following additions:

- USER files (storage of application-specific data in one data format)
- IEC memory bytes (permanent storage of application-specific data at application-specific addresses)

23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

FLASH memory	RAM	remanent memory
physical: 9.0 Mbytes available: 6.0 Mbytes	physical: 2.7 Mbytes available: 1.5 Mbytes	physical: 10 Kbytes available: 10 Kbytes
IEC code (safe) (configurable)	IEC RAM (safe) (configurable)	IEC retain (safe) (2.5 Kbytes)
IEC code (non-safe) (configurable)	IEC RAM (non-safe) (configurable)	IEC memory bytes (safe) (2.5 Kbytes)
user files (1.0 Mbytes)		IEC retain (non-safe) (2.5 Kbytes)
		IEC memory bytes (non-safe) (2.5 Kbytes)

Table: memory areas

Memory allocation variants

22932
23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

The user can select from the pre-defined configurations of the memory partitioning. The configuration enables optimum separation between safety-relevant application and standard application.

Memory Configuration	IEC code safe	IEC RAM safe	IEC code non-safe	IEC RAM non-safe
Configuration 1	1.0 Mbytes	306 Kbytes	4.0 Mbytes	1228 Kbytes
Configuration 2 (preset)	2.0 Mbytes	614 Kbytes	3.0 Mbytes	920 Kbytes
Configuration 3	3.0 Mbytes	920 Kbytes	2.0 Mbytes	614 Kbytes
Configuration 4	4.0 Mbytes	1228 Kbytes	1.0 Mbytes	306 Kbytes

Table: configurable memory partitioning

Description of the allocation of the memory partitions in CODESYS: → ***Allocate memory partition*** (→ p. 65)

5.1.2 Device supply (technology)

Contents

Voltage ranges of the on-board system 33
Start conditions 33
Switch on/off via main switch 34
Switch on/off via ignition lock (Terminal 15) 35

23404

Voltage ranges of the on-board system

22926

The system monitors the voltage ranges of the on-board system.
The voltages mentioned here apply to the specified range ± 1 % (at 36 V).

Voltage [V]		Description
from	to	
	< +5.5	**Undervoltage** VBB15, VBB30: If the controller was in the OPERATING mode, the controller shuts down.
+5.5	< 8.0	**Limited operating range** If the controller was in the OPERATING mode, then it continues to operate in this range without any restrictions in case of voltage dips.
+8.0	+32.0	**Regular operating voltage** Nominal operating voltage all functions available VBB15 > 5 V AND VBB30 > 8 V: The controller is booting
> +32.0	+36.0	**Overvoltage (protected)** The device is not damaged by the voltage deviation. If this condition on VBB15 / VBB30 in the OPERATING mode lasts longer than 10 s, the controller changes to the FATAL ERROR state. If this condition on VBB0...n in the OPERATING mode lasts longer than 10 s, the corresponding output group changes to the COMPONENT ERROR state.
> +36.0		**Overvoltage (unprotected)** In this area, the device is no longer protected and the behaviour is not predictable. The device can be destroyed by this voltage. ► If such voltages are likely to occur in an application, provide for external protection!

Start conditions

22925

The device only boots when sufficient voltage is applied to the power supply connection VBB30 and to VBB15 (= terminal 15).
In vehicles clamp 15 is the plus cable switched by the ignition lock.

This voltage must be provided by the on-board system of the mobile machine.
→ chapter **Monitoring concept** (→ p. 36)

Switch on/off via main switch

22917

To do so: VBB15 is connected with VBB30

Procedure when switching on the main switch

> The system recognises the applied voltage (VBB15 > 5 V AND VBB30 > 8 V) and activates the connection of the controller to the VBB30 potential via solid-state switch.

> The controller boots and starts.

Procedure when switching off the main switch:

> Running tasks will continue till the end.
IMPORTANT: The maximum length of tasks is 50 ms!
Tasks that run for longer will be aborted before the end of the task.

> the system automatically stores the retain data
the input signals are no longer read
the outputs are switched off.

> the system switches off completely

If this behaviour is not wanted, use the circuit via ignition lock:

Switch on/off via ignition lock (Terminal 15)

22918

To do so:

► Connect the VBB15 via the ignition lock (= vehicle terminal 15 *) with the vehicle plus pole.

► Connect VBB30 directly with the vehicle plus pole (= vehicle terminal 30).

*) In vehicles clamp 15 is the plus cable switched by the ignition lock.

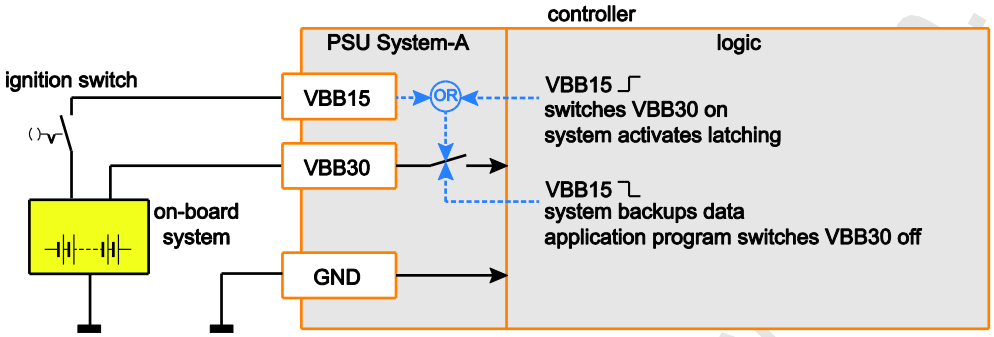


Figure: Delayed switch-off via ignition lock (terminal 15)

Procedure as with switch-on via ignition lock:

> The ignition lock applies voltage to VBB15 (= vehicle terminal 15 *).

> The system recognises the applied voltage (VBB15 > 5 V AND VBB30 > 8 V) and activates the connection of the controller to the VBB30 potential via solid-state switch.

> The ignition lock is bypassed. Latching of the control voltage is established.

> The controller boots and starts.

Procedure for switching off via ignition lock:

► Evaluate VBB15 in the application via FB **SystemSupply** (→ p. 136)
If VBB15 < 5 V:

- execute necessary actions (e.g.:)
  - stop machine gently
  - transmit required data
  - save required data and close
- set the the input `xSwitchOff` of the FB **SupplySwitch** (→ p. 134) to TRUE
- running tasks continue till the end.
- stop the application
- the system automatically stores the retain data
  the input signals are no longer read
  the outputs are switched off.
- the system lifts the latching via VBB30:
- the system switches off completely

5.1.3 Monitoring concept

22919

The controller monitors the supply voltage for overvoltage and undervoltage.
In case of undervoltage, the controller switches off.

Monitoring and securing mechanisms

22941

Switch off outputs via solid-state switch

23273

⚠ WARNING

Danger due to unintentional deactivation of all outputs!

If monitoring routines detect a system error:

> The device switches off the energy for all outputs of the affected output groups.
→ chapter **Output groups**
→ chapter **List of outputs** (→ p. 49)

During the program process, the output switches are under the user's full software control. For further safety, the corresponding applicable national regulations must be complied with.

If an error occurs during the program sequence, it is possible to disconnect the output switches from voltage via the FB **OutputGroup** in order to separate critical plant sections.

11575

⚠ WARNING

Danger due to unintentional and dangerous start of machine or plant sections!

- ► When creating the program, the programmer must ensure that no unintentional and dangerous start of machines or plant sections after a fault (e.g. e-stop) and the following fault elimination can occur!
  ⇨ Realise restart inhibit.
- ► In case of an error, set the outputs concerned to FALSE in the program!

Watchdog

23274

The watchdog has multiple levels:

- IEC task-related watchdog
  This watchdog works in the ifm operating system and is executed in each CPU core. Each task is monitored individually.
  If an error occurs, the system only deactivates the affected PLC and the corresponding outputs.
  Error class = B
- External watchdog
  If an error occurs, this watchdog puts the entire system into the "safe state" (emergency stop). The output groups change to logic "0".
  Error class = A

→ chapter **Error classes**

To eliminate the fault:

► Rebooting the PLC is necessary via voltage on/off.

Configure IEC watchdog

23564

► Familiarise yourself with the following CODESYS functions!
  - Watchdog:
    → Online help > CODESYS Development System > Programming Applications > Task Configuration > Creating a Task Configuration > Tab 'Configuration'
  - Task configuration:
    → Online help > CODESYS Development System > Programming Applications > Task Configuration

To configure the IEC watchdog of a task:

► Open task configuration (→ **Configure task processing (→ p. 81)**)
► Activate watchdog with option field [Enable]
► Enter watchdog [Time]
► Set [Sensitivity]
> Watchdog is configured

The watchdog time must be shorter than the interval time.
The watchdog time must be longer than the runtime of the task.

5.1.4 Inputs (technology)

Contents

Types of inputs 38
List of inputs 42

14090

Types of inputs

Contents

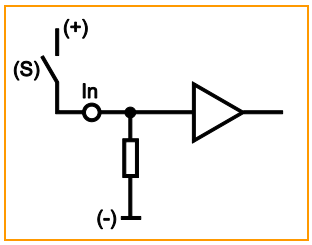
Binary input block diagram plus/minus-switching 38
Input type IN MULTIFUNCTION-A 39
Input type IN FREQUENCY-A/B 39
Input type IN RESISTOR-A 40
Input type IN DIGITAL-A 40
Input type IN DIGITAL-B 41

23078

We differentiate between the following input types:

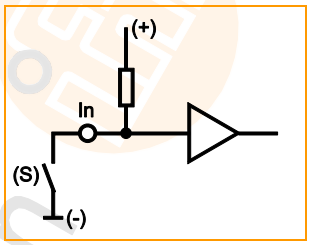
Binary input block diagram plus/minus-switching

23080



CSI = Current Sinking
In = connection of binary input n
(S) = sensor

Binary input block diagram, plus-switching (B_L)
for positive sensor signal
Input = open ⇨ Signal = Low (GND)



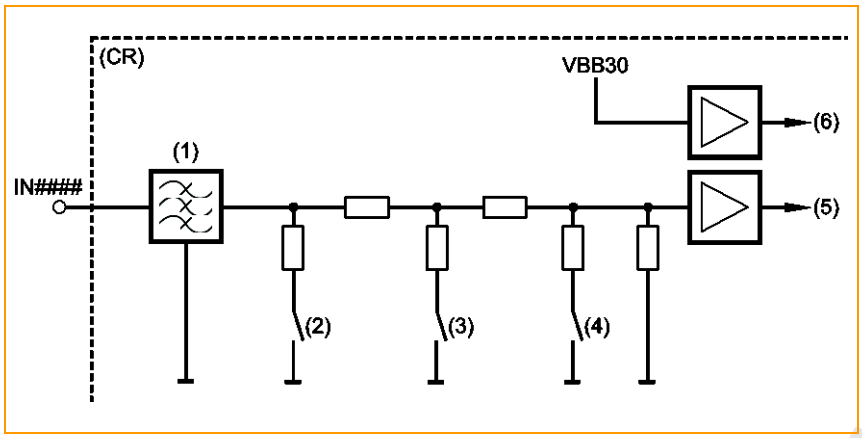
CSO = current sourcing
In = connection of binary input n
(S) = sensor

Binary input block diagram, minus-switching (B_H)
for negative sensor signal:
Input = open ⇨ Signal = High (Supply)

Input type IN MULTIFUNCTION-A

23081

Binary and analogue inputs



Block diagram
IN MULTIFUNCTION-A

IN#### =
Connection of multi-function input no. ####
(CR) = device
(1) = input filter
(2) = analogue current measurement
R_E = 300 Ω
(3) = binary input, plus-switching
R_E = 10 kΩ
(4) = analogue voltage measurement
0...10 V / 0...32 V
(5) = measured value
(6) = reference voltage for ratio

Configure input → Chapter System configuration

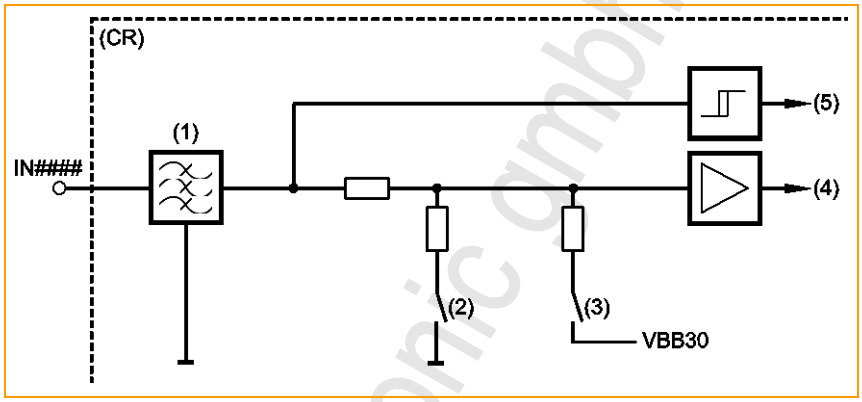
Possible operating modes:

- Binary input CSI (B_L) (R_E = 10 kΩ) or Namur
- Analogue current measurement 0...20 mA
- Analogue voltage measurement 0...10 V
- Analogue voltage measurement 0...32 V
- Analogue voltage measurement, ratiometric to the reference voltage

Input type IN FREQUENCY-A/B

23083

Binary and fast inputs



Block diagram
IN FREQUENCY-A/B

IN#### =
Connection of frequency / counting input no. ####
(CR) = device
(1) = input filter
(2) = CSI binary input
plus-switching
(3) = CSO binary input
minus-switching
(4) = measured value (analogue)
(5) = measured value (binary)

Configure input → Chapter System configuration

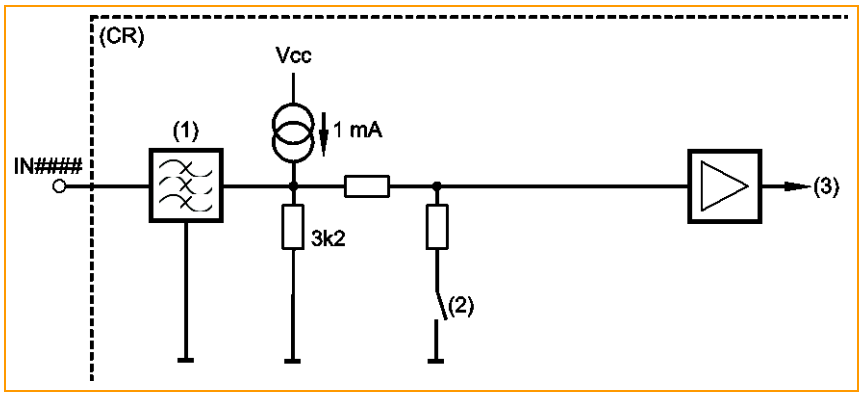
Possible operating modes:

- Binary input CSI (B_L) (R_E = 10 kΩ) or Namur
- Binary input CSO (B_H)
- Analogue voltage measurement 0...10 V (only for input type IN FREQUENCY-B)
- Pulse measurement CSI (B_L) (frequency measurement, ratio measurement, pulse counter)
- Pulse measurement CSO (B_H) (frequency measurement, ratio measurement, pulse counter)

Input type IN RESISTOR-A

23083

Binary inputs and resistance measurement



Block diagram IN RESISTOR-A

IN#### = Connection of frequency / counting input no. ####

(CR) = device

(1) = input filter

(2) = CSI binary input plus-switching

(3) = measured value (analogue)

Configure input → Chapter System configuration

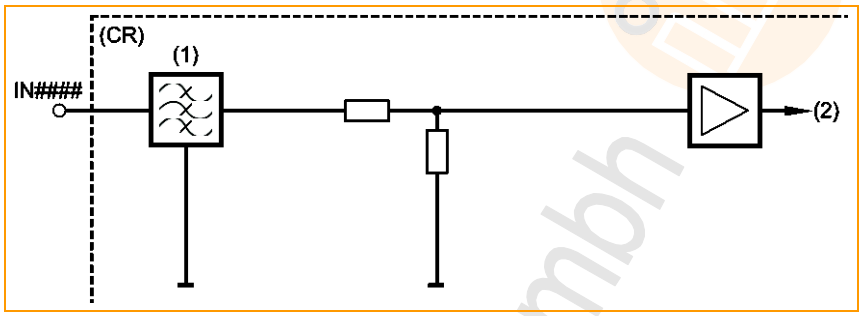
Possible operating modes:

- Binary input CSI (B_L) (R_E = 10 kΩ) or Namur
- Resistance measurement 0...30 kΩ

Input type IN DIGITAL-A

23087

Binary inputs



Block diagram IN DIGITAL-A

IN#### = Connection of binary input no. ####

(CR) = device

(1) = input filter

(2) = measured value (analogue)

Input not configurable

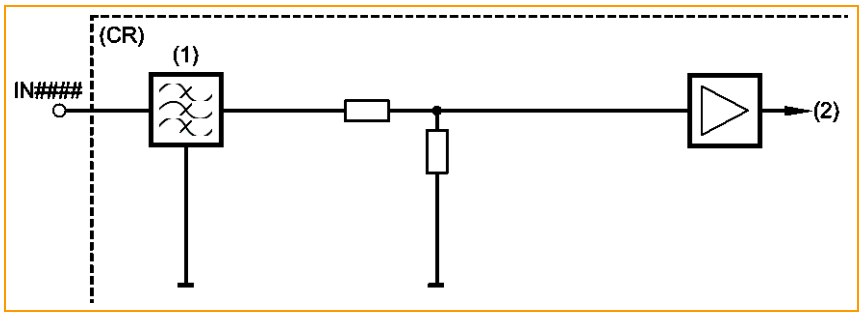
Possible operating modes:

- Binary input CSI (B_L) (R_E = 10 kΩ) or Namur

Input type IN DIGITAL-B

23089

Binary inputs



Block diagram
IN DIGITAL-B
IN#### = Connection of binary input no. ####
(CR) = device
(1) = input filter
(2) = measured value (analogue)

Input not configurable

Possible operating modes:

- Binary input CSI (B_L) (R_E = 3.3 kΩ) or Namur

List of inputs

23123

IEC identifier	Input type
IN0000	IN Frequency-A
IN0001	IN Frequency-A
IN0002	IN Frequency-A
IN0003	IN Frequency-A
IN0100	IN Multifunction-A
IN0101	IN Multifunction-A
IN0102	IN Multifunction-A
IN0103	IN Multifunction-A
IN0200	IN Multifunction-A
IN0201	IN Multifunction-A
IN0202	IN Multifunction-A
IN0203	IN Multifunction-A
IN0300	IN Digital-B
IN0301	IN Digital-B
IN0400	IN Resistor-A
IN0401	IN Resistor-A
IN0500	IN Frequency-A
IN0501	IN Frequency-A
IN0502	IN Frequency-A
IN0503	IN Frequency-A
IN0600	IN Multifunction-A
IN0601	IN Multifunction-A
IN0602	IN Multifunction-A
IN0603	IN Multifunction-A
IN0700	IN Multifunction-A
IN0701	IN Multifunction-A
IN0702	IN Multifunction-A
IN0703	IN Multifunction-A
IN0800	IN Digital-B
IN0801	IN Digital-B
IN0900	IN Resistor-A
IN0901	IN Resistor-A
IN1000	IN Frequency-A
IN1001	IN Frequency-A
IN1002	IN Frequency-A
IN1003	IN Frequency-A
IN1100	IN Multifunction-A

IEC identifier	Input type
IN1101	IN Multifunction-A
IN1102	IN Multifunction-A
IN1103	IN Multifunction-A
IN1200	IN Digital-A
IN1201	IN Digital-A
IN1202	IN Digital-A
IN1203	IN Digital-A
IN1300	IN Digital-B
IN1301	IN Digital-B
IN1302	IN Digital-B
IN1303	IN Digital-B
IN1400	IN Digital-A
IN1401	IN Digital-A
IN1402	IN Digital-A
IN1403	IN Digital-A
IN1500	IN Frequency-A
IN1501	IN Frequency-A
IN1502	IN Frequency-A
IN1503	IN Frequency-A
IN1600	IN Multifunction-A
IN1601	IN Multifunction-A
IN1602	IN Multifunction-A
IN1603	IN Multifunction-A

5.1.5 Output types

Contents

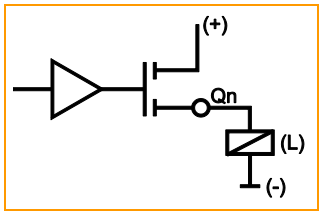
Binary output block diagram plus/minus-switching 44
Output type OUT PWM-n-A..... 45
Output type OUT PWM-n-B..... 46
Output type OUT PWM-n-BRIDGE-A..... 47
Output type OUT Supply-A..... 48
Output type OUT Voltage-A 49

23079

We differentiate between the following output types:

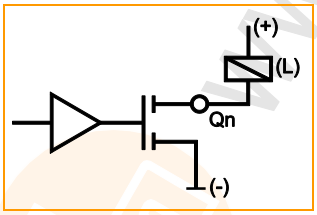
Binary output block diagram plus/minus-switching

23090



CSO = current sourcing
Qn = connection of output n
(L) = load

Output block diagram plus-switching (B_H)
for positive output signal



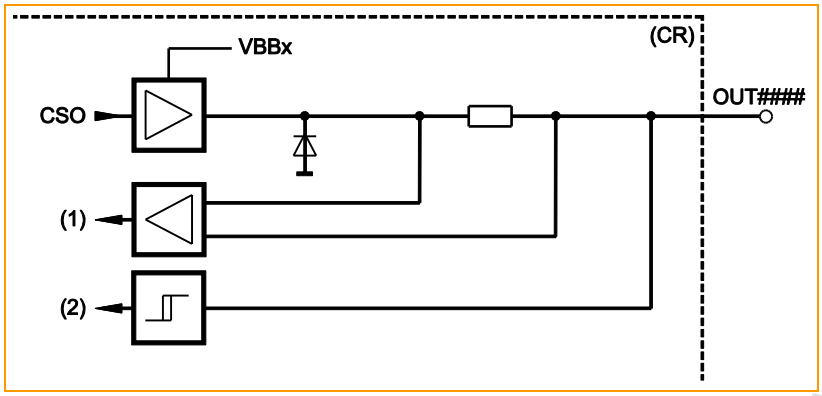
CSI = current sinking
Qn = connection of output n
(L) = load

Output block diagram, minus-switching (B_L)
for negative output signal

Output type OUT PWM-n-A

23091

n = current rating
Example: n = 25 ⇨ I_{max} = 2.5 A
Binary output or
analogue output with pulse width modulation (PWM), optionally current-controlled (PWM_I)



Block diagram
OUT PWM-n-A
OUT#### = connection of PWM output no. ####
(CR) = device
(1) = measured value (analogue)
(2) = measured value (binary)

Configure input → Chapter System configuration

Possible operating modes:

- Binary output CSO (B_H)
- analogue output CSO with pulse width modulation (PWM_H)
- analogue output CSO with pulse width modulation, current-controlled (PWM_I)

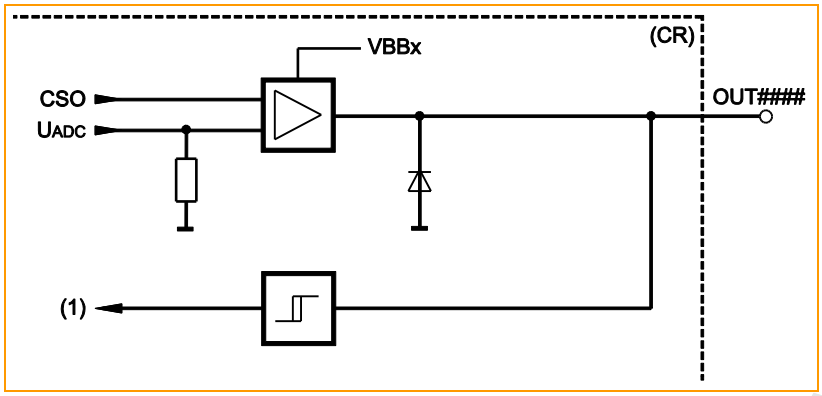
Setting and measurement via:

- FB **Output** (→ p. 129)for binary output
- FB **PWM1000** (→ p. 157) for PWM
- FB **CurrentControl** (→ p. 154) for current control (PWM_I)

Output type OUT PWM-n-B

23093

n = current rating
Example: n = 25 ⇨ I_{max} = 2.5 A
Binary output or
analogue output with pulse width modulation (PWM)



Block diagram
OUT PWM-n-B
OUT#### = connection of PWM output no. ####
(CR) = device
(1) = measured value (binary)
U_{ADC} = current mirror

Configure input → Chapter System configuration

Possible operating modes:

- Binary output CSO (B_H) with restricted current measurement
- analogue output CSO witch pulse width modulation (PWM_H), without current measurement

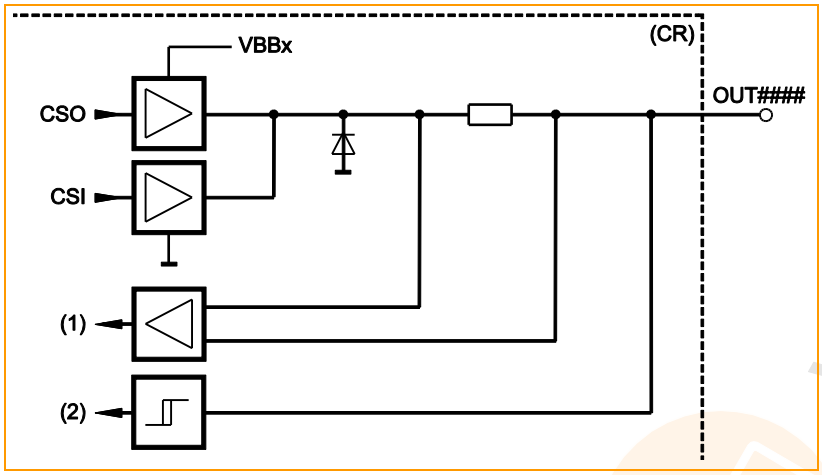
Setting and measurement via:

- Function block **Output** (→ p. 129) for binary output
- Function block **PWM1000** (→ p. 157) for PWM

Output type OUT PWM-n-BRIDGE-A

23094

n = current rating
Example: n = 25 ⇨ I_{max} = 2.5 A
Binary output or
analogue output with pulse width modulation (PWM), optionally current-controlled (PWM_I)
or bridge output (via PWM)



Block diagram
OUT PWM-n-A
OUT#### = connection of PWM output no. ####
(CR) = device
(1) = measured value (analogue)
(2) = measured value (binary)

Configure input → Chapter System configuration

Possible operating modes:

- Binary output CSO (B_H)
- Binary output CSI (B_L)
- analogue output CSO with pulse width modulation (PWM_H), current controlled (PWM_I)
- analogue output CSI with pulse width modulation (PWM_L)
- analogue output CSO with pulse width modulation, current-controlled (PWM_I)
- Pair of outputs as bridge with pulse width modulation (PWM)

Setting and measurement via:

- FB **Output** (→ p. 129) for binary output
- FB **PWM1000** (→ p. 157) for PWM
- FB **CurrentControl** (→ p. 154) for current control (PWM_I)
- FB **HBridge** (→ p. 149) for bridge output

Output type OUT Supply-A

23125

The output `OUT3000` is used to supply sensors with a stable voltage (5 V or 10 V) that is not affected by fluctuations of the supply voltage.

13402

NOTICE

Reference voltage output can get damaged!

► Do NOT apply any external voltage!

Setting and measurement via FB **Output** (→ p. 129) or via system configuration:

Setting / measurement via system configuration

22942

Setting the reference voltage:

► In the device tree, select [Local_IO] > [Outputs] > tab [Parameter] > [OUT3000]

► Activate the required list element in the column [Value]:
for 5 V: [`OUT_SENSOR_05`] or
for 10 V: [`OUT_SENSOR_10`]

Monitoring of the values at the reference voltage output:

► In the device tree, select [Local_IO] > [Outputs] > tab [IO-Mapping] > [OUT3000_I]

> [OutVoltageDiag] indicates the measured voltage in [mV]
[OutCurrentDiag] indicates the measured voltage in [mA]

Setting /measurement via FB Output

23422

Setting the reference voltage:

► Use the inputs in the FB output as follows:
[uiChannel] = 3000
[eMode] = [`OUT_SENSOR_05`] (for 5 V) or
[eMode] = [`OUT_SENSOR_10`] (for 10 V)

Monitoring of the values at the reference voltage output:

► Read the outputs in the FB output as follows:

> [uiOutVoltage] indicates the measured voltage in [mV]
[uiOutCurrent] indicates the measured current in [mA]

Details → FB **Output** (→ p. 129).

Output type OUT Voltage-A

23126

The output provides 0...10 V e.g. for further controllers or actuators.
M3071n / CR071n: only `OUT3001`
M3072n / CR072n: `OUT3001` and `OUT3002`

The output is protected against overload and automatically switches off if overloaded.

13402

NOTICE

Reference voltage output can get damaged!

► Do NOT apply any external voltage!

Setting and measurement via FB Output or via system configuration:

Setting / measurement via system configuration

23424

Setting the reference voltage:

► Select [Local_IO] > [Outputs] > tab [Parameter] > [OUT3001 / OUT3002] in the device tree

► Enter the required value [V] in the column [Value]
permissible = 0...10

Monitoring of the values at the reference voltage output:

► Select [Local_IO] > [Outputs] > tab [IO-Mapping] > [OUT3001_I / OUT3002_I] in the device tree

> [OutVoltageDiag] indicates the measured voltage in [mV]

Setting /measurement via FB Output

23425

Setting the reference voltage:

► Use the inputs in the FB output as follows:
[uiChannel] = 3001 / 3002
[uiValue] = required voltage in [mV]
permissible = 0...10000

Monitoring of the values at the reference voltage output:

► Read the outputs in the FB output as follows:

> [uiOutVoltage] indicates the measured voltage in [mV]
[uiOutCurrent] indicates the measured current in [mA]

Details → FB **Output** (→ p. 129).

List of outputs

23124

IEC identifier	Output type
OUT0000	OUT PWM-25-A
OUT0001	OUT PWM-25-B
OUT0002	OUT PWM-25-A
OUT0003	OUT PWM-25-B
OUT0004	OUT PWM-25-A
OUT0005	OUT PWM-25-B
OUT0006	OUT PWM-40-Bridge-A
OUT0007	OUT PWM-40-Bridge-A
OUT0008	OUT PWM-40-A
OUT0100	OUT PWM-25-A
OUT0101	OUT PWM-25-B
OUT0102	OUT PWM-25-A
OUT0103	OUT PWM-25-B
OUT0104	OUT PWM-25-A
OUT0105	OUT PWM-25-B
OUT0106	OUT PWM-40-Bridge-A
OUT0107	OUT PWM-40-Bridge-A
OUT0108	OUT PWM-40-A
OUT0200	OUT PWM-25-A
OUT0201	OUT PWM-25-B
OUT0202	OUT PWM-25-A
OUT0203	OUT PWM-25-B
OUT0204	OUT PWM-25-A
OUT0205	OUT PWM-25-B
OUT0206	OUT PWM-40-Bridge-A
OUT0207	OUT PWM-40-Bridge-A
OUT0208	OUT PWM-40-A
OUT0300	OUT PWM-25-A
OUT0301	OUT PWM-25-B
OUT0302	OUT PWM-25-A
OUT0303	OUT PWM-25-B
OUT0304	OUT PWM-25-A
OUT0305	OUT PWM-25-B
OUT0306	OUT PWM-40-Bridge-A
OUT0307	OUT PWM-40-Bridge-A
OUT0308	OUT PWM-40-A
OUT3000	OUT Supply-A
OUT3001	OUT Voltage-A
OUT3002	OUT Voltage-A

5.1.6 Feedback in case of externally supplied outputs

23874

Do not apply any external voltage to the outputs!

> As soon as output group switch VBBn_SW_Q = FALSE:
The internal device monitoring checks the voltage on the contact bar after the output group switch.
If then a voltage of > 0.4 VBBn is measured:
- the controller reports error class C,
- the controller switches the group to the safe state.

Safe state of the group = all outputs are switched off
- all outputs will be switched off
- the controller reports the error to the IEC application

To reboot the device:

► remove the error cause

► do a power-on reset.

OR error handling in the IEC application:

► remove the error cause

► Remove error of the group via xResetError

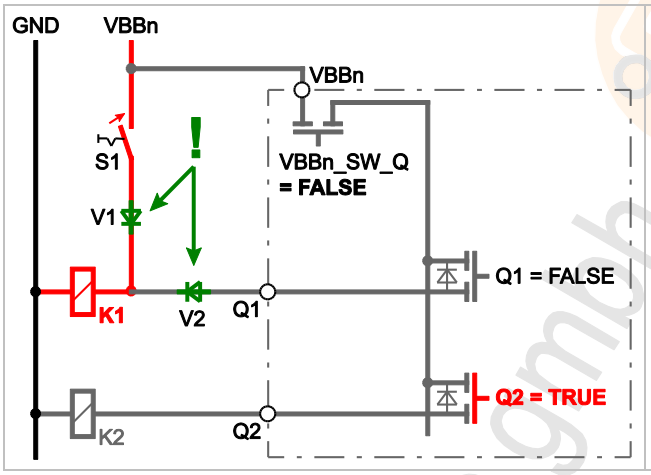


Figure: Example of wiring with blocking diodes due to the risk of feedback

Remedy:
Insert the blocking diodes V1 and V2 (→ green arrows)!

Successful:
If VBBn_SW_Q = FALSE, the controller does not go to error class C when the S1 contact is closed.

NOTE

Help for externally connected outputs

► Decouple the externally connected outputs by means of diodes so that no external voltage can be connected to the output terminal of the controller!

5.2 Interfaces

23132

The device includes the interfaces described in the following.

Position of the connections on the device and technical data: → Installation instructions, data sheet

5.2.1 Serial interface

23133

This device features a serial interface.

The serial interface can generally be used in combination with the following functions:
- program download
- debugging

Connections and data → data sheet

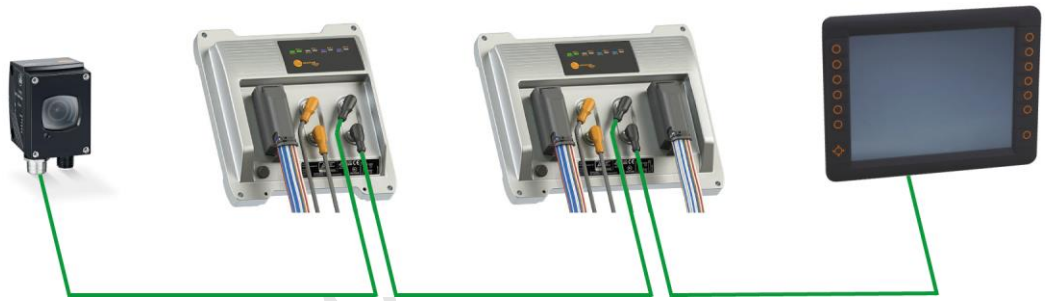
5.2.2 Ethernet interface

23134
23450

NOTICE

Only use the Ethernet interface in a secure network environment (e.g. separate network or VPN)!
Otherwise, unauthorised persons can read or manipulate data or tamper with the functions of the device.

This device features an Ethernet interface with 2 ports via an internal switch.
This enables line wiring between several devices.



The Ethernet interface supports the following standards:
- transmission rate 10/100 Mbits/s

The Ethernet interface supports the following protocols:
- TCP/IP
- UDP/IP
- Modbus TCP slave
- Modbus/TCP master
- network variables UDP

Connections and data → data sheet

5.2.3 CAN: Interfaces and protocols

23147

► Familiarise yourself with the following CODESYS functions!
- CAN-based fieldbuses
  → Online help > Fieldbus support > CAN-based fieldbuses

The device has 4 CAN interfaces. Each CAN interface supports the following protocols:
- RawCAN (CAN Layer 2)
- CANopen Manager
- CANopen Device
- CANopen Safety Manager
- CANopen Safety Device
- J1939 Manager

23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

5.3 Software description

Contents

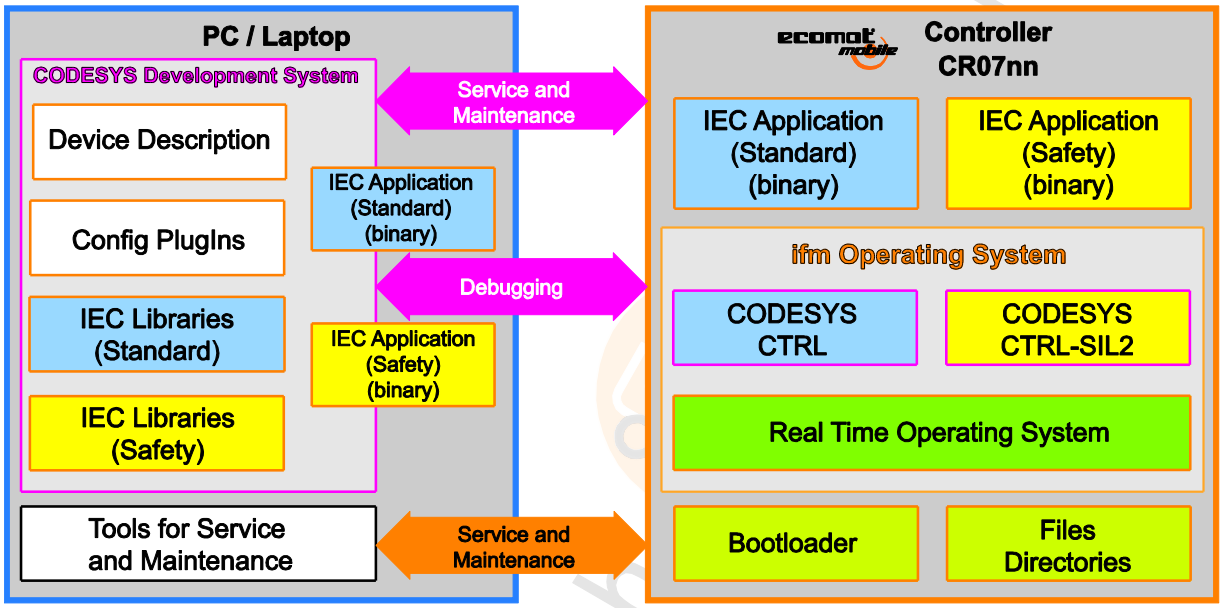
Overview: Software 54
Software module for the device 56

23148

5.3.1 Overview: Software

23511

We differentiate between the following software components:



Software on the PC/notebook.

23518

The programming environment CODESYS Development System is installed in the PC/notebook to create and debug both applications. The controller supports service and maintenance via CODESYS or via other tools.

The CODESYS functions are extended with Config plugins. Thereby, additional setting options for memory and inputs/outputs become available. **ifm electronic** provides adequate device descriptions for the CODESYS Development System for each derivative. IEC libraries for the safe and non-safe applications provide CODESYS and the programmer with access options to the functions of the controller.

Software in the controller

23519

The controller processes the applications by means of several software components.

The ifm operating system with the CODESYS CTRL-SIL2 and the CODESYS CTRL constitutes, among other things, the runtime environment that executes both applications. The real-time operating system enables separate execution of the safe and the non-safe software components in the controller.

23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

Using the configuration file `comconf.cfg`, the programmer can control the interface.

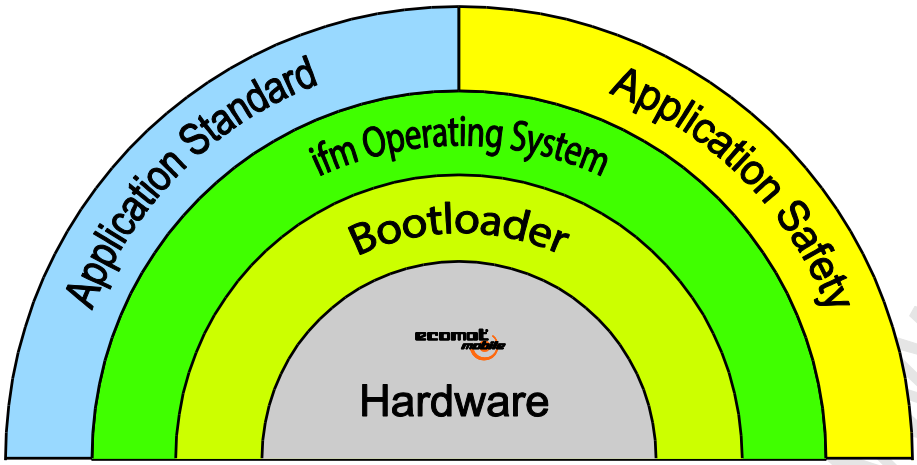
The programmer can store files and directories on the controller and use them in the application. Or the applications themselves create files and store them in the controller.

The bootloader is a fallback level for cases in which the ifm operating system is no (longer) available or corrupt.

5.3.2 Software module for the device

23500

The software in this device communicates with the hardware as follows:



Software module	Can the user change the module?	Using what?
Application with libraries a) for standard PLC b) for safety PLC	yes	CODESYS, Tools for service and maintenance
ifm operating system	Upgrade yes Downgrade no	CODESYS, Tools for service and maintenance
Bootloader	no	---
(Hardware)	no	---

We describe this software module in the following:

Bootloader

23503

The bootloader is a start program with which the operating system and the application can be reloaded on the device.

23561

Only execute the bootloader update when explicitly requested by ifm!

Operating system

23504

Basic program in the device, establishes the connection between the hardware of the device and the application.
→ Chapter **Software module for the device**

The device is supplied with the installed operating system.
Verifying and changing the operating system version → Chapter **Check the operating system version of the device** (→ p. 22)

The operating system only needs to be downloaded once - if at all. The application can then be loaded (also several times) in a PLC without affecting the operating system.

The operating system can be downloaded from **ifm electronic gmbh**'s website:
→ **ifm weltweit • ifm worldwide • ifm à l'échelle internationale** (→ p. 207)

Application

23505

Software specific to the application, implemented by the machine manufacturer, generally containing logic sequences, limits and expressions that control the appropriate inputs, outputs, calculations and decisions.

8340

⚠ WARNING

The user is responsible for the reliable function of the application programs he designed. If necessary, he must additionally carry out an approval test by corresponding supervisory and test organisations according to the national regulations.

Libraries

23505
23458

ifm electronic provides the following function libraries for the programming of the device under CODESYS 3.5:

Name	Description
ifmCANopenManager	Functions for use of the CAN interfaces as CANopen Manager
ifmDeviceCR720S	Data structures, enumeration types and global variables
ifmFastInput	Functions to access the fast inputs of the device
ifmIOcommon	Functions for access to the inputs and outputs of the device
ifmIOconfigDiagProt	Functions to configure the I/O-related diagnostic and protective functions
ifmOutGroup	Functions to control output group switches
ifmOutHBridge	Functions to access H-bridge outputs
ifmOutPWM	Functions to access PWM outputs
ifmRawCAN	Functions for use of the CAN interfaces as CAN Layer 2
ifmSysInfo	Functions to set / read system information
ifmTypes	Global types and interfaces for other ifm libraries

Detailed information about the **ifm** function libraries: → **ifm function libraries** (→ p. 103)

6 Getting started

Contents

Start CODESYS 58
Create CODESYS project 58
Use CODESYS user manual 61
Configure programming interface 62
Add ifm function libraries to the application 63
Activate the access protection for a project 64

15858

This chapter contains information about the first steps to program the device with CODESYS.

6.1 Start CODESYS

23383

Requirements

> Software components are correctly installed (→ **Installation**).

Start CODESYS

► Double-click on [3.5 SP11] symbol

> CODESYS starts.

> CODESYS user interface appears.

6.2 Create CODESYS project

23384

► Familiarise yourself with the following CODESYS functions!

- Create a project
  → Online help > CODESYS Development System > Creating and Configuring a Project
- Manage a project
  → Online help > CODESYS Development System > Protecting and Saving the Project

ifm electronic provides a special template for each model of the device family. The user can select the corresponding template when the project is created.

6.2.1 Create new project with CR720S

23452

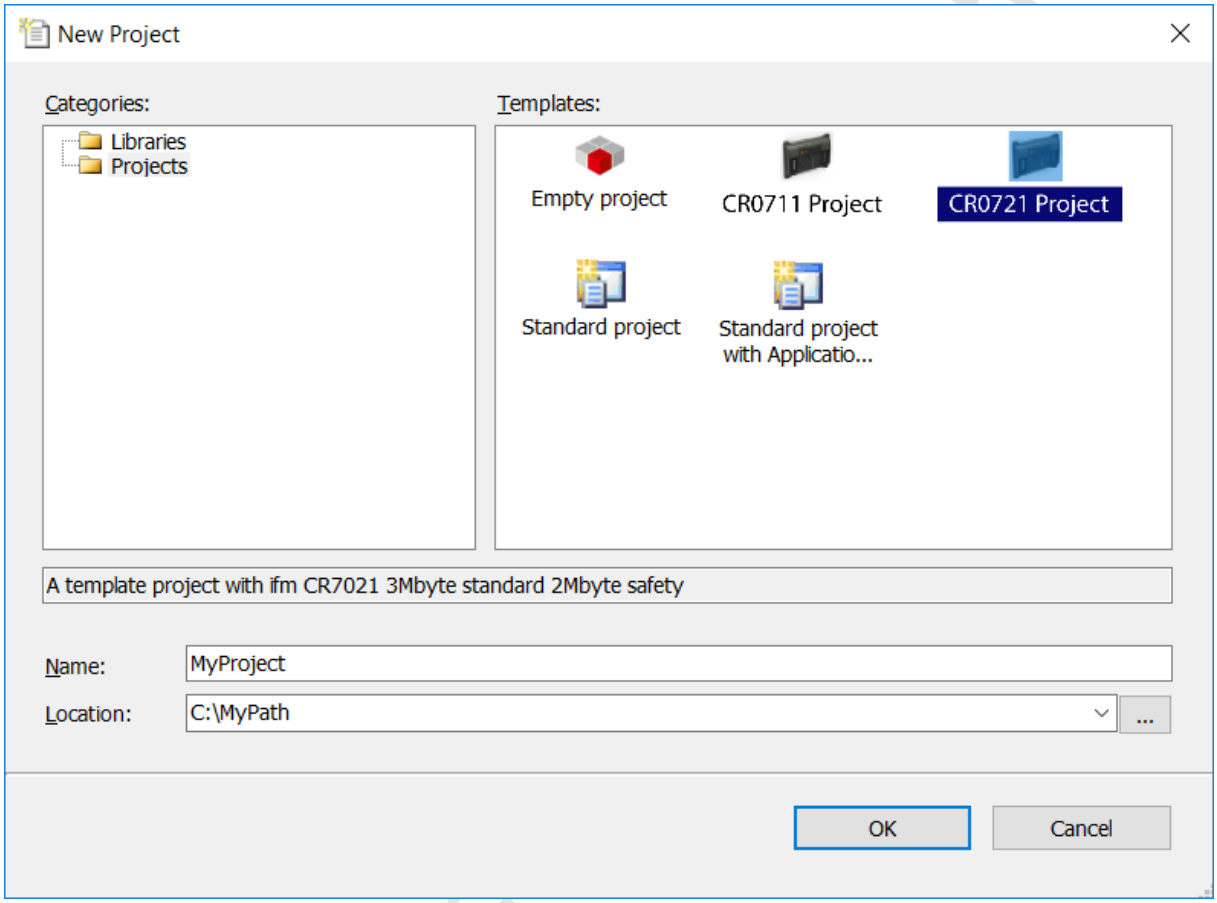
Requirements

> ifm package "CODESYS for ifm R360III Products" has been correctly installed (→ **Installation**).

1 Create new CR720S project

► Select [File] > [New Project...].

> The window [New Project] appears.



► Set the following values:
1. [Templates]: Select the device project template, e.g. [CR0721 Project]
2. [Name]: Enter the project name
3. [Location]: Select storage location for the project file

► Click on [OK] to adopt the selected values.

> CODESYS creates a new CR720S project.

> The window [Devices] shows the device tree of the project (→ **Overview: Project structure with CR720S**).

2 Save the project

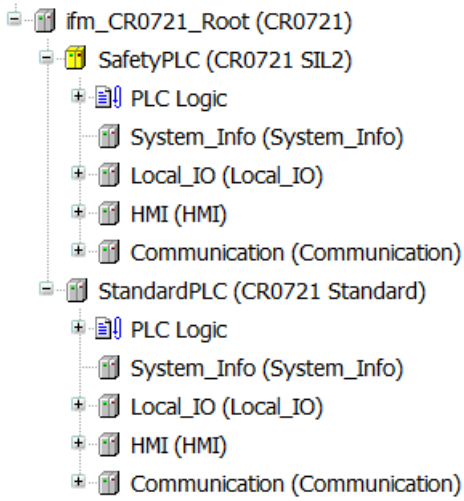
► Select [File] > [Save Project].

> CODESYS saves the project.

6.2.2 Overview: Project structure with CR720S

23387

A CODESYS project contains all components for configuration, management and programming of the CR720S. All components of a project are shown in the window [Devices] in a hierarchic tree view. CODESYS projects with a CR720S have the following structure:



Legend:

ifm_CR720S_Root (ifm CR720S Root)	provides access to the settings of the CR720S → **Configure PLC**
SafetyPLC (ifm CR720S SIL2)	Content of the PLC called "SafetyPLC"
StandardPLC (ifm CR720S Standard)	Content of the PLC called "StandardPLC"
PLC logic	contains the applications of the CR720S → **Objects of a PLC application** (→ p. 78)
System info	provides access to the device information → **Display system information** (→ p. 102)
Local_IO	provides access to the configuration options of the inputs and outputs → **Configure inputs and outputs** (→ p. 68)
HMI	provides access to the configuration options of the operating and display elements
Communication	provides access to the configuration options of the communication interfaces → **Configure CAN interfaces** (→ p. 71)

The programmer can adjust the terms in the structure before the expression in brackets:

► Right mouse click on the term > [Properties...]
> The window [Properties] appears > tab [Common]
► enter a term
► confirm with [OK]

6.3 Use CODESYS user manual

15221

This manual only describes the integration, configuration and the programming of the CR720S using the CODESYS development system.

For the description of user actions and user interface elements the CODESYS terminology will be used.

Standard functions and methods of CODESYS will not be described. At the beginning of each section there will be a reference to the corresponding chapters of the CODESYS online help.

To access the online help of the CODESYS development system:

► Start CODESYS.

> The CODESYS user interface appears.

► Press [F1].

> Online help of the CODESYS development system appears.

► Familiarise yourself with the CODESYS development system! In particular with the following topics:
- Names and functions of the user interface elements
- Basic menu functions
- Programming techniques and methods for data retention

6.4 Configure programming interface

23495

Programming of the device-internal PLC is made via the Ethernet interface of the device (position of the connections: → Installation instructions).

Device and PC/laptop can be coupled directly or indirectly via an Ethernet network.

- ► Only use the recommended accessories for connection of the Ethernet interfaces! (→ Installation instructions).
- ► For connection in the network, an experienced user or system administrator should set up the network addresses and do the configuration.

23450

NOTICE

Only use the Ethernet interface in a secure network environment (e.g. separate network or VPN)!

Otherwise, unauthorised persons can read or manipulate data or tamper with the functions of the device.

6.4.1 Set communication path of PLC

13901

To configure the communication path between the programming system CODESYS and the device-internal PLC:

Preparations

> CODESYS PC/laptop and Ethernet interface of the device are connected.

> Optional: Adjust IP settings of the Ethernet interface.

1 Select communication settings

- ► In the device tree: Double-click on symbol [Device (CR720S)]

> In the editor window: Select tab [Communication].

> Editor window shows communication settings.

2 Select gateway

- ► Select the requested gateway in the list [Gateway].

> List shows selected gateway.

3 Set communication path

- ► Activate [Scan Network …].

> Window [Select Device] appears.

- ► Select gateway node and start scan process with [Scan network].

> CODESYS scans network for devices.

> Window shows network path and detected devices.

- ► Select node of the device and activate [OK] to set the communication path to the device-internal PLC.

> CODESYS can transfer data to the device-internal PLC.

6.5 Add ifm function libraries to the application

23390

► Familiarise yourself with the following CODESYS functions!
  - Library manager
    → Online help > CODESYS Development System > Managing Libraries > Adding a library to the application

The ifm package includes function libraries for the programming of the device under CODESYS. The libraries are installed in CODESYS together with the ifm package.

The user can add the libraries individually to an application he/she needs for the programming.

By means of the container library `ifmR360-3.library`, the user can add all functions available for the device to the project.

To integrate a library into a project:

Requirements:

> ifm package is correctly installed (→ **Install package (PC/laptop)** (→ p. 21)).

Load container library

► In the device tree: Double-click on [PLC Logic] > [Application] > [Library Manager]
> Editor window shows table of added libraries.
► Click on [Add library].
> Dialogue window [Add library] appears.
► Select requested library and click on [OK] to add the selected library to the application.
> CODESYS adds the selected library to the project.
> Editor window shows the library.

6.6 Activate the access protection for a project

21783

► Familiarise yourself with the following CODESYS functions!
  - Protect and save project
    → Online help > CODESYS Development System > Protect and save project

The user can use a password to protect the device from unauthorised access.

► Select [Project] > [Project Settings...].
> Window [Project Settings] appears.
► Select [Security].
► Activate checkbox [Enable project file encryption].
► Enter the requested password in the field [New password].
► Enter the entered password again in the field [Confirm new password].
► Select [OK] to activate the access protection for the project.
> Access protection is activated. Project is encrypted.

7 System configuration

Contents

Configure PLC 65
Configure inputs and outputs 68
Configure interfaces 69

23096

7.1 Configure PLC

23097
23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

7.1.1 Allocate memory partition

23483
23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

Further information: → **Memory allocation variants** (→ p. 32)

To allocate the memory partitions to the PLCs:

1 Select memory partition

► In the device tree: Double-click on symbol [Device (CR720S)]

► In the editor window: Select [Memory Layout] tab.

> The editor window shows the partitioning of the memory:

Memory layout	Partitioning Safety PLC / PLC
MemoryLayout_4s_1	4 MB / 1 MB
MemoryLayout_3s_2	3 MB / 2 MB
MemoryLayout_2s_3	2 MB / 3 MB
MemoryLayout_1s_4	1 MB / 4 MB

2 Set memory partitioning

► Highlight the required memory partition

► Click on the [Update Devices] button

> Memory partitioning is adopted in CODESYS

3 Load memory partition into the device

► Click on the [Download Configuration] button

> Memory partition is downloaded to the device

7.1.2 Allocate inputs/outputs

23482
23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

To allocate the I/Os to the PLCs:

Before the programming of the application may even begin:

1 Select I/O allocation

► In the CODESYS device tree: Double-click on symbol [Device (CR720S)]

► In the editor window: Select [I/O Assignment] tab.

> Editor window shows the PLC allocation of the inputs/outputs (excerpt):

Section	Element	Parameter	Standard PLC	Safety PLC
System info	IP Settings	--	⊙	O
Local_IO	Inputs	IN0000	⊙	O
		IN0001	⊙	O
		IN0002	⊙	O
		…	…	…
	Outputs	OUT0000	⊙	O
		OUT0001	⊙	O
		OUT0002	⊙	O
		…	…	…
	System_Outputs	VBB0_SW	⊙	O
		VBB1_SW	⊙	O
		…	…	…
		Supply_Switch	⊙	O
HMI	User_LEDs	User LED 0	⊙	O
		User LED 1	⊙	O
		User LED 2	⊙	O
		User LED 3	⊙	O

2 Set I/O allocation

► Highlight all I/Os in the column [StandardPLC] to allocate it to the standard PLC

► Highlight all I/Os in the column [SafetyPLC] to allocate it to the safety PLC

> The I/Os are allocated

7.1.3 Manage files

23520

To transfer files between PC and device:

1 Select file view

- ► In the device tree: Double-click on symbol [Device (CR720S)]
- ► In the editor window: Select the [Files] tab.
- > The editor window shows the file structure on the PC on the left and on the device on the right

2 Transfer file from PC to device

- ► Highlight the file on the left
- ► Select device target directory on the right
- ► Star transfer using the [>>] button
- > The file is transferred to the device

3 Transfer the file from the device to the PC

- ► Highlight the file on the right
- ► Select PC target directory on the left
- ► Start the transfer using the [<<] button
- > The fle is transferred to the PC

7.1.4 Manage users and groups

23521

This function has not yet been implemented.

7.2 Configure inputs and outputs

23099

The inputs and outputs can be configured applying two methods:

7.2.1 via system configuration

23149

This method is useful if the configuration is not supposed to be changed again during the runtime of the application.

Note: Only the I/Os allocated to the PLC can be configured!

Procedure using the example of the operating mode of an input / output:

► In the CODESYS device tree: Extend required PLC > Element [Local_IO]
► Double-click on [Inputs] / [Outputs]
► Click on [Parameters] tab
> The parameter setting view of the inputs / outputs appears
► Select input / output from the list
► Double-click in the column [Value] of the parameter [Mode]
► Click on arrow symbol
> List of possible modes appears
► Click on the required mode
> The mode for the input / output is set
► If needed, set further parameters as described, e.g. filters, periods, frequency, etc.

7.2.2 via function block

23150

This method is useful if the configuration is supposed to be changed during the runtime of the application.

The operating type of the inputs and outputs is set via the block input eMode of the following FBs.

Examples:

- FB **Input** (→ p. 126) > Input eMode
- FB **Output** (→ p. 129) > Input eMode
- FB **OutputGroup** > Input eMode

7.3 Configure interfaces

23100

7.3.1 Configure serial interface

23151

The CODESYS service communication via RS232 only works with the preset baud rate.

For other purposes, the device supports the following baud rates:
9 600 baud
19 200 baud
28 800 baud
38 400 baud
57 600 baud
115 200 baud (preset)

Setting the interface: → **Interface configuration file comconf.cfg** (→ p. 77)

7.3.2 Configure Ethernet interface

23152

Setting the interface: → **Interface configuration file comconf.cfg** (→ p. 77)

Factory setting:
IP address = 192.168.82.247
Subnet mask = 255.255.255.0
Gateway address = 192.168.82.21
UDP port = 12345

23450

NOTICE

Only use the Ethernet interface in a secure network environment (e.g. separate network or VPN)!

Otherwise, unauthorised persons can read or manipulate data or tamper with the functions of the device.

The the IP parameter of the Ethernet interface

23455

In order to update the runtime system of the CR720S via a network, the device must be connected to the corresponding network. For the configuration of the Ethernet interface, the following options are available:

- Manual — The user defines the parameters of the Ethernet interface manually:
  IP address,
  Subnet mask,
  gateway address
  ► Observe the **Address assignment in Ethernet networks** (→ p. 70) in Ethernet networks!
- Automatic — The interface parameters are set via the Dynamic Host Configuration Protocol (DHCP).
  (the development of this function is still in progress)

Setting the interface: → **Interface configuration file comconf.cfg** (→ p. 77)

Address assignment in Ethernet networks

14436

In the Ethernet network every IP address MUST be unique.

The following IP addresses are reserved for network-internal purposes and are therefore not allowed as an address for participants: nnn.nnn.nnn.0 | nnn.nnn.nnn.255.

Only network participants whose subnet mask is identical and whose IP addresses are identical with respect to the subnet mask can communicate with each other.

Rule:
If part of the subnet mask = 255, the corresponding IP address parts must be identical.
If part of the subnet mask = 0, the corresponding IP address parts must be different.

If the subnet mask = 255.255.255.0, 254 participants communicating with each other are possible in the network.
If the subnet mask = 255.255.0.0, 256x254 = 65 024 participants communicating with each other are possible in the network.

In the same physical network different subnet masks of the participants are allowed. They form different groups of participants which cannot communicate with groups of participants having other subnet masks.

In case of doubt or problems please contact your system administrator.

Examples:

Participant A IP address	Participant A Subnet mask	Participant B IP address	Participant B Subnet mask	Communication of participants possible?
192.168.82.247	255.255.255.0	192.168.82.10	255.255.255.0	Yes, 254 participants possible
192.168.82.**247**	255.255.255.0	192.168.82.**247**	255.255.255.0	No (same IP address)
192.168.82.247	255.255.**255**.0	192.168.82.10	255.255.**0**.0	No (different subnet mask)
192.168.**82**.247	255.255.255.0	192.168.**116**.10	255.255.255.0	No (different IP address range: 82 vs. 116)
192.168.222.213	255.255.0.0	192.168.222.123	255.255.0.0	Yes, 65 024 participants possible
192.168.111.213	255.255.0.0	192.168.222.123	255.255.0.0	Yes, 65 024 participants possible
192.168.82.247	255.255.255.0	192.168.82.**0**	255.255.255.0	No; the whole network is disturbed because the IP address xxx.xxx.xxx.0 is not allowed

7.3.3 Configure CAN interfaces

23156

The CAN interfaces are configurable as follows:

- via system configuration:
  - CANopen
  - SAE J1939
- via function block:
  - RAW-CAN

Under Vendor = 3S, you will find, among others, the following entries:

```
CIA CANopen
 +- CIA CANopenManager
     +- CANopen_Manager
     +- CANopen_Manager_SIL2
 +- CIA Local Device
     +- CANopen Device
     +- CANopen Device SIL2
SAE J1939
 +- SAE J1939 Manager
     +- J1939_Manager
```

23368

⚠ WARNING

The safety functionality is in preparation.
At present the device has NO safety functionality!

► Do NOT use the device for safety-related functions!

The functions of the following protocols are available, but NOT yet suitable for safety applications:
- CANopen_Manager_SIL2
- CANopen_Device_SIL2

via system configuration: CANopen Manager

23159

In the CODESYS device tree, you will find the following entry under each PLC:
[Communication] > [CAN]

Configure each interface only at ONE position!

- **Attach CAN bus**

► In the CODESYS device tree: Right mouse click on [Communication] > [CAN].
► Select [Add Device...].
> Window [Add Device] appears.
► Select [Vendor:] [ifm electronic].
► In the list below: Select [ifmCANbus].
► Confirm the selection with [Add Device].
> Close the window [Add Device] with the [Close] button.

- **Assign CAN interface**

► In the CODESYS device tree: Double-click on [Communication] > [CAN] > [ifmCANBus].
► Tab [General] > [General] > [Network]:
assign this setting with ▲/▼ to a CAN interface.
permissible = 0...3
► Select the required value for baud rate [Baudrate (bit/s)] from the list field.

- **Attach CANopen manager**

► In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus].
► Select [Add Device...].
> Window [Add Device] appears.
► Select [Vendor:] [<All vendors>].
► In the list below: Select [Fieldbusses] > [CiA CANopen] > [CiA CANopenManager] > [CANopenManager].
► Confirm the selection with [Add Device].
► Close the window [Add Device] with the [Close] button.

- **Set CANopen manager parameters**

► In the CODESYS device tree: Double-click on [Communication] > [CAN] > [CiA CANopenManager] > [CANopenManager].

► Tab [General] > [General] > [Node ID]:
assign a node ID to this interface using ▲/▼.
permissible = 1...127

► Select the further parameters according to the requirements, e.g.:

- Configure the heartbeat protocol in the section [Nodeguarding]:
  - ► Click on the field of options in order to activate [Heartbeat Producing]
  - ► Set the parameters [Node ID] and [Producer Time (ms)]
- Configure the sync protocol in the section [Sync]:
  - ► Click on the field of options to activate [Enable Sync Producing], if necessary.
  - ► Set the parameters [COB-ID (Hex)], [Cycle Period (µs)] and [Window Length (µs)]
- In the section [Time]: The time protocol is not supported.

> With the menu [File] > [Save Project], the values become valid.

The sync protocol triggers the receiving/sending of data of the CANopen devices (input: SDO 16#1800/ output: SDO 16#1400).

via system configuration: CANopen device

23523

In the CODESYS device tree, you will find the following entry under each PLC:
[Communication] > [CAN]
These entries are equivalent.

- **Attach CAN bus**

► In the CODESYS device tree: Right mouse click on [Communication] > [CAN].
► Select [Add Device...].
> Window [Add Device] appears.
► Select [Vendor:] [ifm electronic].
► In the list below: Select [ifmCANbus].
► Confirm the selection with [Add device].
► Close the window [Add device] with the [Close] button.

- **Assign CAN interface**

► In the CODESYS device tree: Double-click on [Communication] > [CAN] > [ifmCANBus].
► Tab [General] > [General] > [Network]:
assign this setting with ▲/▼ to a CAN interface.
permissible = 0...3
► Select the required value for baud rate [Baudrate (bit/s)] from the list field.

- **Attach CANopen device**

► In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus].
► [Add Device...] Select .
> Window [Add Device] appears.
► Select [Vendor:] <All vendors>.
► In the list below: Select [Fieldbusses] > [CiA CANopen] > [Local Device] > [CANopenDevice].
► Confirm the selection with [Add Device].
► Close the window [Add Device] with the [Close] button.

- **Set CANopen device parameters**

► In the CODESYS device tree: Double-click on [Communication] > [CAN] > [CANopenDevice].
► Tab [General] > [General] > [Node ID]:
assign a node ID to this interface using ▲/▼.
permissible = 1...127
► Select the further parameters according to the requirements.
> With the menu [File] > [Save Project], the values become valid.

When activating the option field [Werkseinstellungen]: Each time the controller is switched on or the program is downloaded, the settings are reset to factory settings. Thereby, user settings can be overwritten. The type and the scope of reset settings depend on the CANopen device.

In the CANopen Device, the correct baud rate and the node ID must be set, so that the CANopen master will recognise the device.

via system configuration: J1939 manager

23522

In the CODESYS device tree, you will find the following entry under each PLC:
[Communication] > [CAN]
These entries are equivalent.

- **Attach CAN bus**

► In the CODESYS device tree: Right mouse click on [Communication] > [CAN].
► Select [Add Device...].
> Window [Add Device] appears.
► Select [Vendor:] [ifm electronic].
► In the list below: Select [ifmCANbus].
► Confirm the selection with [Add Device].
> Close the window [Add Device] with the [Close] button.

- **Assign CAN interface**

► In the CODESYS device tree: Double-click on [Communication] > [CAN] > [ifmCANBus].
► Tab [General] > [General] > [Network]:
assign this setting with ▲/▼ to a CAN interface.
permissible = 0...3
► Select the required value for baud rate [Baudrate (bit/s)] from the list field.

- **Attach J1939 manager**

► In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus].
► Select [Add Device...].
> Window [Add Device] appears.
► Select [Vendor:] <All Vendors>.
► In the list below: Select [Fieldbusses] > [SAE J1939] > [J1939 Manager] > [J1939_Manager].
► Confirm the selection with [Add Device].
► Close the window [Add Device] with the [Close] button.

- **Set J1939 manager parameters**

► In the CODESYS device tree: Double-click [Communication] > [CAN] > [J1939_Manager].
► Tab [General] > [Database] > [Database]:
select the list from the required database.
default = J1939Default

Users can use their own databases.
They must be at the following storage location:
`C:\ProgramData\CODESYS\J1939 Databases`
The directory `ProgramData` is hidden by default.

> With the menu [File] > [Save Project], the values become valid.

- **Attach J1939-ECU**

► In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus] > [J1939_Manager].
► Select [Add Device...].
> Window [Add Device] appears.
► In the area [Device]: [Vendor:] select <All vendors>.
► In the list below: Select [Fieldbusses] > [J1939] > [J1939_ECU] > [J1939_ECU].
► Confirm the selection with [Add Device].
► Close the window [Add Device] with the [Close] button.

- **Set J1939-ECU parameters**

► In the CODESYS device tree: Double-click on [Communication] > [CAN] > [J1939_Manager] > [J1939_ECU].
► Make the following settings in the tab [General] in the section [General] according to the specific application:

user case	[Local Device]		Significance [Preffered Address]
▪ Receiving broadcast data of the ECU ▪ No transmission	☐	deactivated	Address of the ECU from which the data is to be received
▪ Sending data (broadcast and P2P) ▪ Receiving P2P data	☑	activated	Address of the ifm controller

► Add parameter groups in the tab [TX-Signals] by clicking on [Add PG].
> The settings become valid with menu [File] > [Save Project].

via function block: RAW-CAN

23160

The **Library ifmRawCAN.library** (→ p. 162) features several function blocks for this application.

7.3.4 Interface configuration file comconf.cfg

22929

The file directory /com of the device contains the file comconf.cfg.
To change the configuration data of the following interfaces, this file must be written into the device with the corresponding changes:

- Serial interface
- Ethernet interface
- CAN interfaces

Factory setting of the content:

```
[ETHERNET]
Number=1

[ETHERNET0]
IpV4Address=192.168.82.247
IpV4SubnetMask=255.255.255.0
IpV4Gateway=192.168.82.21
UDPPort=12345

[CAN]
Number=4

[CAN0]
Baud rate=250000
NodeId=127

[CAN1]
Baud rate=250000
NodeId=126

[CAN2]
Baud rate=250000
NodeId=125

[CAN3]
Baud rate=250000
NodeId=124

[COM]
Number=1

[COM0]
Baud rate=115200
Bits=8
Parity=0
Stop=1
```

► To start the (deactivated) device with these default settings:
(the file comconf.cfg is not taken into consideration)
TRUE on connection RESET-COM (Pin 72) simultaneous wtih POWER-ON
After the start-up: FALSE on RESET-COM

► To start the (deactivated) device with the content of the (new) file comconf.cfg:
FALSE on connection RESET-COM (Pin 72) simultaneously with POWER-ON

8 Programming

Contents

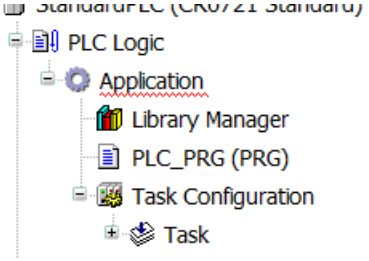
Objects of a PLC application 78
Create PLC application 79
Use ifm function libraries 82
Use IO mapping 84
Use RawCAN (CAN Layer 2) 88
Use CANopen 90
Use SAE J1939 91

14603

8.1 Objects of a PLC application

22915

All objects of a PLC application are listed as sub-elements of the node [Application] in the device tree. In the basic configuration, a PLC application contains the following objects:



[Application]	Container for objects of a PLC applications
[Library manager]	Provides access to the standard and device-specific function libraries: → **Use ifm function libraries**
[PLC_PRG(PRG)]	Provides access to the editor of the PLC application → **Create PLC application** (→ p. 79)
[Task configuration]	Provides access to the settings of the task processing: → **Configure task processing** (→ p. 81)

If necessary, the user can add further objects to the PLC application.

8.2 Create PLC application

23457

► Familiarise yourself with the following CODESYS functions!
- → Online help > CODESYS Development System > Programming Applications

CODESYS automatically generates the function block PLC_PRG (PRG) during project creation. The function block is processed cyclically. Other programs are called in this function block.

To create a PLC application:

► In the device tree: Double-click on [Application] > [PLC_PRG (PRG)]

> Editor window shows input mask of the selected programming language.

► Enter program code.

8.2.1 Supported programming languages

23454

The following table shows which ifm function libraries support which programming languages according to IEC 61131:

Library	function block diagram (FBD)	sequential function chart (SFC)	instruction list (IL)	continuous function chart (CFC)	ladder diagram (LD)	structured text (ST)
ifmDeviceCR720S.library	X	X	X	X	X	X
ifmCANopenManager.library	X	X	X	X	X	X
ifmRawCAN.library	X	X	X	X	X	X
ifmFastInput.library	X	X	X	X	X	X
ifmIOcommon.library	X	X	X	X	X	X
ifmOutHBridge.library	X	X	X	X	X	X
ifmOutGroup.library	X	X	X	X	X	X
ifmOutPWM.library	X	X	X	X	X	X

Legend:
X = is supported
– = is not supported

8.2.2 Supported variable types

Contents

Retain variables 80

23456

► Familiarise yourself with the following CODESYS functions!
- Local variables
  → Online help > CODESYS Development System > Reference Programming > Variable Types and special Variables > Local Variables - VAR
- Global variable list
  → Online help > CODESYS Development System > Reference Programming > Variable Types and special Variables > Global Variables - VAR_GLOBAL
- Network variables *(currently not supported)*
  → Online help > CODESYS Development System > Exchanging Data on the Network > Network Variables

The device supports the following variable types:

Variable type	Declaration	Scope of validity	Memory behaviour
local	In the declaration part of the POU	Applies only to the POU in which it has been declared	volatile
local retain			non volatile
global	in the global variable list (GVL)	applies to all POUs of the project	volatile
global retain			non volatile
network	In network variable lists	Values are available to all projects in the whole network if the variable is contained in their network variables lists.	Volatile
Network retain			non volatile

For performance reasons, do not overuse 64 bit variables!
CAN network variables are not supported!

Retain variables

23613

Variables declared as `RETAIN` generate remanent data. Retain variables keep the values saved in them when the device is switched on/off or when an online reset is made.

Typical applications for retain variables are for example:
• operating hours which are counted up and retained while the machine is in operation,
• position values of incremental encoders,
• preset values entered in the screen device,
• machine parameters,
i.e. all variables whose values must not get lost when the device is switched off.

8.2.3 Options to access input and output data

17621

In a CODESYS project, each input and output has a physical address according to the IEC standard (e.g. %IW5). CODESYS offers the following options to access this address from a PLC application and thereby to access the input and outputs data of the device:

- Access to IEC address via AT declaration
- Definition of an ALIAS for an IEC address
- Link a program variable to an IEC address (mapping)

8.2.4 Configure task processing

23487

► Familiarise yourself with the following CODESYS functions!
  - Task configuration:
    → Online help > CODESYS Development System > Programming Applications > Task Configuration

The processing of the tasks is controlled by parameters. The user can set the parameters for each task.

CODESYS automatically creates the following task when the project is created:

Name	Description
Task	Task for the processing of the main program [PLC_PRG (PRG)]

For subprograms with POUs that are to be executed several times per PLC cycle:

► Create new task.

► Configure task properties:
1. [Priority]: permissible = 0 (high) ... 3 (low)
   (select a priority for each task)
2. [Type]: Cyclic
3. [Interval]: Interval of the task call-ups in [ms]
The interval time must be longer than the runtime of the task.

► Recommended: Activate watchdog: → **Configure IEC watchdog** (→ p. 37)
The watchdog time must be shorter than the interval time.
The watchdog time must be longer than the runtime of the task.

► Assign subprogram with POUs to the newly created task.

If the CAN buses are heavily utilised:

► Configure task properties:
1. [Priority]: high
2. [Type]: Cyclic
3. [Interval]: requested cycle time (=transmission interval)

► Assign subprograms with the POUs for CAN communication to the CAN tasks.

8.3 Use ifm function libraries

23458

ifm electronic provides the following function libraries for the programming of the device under CODESYS 3.5:

Name	Description
ifmCANopenManager	Functions for use of the CAN interfaces as CANopen Manager
ifmDeviceCR720S	Data structures, enumeration types and global variables
ifmFastInput	Functions to access the fast inputs of the device
ifmIOcommon	Functions for access to the inputs and outputs of the device
ifmIOconfigDiagProt	Functions to configure the I/O-related diagnostic and protective functions
ifmOutGroup	Functions to control output group switches
ifmOutHBridge	Functions to access H-bridge outputs
ifmOutPWM	Functions to access PWM outputs
ifmRawCAN	Functions for use of the CAN interfaces as CAN Layer 2
ifmSysInfo	Functions to set / read system information
ifmTypes	Global types and interfaces for other ifm libraries

Detailed information about the ifm function libraries: → ifm function libraries (→ p. 103)

8.3.1 Access to inputs

23600

To access the inputs of the device, the following functional elements are available:

Function element	Short description
Input (→ p. 126)	Assigns an operating mode to an input channel Provides the current state of the selected channel
FastCount (→ p. 117)	Counter block for fast input pulses
IncEncoder (→ p. 119)	Up/down counter function to evaluate encoders
Period (→ p. 121)	• measures at the indicated channel: the frequency and the period length (cycle time) in [µs], • measures at the indicated channel pair: the phase shift in [°] between channel A and channel B

8.3.2 Access to outputs

23608

To access the outputs of the device, the following functional elements are available:

Function element	Short description
Output (→ p. 129)	Assigns an operating mode to an output channel Provides the current state of the selected channel
OutputGroup	controls the activation status of an output group and provides diagnostic information about the group and the connected outputs. Using the FB, an output group including the corresponding outputs can be switched on or off.
HBridge (→ p. 149)	H bridge on a PWM channel pair
PWM1000 (→ p. 157)	Initialises and configures a PWM-capable output channel the mark-to-space ratio can be indicated in steps of 1 ‰
CurrentControl (→ p. 154)	Current controller for a PWMi output channel

8.3.3 Control device

23278

The following function elements are available to control the device:

Function element	Short description
SupplySwitch (→ p. 134)	Switch off the unit
SetLED (→ p. 132)	Change the frequency and the colour of the status LED in the application program

8.3.4 Read device information

23610

To read information from the device the following functional elements are available:

Function element	Short description
SystemSupply (→ p. 136)	indicates the value of the system voltage
Temperature (→ p. 138)	indicates the value of the system temperature

8.4 Use IO mapping

Contents

Access inputs 85
Access outputs 86
Read diagnostic data of the device 87

23498

During the IO mapping (I/O image), global variables are coupled to the IEC addresses (%Ixx, %Qxx). Via symbol names, the user has access to the following elements from the application:

- inputs and outputs
- Functions of the operating elements
- Functions of the display elements
- States of system components and characteristic values

The addresses of the system flags can change if the PLC configuration is extended.
► While programming only use the symbol names of the system flags!

8.4.1 Access inputs

23499

The user can use the following global variables to access the operating modes and the values of the inputs of the device.

Variable	Data type	Access	Description	Possible values	
INnnnn_I.					
ValueAnalogue	UINT	r	Value of the analogue input	0 ... 65535	
ValueDigital	BIT	r	Value of the digital input	FALSE	Input deactivated
				TRUE	Input activated
ValueCount	UDINT	r	Value of the counting input	0 ... 4294967295	
ValueCountIncEnc	DINT	r	Value of the encoder input	-2147483648 ... 2147483647	
ValueLastCountWasUp	BIT	r	Counting direction upwards	FALSE	not active
				TRUE	active
ValueLastCountWasDown	BIT	r	Counting direction downwards	FALSE	not active
				TRUE	active
ValueCycle	UDINT	r	cCcle time	0 ... 4294967295	
ValueFreq	REAL	r	Frequency	1.401e-45 ... 3.403e+38	
ValueTime	UDINT	r	Elapsed time since the last edge evaluation	0 ... 4294967295	
ValueRatio	UINT	r	Pulse/pause ratio	0 ... 65535	
Error	BIT	r	Error	FALSE	no error
				TRUE	Error
INnnnn_Q.					
CountDirection	ENUM of INT	r/w	Read/set counting direction	COUNT_OFF	Counting off
				COUNT_UP	Counting upward
				COUNT_DOWN	Zählen abwärts
Counting downward	BIT	w	Set preset value	FALSE	no action
				TRUE	Set preset value

Legend:
r = read only
r/w = read and write

The valid value ranges and the type and number of the variables of the input depend on the active operating mode of the input.

► Observe configuration of the inputs! → **Configure inputs and outputs** (→ p. 68)

8.4.2 Access outputs

23515

The user can use the following global variables to access the operating modes and the values of the outputs of the device.

Variable	Data type	Access	Description	Possible values	
OUTnnnn_I					
OutCurrent	UINT	r/w	Current value of the analogue output	0 ... 65535	
Ratio	UINT	r/w	PWM Ratio	0 ... 65535	
OutVoltageDiag	UINT	r	Measured voltage value of the analogue output in mV	0 ... 65535	
OutCurrentDiag	UINT	r	Measured current value of the analogue output in mA	0 ... 65535	
OutState	BIT	r/w	Output status	0 ... 4294967295	
Error	BIT	r/w	Error	FALSE	no error
				TRUE	Error
OUTnnnn_Q					
ValueAnalogue	UINT	r/w	Analogue output value	0 ... 65535	
ValueDigital	UINT	r/w	Digital output value	0 ... 65535	
OutVoltage	UINT	r/w	Output voltage	0 ... 65535	
Error	BIT	r/w	Error	FALSE	no error
				TRUE	Error

Legend:
r = read only
r/w = read and write

The valid value ranges and the type and number of the variables of the output depend on the active operating mode of the output.

► Observe configuration of the outputs! → **Configure inputs and outputs** (→ p. 68)

8.4.3 Read diagnostic data of the device

23528

The user can use the following global variables to access the current diagnostic data of the device:

Name	Data type	Access	Description	Possible values	
iTemperature0	INT	r	Temperature on the system board (value in °C)	-32768 ... 32767	
iTemperature1	INT	r	Temperature on the system board (value in °C)	-32768 ... 32767	
uiVoltageVBB15	UINT	r	Voltage at power input VBB15 (value in mV)	0 ... 65535	0 mV ... 65535 mV
uiVoltageVBB30	UINT	r	Voltage at power input VBB30 (value in mV)	0 ... 65535	0 mV ... 65535 mV

Legend:
r = read only

8.5 Use RawCAN (CAN Layer 2)

Contents

RawCAN: Control CAN network nodes 88
RawCAN: Send and receive CAN messages 88
RawCAN: Request and send remote CAN messages 89

23545

► Observe the notes on task configuration! (→ **Configure task processing** (→ p. 81))

In order to access one of the CAN interfaces configured for CANopen operation, the following POUs are available.

Requirements:

- The CAN interface is configured for operation as RawCAN (CAN Layer 2) (→ **Configure CAN interfaces** (→ p. 71)).

8.5.1 RawCAN: Control CAN network nodes

23546

The following POUs are available to control a node in a CAN network:

Function element	Short description
CAN_Enable (→ p. 163)	initialises the specified CAN interface configures the CAN baud rate
CAN_Recover (→ p. 165)	controls the processing of a failure of the specified CAN channel If the CAN channel fails, reset the CAN interface and reboot

8.5.2 RawCAN: Send and receive CAN messages

23547

The following POUs are available to send or receive messages in a CAN network:

Function element	Short description
CAN_Rx (→ p. 171)	configures a data receive object and reads the receive buffer of the data object
CAN_RxMask (→ p. 173)	receives CAN messages of a non-coherent area The area is defined via a bit pattern and a bit mask
CAN_RxRange (→ p. 175)	receives CAN messages of a coherent area The area is defined via an upper and lower limit
CAN_Tx (→ p. 177)	asynchronous transmission of CAN messages

8.5.3 RawCAN: Request and send remote CAN messages

23548

The following POUs are available to request remote messages in a CAN network or to send replies to a remote request:

Function element	Short description
CAN_RemoteRequest (→ p. 167)	Send a request for a remote message
CAN_RemoteResponse (→ p. 169)	reply to the request of a remote message

8.6 Use CANopen

Contents

CANopen: Send and receive SDO 90
CANopen: Network Management (NMT) 90

23544

- ► Observe the notes on task configuration! (→ **Configure task processing** (→ p. 81))
- ► Observe the notes about CANopen! (→ System manual)

In order to access one of the CAN interfaces configured for CANopen operation, the following POUs are available in ifm libraries.

Further POUs are available in CODESYS libraries from 3S.

Requirements

- The device is configured as CANopen manager (master) (→ **via system configuration: CANopen Manager** (→ p. 72)).

8.6.1 CANopen: Send and receive SDO

23537

The following POUs are available to send or receive Service Data Objects (SDO):

Function element	Short description
COP_SDOread (→ p. 106)	Read Service Data Object (SDO)
COP_SDOwrite (→ p. 108)	Write Service Data Object (SDO

8.6.2 CANopen: Network Management (NMT)

23539

The following POUs are available for the management of the CANopen network:

Function element	Short description
COP_GetNodeState (→ p. 104)	Request state of one or several CANopen devices
COP_SendNMT (→ p. 110)	Send an NMT control command to a CANopen device

8.7 Use SAE J1939

23802

To use the SAE J1939 network protocol, 3S provides the library `IoDrvJ1939`.

9 Operation

Contents

Transfer CODESYS project to device 92
Operating states 94
Status LEDs 95
Reset 98
Data transmission for series production 100
Display system information 102

23280

► Familiarise yourself with the following CODESYS functions!

- Translate project/application and transfer to the device
  → Online help > CODESYS Development System > Transferring Applications to the PLC

9.1 Transfer CODESYS project to device

Contents

Load the application to the device 92
Delete application from CR720S 93

23493

To save the CODESYS project on the device, transfer the following component:

- Application (→ **Load the application to the device** (→ p. 92))

► Observe notes on the operating modes of the PLC of the device!
→ **Operating states of CR720S**

9.1.1 Load the application to the device

23494

To transfer the created application as boot project to the device:

Requirements:

> Communication path is set (→ **Set communication path of PLC** (→ p. 62)).
> Project tested.

1 Translate application

► In the device tree: highlight application as active application.
► Use [Build] > [Rebuild] to translate the active application.
> CODESYS generates program code.

2 Load application to the device

► Only for safety PLC: Change to debug mode with [SIL2] > [Enter debug mode…].
► Use [Online] > [Login] connect with the device.
> Active application is loaded to the device (download).
> Application on the device is in the STOP state.

3 Start application

► Use [Debug] > [Start] to start the application.
> Application goes to the RUN state.

9.1.2 Delete application from CR720S

18030

To delete an application stored on the device:

1 Connect with the device

► In the device tree: highlight application as active application.

► Use [Online] > [Login] to establish connection to the device.

> CODESYS is in the online mode.

2 Delete application

► In the editor window: Select [Device] > [Applications] tab.

► Press [Refresh List] to refresh the view.

> List shows the applications that are stored on the device.

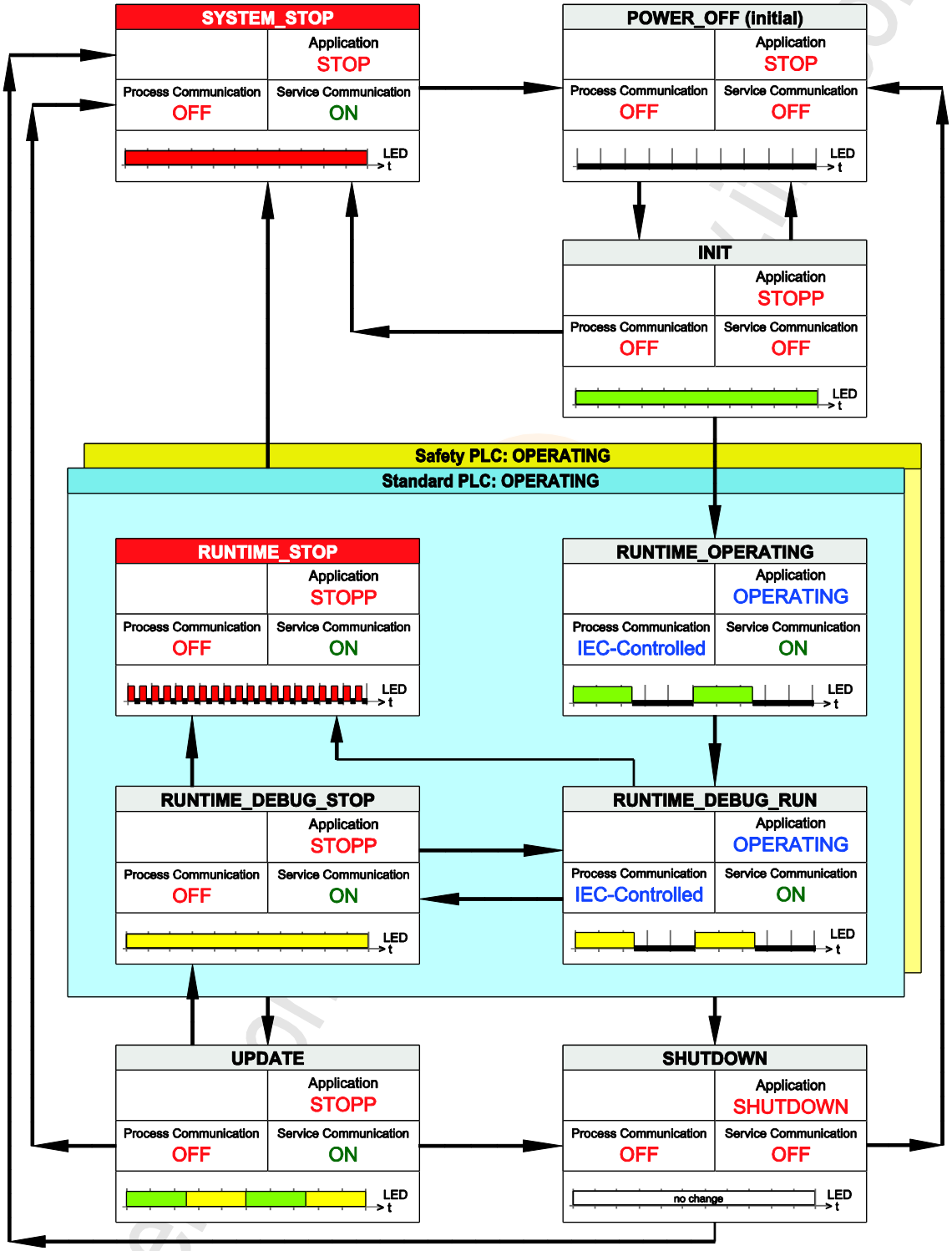
► Delete all applications in the device with [Remove All].
OR:
Highlight requested application and press [Remove] to delete it from the device.

> Selected application will be deleted.

9.2 Operating states

23421

The following figure shows the possible operating modes of the device:



It contains:
- status of the application
- status of the process communication (inputs/outputs, CAN bus)
- status of the service communication (connection with the programming device)
- display of the LEDs SYS0 / SYS1

9.3 Status LEDs

22920

The device has the following LEDs:

LED	Description
SYS0	Status of the standard PLC Status of the ifm operating system Status of the bootloader
SYS1	Status of the safety PLC Status of the ifm operating system
ETH0	Status of the Ethernet interface 0
ETH1	Status of the Ethernet interface 1
APPL0 APPL1 APPL2 APPL3	LEDs for free use in the application

9.3.1 Status LED: system ifm operating system (SYS0+SYS1)

23429

For the status of the ifm operating system, both LEDs SYS0 and SYS1 are lit simultaneously:

LED colour	Display	Description
off	permanently off	ifm operating system on the unit: POWER_OFF
		
Green	permanently on	ifm operating system on the unit: INIT
		
Red	permanently on	ifm operating system on the unit: SYSTEM_STOP Error class = A
		
no change	no change	ifm operating system on the unit: SHUTDOWN
	no change	
green-yellow	Flashing with 2 Hz	ifm operating system on the unit: UPDATE
	 (time frame = 200 ms)	

9.3.2 Status LED: system PLC (SYS0, SYS1)

23436

The SYS0 LED is for the "standard PLC".
The SYS1 LED is for the "safety PLC".
The status of one of the PLCs has no influence on the display of the other PLC.

LED colour	Display	Description
Green	permanently on	RUNTIME_OPERATING no application loaded
		
Green	Flashing with 2 Hz	RUNTIME_OPERATING Application = RUN
	 (time frame = 200 ms)	
Yellow	Flashing with 2 Hz	RUNTIME_DEBUG_RUN Application = RUN
	 (time frame = 200 ms)	
Yellow	permanently on	RUNTIME_DEBUG_STOP Application = STOP
		
Red	flashes with 10 Hz	RUNTIME_STOP Error class = B
	 (time frame = 200 ms)	

9.3.3 Status LED: System bootloader (SYS0)

23426

The SYS0 LED is for the bootloader status only.
The SYS1 LED is switched off in these cases.

23561

Only execute the bootloader update when explicitly requested by ifm!

LED colour	Display	Description
Green	Flashing with 5 Hz	no runtime system loaded
	 (time frame = 200 ms)	
green-yellow	flashes with 5 Hz	Bootloader update process active
	 (time frame = 200 ms)	

9.3.4 Status LED: Ethernet interfaces (ETH0, ETH1)

23445

The two Ethernet interfaces indicate their status as follows:

LED colour	Display	Description
Green	permanently on	Ethernet connection is established non data traffic
		
Green	blinks	Ethernet connection is established with data traffic
		

9.3.5 Controlling LEDs in the applications

23449

The LEDs APPL0 to APPL3 are for free use in the applications.
This is the function of the FB **SetLED** (→ p. 132).

Possible colours: → **LED_COLOUR (ENUM)** (→ p. 115)
Possible frequencies: → **LED_FLASH_FREQ (ENUM)** (→ p. 115)

9.4 Reset

Contents

Supported reset variants 98
Reset application (warm) 98
Reset application (cold) 99
Reset application (origin) 99

18025

9.4.1 Supported reset variants

18613

The following table shows the reset variants supported by the device-internal CODESYS PLC and the resulting system behaviour:

Type of reset	System behaviour	Triggering actions
Reset (warm)	▪ application goes to STOP state. ▪ Standard variables (VAR) of the application are initialised. ▪ Remanent variables (VAR RETAIN) of the application keep their current values.	→ **Reset application (warm)** (→ p. 98)
Reset (cold)	▪ application changes to the STOP state. ▪ All variables (VAR, VAR RETAIN) of the application are initialised.	→ **Reset application (cold)** (→ p. 99)
Reset (default)	▪ application goes to STOP state. ▪ The application on the PLC is deleted. ▪ All variables (VAR, VAR RETAIN) of the application are initialised. ▪ PLC is reset to the default state.	→ **Reset application (origin)** (→ p. 99)

A variable that has been declared without an initialisation value is initialised with the variable-specific standard value (e.g. `INT = 0`).

9.4.2 Reset application (warm)

7233

To reset the application:

► In the device tree: Select [Application] and select
► [Online] > [Login] as active application.
> CODESYS changes to the online mode.
► Select [Online] > [Reset warm] to reset the application.
> Application changes to the STOP state.
> Standard variables are newly initialised.
> Retain variables keep their values.

9.4.3 Reset application (cold)

7230

To reset the application:

► In the device tree: Select [Application].
► Select [Online] > [Login].
> CODESYS changes to the online mode.
► Select [Online] > [Reset cold] to reset the application.
> Application changes to the STOP state.
> All variables are newly initialised

9.4.4 Reset application (origin)

22672

To reset the application:

► In the device tree: Select [Application].
► Select [Online] > [Login].
> CODESYS changes to the online mode.
► Select [Online] > [Reset origin] to reset the application.
> Application changes to the STOP state and is deleted.
> All variables are newly initialised
> PLC is reset to the original state.

9.5 Data transmission for series production

23577

For the series production, application data and stored data can be transferred to the PC and then transferred from the PC to further devices.

The data transmission takes place in two steps:

1 Data backup from the device to the PC

2 Distribution of the backed up data to the target devices

9.5.1 Transmission of the files with CODESYS

23579
23520

To transfer files between PC and device:

1 **Select file view**
- ► In the device tree: Double-click on symbol [Device (CR720S)]
- ► In the editor window: Select the [Files] tab.
- > The editor window shows the file structure on the PC on the left and on the device on the right

2 **Transfer file from PC to device**
- ► Highlight the file on the left
- ► Select device target directory on the right
- ► Star transfer using the [>>] button
- > The file is transferred to the device

3 **Transfer the file from the device to the PC**
- ► Highlight the file on the right
- ► Select PC target directory on the left
- ► Start the transfer using the [<<] button
- > The fle is transferred to the PC

9.5.2 Data transmission with TFTP

23580

With the aid of the program TFTP, files can be transferred.

Transfer file from device to PC:

IP address = address of the source device, e.g. 192.168.82.247
Source = source file on the device
Target = target file on the PC

Transfer file from PC to device:

```
tftp -i IP-Adresse PUT source target
```

IP address = address of the source device, e.g. 192.168.82.247
Source = source file on the PC
Target = target file on the device

Example:

```
tftp -i 192.168.82.247 PUT [Windows-Pfad]\ifmOS.ifm \os\ifmOS.ifm
```

9.5.3 Files for series production

23578

The following files must be transferred:

Data name / path	Description
apps	Folder
▪ standard.app	Application non-safe
▪ safe.app	Application safe
os	Folder
▪ ifmOS.ifm	ifmOS
cfg	Folder
▪ comconf.cfg	Communication configuration
▪ memconf.ifm	Memory configuration

The following file must be transferred according to the kind of application (retain data and free user data):

Data name / path	Description
retain	Folder
▪ standard.ret	Application retain non-safe
▪ standard.mb	Application memory bytes non-safe
▪ safe.ret	Application retain safe
▪ safe.mb	Application memory bytes safe
data	Folder
▪ *.*	Memory space for user-defined data

9.6 Display system information

14161

In the online mode the device tree displays the current values of the following system parameters:

Parameter	Description	Possible values
[IP Settings]	IP settings	--
▪ [IP Address]	IP address of the device	E.g. 192.168.0.100
▪ [IP Mask]	Subnet mask of the network	E.g. 255.255.255.0
▪ [Gateway Address]	IP address of the network gateway	E.g. 192.168.0.2
[Version Firmware]	Version of the installed firmware	E.g. V1.4.0
[Serial Number Device]	Serial number of the device	E.g. 1511AB019

To display the system information of the device:

► Establish connection between CODESYS and CR720S.
► Select [Online] > [Login].
► CODESYS changes to the online mode.
► In the device tree: Double-click on [System_Info]
► In the editor window: Select tab [Parameter].
> In the editor window: Table shows current values of the system parameters.

10 ifm function libraries

Contents

General ... 103
Library ifmCANopenManager.library ... 103
Library ifmDeviceCR0721.library ... 113
ifmFastInput.library ... 116
Library ifmIOcommon.library ... 125
Library ifmOutGroup ... 143
Library ifmOutHBridge ... 148
Library ifmOutPWM ... 153
Library ifmRawCAN.library ... 162

7034

This chapter contains the detailed description of the function libraries provided by ifm electronic for programming the device under CODESYS 3.5.

10.1 General

23828

General information about:

- → **Messages / diagnostic codes of the function blocks** (→ p. 181)
- → **ifm behaviour models for function blocks** (→ p. 183)

10.2 Library ifmCANopenManager.library

Contents

COP_GetNodeState ... 104
COP_SDOread ... 106
COP_SDOwrite ... 108
COP_SendNMT ... 110
NMT_SERVICE (ENUM) ... 112
NMT_STATES (ENUM) ... 112

18443

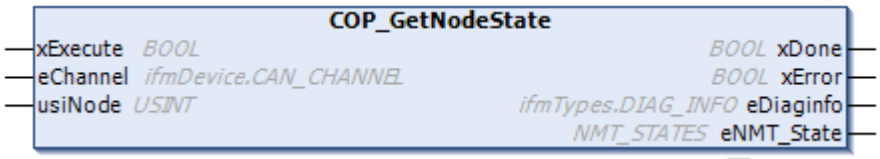
The library contains program blocks (POU) and data structures for the programming of the functionality of a CANopen Manager.

10.2.1 COP_GetNodeState

15956

Function block type: Function block (FB)
Behaviour model: EXECUTE

Library: ifmCANopenManager.library
Symbol in CODESYS:



Description

18445

The FB indicates the current state of a CANopen node.

Input parameter

18446

Parameter	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE ⇨ TRUE	FB is executed once
			Other	No impact on FB processing
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
usiNode	USINT	ID of the CANopen node	0	Local device
			1 ... 127	ID of the CANopen node

Output parameter

18447

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	▪ FB successfully executed ▪ FB can be called again
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
eNMT_State	NMT_ STATES	State of the CANopen node	→ **NMT_STATES (ENUM)** (→ p. 112)	

Diagnostic codes:

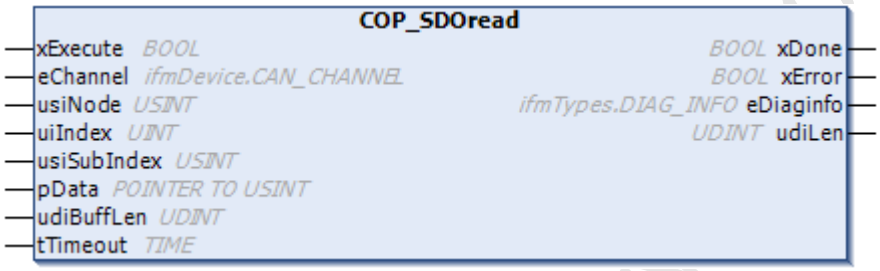
- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_BUSY — State: FB/Function is currently executed.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_BUS_OFF — Error: CAN interface is in the "BUS OFF" state
- ERR_COMMUNICATION — Error: no Connection to the network node or network node not available

10.2.2 COP_SDOread

18448

Function block type: Function block (FB)
Behaviour model: EXECUTE

Library: ifmCANopenManager.library
Symbol in CODESYS:



Description

7144

The FB reads the contents of a Service Data Object (SDO) and writes them into a buffer storage. The SDO is selected via the CAN interface, the ID of the CANopen node, as well as index and subindex of the object directory.

The CANopen node has to reply to the request of the FB within a period of time defined by the user.

Input parameters

19832

Parameters	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE ⇨ TRUE	FB is executed once
			Other	No impact on FB processing
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
usiNode	USINT	ID of the CANopen node	0	Local device
			1 ... 127	ID of the CANopen node
uiIndex	UINT	Index in the object directory		
usiSubIndex	USINT	Subindex of the index in the object directory		
pData	Pointer to USINT	Pointer on buffer storage		
udiBuffLen	UDINT	Size of the buffer storage (in byte)		
tTimeout	TIME	Max. response time	E.g. `T#25ms`	

Output parameters

11271

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	▪ FB successfully executed ▪ FB can be called again
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
udiLen	UDINT	Number of received bytes		

Diagnostic codes:

- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_CHAN_UNKNOWN — Error: Selected communication channel unknown / not configured
- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_BUFFER_OVERFLOW — Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR_INVALID_OBJ_ENTRY — Error: Object directory entry is invalid.
- ERR_TIMEOUT — Error: The maximum permissible execution time was exceeded. The action was not finished.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!

10.2.3 COP_SDOwrite

17128

Function block type: Function block (FB)
Behaviour model: EXECUTE

Library: ifmCANopenManager.library
Symbol in CODESYS:

Description

19833

The FB writes the contents of a Service Data Object (SDO). The SDO is selected via the CAN interface, the ID of the CANopen node, as well as index and subindex of the object directory.

Input parameters

7011

Parameters	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE ⇨ TRUE	FB is executed once
			Other	No impact on FB processing
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
usiNode	USINT	ID of the CANopen node	0	Local device
			1 ... 127	ID of the CANopen node
uiIndex	UINT	Index in the object directory		
usiSubIndex	USINT	Subindex of the index in the object directory		
pData	Pointer to USINT	Pointer on buffer storage		
udiLen	UDINT	Number of received bytes		
tTimeout	TIME	Max. response time	E.g. `T#25ms`	

Output parameters

7005

Parameters	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	▪ FB successfully executed ▪ FB can be called again
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	

Diagnostic codes:

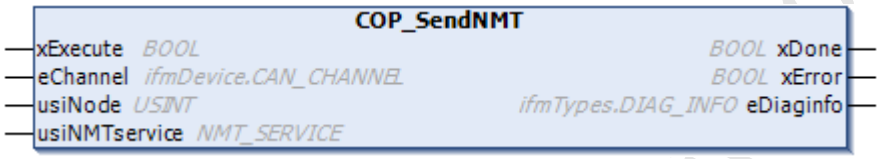
- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_CHAN_UNKNOWN — Error: Selected communication channel unknown / not configured
- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_BUFFER_OVERFLOW — Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR_INVALID_OBJ_ENTRY — Error: Object directory entry is invalid.
- ERR_TIMEOUT — Error: The maximum permissible execution time was exceeded. The action was not finished.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!

10.2.4 COP_SendNMT

7006

Function block type: Function block (FB)
Behaviour model: EXECUTE

Library: ifmCANopenManager.library
Symbol in CODESYS:



Description

7001

The FB sends a command for the control of a CANopen node.

Input parameter

6981

Parameter	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE ⇨ TRUE	FB is executed once
			Other	No impact on FB processing
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
usiNode	USINT	ID of the CANopen node	0	Local device
			1 ... 127	ID of the CANopen node
usiNMTservice	NMT_ SERVICE	Command for the control of a CANopen node	→ **NMT_SERVICE (ENUM)** (→ p. 112)	

Output parameters

7147

Parameters	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	▪ FB successfully executed ▪ FB can be called again
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	

Diagnostic codes:

- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_CHAN_UNKNOWN — Error: Selected communication channel unknown / not configured
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!

10.2.5 NMT_SERVICE (ENUM)

7132

Name	Description	Possible values		Data type	Value
NMT_SERVICE	Command for the control of a CANopen node	INIT_NODE	Initiate CAN node	INT	0
		SET_PRE_OPERATIONAL	Set preoperational state	INT	1
		SET_OPERATIONAL	Set operational state	INT	2
		RESET_NODE	Reset CAN node	INT	3
		RESET_COMM	Reset communication	INT	4
		STOP_NODE	Stop CAN node	INT	5

10.2.6 NMT_STATES (ENUM)

18496

Name	Description	Possible values		Data type	Value
NMT_STATES	State of the CAN network	INIT	Initialisation	INT	0
		PREOP	Preopertional	INT	1
		OPERATIONAL	Operational	INT	2
		STOP	STOP	INT	3
		NOT_AVAILABLE	Not available	INT	4
		UNKNOWN	Unknown	INT	5

10.3 Library ifmDeviceCR0721.library

Contents

CAN_BAUDRATE (ENUM) 114
CAN_CHANNEL (ENUM) 114
CANconstants (GVL) 114
SysInfo (GVL) 114
SysInfoStruct (STRUCT) 115
LED_COLOUR (ENUM) 115
LED_FLASH_FREQ (ENUM) 115

23255

The library contains all device-specific data structures, enumeration types, global variables and constants.

10.3.1 CAN_BAUDRATE (ENUM)

23253

Name	Description	Possible values		Data type	Value
CAN baud rate	Data transmission rate of the CAN interface	KBAUD_20	20 kilobaud	INT	20
		KBAUD_33	33.3 kilobaud	INT	33
		KBAUD_50	50 kilobaud	INT	50
		KBAUD_83	83.3 kilobaud	INT	83
		KBAUD_100	100 kilobaud	INT	100
		KBAUD_125	125 kilobaud	INT	125
		KBAUD_250	250 kilobaud	INT	250
		KBAUD_500	500 kilobaud	INT	500
		KBAUD_800	800 kilobaud	INT	800
		KBAUD_1000	1000 kilobaud	INT	1000

10.3.2 CAN_CHANNEL (ENUM)

17131

Name	Description	Possible values		Data type	Value
CAN_CHANNEL	Identifier of the CAN Interface	CHAN_0	CAN interface 0	INT	0
		CHAN_1	CAN interface 1	INT	1
		CHAN_2	CAN interface 2	INT	2
		CHAN_3	CAN interface 3	INT	3

10.3.3 CANconstants (GVL)

20936

Name	Description	Data type	Value
usiNumberCANitf	Number of the CAN interfaces of the devices	UINT	4

10.3.4 SysInfo (GVL)

20937

Name	Description	Data type	Value
usiNumberOfSysInfo	Number of system components of the device	USINT	8
aSysInfoList	Variable with list of the system components (→ **aSysInfoList (GVL)**)	ARRAY[0..8] OF **SysInfoStruct (STRUCT)** (→ p. 115)	

10.3.5 SysInfoStruct (STRUCT)

21317

Designation	Data type	Description	Possible values
eInfoType	INFO_TYPE	System component	E.g. FIRMWARE_DEVICE
sValue	STRING (255)	Value of the system component	E.g. 3.1
sName	STRING (32)	Name of the system component	E.g. FW Device

10.3.6 LED_COLOUR (ENUM)

23232

Name	Description	Possible values		Data type	Value
LED_COLOUR	Colour of the LED (RGB code)	BLACK (OFF)	Off	UINT	0x00 0000
		WHITE	White	UINT	0xFF FFFF
		RED	Red	UINT	0xFF 0000
		GREEN	Green	UINT	0x00 FF00
		BLUE	Blue	UINT	0x00 00FF
		YELLOW	Yellow	UINT	0xFF FF00
		MAGENTA	Magenta	UINT	0xFF 00FF
		CYAN	Cyan	UINT	0x00 FFFF

10.3.7 LED_FLASH_FREQ (ENUM)

23233

Name	Description	Possible values		Data type	Value
LED_FLASH_FREQ	Flashing frequency of the status LED	FRQ_0Hz	off	INT	0
		FRQ_05Hz	0.5 Hz	INT	1
		FRQ_1Hz	1 Hz	INT	2
		FRQ_2Hz	2Hz	INT	4
		FRQ_5Hz	5 Hz	INT	7
		FRQ_10Hz	10Hz	INT	8

10.4 ifmFastInput.library

Contents

FastCount 117
IncEncoder 119
Period 121
COUNT_DIRECTION (ENUM) 123
ENCODER_RESOLUTION (ENUM) 123
FREQ_SENSE_PERIODS (ENUM) 123
MODE_FAST_COUNT (ENUM) 123
MODE_INC_ENCODER (ENUM) 124
MODE_PERIOD (ENUM) 124

23257

The library contains function blocks (POU) and enumeration types to control the quick inputs of the device.

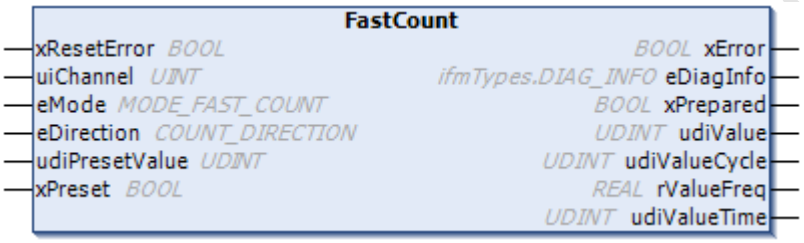
10.4.1 FastCount

23262

Function block type: Function block (FB)

Library: ifmIFastInput.library

Symbol in CODESYS:



Description

23259

The FB functions as a counter block for pulses on fast input channels.

Input parameters

23260

Parameters	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE TRUE	When switching from FALSE ⇨ TRUE: Reset request to the lower level system
uiChannel	UINT	Input channel	Group + channel → Data sheet → **Note on wiring** (→ p. 29)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_FAST_COUNT	Operating mode of the input channel	→ **MODE_FAST_COUNT (ENUM)** (→ p. 123)	
eDirection	COUNT_DIRECTION	Counting direction	→ **COUNT_DIRECTION (ENUM)** (→ p. 123)	
udiPresetValue	UDINT	Preset counter value	permissible = 0...4 294 967 295	
xPreset	BOOL	Changeover switch: counter function active / adopt preset counter value	FALSE	counter active; the number of counted pulses is issued to udiValue.
			TRUE	The preset counter value is adopted; udiValue = udiPresetValue

Output parameters

23261

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
udiValue	UDINT	Counter value; number of detected pulses	permissible = 0...4 294 967 295	
udiValueCycle	UDINT	Cycle time of the input signal in [µs]		
rValueFreq	REAL	frequency of the input signal in [Hz]		
udiValueTime	UDINT	Time elapsed since the last edge evaluation in [µs]	0...4 294 967 295	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_TIMING — reserved
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.
- DIAG_INTERNAL — Internal system error.
- DIAG_ACCESS — FB/Function cannot access the required resource; Resource is blocked by another task.
- DIAG_CHANGEOVER_TIME — Minimum changeover time for the highside-lowside selection of the drivers has not yet expired.
- DIAG_SLOW_SIGNAL — Input signal is too slow for the measurement.

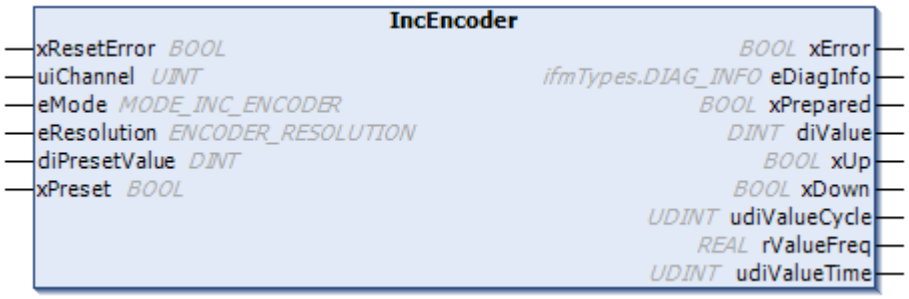
10.4.2 IncEncoder

23298

Function block type: Function block (FB)

Library: ifmIFastInput.library

Symbol in CODESYS:



Description

23299

The FB is used to configure and to operate a digital input pair to record and count incremental encoder pulses.

Two frequency inputs constitute the input pair (channel A and channel B) that is configured and evaluated via the FB.

Behaviour at the counter limits

If the applicable value range is exceeded, the output switches to the minimum value of the applicable area. (= overflow)

If the applicable value range is not reached, output switches to the maximum value of the applicable area. (= outside range)

Input parameter

23300

Parameters	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE TRUE	When switching from FALSE ⇨ TRUE: Reset request to the lower level system
uiChannel	UINT	1. Input channel (channel A) of the pair of input channels	Group + channel → Data sheet → **Note on wiring** (→ p. 29)	
			Examples:	
			703	Group 7 + channel 3
			1203	Group 12 + channel 3
eMode	MODE_INC_ENCODER	Operating mode of the input channel	→ **MODE_INC_ENCODER (ENUM)** (→ p. 124)	
eResolution	ENCODER_RESOLUTION	Resolution / encoder mode	→ **ENCODER_RESOLUTION (ENUM)** (→ p. 123)	
diPresetValue	DINT	Preset counter value	-2 147 483 648...2 147 483 647	
xPreset	BOOL	Changeover switch: counter function active / adopt preset counter value	FALSE	counter active; the number of counted pulses is issued to udiValue.
			TRUE	The preset counter value is adopted; udiValue = udiPresetValue

Output parameters

23301

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
diValue	DINT	Counter value; number of detected pulses	permissible = - 2 147 483 648...2 147 483 647	
xUp	BOOL	Code sequence upwards	FALSE	No count-up since the last call up
			TRUE	Count-up or overflow since that last call-up
xDown	BOOL	Code sequence downwards	FALSE	No count-down since the last call-up
			TRUE	Count-up or underflow since the last call-up
udiValueCycle	UDINT	Cycle time of the input signal in [µs]		
rValueFreq	REAL	frequency of the input signal in [Hz]		
udiValueTime	UDINT	Time elapsed since the last edge evaluation in [µs]	0...4 294 967 295	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_TIMING — reserved
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.
- DIAG_INTERNAL — Internal system error.
- DIAG_ACCESS — FB/Function cannot access the required resource; Resource is blocked by another task.
- DIAG_CHANGEOVER_TIME — Minimum changeover time for the highside-lowside selection of the drivers has not yet expired.
- DIAG_SLOW_SIGNAL — Input signal is too slow for the measurement.

10.4.3 Period

23313

Function block type: Function block (FB)

Library: ifmIFastInput.library

Symbol in CODESYS:

Description

23314

TheFB is used to configure and to operate an input channel or a pair of input channels to detect and count pulses.

In the operating modes IN_PHASE_CSI and IN_PHASE_CSO (to be set at the eMode function block input), a phase measurement is carried out on one input channel pair. The input channel pair is defined by indicating the channel with the even number of the input channel pair (channel A) at the input uiChannel.

In the other operating modes, a signal evaluation is carried out at the input channel defined at the uiChannel input.

Input parameters

23315

Parameters	Data type	Description	Possible values
xResetError	BOOL	Reset request for an occurring error	FALSE / TRUE: When switching from FALSE ⇨ TRUE: Reset request to the lower level system
uiChannel	UINT	Input channel	Group + channel → Data sheet → **Note on wiring** (→ p. 29) Examples: 403: Group 4 + channel 3 502: Group 5 + channel 2
eMode	MODE_PERIOD	Operating mode of the input channel	→ **MODE_PERIOD (ENUM)** (→ p. 124)
ePeriod	FREQ_SENSE_PERIODS	Number of pulse periods for averaging	→ **FREQ_SENSE_PERIODS (ENUM)** (→ p. 123)
udiTimebase	UDINT	Time base for frequency calculation in [ms] Only used in eMode: ▪ IN_FREQUENCY_CSI ▪ IN_FREQUENCY_CSO	→ **MODE_PERIOD (ENUM)** (→ p. 124)

Output parameters

23316

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
udiValueCycle	UDINT	Cycle time of the input signal in [µs]		
rValueFreq	REAL	frequency of the input signal in [Hz]		
udiValueTime	UDINT	Time elapsed since the last edge evaluation in [µs]	0...4 294 967 295	
uiValueRatio	UINT	Depends on the mode that is set in the eMode input. Pulse/pause ratio of the input signal in [‰] at: ▪ IN_PERIOD_RATIO_CSI ▪ IN_PERIOD_RATIO_CSO		
		Phase shift of the input signal at the B channel to the signal at the A channel in [°] ▪ IN_PHASE_CSI ▪ IN_PHASE_CSO		

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- STAT_PREPARING — State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_EXCEEDED_RANGE — Error: The value exceeds the value range of its data type.
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_TIMING — reserved
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.
- DIAG_INTERNAL — Internal system error.
- DIAG_ACCESS — FB/Function cannot access the required resource; Resource is blocked by another task.
- DIAG_CHANGEOVER_TIME — Minimum changeover time for the highside-lowside selection of the drivers has not yet expired.
- DIAG_SLOW_SIGNAL — Input signal is too slow for the measurement.

10.4.4 COUNT_DIRECTION (ENUM)

23267

Name	Description	Possible values	
COUNT_DIRECTION	Counting direction	COUNT_OFF	Counting function off
		COUNT_UP	Counting function up
		COUNT_DOWN	Counting function down

10.4.5 ENCODER_RESOLUTION (ENUM)

23269

Name	Description	Possible values	
ENCODER_RESOLUTION	Resolution	FULL_PERIOD	Counts each rising edge on one channel (A)
		HALF_PERIOD	Counts each rising and falling edge on one channel (A)
		EVERY_EDGE	Counts each rising and falling edge on al channels (A and B)

10.4.6 FREQ_SENSE_PERIODS (ENUM)

23270

Name	Description	Possible values	
FREQ_SENSE_PERIODS	Number of clock periods for the averaging	PERIODS_n mit n = 1	No averaging
		PERIODS_n mit n = 2...16	Averaging via n periods

10.4.7 MODE_FAST_COUNT (ENUM)

23264

Name	Description	Possible values	
MODE_FAST_COUNT	Operating mode of the inputs	UNCHANGED	Setting remains unchanged
		IN_COUNT_CSI	Input to count fast signal edges; CSI
		IN_COUNT_CSO	Input to count fast signal edges; CSO
		MONITOR	▪ Only output data will be updated. ▪ Values, configurations ans process data are not written. ▪ For applications that are not owners of the resource.

10.4.8 MODE_INC_ENCODER (ENUM)

23271

Name	Description	Possible values	
MOTE_INC_ENCODER	Operating mode of the input	UNCHANGED	Setting remains unchanged
		IN_INC_ENCODER_CSI	Input for the evaluation of an incremental encoder, channel A; CSI
		IN_INC_ENCODER_CSO	Input for the evaluation of an incremental encoder, channel A; CSO
		MONITOR	▪ Only output data will be updated. ▪ Values, configurations ans process data are not written. ▪ For applications that are not owners of the resource.

10.4.9 MODE_PERIOD (ENUM)

23272

Name	Description	Possible values	
MODE_PERIOD	Operating mode of the period input	UNCHANGED	Setting remains unchanged
		IN_FREQUENCY_CSI	Input for frequency measurement; CSI
		IN_FREQUENCY_CSO	Input for frequency measurement; CSO
		IN_PERIOD_RATIO_CSI	Input for absolute and ratiometric period measurement; CSI
		IN_PERIOD_RATIO_CSO	Input for absolute and ratiometric period measurement; CSO
		IN_PHASE_CSI	Input pair for phase measurement, CSI
		IN_PHASE_CSO	Input pair for phase measurement, CSO
		MONITOR	▪ Only output data will be updated. ▪ Values, configurations ans process data are not written. ▪ For applications that are not owners of the resource.

10.5 Library ifmIOcommon.library

Contents

Input ... 126
Output ... 129
SetLED ... 132
SupplySwitch ... 134
SystemSupply ... 136
Temperature ... 138
FILTER_INPUT (ENUM) ... 140
FILTER_OUTPUT (ENUM) ... 140
MODE_INPUT (ENUM) ... 141
MODE_OUTPUT (ENUM) ... 142
SYS_VOLTAGE_CHANNEL (ENUM) ... 142

21286

The library contains program blocks (POU) and enumeration types for the control of the inputs and outputs of the device.

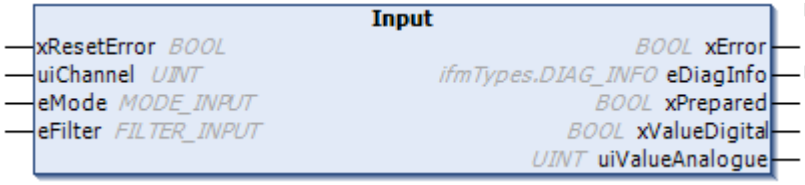
10.5.1 Input

23155

Function block type: Function block (FB)

Library: ifmIOcommon.library

Symbol in CODESYS:



Description

23164

The FB is used to configure and read a digital or analogue input channel.

Filter:
The input signal can be changed with a digital low-pass filter.
Configure the filter via the input eFilter.

Input parameters

23157

Parameters	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE TRUE	When switching from FALSE ⇨ TRUE: Reset request to the lower level system
uiChannel	UINT	Input channel	Group + channel → Data sheet → **Note on wiring** (→ p. 29)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_INPUT	Operating mode of the input channel	→ **MODE_INPUT (ENUM)** (→ p. 141)	
eFilter	FILTER_INPUT	Filter definition of the input channel	→ **FILTER_INPUT (ENUM)** (→ p. 140)	

Output parameters

23158

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
xValueDigital	BOOL	Logical state of the input in digital operating mode In case of analogue operation, the output is FALSE	FALSE	Low Level
			TRUE	High Level
uiValueAnalogue	UINT	Measured input value in analogue operating mode. The interpretation of the input value depends on the setting at the eMode input. **MODE_INPUT (ENUM)** (→ p. 141)	permissible = 0...65 535	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_SHORT_CIRCUIT — Error: Short circuit with GND or VBBx.
- ERR_OPEN_CIRCUIT — Error: Open circuit detected. Possible cause: Wire break.
- ERR_OVERLOAD_CURRENT — Error: Maximum current exceeded.
- ERR_STUCK_AT — Error: Signal is frozen.
- ERR_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- ERR_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- ERR_OVERVOLTAGE — Error: Maximum signal voltage exceeded.
- ERR_UNDERVOLTAGE — Error: Minimum signal voltage not reached.
- ERR_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- ERR_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.
- DIAG_INTERNAL — Internal system error.
- DIAG_OPEN_CIRCUIT — Open circuit detected. Possible cause: Wire break.
- DIAG_SHORT_CIRCUIT — Error: Short circuit with GND or VBBx.
- DIAG_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_OVERLOAD_CURRENT — Maximum current exceeded.
- DIAG_STUCK_AT — Error: Signal is frozen.
- DIAG_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- DIAG_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- DIAG_OVERVOLTAGE — Maximum signal voltage exceeded.
- DIAG_UNDERVOLTAGE — Error: Minimum signal voltage not reached.
- DIAG_NO_CALIB — The selected resource has no valid calibration.
  The displayed values are maybe faulty.
- DIAG_ACCESS — FB/Function cannot access the required resource; Resource is blocked by another task.
- DIAG_CHANGEOVER_TIME — Minimum changeover time for the highside-lowside selection of the drivers has not yet expired.

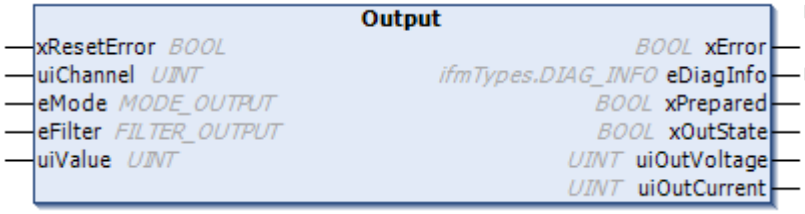
10.5.2 Output

23161

Function block type: Function block (FB)

Library: ifmIOcommon.library

Symbol in CODESYS:



Description

23206

The FB is used to configure and control a digital or analogue output channel.

Input parameters

23162

Parameter	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE TRUE	When switching from FALSE ⇨ TRUE: Reset request to the lower level system
uiChannel	UINT	Output channel	Group + channel → Data sheet → **Note on wiring** (→ p. 29)	
			Examples:	
			703	Group 7 + channel 3
			1203	Group 12 + channel 3
eMode	MODE_OUTPUT	Operating type of the output channel	→ **MODE_OUTPUT (ENUM)** (→ p. 142)	
eFilter	FILTER_ OUTPUT	Filter definition of the output channel	→ **FILTER_OUTPUT (ENUM)** (→ p. 140)	
uiValue	UINT	Value that is to be written to the output		
		In the digital mode or sensor supply mode; if setting at the eMode input = ▪ OUT_DIGITAL_CSI ▪ OUT_DIGITAL_CSO ▪ OUT_SENSOR_05 ▪ OUT_SENSOR_10	FALSE TRUE	Output deactivated Output activated
		In analogue mode; if setting at the eMode input = ▪ OUT_ANALOGUE_10 Values indicated in [mV]	0...10 000	

Output parameters

23163

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
xOutState	BOOL	Return value activation state of the selected output (!) The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output is deactivated
			TRUE	Output is activated
uiOutVoltage	UINT	Current output voltage in [mV] Only available for the operating modes "analogue" and "sensor"	0	Operating mode neither "analogue" nor "sensor"
			≠ 0	Operating mode "analogue" or "sensor"
uiOutCurrent	UINT	Present output current in [mA] Not available for the operating types OUT_DIGITAL_CSI and OUT_ANALOGUE_10	available = 0...final value of the measuring range	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_SHORT_CIRCUIT — Error: Short circuit with GND or VBBx.
- ERR_STUCK_AT — Error: Signal is frozen.
- ERR_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- ERR_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- ERR_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- ERR_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.
- DIAG_INTERNAL — Internal system error.
- DIAG_ACCESS — FB/Function cannot access the required resource; Resource is blocked by another task.
- DIAG_CHANGEOVER_TIME — Minimum changeover time for the highside-lowside selection of the drivers has not yet expired.
- DIAG_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_STUCK_AT — Error: Signal is frozen.
- DIAG_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- DIAG_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- DIAG_SHORT_CIRCUIT — Error: Short circuit with GND or VBBx.
- DIAG_NO_CALIB — The selected resource has no valid calibration.
  The displayed values are maybe faulty.

10.5.3 SetLED

23220

Function block type: Function block (FB)

Library: ifmIOcommon.library

Symbol in CODESYS:

Description

23221

The FB is used to configure and control an LED.

Input parameters

23222

Parameters	Data type	Description	Possible values	
uiChannel	UINT	Output channel of the ED	0...3	Device LED APP 0...3
eColour1	ENUM	LED colour status 1	→ **LED_COLOUR (ENUM)** (→ p. 115)	
eColour2	ENUM	LED colour status 0	→ **LED_COLOUR (ENUM)** (→ p. 115)	
eFrequency	ENUM	Flashing frequency of the status LED	→ **LED_FLASH_FREQ (ENUM)** (→ p. 115)	

Output parameters

23223

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_BUSY — State: FB/Function is currently executed.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT_PREPARING — State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- ERR_INVALID_FREQUENCY — Error: Unsupported frequency.
- ERR_INVALID_COLOUR — Error: Unsupported colour.
- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_INSTANCE — Error: Instance is ZERO or invalid.
- ERR_ACCESS — Error: FB/Funktion cannot access the required resource; Resource is blocked by another task.
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.

10.5.4 SupplySwitch

8034

Function block type: Function block (FB)

Library: ifmIOcommon.library

Symbol in CODESYS:

Description

23252

The FB stops all running applications and switches off the voltage supply latching (terminal 30) in order to shut down the device safely.

The voltage supply latching is only deactivated if the following conditions are met:

- Voltage VBB15 < 5.5 V (undervoltage)

The separation from the VBB30 takes place when all IEC tasks are finished.

Input parameters

23153

Parameters	Data type	Description	Possible values	
xSwitchOff	BOOL	Deactivate latching switch of the device	FALSE	No action
			TRUE	Request deactivation of the latching switch

Output parameters

23154

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.
- DIAG_INTERNAL — Internal system error.
- DIAG_ACCESS — FB/Function cannot access the required resource; Resource is blocked by another task.

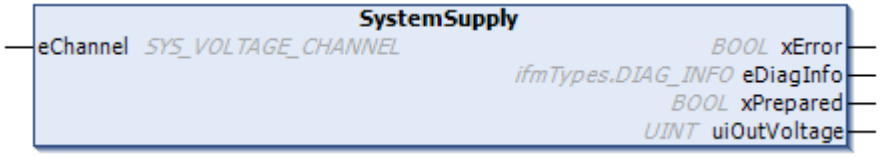
10.5.5 SystemSupply

23242

Function block type: Function block (FB)

Library: ifmIOcommon.library

Symbol in CODESYS:



Description

23237

The FB indicates the value of the system voltage.

Input parameters

23238

Parameter	Data type	Description	Possible values
eChannel	ENUM	System voltage channel	→ SYS_VOLTAGE_CHANNEL (ENUM) (→ p. 142)

Output parameters

23239

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
uiOutVoltage	UINT	Current output voltage of the selected system voltage channel in [mV]	permissible = 0...maximum operating voltage	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

▪ STAT_DONE	State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
▪ STAT_PREPARING	State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
▪ ERR_INVALID_VALUE	Error: At least one input parameter is invalid or outside the value range.
▪ ERR_INTERNAL	Error: Internal system error ► Contact the ifm Service Center!
▪ ERR_UNDEFINED	Error: Unknown error ► Contact the ifm Service Center!
▪ ERR_SHORT_CIRCUIT	Error: Short circuit with GND or VBBx.
▪ ERR_OPEN_CIRCUIT	Error: Open circuit detected. Possible cause: Wire break.
▪ ERR_OVERLOAD_CURRENT	Error: Maximum current exceeded.
▪ ERR_STUCK_AT	Error: Signal is frozen.
▪ ERR_STUCK_AT_HIGH	Error: Signal frozen, signal state high.
▪ ERR_STUCK_AT_LOW	Error: Signal frozen, signal state low.
▪ ERR_OVERVOLTAGE	Error: Maximum signal voltage exceeded.
▪ ERR_UNDERVOLTAGE	Error: Minimum signal voltage not reached.
▪ ERR_UNDERVOLTAGE_VBBX	▪ For inputs: Error: Reference voltage not reached. ▪ For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
▪ ERR_OVERVOLTAGE_VBBX	▪ For inputs: Error: Reference voltage exceeded. ▪ For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
▪ ERR_SHORT_CIRCUIT	Error: Short circuit with GND or VBBx.
▪ ERR_OPEN_CIRCUIT	Error: Open circuit detected. Possible cause: Wire break.
▪ DIAG_INVALID_VALUE	At least one input parameter is invalid or exceeds the permissible area.
▪ DIAG_INTERNAL	Internal system error.
▪ DIAG_OPEN_CIRCUIT	Open circuit detected. Possible cause: Wire break.
▪ DIAG_SHORT_CIRCUIT	Error: Short circuit with GND or VBBx.
▪ DIAG_UNDERVOLTAGE_VBBX	▪ For inputs: Error: Reference voltage not reached. ▪ For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
▪ DIAG_OVERVOLTAGE_VBBX	▪ For inputs: Error: Reference voltage exceeded. ▪ For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
▪ DIAG_OVERLOAD_CURRENT	Maximum current exceeded.
▪ DIAG_STUCK_AT	Error: Signal is frozen.
▪ DIAG_STUCK_AT_HIGH	Error: Signal frozen, signal state high.
▪ DIAG_STUCK_AT_LOW	Error: Signal frozen, signal state low.
▪ DIAG_OVERVOLTAGE	Maximum signal voltage exceeded.
▪ DIAG_UNDERVOLTAGE	Error: Minimum signal voltage not reached.
▪ DIAG_NO_CALIB	The selected resource has no valid calibration. The displayed values are maybe faulty.

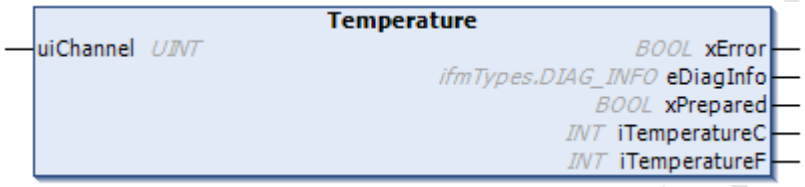
10.5.6 Temperature

23245

Function block type: Function block (FB)

Library: ifmIOcommon.library

Symbol in CODESYS:



Description

23247

The FB indicates the value of the system temperature.

Input parameters

23246

Parameters	Data type	Description	Possible values	
uiChannel	UINT	Input channel	Group + channel → Data sheet → **Note on wiring** (→ p. 29)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2

Output parameters

23248

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
iTemperatureC	INT	Measured temperature in [°C]	e.g. 35	
iTemperatureF	INT	Measured temperature in [°F]	e.g. 95	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- STAT_PREPARING — State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- DIAG_INTERNAL — Internal system error.
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.

10.5.7 FILTER_INPUT (ENUM)

23166

The input signal can be changed with a digital low-pass filter.

For the output signal of the function bloc, the delay time is changed to the input signal change by the filter. This applies both to the switch-on and the switch-off pulse.

Name	Description	Possible values	Digital signal delay	Analogue signal delay
FILTER_INPUT	Valid filters for inputs of the FBs	UNCHANGED	No change of settings	
		FILTER_0	0.6 ms (no digital low-pass filter is set)	1.7 ms (no digital low-pass filter is set)
		FILTER_1	0.9 ms	3.3 ms
		FILTER_2	2.1 ms	7.0 ms
		FILTER_3	4.0 ms	14.1 ms
		FILTER_4	7.6 ms	28.9 ms
		FILTER_5	15.2 ms	58.4 ms
		FILTER_6	30.8 ms	117.2 ms
		FILTER_7	61.6 ms	235.2 ms
		FILTER_8	123.2 ms	470.8 ms
		FILTER_9	246.4 ms	942.4 ms
		FILTER_10	493.2 ms	1885.6 ms
		FILTER_11	986.4 ms	3772.0 ms
		FILTER_12	1972.4 ms	7544.4 ms

10.5.8 FILTER_OUTPUT (ENUM)

23167

Filter setting for the current measurement of an output.

The signal of the current measurement is damped via a first-order low-pass filter.

Name	Description	Possible values	
FILTER_OUTPUT	Valid filter for the outputs of the FBs	UNCHANGED	No change of settings
		FILTER_0	1.7 ms
		FILTER_1	1.8 ms
		FILTER_2	2.4 ms
		FILTER_3	3.9 ms
		FILTER_4	7.4 ms
		FILTER_5	14.7 ms
		FILTER_6	29.3 ms
		FILTER_7	58.8 ms
		FILTER_8	117.7 ms
		FILTER_9	235.6 ms
		FILTER_10	471.4 ms
		FILTER_11	943.0 ms
		FILTER_12	1886.1 ms

10.5.9 MODE_INPUT (ENUM)

23168

Name	Description	Possible values	
MODE_INPUT	Operating mode of the inputs	UNCHANGED	Preset mode is maintained
		IN_DIGITAL_CSI	Input for analogue value measurement and digital evaluation without diagnostics; CSI
		IN_DIGITAL_CSI_NAMUR	Input for analogue value measurement and digital evaluation with NAMUR-capable diagnostics; CSI
		IN_VOLTAGE_10	Input for analogue current measurement 0...10 V; CSI
		IN_VOLTAGE_32	Input for analogue current measurement 0...32 V; CSI
		IN_VOLTAGE_RATIO	Input for ratiometric current measurement in relation to VBB30; CSI
		IN_CURRENT_CSI	Input for current measurement 0...20 mA; CSI
		IN_RESISTOR	Input for resistance measurement; CSO
		IN_DIGITAL_CSO	Input for analogue value measurement and digital evaluation without diagnostics; CSO
		IN_DIGITAL_CSO_DIAG	Input for analogue value measurement and digital evaluation with diagnostics similar to NAMUR; CSO
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.

10.5.10 MODE_OUTPUT (ENUM)

23169

Name	Description	Possible values	
MODE_OUTPUT	Operating mode of the outputs	UNCHANGED	Preset mode is maintained
		OUT_DIGITAL_CSI	Digital output without diagnostics; CSI
		OUT_DIGITAL_CSO	Digital output without diagnostics; CSO
		OUT_ANALOGUE_10	Analogue output to generate a selectable voltage 0...10 V without diagnostics. Generated with the help of a filtered PWM signal. CSO
		OUT_SENSOR_05	Output with fixed output voltage 5 V for the sensor supply without diagnostics and without protection. CSO
		OUT_SENSOR_10	Output with fixed output voltage 10 V for the sensor supply without diagnostics and without protection. CSO
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.

10.5.11 SYS_VOLTAGE_CHANNEL (ENUM)

23170

Name	Description	Possible values	
SYS_VOLTAGE_CHANNEL	List of all available system voltages.	VBB30	Terminal 30 system voltage
		VBB15	Terminal 15 system voltage of the ignition switch

10.6 Library ifmOutGroup

Contents

OutputGroup 144
FILTER_OUTPUT_GROUP (ENUM) 147
MODE_OUTPUT_GROUP (ENUM) 147

23341

The library contains function blocks (POU) to control extended output functions.

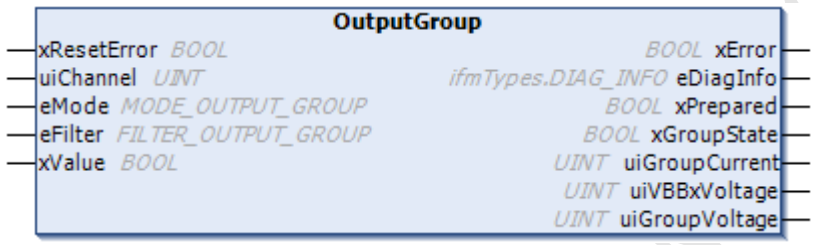
10.6.1 OutputGroup

23326

Function block type: Function block (FB)

Library: ifmIOutGroup.library

Symbol in CODESYS:



Description

23327

The FB controls the activation status of an output group and provides diagnostic information about the group and the connected outputs. Using the FB, an output group including the corresponding outputs can be switched on or off.

Input parameters

23328

Parameter	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE TRUE	When switching from FALSE ⇨ TRUE: Reset request to the lower level system
uiChannel	UINT	Output channel group	→ Data sheet → **Note on wiring** (→ p. 29)	
eMode	MODE_ OUTPUT_ GROUP	Operating type of the output channel group	→ **MODE_OUTPUT_GROUP (ENUM)** (→ p. 147)	
eFilter	FILTER_ OUTPUT_ GROUP	Defines the limit frequency of the output filter	→ **FILTER_OUTPUT_GROUP (ENUM)** (→ p. 147)	
xValue	BOOL	Activation requirement for the output group	FALSE TRUE	Deactivate output group Activate output group

23337

The error of an output group will only be reset if all corresponding outputs are error-free.

Output parameters

23329

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
xGroupState	BOOL	Return value activation state of the selected output group (!) The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output group is deactivated
			TRUE	Output value is activated
uiGroupCurrent	UINT	Measured output current of the entire group in [mA]	available = 0...final value of the measuring range	
uiVBBxVoltage	UINT	Measured voltage before the group switch in [mV]	available = 0...final value of the measuring range	
uiGroupVoltage	UINT	Measured voltage after the group switch in [mV]	available = 0...final value of the measuring range	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- ERR_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- ERR_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- ERR_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.
- DIAG_INTERNAL — Internal system error.
- DIAG_ACCESS — FB/Function cannot access the required resource; Resource is blocked by another task.
- DIAG_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- DIAG_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- DIAG_NO_CALIB — The selected resource has no valid calibration.
  The displayed values are maybe faulty.
- DIAG_OVERLOAD_CURRENT — Maximum current exceeded.
- ERR_OVERLOAD_CURRENT — Error: Maximum current exceeded.
- DIAG_AT_GROUP_OUTPUT — At least one output of the output group is in an error state.
- ERR_AT_GROUP_OUTPUT — Error: At least one output of the output group is in an error state.

10.6.2 FILTER_OUTPUT_GROUP (ENUM)

23338

Filter setting for voltage measurement in an output group.
The signal of the voltage measurement is damped via a first-order low-pass filter.

Name	Description	Possible values	
FILTER_OUTPUT_GROUP		UNCHANGED	No change of settings
		FILTER_0	1.7 ms
		FILTER_1	1.8 ms
		FILTER_2	2.4 ms
		FILTER_3	3.9 ms
		FILTER_4	7.4 ms
		FILTER_5	14.7 ms
		FILTER_6	29.3 ms
		FILTER_7	58.8 ms
		FILTER_8	117.7 ms
		FILTER_9	235.6 ms
		FILTER_10	471.4 ms
		FILTER_11	943.0 ms
		FILTER_12	1886.1 ms

10.6.3 MODE_OUTPUT_GROUP (ENUM)

23270

Name	Description	Possible values	
MODE_OUTPUT_GROUP	Operating type of the output group	UNCHANGED	Setting remains unchanged
		OUT_DIGITAL_CSO	Digital output without diagnostics and without protection; CSO
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.

10.7 Library ifmOutHBridge

Contents

HBridge 149
MODE_BRAKE (ENUM) 152

23467

The library contains function blocks (POU) to control extended output functions via an HBridge.

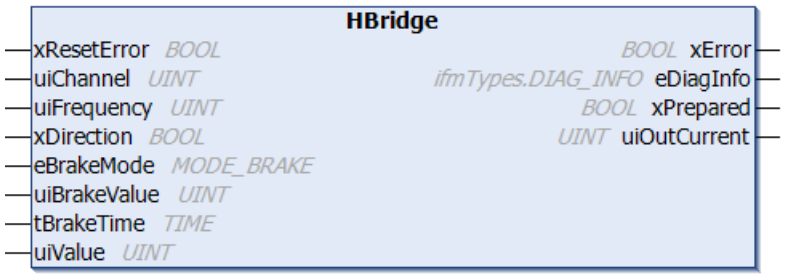
10.7.1 HBridge

23469

Function block type: Function block (FB)

Library: ifmIOutHBridge.library

Symbol in CODESYS:



Description

23470

The FB configures and controls a pair of output channels in the "HBridge" operating type to control a motor.

Input parameters

23471

Parameter	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE TRUE	When switching from FALSE ⇨ TRUE: Reset request to the lower level system
uiChannel	UINT	1. Output channel (channel A) of the output channel pair	Group + channel → Data sheet → **Note on wiring** (→ p. 29)	
			Examples:	
			703	Group 7 + channel 3
			1203	Group 12 + channel 3
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet	
xDirection	BOOL	The direction in which the current flows via the bridge connections. Determines the direction of rotation of the connected motor.	FALSE	PWM Current Sourcing (CSO) is on channel A
			TRUE	PWM Current Sourcing (CSO) is on channel B
eBrakeMode	MODE_BRAKE	Brake mode that applies when the direction of rotation is changed or when stopping	→ **MODE_BRAKE (ENUM)** (→ p. 152)	
eBrakeValue	UINT	Pulse/pause ration of the PWM output signal at the corresponding current sinking output of the bridge in [‰] The input is only relevant in the eBrakeModes that end with "_DYNAMIC" (= dynamic brake).	permissible = 0...1	
tBrakeTime	TIME	Indicates the braking time for the current sinking side of the bridge The input is only relevant in eBrakeModes ending with "_BTIME".	permissible = 0...1 h	
uiValue	UNIT	Pulse/pause ration of the PWM output signal in [‰]	permissible = 0...1	

Output parameters

23472

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
uiOutCurrent	UINT	Measured current at the PWM output during normal operation in [mA] When braking, uiOutCurrent is = 0 because no regular current exists in the lowside path.	available = 0...final value of the measuring range	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- STAT_PREPARING — State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_SHORT_CIRCUIT — Error: Short circuit with GND or VBBx.
- ERR_STUCK_AT — Error: Signal is frozen.
- ERR_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- ERR_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- ERR_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- ERR_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.
- DIAG_INTERNAL — Internal system error.
- DIAG_ACCESS — FB/Function cannot access the required resource; Resource is blocked by another task.
- DIAG_CHANGEOVER_TIME — Minimum changeover time for the highside-lowside selection of the drivers has not yet expired.
- DIAG_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_STUCK_AT — Error: Signal is frozen.
- DIAG_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- DIAG_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- DIAG_SHORT_CIRCUIT — Error: Short circuit with GND or VBBx.
- DIAG_NO_CALIB — The selected resource has no valid calibration.
  The displayed values are maybe faulty.
- DIAG_CONTROL_DITHER — The requested dither value cannot be set because the calculation from dither and PWM value is higher than 1000 per mill.

10.7.2 MODE_BRAKE (ENUM)

23480

Name	Description	Possible values	
MODE_BRAKE	Braking mode that is applied when changing the direction (xDirection) or when stopping (uiValue = 0).	UNCHANGED	Setting remains unchanged
		BRAKE_OFF	No braking. The voltage direction is changed immediately.
		BRAKE_EMCY	Emergency brakes: ▪ In case of change of direction: Braking only during tBrakeTime. ▪ When stopping: Braking during and after the tBrakeTime is elapsed.
		BRAKE_EMCY_BTIME	Emergency brakes, but only during tBrakeTime.
		BRAKE_DYNAMIC	Like BRAKE_EMCY mode, but dynamic braking with the uiBrakeValue.
		BRAKE_DYNAMIC_BTIME	Like BRAKE_EMCY_BTIME mode, but dynamic braking with the uiBrakeValue.

10.8 Library ifmOutPWM

Contents

CurrentControl 154
PWM1000 157
MODE_CURRENT_CONTROL (ENUM) 161
MODE_PWM (ENUM) 161

23381

The library function blocks (POU) and enumeration types for pulse width modulation and current control of output channels.

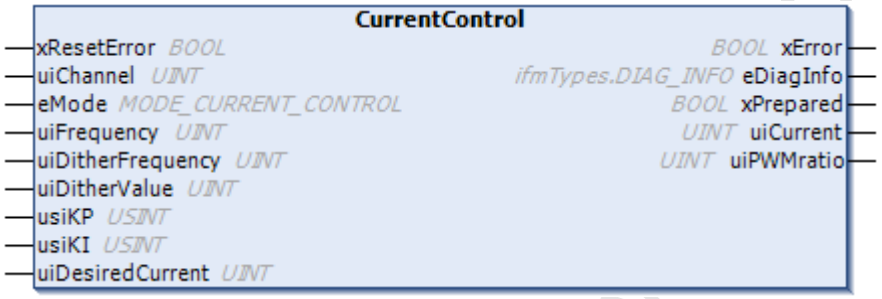
10.8.1 CurrentControl

23359

Function block type: Function block (FB)

Library: ifmIOutPWM.library

Symbol in CODESYS:



Description

23356

The FB is used to configure and operate a current controlled output. The current control is supported by pulse width modulation (PWM). The configuration of PWM frequency and dither is also done with this FB.

Input parameters

23357

Parameters	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE TRUE	When switching from FALSE ⇨ TRUE: Reset request to the lower level system
uiChannel	UINT	Output channel	Group + channel → Data sheet → **Note on wiring** (→ p. 29)	
			Examples:	
			703	Group 7 + channel 3
			1203	Group 12 + channel 3
eMode	MODE_ CURRENT_ CONTROL	Operating type of the output channel	→ **MODE_CURRENT_CONTROL (ENUM)** (→ p. 161)	
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet	
uiDitherFrequency	UNIT	Frequency for the dither signal at the PWM output in [Hz]	permissible = 0...uiFrequency / 2 The value at uiDitherFrequency must be an integer part of the value indicated to uiFrequency. Examples: uiFrequency = 300 Hz uiDitherFrequency = 50 Hz ⇨ 300 /50 = 6 ⇨ even factor, valid uiDitherFrequency = 100 Hz ⇨ 300 /100 = 3 ⇨ uneven factor, invalid Invalid values are corrected to the value that matches the next lower integer factor.	

uiDitherValue	UNIT	Peak-to-peak value of the dither signal which overlays with the PWM signal, in [‰]	permissible = 0...1 If the resulting PWM ratio value is outside the 0...1000 ‰ range, the dither value will be temporarily internally reduced to the minimum/maximum value that is possible, so that the mean value of the PWM ratio corresponds with the required value.
usiKP	USINT	Proportional component of the output signal	permissible = 0...255
usiKI	USINT	Integral component of the output signal	0 ... 255
uiDesiredCurrent	UINT	Default value at the output channel. When 0 is set, the output is immediately deactivated.	0 ... 65535

Output parameters

23358

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
uiCurrent	UINT	Output current signal in [mA]	available = 0...final value of the measuring range	
uiPWMRatio	UINT	PWM pulse ration calculated by the PI controller in [‰]		

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_SHORT_CIRCUIT — Error: Short circuit with GND or VBBx.
- ERR_STUCK_AT — Error: Signal is frozen.
- ERR_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- ERR_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- ERR_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- ERR_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.
- DIAG_INTERNAL — Internal system error.
- DIAG_ACCESS — FB/Function cannot access the required resource; Resource is blocked by another task.
- DIAG_CHANGEOVER_TIME — Minimum changeover time for the highside-lowside selection of the drivers has not yet expired.
- DIAG_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_STUCK_AT — Error: Signal is frozen.
- DIAG_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- DIAG_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- DIAG_SHORT_CIRCUIT — Error: Short circuit with GND or VBBx.
- DIAG_NO_CALIB — The selected resource has no valid calibration.
  The displayed values are maybe faulty.
- DIAG_CONTROL_DITHER — The requested dither value cannot be set because the calculation from dither and PWM value is higher than 1000 per mill.

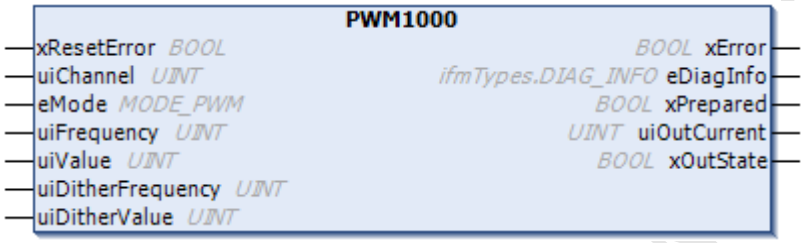
10.8.2 PWM1000

23343

Function block type: Function block (FB)

Library: ifmIOutPWM.library

Symbol in CODESYS:



Description

23344

The FB is used to configure and to operate an output with pulse width modulation.

Input parameters

23345

Parameters	Data type	Description	Possible values
xResetError	BOOL	Reset request for an occurring error	FALSE / TRUE: When switching from FALSE ⇨ TRUE: Reset request to the lower level system
uiChannel	UINT	Input channel	Group + channel → Data sheet → **Note on wiring** (→ p. 29) Examples: 403: Group 4 + channel 3 502: Group 5 + channel 2
eMode	MODE_PWM	Operating type of the output channel	→ **MODE_OUTPUT_GROUP (ENUM)** (→ p. 147)
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet
uiValue	UNIT	Pulse/pause ration of the PWM output signal in [‰]	permissible = 0...1
uiDitherFrequency	UNIT	Frequency for the dither signal at the PWM output in [Hz]	permissible = 0...uiFrequency / 2 The value at uiDitherFrequency must be an integer part of the value indicated to uiFrequency. Examples: uiFrequency = 300 Hz uiDitherFrequency = 50 Hz ⇨ 300 /50 = 6 ⇨ even factor, valid uiDitherFrequency = 100 Hz ⇨ 300 /100 = 3 ⇨ uneven factor, invalid Invalid values are corrected to the value that matches the next lower integer factor.
uiDitherValue	UNIT	Peak-to-peak value of the dither signal which overlays with the PWM signal, in [‰]	permissible = 0...1 If the resulting PWM ratio value is outside the 0...1000 ‰ range, the dither value will be temporarily internally reduced to the minimum/maximum value that is possible, so that the mean value of the PWM ratio corresponds with the required value.

Output parameters

23346

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB outputs still invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
uiGroupCurrent	UINT	Measured output current of the entire group in [mA]	available = 0...final value of the measuring range	
xGroupState	BOOL	Return value activation state of the selected output group The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output group is deactivated
			TRUE	Output value is activated

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 181)):

- ERR_INVALID_VALUE — Error: At least one input parameter is invalid or outside the value range.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_SHORT_CIRCUIT — Error: Short circuit with GND or VBBx.
- ERR_STUCK_AT — Error: Signal is frozen.
- ERR_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- ERR_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- ERR_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- ERR_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_INVALID_VALUE — At least one input parameter is invalid or exceeds the permissible area.
- DIAG_INTERNAL — Internal system error.
- DIAG_ACCESS — FB/Function cannot access the required resource; Resource is blocked by another task.
- DIAG_CHANGEOVER_TIME — Minimum changeover time for the highside-lowside selection of the drivers has not yet expired.
- DIAG_UNDERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG_OVERVOLTAGE_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG_STUCK_AT — Error: Signal is frozen.
- DIAG_STUCK_AT_HIGH — Error: Signal frozen, signal state high.
- DIAG_STUCK_AT_LOW — Error: Signal frozen, signal state low.
- DIAG_SHORT_CIRCUIT — Error: Short circuit with GND or VBBx.
- DIAG_NO_CALIB — The selected resource has no valid calibration.
  The displayed values are maybe faulty.
- DIAG_CONTROL_DITHER — The requested dither value cannot be set because the calculation from dither and PWM value is higher than 1000 per mill.

10.8.3 MODE_CURRENT_CONTROL (ENUM)

23361

Name	Description	Possible values	
MODE_CURRENT_CONTROL	Operating mode of the output	UNCHANGED	Setting is maintained
		OUT_CURRENT_CSO	Output for current control without diagnostics and without protection; CSO
		OUT_CURRENT_CSO_DIAG	Output for current control with diagnostics and without protection; CSO
		OUT_CURRENT_CSO_DIAG_PROT	Output for current control with diagnostics and protection; CSO
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.

10.8.4 MODE_PWM (ENUM)

23348

Name	Description	Possible values	
MODE_PWM	Operating mode of the output	UNCHANGED	Setting is maintained
		OUT_PWM_CSI	PWM output without diagnostics; CSI
		OUT_PWM_CSO	PWM output without diagnostics; CSO
		OUT_PWM_CSO_DIAG	PWM output with diagnostics and without protection; CSO
		OUT_PWM_CSO_DIAG_PROT	PWM output with diagnostics and with protection; CSO
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.

10.9 Library ifmRawCAN.library

Contents

CAN_Enable 163
CAN_Recover 165
CAN_RemoteRequest 167
CAN_RemoteResponse 169
CAN_Rx 171
CAN_RxMask 173
CAN_RxRange 175
CAN_Tx 177
CAN_Info (GVL) 179
CAN_BUS_STATE (STRUCT) 179

8722

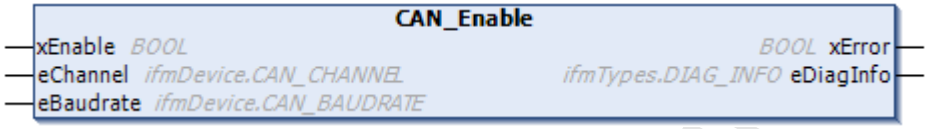
The library contains POUs and data structures for the programming of the CAN Layer 2 level of the CAN interfaces of the device under CODESYS.

10.9.1 CAN_Enable

8709

Function block type: Function block (FB)
Behaviour model: ENABLE

Library: ifmRawCAN.library
Symbol in CODESYS:



Description

7073

The FB activates the CAN Layer 2 functions of a CAN interface with a certain transmission rate. Simultaneously the FB writes information about the current state of the CAN interface into the global variable CAN State.

Changes of the transmission rate or of the CAN interface are applied at once. All existing reception and send buffer storages are deleted.

The FB does not have any influence on a CANopen Manager / CANopen Device at the selected CAN interface. In this case the FB cannot change the transmission rate of the CAN interface.

Input parameter

11241

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	FB is deactivated
			TRUE	FB is activated
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
eBaudrate	CAN_ BAUD RATE	Baud rate of the CAN channel	→ **CAN_BAUDRATE (ENUM)**	

Output parameter

7135

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	

Diagnostic codes:

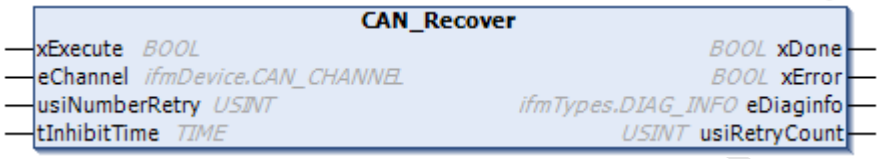
- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_BUS_OFF — Error: CAN interface is in the "BUS OFF" state
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_BAUDRATE_ALREADY_SET — Error: Requested baud rate cannot be set because another baud rate has already been defined.
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

10.9.2 CAN_Recover

11765

Function block type: Function block (FB)
Behaviour model: EXECUTE

Library: ifmRawCAN.library
Symbol in CODESYS:



Description

11771

The FB controls the processing of a failure of the CAN channel.
The call of the FB triggers the following actions:

- If the CAN channel fails the CAN interface is reset and rebooted.
- All buffer storages are emptied.

If the CAN channel keeps failing after the maximum number of recovery attempts has been exceeded, the CAN bus remains in the error state.
► Call FB again to repeat the execution of the recovery function.

Input parameter

11768

Parameter	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE ⇨ TRUE	FB is executed once
			Other	No impact on FB processing
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
usiNumberRetry	USINT	Max. number of retries	E.g. 4	
tInhibitTime	TIME	Time until the CAN interface is started again after the detection of a CAN bus failure	E.g. `#2ms`	

Output parameter

11769

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	▪ FB successfully executed ▪ FB can be called again
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
usiRetryCount	USINT	Counter for retries carried out since the last activation of the FB		

Diagnostic codes:

- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INACTIVE_INTERFACE — Error: Selected CAN channel is deactivated.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

10.9.3 CAN_RemoteRequest

10884

Function block type: Function block (FB)
Behaviour model: EXECUTE

Library: ifmRawCAN.library
Symbol in CODESYS:

Description

10886

The FB sends the request for a CAN Remote message into a CAN network. The FB provides the data of the response message in an array. The FB supports standard and extended frames.

Input parameter

10888

Parameter	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE ⇨ TRUE	FB is executed once
			Other	No impact on FB processing
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
udiID	UDINT	Identifier of the CAN message	▪ for Standard Frame (11 bits identifier): 0 ... 2047 ▪ for Extended-Frame (29 bits identifier): 0 ... 536.870.911	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 7	0 bytes* ... 7 bytes

* ... preset value

Output parameter

10890

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	▪ FB successfully executed ▪ FB can be called again
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
aData	ARRAY [0...7] OF USINT	Array for storage of the data received		

Diagnostic data:

- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT_BUSY — State: FB/Function is currently executed.
- ERR_BUFFER_OVERFLOW — Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_INACTIVE_INTERFACE — Error: Selected CAN channel is deactivated.

10.9.4 CAN_RemoteResponse

19902

Function block type: Function block (FB)
Behaviour model: ENABLE

Library: ifmRawCAN.library
Symbol in CODESYS:

Description

15962

The FB replies as reaction to the request of a CAN Remote message and sends the data required into a CAN network.

As long as the FB is activated it responds to each remote request message (automatic reply).

Several FB calls are possible during one PLC cycle.

Input parameter

9237

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	FB is deactivated
			TRUE	FB is activated
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
udiID	UDINT	Identifier of the CAN message	▪ for Standard Frame (11 bits identifier): 0 ... 2047 ▪ for Extended-Frame (29 bits identifier): 0 ... 536.870.911	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 7	0 bytes* ... 7 bytes

* ... preset value

Output parameter

11740

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
uiRTR_Cnt	UINT	Number of received remote requests after the last FB call		

Diagnostic code:

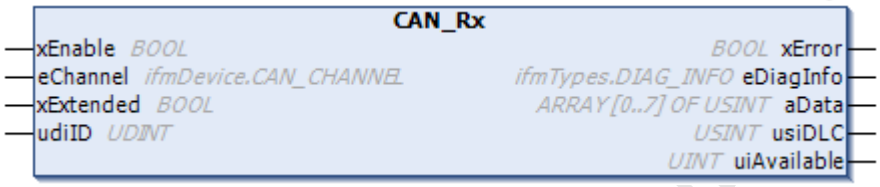
- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INACTIVE_INTERFACE — Error: Selected CAN channel is deactivated.
- ERR_BUFFER_OVERFLOW — Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

10.9.5 CAN_Rx

6939

Function block type: Function block (FB)
Behaviour model: ENABLE

Library: ifmRawCAN.library
Symbol in CODESYS:



Description

11777

The FB receives CAN messages with a defined identifier.
The FB receives all CAN messages with the indicated identifier between 2 FB calls and stores them in a FIFO buffer storage. The number of the received CAN messages is displayed. The CAN message received first is always provided on the output.

Input parameter

11784

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	FB is deactivated
			TRUE	FB is activated
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
udiID	UDINT	Identifier of the CAN message	▪ for Standard Frame (11 bits identifier): 0 ... 2047 ▪ for Extended-Frame (29 bits identifier): 0 ... 536.870.911	

Output parameter

14640

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
aData	ARRAY [0...7] OF USINT	Array for storage of the data received		
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 7	0 bytes* ... 7 bytes
uiAvailable	UINT	▪ Number of received CAN messages since the last FB call ▪ Current CAN message is taken into account	0	No CAN messages received between 2 FB calls
			n	n CAN messages received

Error codes:

- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INACTIVE_INTERFACE — Error: Selected CAN channel is deactivated.
- ERR_BUFFER_OVERFLOW — Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

10.9.6 CAN_RxMask

14643

Function block type: Function block (FB)
Behaviour model: ENABLE

Library: ifmRawCAN.library
Symbol in CODESYS:

Description

14641

The FB receives CAN messages of a non-coherent area. The area is defined by a bit pattern and a bit mask.

The following rules apply to the bit mask:

- 0: The equivalent bit of the CAN identifier can be 0 or 1
- 1: The equivalent bit of the CAN identifier must have the same value as the bit in the bit pattern

Example:

Pattern: `000 0010 0000`
Mask: `000 1111 1111`
Result: `xxx 0010 0000`

All CAN messages with an identifier whose 8 least significant bits have the value "0010 0000" are received.

E.g. 110 **0010 0000** 000 **0010 0000**, 001 **0010 0000**

General behaviour of the FB: → **CAN_Rx** (→ p. 171)

Input parameter

14638

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	FB is deactivated
			TRUE	FB is activated
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
udiIDSet	UDINT	Preset bit pattern for the masking of the identifier of the CAN message	E.g. 000 0010 0000	
udiIDMask	UDINT	Bit pattern of the required area 1 ... bit relevant for selection 0 ... bit not relevant for selection	E.g. 000 1111 1111	

* ... preset value

Output parameter

11736

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
aData	ARRAY [0...7] OF USINT	Array for storage of the data received		
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 7	0 bytes* ... 7 bytes
uiAvailable	UINT	▪ Number of received CAN messages since the last FB call ▪ Current CAN message is taken into account	0	No CAN messages received between 2 FB calls
			n	n CAN messages received
udiID	UDINT	Identifier of the CAN message	▪ for Standard Frame (11 bits identifier): 0 ... 2047 ▪ for Extended-Frame (29 bits identifier): 0 ... 536.870.911	

Diagnostic codes:

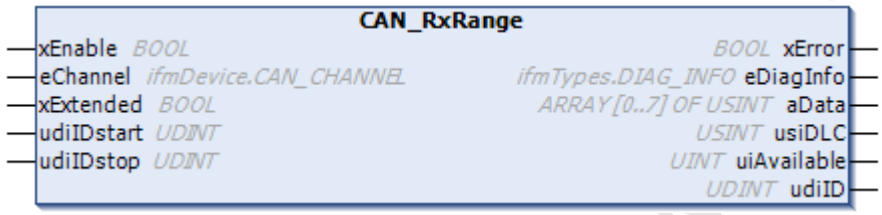
- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INACTIVE_INTERFACE — Error: Selected CAN channel is deactivated.
- ERR_BUFFER_OVERFLOW — Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

10.9.7 CAN_RxRange

11731

Function block type: Function block (FB)
Behaviour model: ENABLE

Library: ifmRawCAN.library
Symbol in CODESYS:



Description

11732

The FB receives CAN messages of a coherent area. The area is defined by an upper and lower limit.
The following rules apply to the definition of this area:

- Lower and upper limit:
  Standard Frames: 0 ... 2047 (11-bit identifier)
  Extended Frames: 0 ... 536 870 911 (29-bit identifier)
- The value for the lower limit must be <= the value of the upper limit.

Example:

Lower limit: `000 0000 0010`

Upper limit: `000 0000 1000`

Result: All CAN messages with an identifier whose 4 least significant bits have a value between "0010" and "1000" are received.

General behaviour of the FB: → **CAN_Rx** (→ p. 171)

Input parameter

14639

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	FB is deactivated
			TRUE	FB is activated
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ **CAN_CHANNEL (ENUM)** (→ p. 114)	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
udiIDStart	UDINT	Start of the required area	E.g. 000 0000 0010	
udiIDStop	UDINT	End of the required area	E.g. 000 0000 1000	

* ... preset value

Output parameter

14642

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	
aData	ARRAY [0...7] OF USINT	Array for storage of the data received		
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 7	0 bytes* ... 7 bytes
uiAvailable	UINT	▪ Number of received CAN messages since the last FB call ▪ Current CAN message is taken into account	0	No CAN messages received between 2 FB calls
			n	n CAN messages received
udiID	UDINT	Identifier of the CAN message	▪ for Standard Frame (11 bits identifier): 0 ... 2047 ▪ for Extended-Frame (29 bits identifier): 0 ... 536.870.911	

Diagnostic codes:

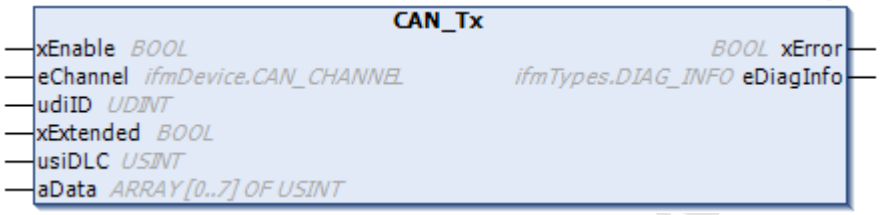
- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INACTIVE_INTERFACE — Error: Selected CAN channel is deactivated.
- ERR_BUFFER_OVERFLOW — Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

10.9.8 CAN_Tx

2269

Function block type: Function block (FB)
Behaviour model: ENABLE

Library: ifmRawCAN.library
Symbol in CODESYS:



Description

7401

By means of this FB CAN messages can be sent asynchronously. The FB writes the configured CAN message into the buffer storage of the selected CAN channel. When the CAN message is transmitted depends on the state of the CAN channel and the buffer storage. The FB and the PLC cycle do not have any influence on this.

The FB can be called several times during a PLC cycle.
The repeated call of the FB during a PLC cycle triggers a repeated transmission of the CAN message within the PLC cycle.

Input parameters

14057

Parameters	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	FB is deactivated
			TRUE	FB is activated
eChannel	CAN_ CHANNEL	Identifier of the CAN Interface	→ CAN_CHANNEL (ENUM) (→ p. 114)	
udiID	UDINT	Identifier of the CAN message	▪ for Standard Frame (11 bits identifier): 0 ... 2047 ▪ for Extended-Frame (29 bits identifier): 0 ... 536.870.911	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 7	0 bytes* ... 7 bytes
aData	ARRAY [0...7] OF USINT	Array with the data to be sent		

* ... preset value

Output parameter

13821

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	

Diagnostic codes:

- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INACTIVE_INTERFACE — Error: Selected CAN channel is deactivated.
- ERR_BUFFER_OVERFLOW — Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

10.9.9 CAN_Info (GVL)

12281

Name	Description	Data type	Possible values	
eBusState	Status of the CAN interface to CiA 11898	→ CAN_BUS_STATE (STRUCT) (→ p. 179)	Undefined	
uiBaudRate	Current baud rate	UINT	0*...65535	
udiRxCount	Counter for all messages detected at the CAN interface	UINT	0*...65535	
uiErrorCntRx	Error counter Rx (receive)	UINT	0*...65535	
uiErrorCntTx	Error counter Tx (send)	UINT	0*...65535	
xWarningRx	Warning signal for error counter Rx	BOOL	FALSE*	uiErrorCntRx < 96
			TRUE	uiErrorcntRx ≥ 96
xWarningTx	Warning signal for error counter Tx	BOOL	FALSE*	uiErrorCntRx < 96
			TRUE	uiErrorcntRx ≥ 96

* = preset value

10.9.10 CAN_BUS_STATE (STRUCT)

18497

Name	Description	Possible values		Data type	Value
CAN_BUS_STATE	State of the CAN interface	UNDEFINED	Interface not available or not configured	INT	0
		ERROR_ACTIVE	Error counter Tx/Rx <= 127	INT	1
		ERROR_PASSIVE	Error counter Tx/Rx > 127 and Error counter Tx > 255	INT	2
		BUS_OFF	Error counter Tx > 255	INT	65535

11 Troubleshooting

Contents

Error classes 180
Error messages 181
Messages / diagnostic codes of the function blocks 181

23282

11.1 Error classes

23276

An error is classified according to its possible impact. The error class determines how the system reacts when a specific error occurs.

Error class		Description	Reaction
A	**Fatal Error**	The overall integrity of the controller is no longer guaranteed. Errors in the central components of the controller that affect the behaviour of other components.	• The controller deactivates components allocated to the PLC. • The controller deactivates the concerned PLC. • The controller saves the information in the error log.
B	**Serious error**	One or several PLCs can no longer be executed.	• The controller deactivates components allocated to the PLC. • The controller deactivates the concerned PLC. • The controller saves the information in the error log.
C	**Component errors**	Error in a controller component; The function of one or several components of the controller is no longer guaranteed.	• The controller puts the affected function into a defined state. • The controller reports the error to the application. • The controller saves the information in the error log.
D	**Periphery errors**	Errors on or in the periphery; a function can no longer be executed.	• The controller puts the affected function into a defined state. • The controller reports the error to the application. • The controller saves the information in the error log. • The error can be reset in the application (as often as required).

11.2 Error messages

23459

(Most) FBs provide, among others, the following signals at their outputs.

► Evaluate these signals in the application!

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	▪ Error occurred ▪ Action could not be executed ▪ Note diagnostic information
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)	

Lists of diagnostic codes are part of the function block descriptions →↓**ifm function libraries** (→ p. 103)

11.3 Messages / diagnostic codes of the function blocks

23460

Status/diagnostic/error messages of the function blocs are defined in the global Enum DIAG_INFO.
They have one of the following prefixes depending on the type of message:

Prefix	Type of message	Description
STAT	Status message	Status messages contain information about the condition of the function block during the normal procedure.
DIAG	Diagnostic message	Diagnostic messages contain information about a failure event. They reset themselves after the failure event has disappeared and can optionally be evaluated by the application.
ERR	Error message	Error messages contain information about a failure event. They must be reset in the application after the failure event has disappeared.

Examples for messages / diagnostic codes:

- STAT_INACTIVE
- DIAG_OPEN_CIRCUIT
- ERR_OVERVOLTAGE

Lists of diagnostic codes are part of the function block descriptions →↓**ifm function libraries** (→ p. 103)

12 Appendix

Contents

Directory structure and file overview 182
ifm behaviour models for function blocks 183

10354

12.1 Directory structure and file overview

23443

The following directories and files are stored in the device:

Data name / path	Description
apps	Folder
▪ standard.app	Application non-safe
▪ safe.app	Application safe
os	Folder
▪ ifmOS.ifm	ifmOS
boot	Folder
▪ boot.ifm	Bootloader
sis	Folder
▪ sissys.ifm	SIS-SYS
cfg	Folder
▪ comconf.cfg	Communication configuration
▪ memconf.ifm	Memory configuration
retain	Folder
▪ standard.ret	Application retain non-safe
▪ standard.mb	Application memory bytes non-safe
▪ safe.ret	Application retain safe
▪ safe.mb	Application memory bytes non-safe
data	folder
▪ *.*	Memory space for user-defined data
compat	Folder
▪ compat.ifm	Compatibility File
cmd	Folder
▪ cmd.ifm	Command File
info	Folder
▪ devinfo.txt	Device information
▪ swinfo.txt	Software information

12.2 ifm behaviour models for function blocks

Contents

General 183
Behaviour model ENABLE 183
Behaviour model EXECUTE 184

23705

This chapter describes the ifm behaviour models for function blocks.

12.2.1 General

23801

ifm function blocks feature the following outputs to return status and error information:

Output	Description	
xError	TRUE	An error has occurred.
	FALSE	No error has occurred.
eDiagInfo	Diagnostic/error information → **Messages / diagnostic codes of the function blocks** (→ p. 181)	

All inputs and outputs in the function block that belong to the ifm behaviour model are featured at the top.

12.2.2 Behaviour model ENABLE

23705

Function blocks that use the behaviour model ENABLE are cyclically processed as long as the status at the input is xEnable = TRUE.

If xEnable = FALSE, the function block will not be executed. All function block outputs are reset to their preset default values and will not be updated. In this case the following applies: xError = FALSE and eDiagInfo = STAT_INACTIVE.

Function blocks that have no xEnable input are processed cyclically when the application is started. The processing is only terminated when the application is stopped. The behaviour corresponds with the behaviour of a function block with a permanent TRUE at the xEnable input.

Response to errors

23815

In case of an error, xError is set to TRUE and eDiagInfo indicates the diagnostic code as long as xEnable is = TRUE.

Irrespective of the data type, all other outputs of the function block will be reset to the following values:

Data type	Value
numerical	0 / 0.0
String	Empty string
BOOL/Bit	FALSE

12.2.3 Behaviour model EXECUTE

23800

Function blocks that have the EXECUTE behaviour model are processed once after a rising edge at the xExecute input.

If the function block has executed its function successfully, the output is set xDone = TRUE.

Response to errors

23827

In case of an error, xError is set to TRUE and eDiagInfo indicates the error status as long as xExecute is = TRUE.

The output xDone is set to FALSE since the execution could not be finished successfully.

Irrespective of the data type, all other outputs of the function block will be reset to the following values:

Data type	Value
numerical	0 / 0.0
String	Empty string
BOOL/Bit	FALSE

13 Glossary of Terms

A

Address

This is the "name" of the bus participant. All participants need a unique address so that the signals can be exchanged without problem.

Application

Software that is programmed by the manufacturer into the machine specifically for the application. The software usually contains logic sequences, limit values and expressions to control the corresponding inputs and outputs, calculations and decisions.

Architecture

Specific configuration of hardware and/or software elements in a system.

B

Baud

Baud, abbrev.: Bd = unit for the data transmission speed. Do not confuse baud with "bits per second" (bps, bits/s). Baud indicates the number of changes of state (steps, cycles) per second over a transmission length. But it is not defined how many bits per step are transmitted. The name baud can be traced back to the French inventor J. M. Baudot whose code was used for telex machines.
1 MBd = 1024 x 1024 Bd = 1 048 576 Bd

Bootloader

The bootloader is a start program with which the operating system and the application can be reloaded on the device.

Bus

Serial data transmission of several participants on the same cable.

C

CAN

CAN = **C**ontroller **A**rea **N**etwork
CAN is a priority-controlled fieldbus system for large data volumes. There are several higher-level protocols that are based on CAN, e.g. 'CANopen' or 'J1939'.

CAN stack

CAN stack = software component that deals with processing CAN messages.

CiA

CiA = CAN in Automation e.V.
User and manufacturer organisation in Germany / Erlangen. Definition and control body for CAN and CAN-based network protocols.
Homepage → www.can-cia.org

CiA DS 304

DS = **D**raft **S**tandard
CANopen device profile for safety communication

CiA DS 401

DS = **D**raft **S**tandard
CANopen device profile for binary and analogue I/O modules

CiA DS 402

DS = **D**raft **S**tandard
CANopen device profile for drives

CiA DS 403

DS = **D**raft **S**tandard
CANopen device profile for HMI

CiA DS 404

DS = **D**raft **S**tandard
CANopen device profile for measurement and control technology

CiA DS 405

DS = **D**raft **S**tandard
CANopen specification of the interface to programmable controllers (IEC 61131-3)

CiA DS 406

DS = **D**raft **S**tandard
CANopen device profile for encoders

CiA DS 407

DS = **D**raft **S**tandard
CANopen application profile for local public transport

Clamp 15

In vehicles clamp 15 is the plus cable switched by the ignition lock.

COB ID

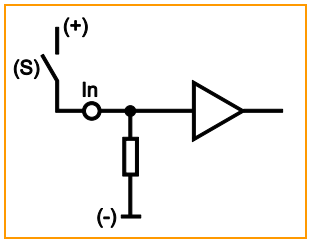
COB = **C**ommunication **Ob**ject
ID = **Id**entifier
ID of a CANopen communication object
Corresponds to the identifier of the CAN message with which the communication project is sent via the CAN bus.

CODESYS

CODESYS® is a registered trademark of 3S – Smart Software Solutions GmbH, Germany.
'CODESYS for Automation Alliance' associates companies of the automation industry whose hardware devices are all programmed with the widely used IEC 61131-3 development tool CODESYS®.
Homepage → www.codesys.com

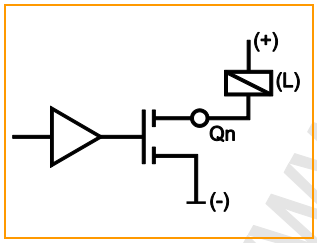
CSI

CSI = Current Sinking



Current Sinking Input
In = connection of binary input n
(S) = sensor

Binary input block diagram, plus-switching
for positive sensor signal
Input = open ⇨ Signal = Low (GND)
→ Low-Side Input (B_L)

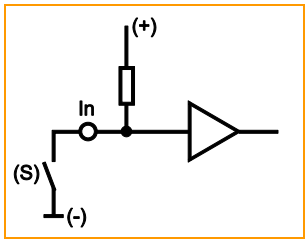


Current Sinking Output
Qn = connection of output n
(L) = load

Output block diagram, minus-switching (B_L)
for negative output signal
Low-Side Output (B_L)

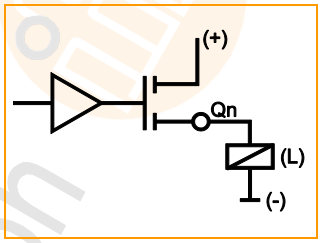
CSO

CSO = Current Sourcing



Current Sourcing Input
In = connection of binary input n
(S) = Sensor

Binary input block diagram, minus-switching
for negative sensor signal:
Input = open ⇨ Signal = High (Supply)
→ High-Side Input (B_H)



Current Sourcing Output
Qn = connection of output n
(L) = load

Output block diagram, plus-switching
for positive output signal
High-Side Output (B_H)

CSV file

CSV = **C**omma **S**eparated **V**alues (also: **C**haracter **S**eparated **V**alues)
A CSV file is a text file for storing or exchanging simply structured data.
The file extension is `.csv`.

Example: Source table with numerical values:

value 1.0	value 1.1	value 1.2	value 1.3
value 2.0	value 2.1	value 2.2	value 2.3
value 3.0	value 3.1	value 3.2	value 3.3

This results in the following CSV file:

```
value 1.0;value 1.1;value 1.2;value 1.3
value 2.0;value 2.1;value 2.2;value 2.3
value 3.0;value 3.1;value 3.2;value 3.3
```

Cycle time

This is the time for a cycle. The PLC program performs one complete run.
Depending on event-controlled branchings in the program this can take longer or shorter.

D

Data type

Depending on the data type, values of different sizes can be stored.

Data type	min. value	max. value	size in the memory
BOOL	FALSE	TRUE	8 bits = 1 byte
BYTE	0	255	8 bits = 1 byte
WORD	0	65 535	16 bits = 2 bytes
DWORD	0	4 294 967 295	32 bits = 4 bytes
SINT	-128	127	8 bits = 1 byte
USINT	0	255	8 bits = 1 byte
INT	-32 768	32 767	16 bits = 2 bytes
UINT	0	65 535	16 bits = 2 bytes
DINT	-2 147 483 648	2 147 483 647	32 bits = 4 bytes
UDINT	0	4 294 967 295	32 bits = 4 bytes
REAL	$-3.402823466 \cdot 10^{38}$	$3.402823466 \cdot 10^{38}$	32 bits = 4 bytes
ULINT	0	18 446 744 073 709 551 615	64 Bit = 8 Bytes
STRING			number of char. + 1

DC

Direct **C**urrent

Diagnosis

During the diagnosis, the "state of health" of the device is checked. It is to be found out if and what →faults are given in the device.

Depending on the device, the inputs and outputs can also be monitored for their correct function.
- wire break,
- short circuit,
- value outside range.

For diagnosis, configuration and log data can be used, created during the "normal" operation of the device.
The correct start of the system components is monitored during the initialisation and start phase. Errors are recorded in the log file.
For further diagnosis, self-tests can also be carried out.

Dither

Dither is a component of the →PWM signals to control hydraulic valves. It has shown for electromagnetic drives of hydraulic valves that it is much easier for controlling the valves if the control signal (PWM pulse) is superimposed by a certain frequency of the PWM frequency. This dither frequency must be an integer part of the PWM frequency.

DLC

Data **L**ength **C**ode = in CANopen the number of the data bytes in a message.
For →SDO: DLC = 8

DRAM

DRAM = **D**ynamic **R**andom **A**ccess **M**emory.
Technology for an electronic memory module with random access (Random Access Memory, RAM). The memory element is a capacitor which is either charged or discharged. It becomes accessible via a switching transistor and is either read or overwritten with new contents. The memory contents are volatile: the stored information is lost in case of lacking operating voltage or too late restart.

DTC

DTC = **D**iagnostic **T**rouble **C**ode = error code
In the protocol J1939 faults and errors well be managed and reported via assigned numbers – the DTCs.

E

ECU

(1) **E**lectronic **C**ontrol **U**nit = control unit or microcontroller
(2) **E**ngine **C**ontrol **U**nit = control device of a engine

EDS-file

EDS = **E**lectronic **D**ata **S**heet, e.g. for:
- File for the object directory in the CANopen master,
- CANopen device descriptions.

Via EDS devices and programs can exchange their specifications and consider them in a simplified way.

Embedded software

System software, basic program in the device, virtually the →runtime system.
The firmware establishes the connection between the hardware of the device and the application program. The firmware is provided by the manufacturer of the controller as a part of the system and cannot be changed by the user.

EMC

EMC = **E**lectro **M**agnetic **C**ompatibility.
According to the EC directive (2004/108/EEC) concerning electromagnetic compatibility (in short EMC directive) requirements are made for electrical and electronic apparatus, equipment, systems or components to operate satisfactorily in the existing electromagnetic environment. The devices must not interfere with their environment and must not be adversely influenced by external electromagnetic interference.

EMCY

Abbreviation for emergency
Message in the CANopen protocol with which errors are signalled.

Ethernet

Ethernet is a widely used, manufacturer-independent technology which enables data transmission in the network at a speed of 10...10 000 million bits per second (Mbps). Ethernet belongs to the family of so-called "optimum data transmission" on a non exclusive transmission medium. The concept was developed in 1972 and specified as IEEE 802.3 in 1985.

EUC

EUC = **E**quipment **U**nder **C**ontrol.
EUC is equipment, machinery, apparatus or plant used for manufacturing, process, transportation, medical or other activities (→ IEC 61508-4, section 3.2.3). Therefore, the EUC is the set of all equipment, machinery, apparatus or plant that gives rise to hazards for which the safety-related system is required.
If any reasonably foreseeable action or inaction leads to →hazards with an intolerable risk arising from the EUC, then safety functions are necessary to achieve or maintain a safe state for the EUC. These safety functions are performed by one or more safety-related systems.

F

FiFo

FIFO (**F**irst **I**n, **F**irst **O**ut) = Operating principle of the stack memory: The data packet that was written into the stack memory first, will also be read first. Each identifier has such a buffer (queue).

Flash memory

Flash ROM (or flash EPROM or flash memory) combines the advantages of semiconductor memory and hard disks. Similar to a hard disk, the data are however written and deleted blockwise in data blocks up to 64, 128, 256, 1024, ... bytes at the same time.

Advantages of flash memories

- The stored data are maintained even if there is no supply voltage.
- Due to the absence of moving parts, flash is noiseless and insensitive to shocks and magnetic fields.

Disadvantages of flash memories

- A storage cell can tolerate a limited number of write and delete processes:
  - Multi-level cells: typ. 10 000 cycles
  - Single level cells: typ. 100 000 cycles
- Given that a write process writes memory blocks of between 16 and 128 Kbytes at the same time, memory cells which require no change are used as well.

FRAM

FRAM, or also FeRAM, means **Fe**rroelectric **R**andom **A**ccess **M**emory. The storage operation and erasing operation is carried out by a polarisation change in a ferroelectric layer.
Advantages of FRAM as compared to conventional read-only memories:
- non-volatile,
- compatible with common EEPROMs, but:
- access time approx. 100 ns,
- nearly unlimited access cycles possible.

H

Heartbeat

The participants regularly send short signals. In this way the other participants can verify if a participant has failed.

HMI

HMI = **H**uman **M**achine **I**nterface

I

ID

ID = **Id**entifier

Name to differentiate the devices / participants connected to a system or the message packets transmitted between the participants.

IEC 61131

Standard: Basics of programmable logic controllers
- Part 1: General information
- Part 2: Production equipment requirements and tests
- Part 3: Programming languages
- Part 5: Communication
- Part 7: Fuzzy Control Programming

IEC user cycle

IEC user cycle = PLC cycle in the CODESYS application program.

Instructions

Superordinate word for one of the following terms:
installation instructions, data sheet, user information, operating instructions, device manual, installation information, online help, system manual, programming manual, etc.

Intended use

Use of a product in accordance with the information provided in the instructions for use.

IP address

IP = **I**nternet **P**rotocol.
The IP address is a number which is necessary to clearly identify an internet participant. For the sake of clarity the number is written in 4 decimal values, e.g. 127.215.205.156.

ISO 11898

Standard: Road vehicles – Controller area network
- Part 1: Data link layer and physical signalling
- Part 2: High-speed medium access unit
- Part 3: Low-speed, fault-tolerant, medium dependent interface
- Part 4: Time-triggered communication
- Part 5: High-speed medium access unit with low-power mode

ISO 11992

Standard: Interchange of digital information on electrical connections between towing and towed vehicles
- Part 1: Physical and data-link layers
- Part 2: Application layer for brakes and running gear
- Part 3: Application layer for equipment other than brakes and running gear
- Part 4: Diagnostics

ISO 16845
Standard: Road vehicles – Controller area network (CAN) – Conformance test plan

J

J1939
→ SAE J1939

L

LED
LED = **L**ight **E**mitting **D**iode.
Light emitting diode, also called luminescent diode, an electronic element of high coloured luminosity at small volume with negligible power loss.

Link
A link is a cross-reference to another part in the document or to an external document.

LSB
Least **S**ignificant **B**it/Byte

M

MAC-ID
MAC = **M**anufacturer‘s **A**ddress **C**ode
= manufacturer's serial number.
→ID = **Id**entifier
Every network card has a MAC address, a clearly defined worldwide unique numerical code, more or less a kind of serial number. Such a MAC address is a sequence of 6 hexadecimal numbers, e.g. "00-0C-6E-D0-02-3F".

Master
Handles the complete organisation on the bus. The master decides on the bus access time and polls the →slaves cyclically.

Misuse
The use of a product in a way not intended by the designer.
The manufacturer of the product has to warn against readily predictable misuse in his user information.

MMI
→ **HMI** (→ p. 190)

MRAM

MRAM = **M**agnetoresistive **R**andom **A**ccess **M**emory
The information is stored by means of magnetic storage elements. The property of certain materials is used to change their electrical resistance when exposed to magnetic fields.
Advantages of MRAM as compared to conventional RAM memories:
- non volatile (like FRAM), but:
- access time only approx. 35 ns,
- unlimited number of access cycles possible.

MSB

Most **S**ignificant **B**it/Byte

N

NMT

NMT = **N**etwork **M**anagement = (here: in the CANopen protocol).
The NMT master controls the operating states of the NMT slaves.

Node

This means a participant in the network.

Node Guarding

Node = here: network participant
Configurable cyclic monitoring of each →slave configured accordingly. The →master verfies if the slaves reply in time. The slaves verify if the master regularly sends requests. In this way failed network participants can be quickly identified and reported.

O

Obj / object

Term for data / messages which can be exchanged in the CANopen network.

Object directory

Contains all CANopen communication parameters of a device as well as device-specific parameters and data.

OBV

Contains all CANopen communication parameters of a device as well as device-specific parameters and data.

OPC

OPC = **O**LE for **P**rocess **C**ontrol
Standardised software interface for manufacturer-independent communication in automation technology
OPC client (e.g. device for parameter setting or programming) automatically logs on to OPC server (e.g. automation device) when connected and communicates with it.

Operating system

Basic program in the device that establishes the connection between the hardware of the device and the application program.
→ Chapter **Software module for the device**

Operational

Operating state of a CANopen participant. In this mode →SDOs, →NMT commands and →PDOs can be transferred.

P

PC card

→PCMCIA card

PCMCIA card

PCMCIA = Personal Computer Memory Card International Association, a standard for expansion cards of mobile computers.
Since the introduction of the cardbus standard in 1995 PCMCIA cards have also been called PC card.

PDM

PDM = **P**rocess and **D**ialogue **M**odule.
Device for communication of the operator with the machine / plant.

PDO

PDO = **P**rocess **D**ata **O**bject.
The time-critical process data is transferred by means of the "process data objects" (PDOs). The PDOs can be freely exchanged between the individual nodes (PDO linking). In addition it is defined whether data exchange is to be event-controlled (asynchronous) or synchronised. Depending on the type of data to be transferred the correct selection of the type of transmission can lead to considerable relief for the →CAN bus.
According to the protocol, these services are unconfirmed data transmission: it is not checked whether the receiver receives the message. Exchange of network variables corresponds to a "1 to n connection" (1 transmitter to n receivers).

PDU

PDU = **P**rotocol **D**ata **U**nit = protocol data unit.
The PDU is a term from the →CAN protocol →SAE J1939. It refers to a component of the target address (PDU format 1, connection-oriented) or the group extension (PDU format 2, message-oriented).

PES

Programmable **E**lectronic **S**ystem ...
- for control, protection or monitoring,
- dependent for its operation on one or more programmable electronic devices,
- including all elements of the system such as input and output devices.

PGN

PGN = **P**arameter **G**roup **N**umber
PGN = 6 zero bits + 1 bit reserved + 1 bit data page + 8 bit PDU Format (PF) + 8 PDU Specific (PS)
The parameter group number is a term from the →CAN protocol →SAE J1939.

Pictogram

Pictograms are figurative symbols which convey information by a simplified graphic representation.
(→ chapter **What do the symbols and formats mean?**)

PID controller

The PID controller (proportional–integral–derivative controller) consists of the following parts:
• P = proportional part
• I = integral part
• D = differential part (but not for the controller CR04nn, CR253n).

PLC configuration

Part of the CODESYS user interface.

► The programmer tells the programming system which hardware is to be programmed.
> CODESYS loads the corresponding libraries.
> Reading and writing the periphery states (inputs/outputs) is possible.

Pre-Op

Pre-Op = PRE-OPERATIONAL mode.
Operating status of a CANopen participant. After application of the supply voltage each participant automatically passes into this state. In the CANopen network only →SDOs and →NMT commands can be transferred in this mode but no process data.

Process image

Process image is the status of the inputs and outputs the PLC operates with within one →cycle.

- At the beginning of the cycle the PLC reads the conditions of all inputs into the process image. During the cycle the PLC cannot detect changes to the inputs.
- During the cycle the outputs are only changed virtually (in the process image).
- At the end of the cycle the PLC writes the virtual output states to the real outputs.

PWM

PWM = pulse width modulation
The PWM output signal is a pulsed signal between GND and supply voltage.
Within a defined period (PWM frequency) the mark-to-space ratio is varied. Depending on the mark-to-space ratio, the connected load determines the corresponding RMS current.

R

ratiometric

Measurements can also be performed ratiometrically. If the output signal of a sensor is proportional to its suppy voltage then via ratiometric measurement (= measurement proportional to the supply) the influence of the supply's fluctuation can be reduced, in ideal case it can be eliminated.
→ analogue input

RAW-CAN

RAW-CAN means the pure CAN protocol which works without an additional communication protocol on the CAN bus (on ISO/OSI layer 2). The CAN protocol is international defined according to ISO 11898-1 and garantees in ISO 16845 the interchangeability of CAN chips in addition.

remanent

Remanent data is protected against data loss in case of power failure.
The →runtime system for example automatically copies the remanent data to a →flash memory as soon as the voltage supply falls below a critical value. If the voltage supply is available again, the runtime system loads the remanent data back to the RAM memory.
The data in the RAM memory of a controller, however, is volatile and normally lost in case of power failure.

ro

RO = read only for reading only
Unidirectional data transmission: Data can only be read and not changed.

RTC

RTC = **R**eal **T**ime **C**lock
Provides (batter-backed) the current date and time. Frequent use for the storage of error message protocols.

rw

RW = read/ write
Bidirectional data transmission: Data can be read and also changed.

S

SAE J1939

The network protocol SAE J1939 describes the communication on a →CAN bus in commercial vehicles for transmission of diagnosis data (e.g.engine speed, temperature) and control information.
Standard: Recommended Practice for a Serial Control and Communications Vehicle Network
- Part 2: Agricultural and Forestry Off-Road Machinery Control and Communication Network
- Part 3: On Board Diagnostics Implementation Guide
- Part 5: Marine Stern Drive and Inboard Spark-Ignition Engine On-Board Diagnostics Implementation Guide
- Part 11: Physical Layer – 250 kBits/s, Shielded Twisted Pair
- Part 13: Off-Board Diagnostic Connector
- Part 15: Reduced Physical Layer, 250 kBits/s, Un-Shielded Twisted Pair (UTP)
- Part 21: Data Link Layer
- Part 31: Network Layer
- Part 71: Vehicle Application Layer
- Part 73: Application Layer – Diagnostics
- Part 81: Network Management Protocol

SD card

An SD memory card (short for **S**ecure **D**igital Memory Card) is a digital storage medium that operates to the principle of →flash storage.

SDO

SDO = **S**ervice **D**ata **O**bject.
The SDO is used for access to objects in the CANopen object directory. 'Clients' ask for the requested data from 'servers'. The SDOs always consist of 8 bytes.

Examples:
- Automatic configuration of all slaves via →SDOs at the system start,
- reading error messages from the →object directory.

Every SDO is monitored for a response and repeated if the slave does not respond within the monitoring time.

Self-test

Test program that actively tests components or devices. The program is started by the user and takes a certain time. The result is a test protocol (log file) which shows what was tested and if the result is positive or negative.

Slave

Passive participant on the bus, only replies on request of the →master. Slaves have a clearly defined and unique →address in the bus.

stopped

Operating status of a CANopen participant. In this mode only →NMT commands are transferred.

Symbols

Pictograms are figurative symbols which convey information by a simplified graphic representation.
(→ chapter **What do the symbols and formats mean?**)

System variable

Variable to which access can be made via IEC address or symbol name from the PLC.

T

Target

The target contains the hardware description of the target device for CODESYS, e.g.: inputs and outputs, memory, file locations.
Corresponds to an electronic data sheet.

TCP

The **T**ransmission **C**ontrol **P**rotocol is part of the TCP/IP protocol family. Each TCP/IP data connection has a transmitter and a receiver. This principle is a connection-oriented data transmission. In the TCP/IP protocol family the TCP as the connection-oriented protocol assumes the task of data protection, data flow control and takes measures in the event of data loss. (compare: →UDP)

Template

A template can be filled with content.
Here: A structure of pre-configured software elements as basis for an application program.

U

UDP

UDP (**U**ser **D**atagram **P**rotocol) is a minimal connectionless network protocol which belongs to the transport layer of the internet protocol family. The task of UDP is to ensure that data which is transmitted via the internet is passed to the right application.
At present network variables based on →CAN and UDP are implemented. The values of the variables are automatically exchanged on the basis of broadcast messages. In UDP they are implemented as broadcast messages, in CAN as →PDOs.
According to the protocol, these services are unconfirmed data transmission: it is not checked whether the receiver receives the message. Exchange of network variables corresponds to a "1 to n connection" (1 transmitter to n receivers).

Use, intended

Use of a product in accordance with the information provided in the instructions for use.

W

Watchdog

In general the term watchdog is used for a component of a system which watches the function of other components. If a possible malfunction is detected, this is either signalled or suitable program branchings are activated. The signal or branchings serve as a trigger for other co-operating system components to solve the problem.

14 Index

A

About this manual ... 6
Access inputs ... 85
Access outputs ... 86
Access to inputs ... 82
Access to outputs ... 83
Activate the access protection for a project ... 64
Add ifm function libraries to the application ... 63
Address ... 185
Address assignment in Ethernet networks ... 70
Address assignment of the outputs ... 169
Allocate inputs/outputs ... 66
Allocate memory partition ... 65
Appendix ... 182
Application ... 57, 185
Architecture ... 185
Available memory ... 31

B

Baud ... 185
Behaviour model ENABLE ... 183
Behaviour model EXECUTE ... 184
Binary input block diagram plus/minus-switching ... 38
Binary output block diagram plus/minus-switching ... 44
Block diagram of the supply and of the output deactivation ... 29
Bootloader ... 56, 185
Bus ... 185

C

CAN ... 185
  Interfaces and protocols ... 53
CAN stack ... 185
CAN_BAUDRATE (ENUM) ... 114
CAN_BUS_STATE (STRUCT) ... 179
CAN_CHANNEL (ENUM) ... 114
CAN_Enable ... 163
CAN_Info (GVL) ... 179
CAN_Recover ... 165
CAN_RemoteRequest ... 167
CAN_RemoteResponse ... 169
CAN_Rx ... 171
CAN_RxMask ... 173
CAN_RxRange ... 175
CAN_Tx ... 177
CANconstants (GVL) ... 114
CANopen
  Network Management (NMT) ... 90
  Send and receive SDO ... 90
Carry out installation ... 19
Check the hardware version of the device ... 23
Check the operating system version of the device ... 22
CiA ... 185
CiA DS 304 ... 186
CiA DS 401 ... 186
CiA DS 402 ... 186
CiA DS 403 ... 186
CiA DS 404 ... 186
CiA DS 405 ... 186
CiA DS 406 ... 186
CiA DS 407 ... 186
Clamp 15 ... 186
COB ID ... 186
CODESYS ... 187
CODESYS programming software ... 19
Complete package for ecomatController CR720S ... 20
Components of the complete package ... 20
Configure CAN interfaces ... 71
Configure Ethernet interface ... 69
Configure IEC watchdog ... 37
Configure inputs and outputs ... 68
Configure interfaces ... 69
Configure PLC ... 65
Configure programming interface ... 62
Configure serial interface ... 69
Configure task processing ... 81
Control device ... 83
Controlling LEDs in the applications ... 97
COP_GetNodeState ... 104
COP_SDOread ... 106
COP_SDOwrite ... 108
COP_SendNMT ... 110
COUNT_DIRECTION (ENUM) ... 123
Create CODESYS project ... 58
Create new project with CR720S ... 59
Create PLC application ... 79
CSI ... 187
CSO ... 187
CSV file ... 187
CurrentControl ... 154
Cycle time ... 187

D

Data transmission for series production ... 100
Data transmission with TFTP ... 101
Data type ... 188
DC ... 188
Delete application from CR720S ... 93
Description 104, 106, 108, 110, 117, 119, 121, 126, 129, 132, 134, 136, 138, 144, 149, 154, 157, 163, 165, 167, 169, 171, 173, 175, 177
Device supply (technology) ... 33
Diagnosis ... 188
Directory structure and file overview ... 182
Display system information ... 102
Dither ... 188
DLC ... 188
DRAM ... 189
DTC ... 189

E

ECU ... 189
EDS-file ... 189
Embedded software ... 189
EMC ... 189
EMCY ... 189
ENCODER_RESOLUTION (ENUM) ... 123
Error classes ... 180
Error messages ... 181

Ethernet 189
Ethernet interface 52
EUC 190

F

FastCount 117
Feedback in case of externally supplied outputs 51
FiFo 190
Files for series production 101
FILTER_INPUT (ENUM) 140
FILTER_OUTPUT (ENUM) 140
FILTER_OUTPUT_GROUP (ENUM) 147
Flash memory 190
FRAM 190
FREQ_SENSE_PERIODS (ENUM) 123
Functions and features 10

G

General 103, 183
Getting started 58
Group designations 29

H

Hardware 18
Hardware description 26
Hardware structure 26
HBridge 149
Heartbeat 190
History of the document CR0720 9
HMI 190
How is this documentation structured? 9

I

ID 191
IEC 61131 191
IEC user cycle 191
ifm behaviour models for function blocks 183
ifm function libraries 103
ifm weltweit • ifm worldwide • ifm à l'échelle internationale 207
ifmFastInput.library 116
Important standards 13
IncEncoder 119
Input 126
Input parameter 104, 110, 119, 163, 165, 167, 169, 171, 173, 175
Input parameters106, 108, 117, 121, 126, 129, 132, 134, 136, 138, 144, 149, 154, 158, 177
Input type IN DIGITAL-A 40
Input type IN DIGITAL-B 41
Input type IN FREQUENCY-A/B 39
Input type IN MULTIFUNCTION-A 39
Input type IN RESISTOR-A 40
Inputs (technology) 38
Install CODESYS Development System 19
Install package (PC/laptop) 21
Installation 18
Instructions 191
Intended use 191
Interface configuration file comconf.cfg 77
Interfaces 52
IP address 191
ISO 11898 191
ISO 11992 191
ISO 16845 192

J

J1939 192

L

LED 192
LED_COLOUR (ENUM) 115
LED_FLASH_FREQ (ENUM) 115
Legal and copyright information 6
Libraries 57
Library ifmCANopenManager.library 103
Library ifmDeviceCR0721.library 113
Library ifmIOcommon.library 125
Library ifmOutGroup 143
Library ifmOutHBridge 148
Library ifmOutPWM 153
Library ifmRawCAN.library 162
Licensing 18
Link 192
List of inputs 42
List of outputs 50
Load the application to the device 92
LSB 192

M

MAC-ID 192
Manage files 67
Manage users and groups 67
Master 192
Memory allocation 31
Memory allocation variants 32
Memory, available 31
Messages / diagnostic codes of the function blocks 181
Misuse 192
MMI 192
MODE_BRAKE (ENUM) 152
MODE_CURRENT_CONTROL (ENUM) 161
MODE_FAST_COUNT (ENUM) 123
MODE_INC_ENCODER (ENUM) 124
MODE_INPUT (ENUM) 141
MODE_OUTPUT (ENUM) 142
MODE_OUTPUT_GROUP (ENUM) 147
MODE_PERIOD (ENUM) 124
MODE_PWM (ENUM) 161
Monitoring and securing mechanisms 36
Monitoring concept 36
MRAM 193
MSB 193

N

NMT 193
NMT_SERVICE (ENUM) 112
NMT_STATES (ENUM) 112
Node 193
Node Guarding 193
Note on wiring 29
Notes

serial number ... 17
Notizen • Notes • Notes ... 203

O

Obj / object ... 193
Object directory ... 193
Objects of a PLC application ... 78
OBV ... 193
OPC ... 193
Operating states ... 94
Operating system ... 56, 194
Operation ... 92
Operational ... 194
Options to access input and output data ... 81
Organise the creation of safe machinery with the V model ... 14
Output ... 129
Output parameter ... 105, 163, 166, 168, 170, 172, 174, 176, 178
Output parameters ... 107, 109, 111, 118, 120, 122, 127, 130, 133, 135, 136, 139, 145, 150, 155, 159
Output type OUT PWM-n-A ... 45
Output type OUT PWM-n-B ... 46
Output type OUT PWM-n-BRIDGE-A ... 47
Output type OUT Supply-A ... 48
Output type OUT Voltage-A ... 49
Output types ... 44
OutputGroup ... 144
Overview
documentation for CODESYS 3.n ... 8
documentation modules for CR720S ... 8
Hardware ... 27
Project structure with CR720S ... 60
Software ... 54

P

PC card ... 194
PCMCIA card ... 194
PDM ... 194
PDO ... 194
PDU ... 194
Period ... 121
PES ... 194
PGN ... 195
Pictogram ... 195
PID controller ... 195
PLC configuration ... 195
Please note! ... 11
Pre-Op ... 195
Previous knowledge ... 12
Process image ... 195
Programming ... 78
Purpose of the document ... 7
PWM ... 195
PWM1000 ... 157

R

ratiometric ... 195
RawCAN
Control CAN network nodes ... 88
Request and send remote CAN messages ... 89
Send and receive CAN messages ... 88
RAW-CAN ... 196
Read device information ... 83
Read diagnostic data of the device ... 87
remanent ... 196
Reset ... 98
Reset application (cold) ... 99
Reset application (origin) ... 99
Reset application (warm) ... 98
Response to errors ... 183, 184
Retain variables ... 80
ro ... 196
RTC ... 196
rw ... 196

S

SAE J1939 ... 196
Safety instructions ... 11
SD card ... 196
SDO ... 197
Self-test ... 197
Serial interface ... 52
Set communication path of PLC ... 62
SetLED ... 132
Setting / measurement via system configuration ... 48, 49
Setting /measurement via FB Output ... 48, 49
Slave ... 197
Software ... 18
Software description ... 54
Software in the controller ... 55
Software module for the device ... 56
Software on the PC/notebook ... 54
Standard PLC and safety PLC ... 30
Start CODESYS ... 58
Start conditions ... 33
Start-up behaviour of the controller ... 16
Status LED
Ethernet interfaces (ETH0, ETH1) ... 97
System bootloader (SYS0) ... 96
system ifm operating system (SYS0+SYS1) ... 95
system PLC (SYS0, SYS1) ... 96
Status LEDs ... 95
stopped ... 197
SupplySwitch ... 134
Supported programming languages ... 79
Supported reset variants ... 98
Supported variable types ... 80
Switch off outputs via solid-state switch ... 36
Switch on/off via ignition lock (Terminal 15) ... 35
Switch on/off via main switch ... 34
Symbols ... 197
Symbols and formats used ... 7
SYS_VOLTAGE_CHANNEL (ENUM) ... 142
SysInfo (GVL) ... 114
SysInfoStruct (STRUCT) ... 115
System configuration ... 65
System context of the controller ... 27
System description ... 26
System overview ... 28
System requirements ... 18
System variable ... 197
SystemSupply ... 136

T

Target....197
TCP....197
Temperature....138
Template....197
The the IP parameter of the Ethernet interface....69
Transfer CODESYS project to device....92
Transmission of the files with CODESYS....100
Troubleshooting....180
Types of inputs....38

U

UDP....198
Uninstall package (PC/laptop)....22
Update package (PC/laptop)....21
Update the operating system of the device....24
Update the operating system of the device with the batch file....24
Use CANopen....90
Use CODESYS user manual....61
Use ifm function libraries....82
Use IO mapping....84
Use RawCAN (CAN Layer 2)....88
Use SAE J1939....91
Use, intended....198

V

V model....14
via function block....68
RAW-CAN....76
via system configuration....68
CANopen device....74
CANopen Manager....72
J1939 manager....75
Voltage ranges of the on-board system....33

W

Warnings used....16
watchdog....198
Watchdog....37, 198
What previous knowledge is required?....12

15 Notizen • Notes • Notes

16 ifm weltweit • ifm worldwide • ifm à l'échelle internationale

Version: 2016-11-29 8310

ifm electronic gmbh • Friedrichstraße 1 • 45128 Essen
www.ifm.com • Email: info@ifm.com
Service hotline: 0800 / 16 16 16 (only Germany, Mo-Fr 07.00...18.00 h)

ifm Niederlassungen • Sales offices • Agences

D	Niederlassung Nord • 31135 Hildesheim • Tel. 0 51 21 / 76 67-0
	Niederlassung West • 45128 Essen • Tel. 02 01 / 3 64 75 -0
	Niederlassung Mitte-West • 58511 Lüdenscheid • Tel. 0 23 51 / 43 01-0
	Niederlassung Süd-West • 64646 Heppenheim • Tel. 0 62 52 / 79 05-0
	Niederlassung Baden-Württemberg • 73230 Kirchheim • Tel. 0 70 21 / 80 86-0
	Niederlassung Bayern • 82178 Puchheim • Tel. 0 89 / 8 00 91-0
	Niederlassung Ost • 07639 Tautenhain • Tel. 0 36 601 / 771-0
A, SL	ifm electronic gmbh • 1120 Wien • Tel. +43 16 17 45 00
AUS	ifm efector pty ltd. • Mulgrave Vic 3170 • Tel. +61 3 00 365 088
B, L	ifm electronic N.V. • 1731 Zellik • Tel. +32 2 / 4 81 02 20
BG	ifm electronic eood • 1202 Sofia • Tel. +359 2 807 59 69
BR	ifm electronic Ltda. • 03337-000, Sao Paulo SP • Tel. +55 11 / 2672-1730
CH	ifm electronic ag • 4 624 Härkingen • Tel. +41 62 / 388 80 30
CL	ifm electronic SpA • Oficina 5032 Comuna de Conchalí • Tel. +55 11 / 2672-1730
CN	ifm electronic (Shanghai) Co. Ltd. • 201203 Shanghai • Tel. +86 21 / 3813 4800
CND	ifm efector Canada inc. • Oakville, Ontario L6K 3V3 • Tel. +1 800-441-8246
CZ	ifm electronic spol. s.r.o. • 25243 Průhonice • Tel. +420 267 990 211
DK	ifm electronic a/s • 2605 BROENDBY • Tel. +45 70 20 11 08
E	ifm electronic s.a. • 08820 El Prat de Llobregat • Tel. +34 93 479 30 80
F	ifm electronic s.a. • 93192 Noisy-le-Grand Cedex • Tél. +33 0820 22 30 01
FIN	ifm electronic oy • 00440 Helsinki • Tel . +358 75 329 5000
GB, IRL	ifm electronic Ltd. • Hampton, Middlesex TW12 2HD • Tel. +44 208 / 213-0000
GR	ifm electronic Monoprosopi E.P.E. • 15125 Amaroussio • Tel. +30 210 / 6180090
H	ifm electronic kft. • 9028 Györ • Tel. +36 96 / 518-397
I	ifm electronic s.a. • 20041 Agrate-Brianza (MI) • Tel. +39 039 / 68.99.982
IL	Astragal Ltd. • Azur 58001 • Tel. +972 3 -559 1660
IND	ifm electronic India Branch Office • Kolhapur, 416234 • Tel. +91 231-267 27 70
J	efector co., ltd. • Chiba-shi, Chiba 261-7118 • Tel. +81 043-299-2070
MAL	ifm electronic Pte. Ltd • 47100 Puchong Selangor • Tel. +603 8063 9522
MEX	ifm efector S. de R. L. de C. V. • Monterrey, N. L. 64630 • Tel. +52 81 8040-3535
N	Sivilingeniør J. F. Knudtzen A/S • 1396 Billingstad • Tel. +47 66 / 98 33 50
NA	ifm elctronic (pty) Ltd • 25 Dr. W. Kulz Street Windhoek • Tel. +264 61 300984
NL	ifm electronic b.v. • 3843 GA Harderwijk • Tel. +31 341 / 438 438
NZ	ifm efector pty ltd • 930 Great South Road Penrose, Auckland • Tel. +64 95 79 69 91
P	ifm electronic s.a. • 4410-136 São Félix da Marinha • Tel. +351 223 / 71 71 08
PL	ifm electronic Sp. z o.o. • 40-106 Katowice • Tel. +48 32-608 74 54
RA, ROU	ifm electronic s.r.l. • 1107 Buenos Aires • Tel. +54 11 / 5353 3436
RO	ifm electronic s.r.l • Sibiu 557260 • Tel. +40 269 224550
ROK	ifm electronic Ltd. • 140-884 Seoul • Tel. +82 2 / 790 5610
RUS	ifm electronic • 105318 Moscow • Tel. +7 495 921-44-14
S	ifm electronic a b • 41250 Göteborg • Tel. +46 31 / 750 23 00
SGP	ifm electronic Pte. Ltd. • Singapore 609 916 • Tel. +65 6562 8661/2/3
SK	ifm electronic s.r.o. • 835 54 Bratislava • Tel. +421 2 / 44 87 23 29
THA	SCM Allianze Co., Ltd. • Bangkok 10 400 • Tel. +66 02 615 4888
TR	ifm electronic Ltd. Sti. • 34381 Sisli/Istanbul • Tel. +90 212 / 210 50 80
UA	TOV ifm electronic • 02660 Kiev • Tel. +380 44 501 8543
USA	ifm efector inc. • Exton, PA 19341 • Tel. +1 610 / 5 24-2000
VN	ifm electronic • Ho Chi Minh city 700000 • Tel. +84-8-35125177
ZA	ifm electronic (Pty) Ltd. • 0157 Pretoria • Tel. +27 12 345 44 49