ifm electronic

CE

Programming Manual

PDM360 NG 7"

CR1081

CR1085

Firmware: 3.2.x

CODESYS: 3.5.9.4

English

7391075_00_UK 2017-01-10

Contents

1 Preliminary note

Contents

14801

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

7296

This document applies to the following products from ifm electronic:

- Dialogue module PDM360 NG (art. no.: CR1081/CR1085), firmware version V3.2.0.1 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 with CODESYS 3.5
- Programming of the device-internal PLC of the CR1081/CR1085 by means of the CODESYS 3.5 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 Decimal number
0x123 Hexadecimal number
0b010 Binary number

[...] Designation of pushbuttons, buttons or indications

1.4 Overview: User documentation for CR1081/CR1085

9193

ifm electronic provides the following user documentation for the models of the device class "dialogue modules PDM360 NG":

Document	Contents / Description
Data sheet	Technical data in a table
Installation instructions *	▪ Instructions for installation, electrical installation and commissioning ▪ Technical data
Programming manual	▪ Functions of the setup menu of the device ▪ Creation of a CR1081/CR1085 project with CODESYS 3.5 ▪ Target settings with CODESYS 3.5 ▪ Programming of the device-internal PLC with CODESYS 3.5 ▪ Description of the device-specific CODESYS function libraries

Legend:
*... The installation instructions are supplied with the device.

All documents can be downloaded from ifm's website:
→ www.ifm.com > Select country > [Data sheet search] > (Article no.) > [Operating instructions]

1.5 Overview: CODESYS documentation of 3S

9194

The following user documentation is provided by 3S GmbH for programming the CR1081/CR1085 with CODESYS:

Document	Contents / Description
Online help	▪ Context-sensitive help ▪ Description of the CODESYS programming system
CODESYS installation and first steps	▪ Notes about the installation of the programming system CODESYS ▪ First steps for handling the programming system CODESYS

After the installation of the programming system CODESYS 3.5 all documents are stored on the hard disk of the PC/laptop and can be accessed:

- Online help:
  `...\Programme\3S CoDeSys\CoDeSys\Online-Help`
- CODESYS installation and first steps:
  `...\Programme\3S CoDeSys\CoDeSys\Documentation`

1.6 Modification history

18020

Version	Topic	Date
00	New creation of the document	14.08.2016

2 Safety instructions

Contents

213

2.1 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.

2.2 Note!

7328
11212

No characteristics are warranted with the information, notes and examples provided in this manual. With the drawings, representations and examples given no responsibility for the system is assumed and no application-specific particularities are taken into account.

- ► 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 personal 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 at:
  → www.ifm.com > Select your country > [Data sheet search] > (article number) > [Technical data in PDF format]
- ► Note the installation and wiring information as well as the functions and features of the devices!
  → supplied installation instructions or on **ifm's** homepage:
  → www.ifm.com > Select your country > [Data sheet search] > (article number) > [Operating instructions]
- ► Please note the corrections and notes in the release notes for the existing documentation, available on the **ifm** website:
  → www.ifm.com > Select your country > [Data sheet search] > (article number) > [Operating instructions]

NOTICE

Danger of burn-in effects!

If the device is operated with the same visualisation for many hours, the image may be burned into the screen.

► Operate the device with changing visualisations!

2.3 Start-up behaviour of the controller

6827
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 watchdog response because of too long a cycle time
- error elimination after an E-stop

To ensure a safe behaviour of the controller:

► Monitor the voltage supply in the application program.

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

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

► Monitor relay contacts which can cause hazardous movements in the application program (feedback).

► If necessary, ensure that welded relay contacts in the application project cannot trigger or continue hazardous movements.

3 System requirements

Contents

16903

3.1 Hardware

11484

- Device of the product family dialogue module PDM360 NG CR1081/CR1085
  - Hardware version: CR1081: AE | CR1085: AC
  - Runtime system: VV3.2.0.1
- PC/laptop for the programming system CODESYS (→ system requirements CODESYS Development System V3.5)
- Ethernet connection between CODESYS PC/laptop and Ethernet interface of the CR1081/CR1085 (→ installation instructions)

3.2 Software

13915

The following software components are required to program the device-internal PLC of the CR1081/CR1085:

Component	Description	Version
CODESYS Development System	Programming software CODESYS for PLC programming complying with the standard IEC 61131-3	3.5 SP9 patch 4
Package CODESYS for ifm PDM360NG Products	▪ Device and interface description of CR1081/CR1085 ▪ Function libraries for the programming of the device	1.6.2.0

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

On its website ifm electronic provides the software components for download:
→ www.ifm.com > Service > Download > [Systems for mobile machines]

3.3 Licensing

11511

By buying a dialogue module PDM360 NG CR1081/CR1085 the buyer also purchases a licence valid for using the programming system CODESYS 3.5.

4 Installation

Contents

19339

4.1 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.

4.1.1 Install CODESYS Development System

18596

To install the software "CODESYS Development System":

► Install the programming system CODESYS 3.5 SP9 patch 4.
→ CODESYS installation and first steps

> CODESYS 3.5 SP9 patch 4 is installed on the programming PC/laptop.

4.2 ifm package for PDM360 NG

Contents

9888

► Familiarise yourself with the following CODESYS functions!
- Package manager
  → Online help > CODESYS Development System > Manage packages and licenses

4.2.1 Components of the package

9889

ifm provides the CODESYS package CODESYS for ifm PDM360NG Products for the programming of the device-internal PLC. The package contains the following components:

- Device description files
- ifm function libraries and extensions
- EDS files for various ifm devices

Detailed information about the ifm function libraries: → **ifm function libraries** (→ p. 144)

4.2.2 Install ifm package

7341

To install the ifm package CODESYS for ifm PDM360NG Products:

Prerequisites

> CODESYS 3.5 SP9 patch 4 is installed on the PC/laptop.
> ifm package CODESYS for ifm PDM360NG Products is stored on the PC/laptop.

1 Start CODESYS

► Start CODESYS.
> 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 ifm package

► Activate [Install...].
> The file explorer appears.
► Select file `ifm_PDM360NG_V1_6_2_0.package` and carry out a complete installation.
> Window [Package Manager] shows installed ifm package.
► Activate [Close] to close the Package Manager.
► Save the project.

4.2.3 Update ifm package

15219

To update the ifm package CODESYS for ifm PDM360NG Products:

1 Uninstall the old version of the ifm package

► **Uninstall ifm package** (→ p. 19)

2 Install the new version of the ifm package

► **Install ifm package** (→ p. 18)

3 Update device

► In the device tree: Mark node [Device (CR1081/CR1085)].
► Select [Project] > [Update Device...].
> Dialogue window appears.
► Activate [Update Device] to start the updating process.
> CODESYS loads new device libraries.
> Device tree view is updated.
► Activate [Close] to close the Package Manager.
► Save the project.

4.2.4 Uninstall ifm package

15022

To uninstall the ifm package CODESYS for ifm PDM360NG Products:

1 Start package manager

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

2 Uninstall ifm 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.
► Activate [Close] to close the Package Manager.

4.3 Update the runtime system of the device

Contents

18159

4.3.1 Notes

18162

To update the runtime system the device needs to be in recovery mode.

Download the update file

18165

The user can download the current version of the runtime system from the ifm website.

► Access the download from the ifm website.
(→ www.ifm.com > Select your country > [Service] > [Download] > [Systems for mobile machines])

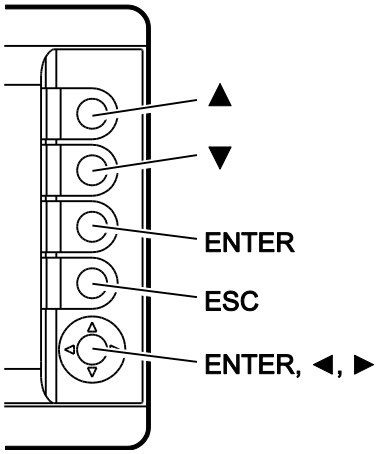
► Download the ZIP archive with the update file and unzip.

> The update file is stored on the PC/laptop (file `*.ifm`).

Key assignment in recovery mode

18163

In the recovery mode, the following key assignment applies:



Components of the runtime system

11649

During the update process the following software compknents of the device will be updated:

1. System software (Linux)
2. Firmware of the keys
3. Firmware of the I/O co-processor
4. Firmware of the watchdog co-processor

4.3.2 Enter the recovery mode

19325

► Switch off the device (disconnect device from the mains).
► **Simultaneously** process the following actions:
  ▪ Press and keep pressed any 3 keys on the device simultaneously.
  ▪ Switch on the device.
    To do so, pin 15 and pin 30 must be connected with plus.
> The device is booting.
► Keep the keys pressed until the splash screen appears.
> The ifm recovery screen appears and displays IP configuration of the device's ethernet interface (IP address, subnet mask and gateway address).
> The device is in recovery mode.

4.3.3 Update the runtime system

18161

NOTICE

Risk of data loss!

Interrupting the update process can result in a loss of the user data stored in the device.

► Do not interrupt the update process!

The runtime system of the device can only be updated in the recovery mode.

The current version of the runtime system can be downloaded from the ifm electronic website (→ **Download the update file** (→ p. 21)).

► Select one of the following options:
  ▪ **Runtime system update via USB memory device** (→ p. 23)
  ▪ **Runtime system update via web browser** (→ p. 24)

Runtime system update via USB memory device

9382

1 Download current release of runtime system

► Download new release of runtime system from the ifm website (→ **Download the update file** (→ p. 21)).

► Unzip the archive and copy the update file into the root directory of the USB memory.

Only ONE update file is allowed in root directory.

2 Connect the USB memory with the device

► Connect the USB memory with the device (→ installation instructions).

3 Enter the recovery mode

► **Enter the recovery mode** (→ p. 22)

4 Install the runtime system

► Use [▲] / [▼] to select the [Install from USB] button.

► Press [ENTER] to start the update process.

> Software components of the device are updated (→ **Notes** (→ p. 21)).

If successful:

> Status message appears: `Software update successful`.

The upgrade process may take a few minutes. Do not press any key on the device during the upgrade process!

5 Reboot the device neu starten

► Use [▲] / [▼] to select the [Reboot] button.

► Press [ENTER] to reboot the device.

> The device reboots.

Runtime system update via web browser

9379

1 Neues Laufzeitsystem herunterladen

► Download new release of runtime system from the ifm website (→ **Download the update file** (→ p. 21)).

2 Enter the recovery mode

► **Enter the recovery mode** (→ p. 22)

3 Connect the device with the PC/laptop

► Establish an Ethernet network connection between the device and the PC/laptop (→ installation instructions).

► Optional: Set the IP configuration of the ethernet interface of the PC/laptop to match the IP configuration of the device.

4 Acitvate the web interface of the device

► Start the web browser on the PC/laptop.

► Enter the IP address of the device in the address line of the browser and press [ENTER] to activate the web interface (IP-Adresse: → Schritt 2).

> Browser shows web interface of the device.

> Web interface shows content of [Software Upgrade] tab.

5 Install the runtime system

► Press the [Select...] button.

> Windows explorer appears.

► Select update file (`Vxx.yy.zz_PDM360NG.ifm`) and press [Open] to proceed.

> [Select image to upload] field shows directory path and file name of the selected update file.

► Press [Send] button to start the update process.

> Software components of the device are updated (→ **Notes** (→ p. 21)).

> Web interface and device display show status messages.

If successful:

> Status message appears: `Software update successful`.

The upgrade process may take a few minutes. Do not press any key on the device during the upgrade process!

6 Reboot the device neu starten

► Use [▲] / [▼] to select the [Reboot] button.

► Press [ENTER] to reboot the device.

> The device reboots.

4.3.4 Configure IP parameter of ethernet interface

11915

In order to update the runtime system of the CR1081/CR1085 via a network infrastructure, the device's ethernet interface needs to be configured. The device provides the following options for configuration of the ethernet interface:

- Manual — The operator sets the interface parameters (IP address, subnet mask, gateway address) manually.
- Automatic — The interface parameters are set automatically by the network's DHCP server (DHCP = Dynamic Host Configuration Protocol).

To configure the IP parameter of the ethernet interface:

► Select one of the following options:
- **Synchronize the IP parameters with DHCP sever** (→ p. 25)
- **Configure the IP parameter manually** (→ p. 26)

Synchronize the IP parameters with DHCP sever

13375

1 Connect device with IP network

► Connect ethernet interface of the device with the IP network.

2 Select menu page

► Enter the Recovery mode (→ **Enter the recovery mode** (→ p. 22)).

► Use [▲] / [▼] to select the [Network Setup] menu and press [ENTER].

> Menu page displays available configuration options and current values of the IP parameters ([IP-address], [netmask], [gateway])

3 Set IP parameters

► Use [▲] / [▼] to select the [DHCP] menu item and press [ENTER].

> Device tries to obtain the IP parameters of the ethernet interface from the DHCP server of the network.

If successful:

> [Network Setup] menu page appears.

> Information bar shows active values of the IP parameters [IP-address], [netmask] und [gateway].

> Device can be accessed via the IP network by using the shown parameters.

If not successful:

> Error message is displayed.

► Press [ESC] to return to the [Network Setup] menu page.

► Remove the error cause and repeat the procedure.

Configure the IP parameter manually

8906

Detailed information about the assignment of IP addresses in Ethernet networks: → **Address assignment in Ethernet networks** (→ p. 143).

1 Select menu page

► Enter the Recovery mode (→ **Enter the recovery mode** (→ p. 22)).
► Use [▲] / [▼] to select the [Network Setup] menu and press [ENTER].
> Menu page displays available configuration options and current values of the IP parameters ([IP-address], [netmask], [gateway])
► Use [▲] / [▼] to select the [Manual Setup] menu item and press [ENTER].
► Menu page displays options for manually setting of the IP parameters of the ethernet interface.

2 Set IP address

► Use [▲] / [▼] to select the [IP Address] menu item and press [ENTER].
> Menu page displays [IP address] input boxes.
> Right input box is in edit mode (dark-grey background).
► Use [▲] / [▼] to increment /decrement the number in the input box until the required value is set.
> Input box displays the new value.
► Use [◀] / [▶] to select the next input box and use [▲] / [▼] to enter the required number.
► Repeat the procedure until the required IP adress is set.
► Press [OK] to apply the new values and to return to the previous menu page.
OR
Press [ESC] to cancel to input procedure, to keep the old values and to return to the previous menu page.

3 Set subnetmask and gateway addess

► Repeat step 1 to set [Network Mask] and [Default Gateway].

4 Apply new values

► Use [▲] / [▼] to select [Apply] menu item and press [ENTER].
> New values are applied.
► Press [ESC] to return to the [Network Setup] menu page.
> Information bar displays current values of the interface parameters [IP-address], [netmask] und [gateway].

4.3.5 Quit the recovery mode

082

► Select ifm recovery screen.
► Use [▲] / [▼] to select the menu item [Reboot] and press [ENTER] to activate it.
► The device reboots.

If there is a valid project stored on the device, this project will be startet automatically after completing the reboot process.
If there is no valid project stored on the device, the CR1081/CR1085 automatically enters the setup mode after completing the reboot process.

5 Device configuration

Contents

19278

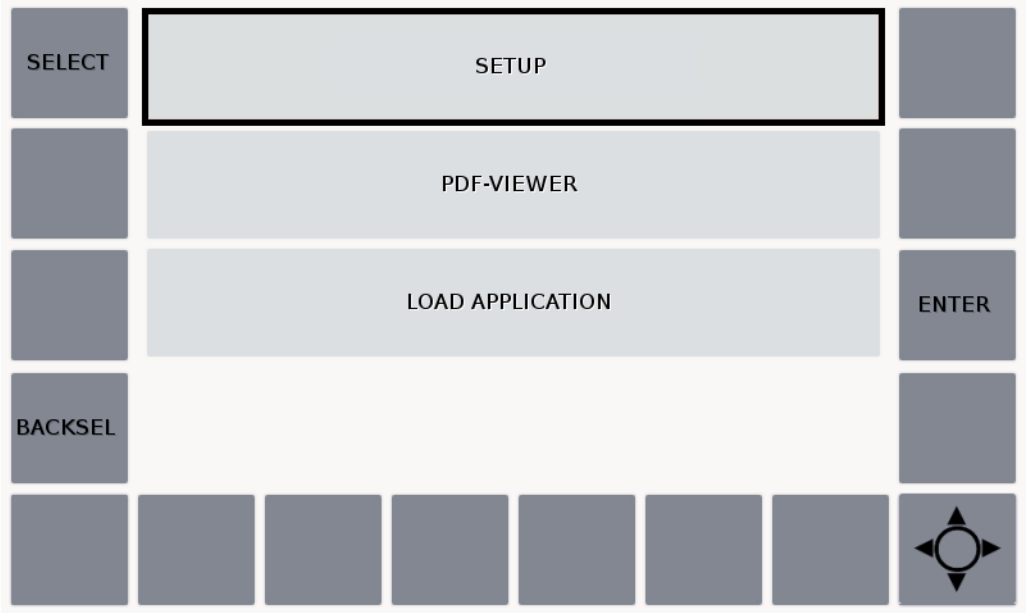
This chapter describes the configuration of the device in the setup mode.

5.1 Enter the setup mode

18152

If no valid application is stored in the device, the device passes automatically in the setup mode after power on.

► Disconnect the device from power supply.
► Perform the following actions simultaneously:
  ▪ Press any 2 keys **simultaneously** and keep them pressed.
  ▪ Switch on the device.
► Keep both keys pressed until the setup menu appears.
> Device boots and passes into the setup mode.
> The start screen of the setup mode appears.



► Select one of the following submenus:

Submenu	Description
[SETUP]	Start setup menu (→ **Setup: start screen** (→ p. 39))
[PDF-VIEWER]	Start the PDF file viewer (→ **PDF file viewer** (→ p. 63))
[LOAD APPLICATION]	Load and start the PLC application (→ **Setup: Start the application** (→ p. 68))

5.2 Operating instructions

Contents

18108

In setup mode, the following rules apply:

5.2.1 Key functionality

18155

The assignment of the function keys is context sensitive. The function of a key is described by the text label of the corresponding button on the screen.

Only active function keys have a labeled button.

Only active function keys are backlit by the key LED.

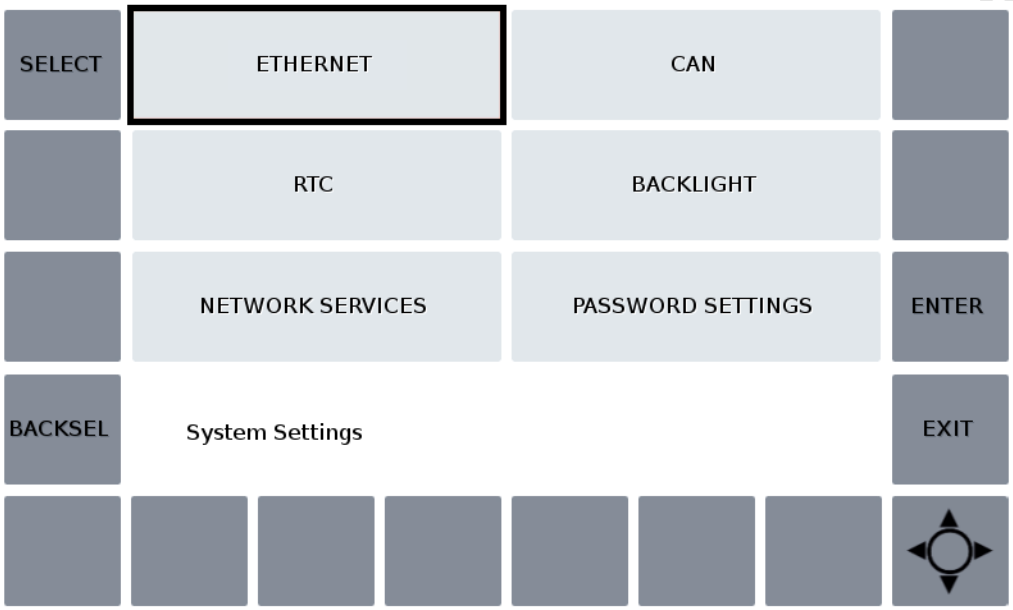
The following table provides an overview of the available key functions:

Button	Description	Example
[SELECT]	Select next GUI element	Select next menu item
[BACKSEL]	Select previous GUI element	Select previous menu item
[ENTER]	Activate function of the selected GUI element	Change to selected menu Show content of the selected directory
[EXIT]	Change to upper menu level	Leave current menu page and change to upper menu level
[LOAD]	Load selected GUI element	Load font file in the device memory
[RELOAD]	Update	Update view of current directory
[SAVE]	Save selected file or configuration	Save IP address
[START]	Start process	Start receiving CAN frames
[STOP]	Stop process	Stop receiving CAN frames
[▲], [▼], [◀], [▶]	▪ move to direction ▪ Increment/decrement value in selected number field	Select character of on-screen keyboard

Operation: Navigate in the setup menu

14674

To navigate in the setup menu:



► Use [SELECT] to select the next menu item (forward).
OR:
Use [BACKSEL] tot select the previous menu item (backward).

> The selected menu item has a black frame.

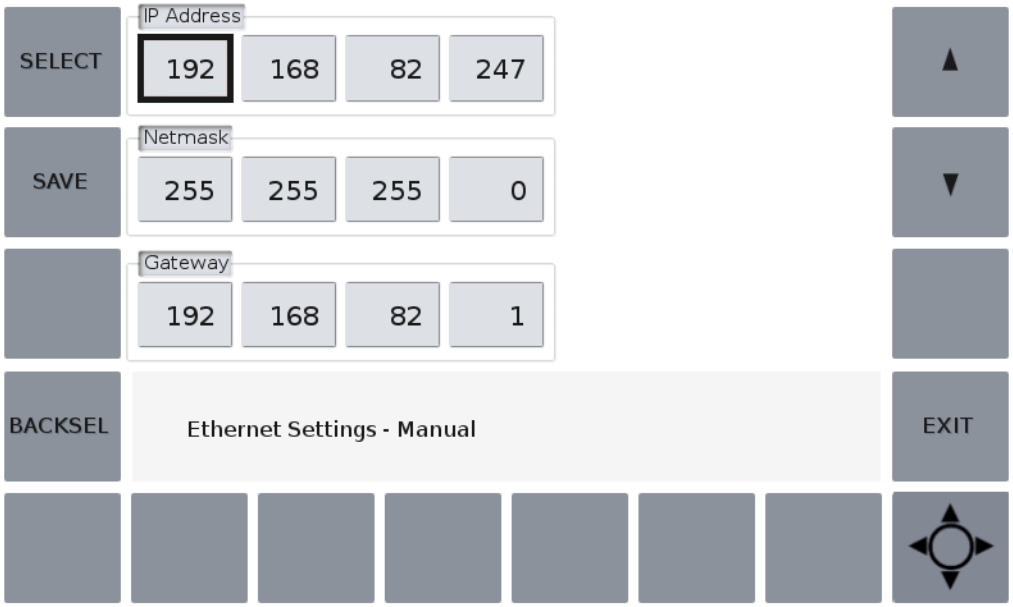
► Use [ENTER] to change to the selected menu page.
OR:
Use [EXIT] to change to the higher-level menu.

> The information field shows information about the current menu page (i.e. `System settings`).

Operation: Navigate within a menu screen

14675

To navigate within a menu page:

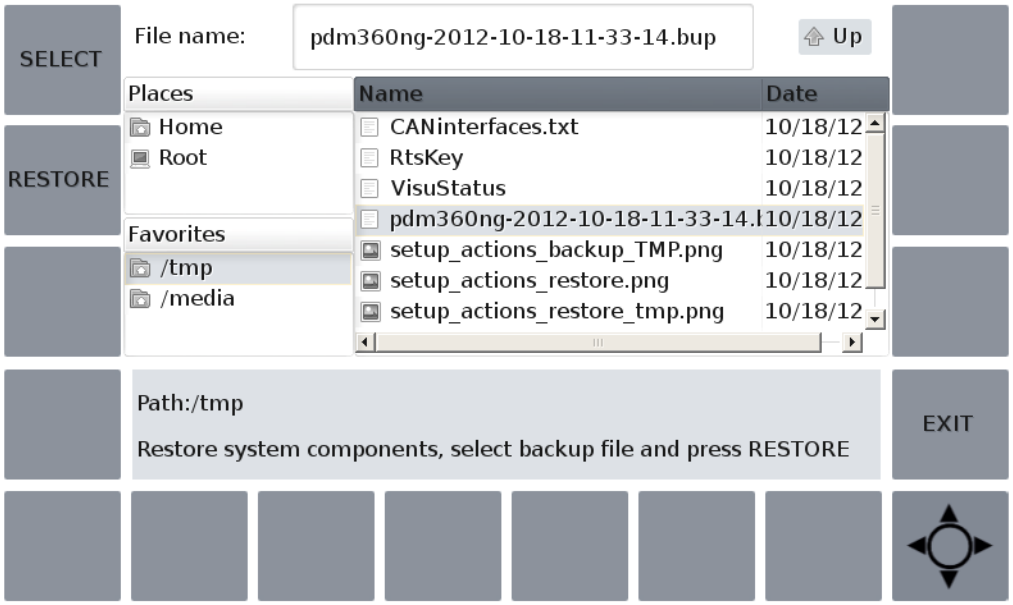


► Use [SELECT] or [BACKSEL] to select the requested GUI element (i.e. number field, option field).
> The selected GUI element has a black frame.
> The selected GUI element is activated and can be changed.
► Use [ENTER] to activate / deactivate the selected GUI element (i.e. option field)
OR:
Use [▲]/[▼] to increment/decrement the value of the selected GUI element (i.e. number field).
> The changes are displayed.
► Use [SAVE] to confirm and apply the changes.
► Use [EXIT] to change to the higher-level menu.

Operation: Navigate in the file directory

14673

To navigate in the file directory:



► Use [SELECT] to select a specific window (i.e. [Name], [Places], [Favorites] and [UP])
> The selected window is focused (black frame or dark-grey header).
► Use [▲]/[▼] to select an specific item within the selected window (i.e. file, directory).
> The selected item has a light-grey background.
► Use [ENTER] to change into the selected directory.
OR
Start action according to key label (i.e. Use [RESTORE] to restore the data saved in the selected backup file).
> Selected action is proceeded.
> Use [EXIT] to change to the higher-level menu.

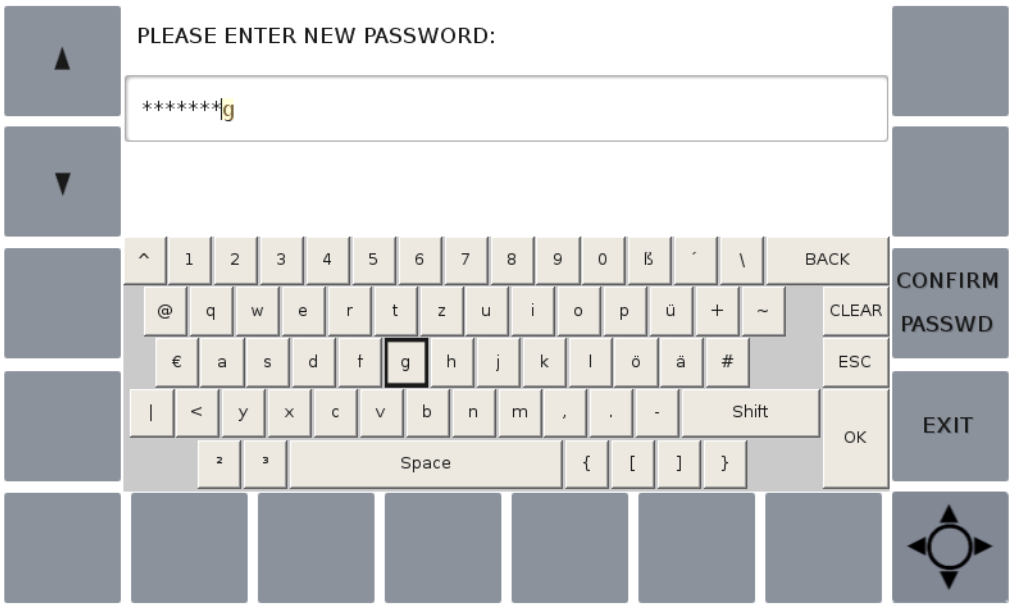
Operation: Enter a text using the on-screen keyboard

11763

The user interface of the runtime system has an on-screen keyboard. It appears automatically as soon as the user is expected to enter characters.

The on-screen keyboard shows the German character set (QWERTZ keyboard layout).
To protect sensitive information only the character entered last appears in plain text. All other characters are replaced by an *.

To enter a text:



► Use [▲] / [▼] and [▶] / [◀] to select the button of the requested character.
> The selected button is highlighted (black frame).
► Use [ENTER] to activate the selected button.
> The entered character appears in the input box.
► Repeat the process for all requested characters.
► Use [CONFIRM PASSWD] to confirm the entry
OR:
Use [EXIT] to cancel and return to the previous setup screen .

5.2.2 Navigation key

18158

The navigation key consists of the following single keys:

- 4 direction keys (▼, ▲, ◀, ▶)
- 1 function key (center key)

The keys of the navigation key can be used as an alternative as well as an equivalent to function keys. The following table shows operating elements with the same function:

Navigation key	Function key	Function
	[▲]	Move up
	[◀]	Move left
	[BACKSEL]	Select previous menu item
	[▶]	Move right
	[SELECT]	Select next menu item
	[▼]	Move down
	[ENTER]	Activate menu item / button

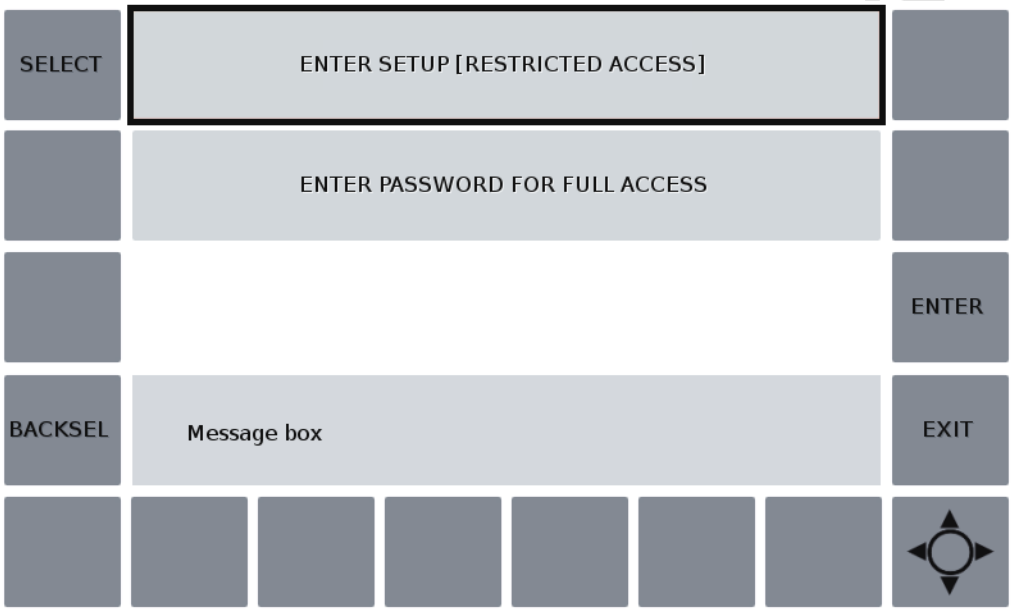
5.2.3 Setup: Working with enabled password protection

11914

If the password protection is enabled, the operator has only accress to the following menu functions:

- [SETUP] > [System Information]
- [SETUP] > [ACTIONS] > [Keyboard Test]

All other menu functions are disabled.
Additionally, after changing to the setup menu the display shows the following setup start screen:



Select one of the following options:

- **Setup: Setup menu with restricted access rights** (→ p. 38)
- **Setup: Enter the password for full access to the setup menu** (→ p. 38)

Setup: Setup menu with restricted access rights

11729

Protected setup functions have the following characteristics:

- They are disabled in all setup submenus (greyed out).
- They cannot be selected with [SELECT] / [BACKSEL].

To call the setup menu with restricted access rights:

1 Select menu page

► Navigation path: [SETUP] > [ENTER SETUP [RESTRICTED ACCESS]]

> The setup menu appears

> Disabled functions are greyed out.

2 Select menu item

► Use [SELECT] / [BACKSEL] to select the requested menu item .

> The selected menu item has a black frame.

► Use [ENTER] to change to the selected setup screen.
OR:
Use [EXIT] return to the previous setup screen.

Setup: Enter the password for full access to the setup menu

11759

To enter the password to have full access to the setup menu:

1 Select menu page

► Navigation path: [SETUP] > [ENTER PASSWORD FOR FULL ACCESS]

> Display shows on-screen keyboard for entering the password.

2 Enter passwort

► Enter the correct password (→ **Operation: Enter a text using the on-screen keyboard** (→ p. 35)).

► Use [CONFIRM PASSWD] to confirm the entry
OR:
Use [EXIT] to cancel and return to the previous setup screen .

> The setup menu appears.

> Information bar displays a confirmation message.

> All setup functions can be selected without any restriction.

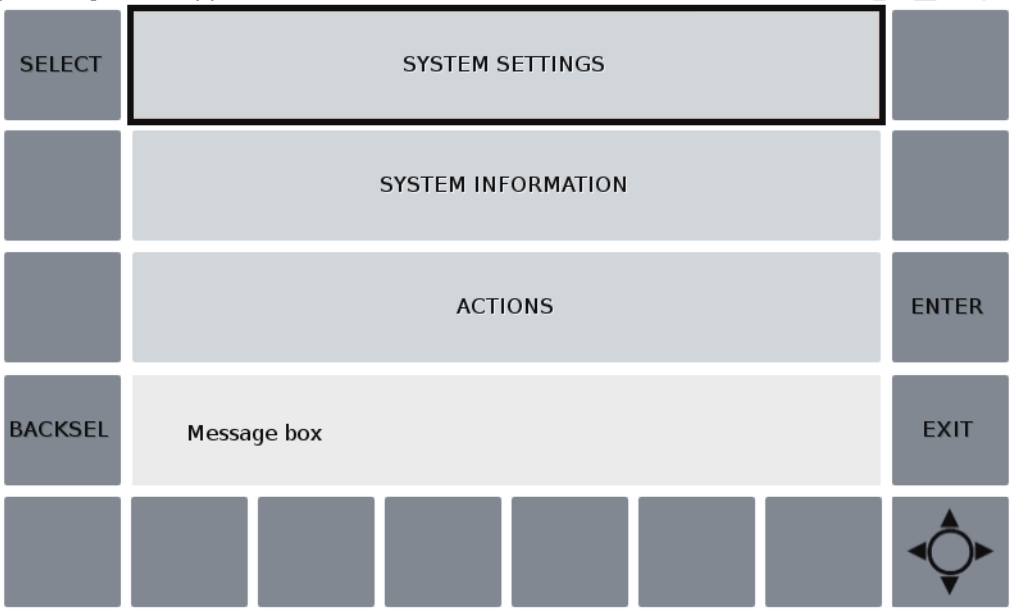
Full access to all setup function is only valid until the user exits the setup menu.

► When the setup menu is called, enter the password again!

5.3 Setup: start screen

14556

► Select [SETUP] and press [ENTER].
> [SETUP] menu appears:



► Select one of the following menu items:

Name	Description
[SYSTEM SETTINGS]	Access system settings (→ **Setup: System settings** (→ p. 40))
[SYSTEM INFORMATION]:	Display system information: (→ **Setup: Display system information** (→ p. 49))
[ACTIONS]	Access additional functions (→ **Setup: More functions** (→ p. 50))

Consider notes regarding the work with enabled password protection!
→ **Setup: Working with enabled password protection** (→ p. 37)

5.3.1 Setup: System settings

9603

The menu [SETUP] > [SYSTEM SETTINGS] provides access to the following functions:

Menu item	Function
[ETHERNET]	→ **Setup: Configure the IP parameters manually** (→ p. 41)
[CAN]	→ **Setup: Set the CAN interfaces** (→ p. 43)
[RTC]	→ **Setup: Set the real-time clock (RTC)** (→ p. 44)
[BACKLIGHT]	→ **Setup: Set the display brightness** (→ p. 45)
[NETWORK SERVICES]	→ **Setup: Set the network services** (→ p. 46)
[PASSWORD SETTINGS]	→ **Setup: Set the password protection** (→ p. 47)

Setup: Configure the Ethernet interface

13857

The device provides the following options for configuration of the Ethernet interface:

- Manual — The operator sets the interface parameters (IP address, subnet mask, gateway address) manually.
- Automatic — The interface parameters are set automatically by the network's DHCP server (DHCP = Dynamic Host Configuration Protocol).

To configure the IP parameter of the ethernet interface:

► Select one of the following options:
  - **Synchronize the IP parameters with DHCP sever** (→ p. 25)
  - **Configure the IP parameter manually** (→ p. 26)

Setup: Configure the IP parameters manually

7408

- IP address (factory settings) = 192.168.82.247
- Login data to access the device via Telnet/SSH and FTP/SCP:
  USERNAME = root
  PASSWORD = pdm360ng
- Further information regarding the address assignment in Ethernet networks:
  → **Address assignment in Ethernet networks** (→ p. 143).

To set the IP parameters of the Ethernet interface manually:

1 Select menu page

► Navigation path: [SETUP] > [SYSTEM SETTINGS] > [ETHERNET] > [MANUAL]

> Menu page displays the following information:

Name	Description	Possible values
[IP Address]	IP address of the Ethernet interaface of the device	i.e. `192.68.82.247`
[Netmask]	Subnetmask of the network segment	i.e. `255.255.255.0`
[Gateway]	IP address of the network gateway	i.e. `192.168.82.1`

2 Set IP parameters

► Use [SELECT] / [BACKSEL] to select the required number field.

► Use [▲] / [▼] to increment / decrement the number in the selected number field until the requested value has been reached.

> Number field shows the required value.

► Optional: Repeat the procedure until the values in all number fields are correctly set.

3 Save changed values

► Press [SAVE] to confirm and save the changed values.

Setup: Configure the IP parameters automatically

13791

To use DHCP, the device must be connected to a DHCP server via the Ethernet interface.

To set the IP parameters of the Ethernet interface by means of the Dynamic Host Configuration Protocol (DHCP).

1 Select menu page

► Navigation path: [SETUP] > [SYSTEM SETTINGS] > [ETHERNET] > [DHCP]

> Menu page displays the following information:

Bezeichung	Bedeutung	Mögliche Werte	
[DHCP Client]	Status of DHCP client of the device	☐	DHCP client disabled
		☑	DHCP client enabled
[IP Address]	IP address of the Ethernet interface of the device	i.e. `192.68.82.247`	
[Netmask]	Subnetmask of the network segment	i.e. `255.255.255.0`	
[Gateway]	IP address of the network gateway	i.e. `192.68.82.1`	

2 Enable DHCP client of the device

► Press [ENTER] to enable the [DHCP Client] checkbox.

> The DHCP client tries to obtain valid IP parameters from the DHCP server.

> Status message appears: `Trying to get DHCP lease...`

If successful:

> [IP Address], [Netmask] and [Gateway] show valid values assigned by the DHCP server.

> Status message appears: `DHCP configured successfully`

If not successful:

> Status message appears: `Failed to get DHCP lease!`

► Use [EXIT] to stop the configuration process.

► Check the connection to the DHCP server!

► Repeat the configuration process (s.a.).

Setup: Set the CAN interfaces

7413

To set the parameters of the CAN interfaces:

1 Select menu page

► Navigation path: [SETUP] > [SYSTEM SETTINGS] > [CAN]

> Menu page displays the following informations:

Name	Description
[CAN Bus 1]	Settings of of CAN interace 1
[CAN Bus 2]	Settings of of CAN interface 2
[CAN Bus 3]	Settings of of CAN interface 3
[CAN Bus 4] *	Settings of of CAN interface 4

* ... only available at devices with 4 CAN interfaces

2 Select CAN interface

► Use [SELECT] / [BACKSEL] to select the requested CAN interface.

► Press [ENTER] to change to the selected menu page.

> Menu page displays the following informations:

Name	Description	Possible values
[Baudrate CAN Bus x] *	Baudrate of the selected CAN interface	0 kBit/s 20 kBit/s 50 kBit/s 100 kBit/s 125 kBit/s 250 kBit/s 500 kBit/s 800 kBit/s 1 MBit/s

* ... x = ID of the selected CAN interface

3 Change baudrate of the CAN interface

► Press [ENTER] to change to the selected menu page.

> Menu page displays the current baudrate of the selected CAN interface.

► Use [▲] / [▼] to set the required value.

► Press [SAVE] to confirm and save the changed values.

► Press [EXIT] to return to the upper next menu level.

> Optional: Repeat steps 2 and 3 to set the values of additional CAN interfaces.

Setup: Set the real-time clock (RTC)

7414

To set the real-time clock.

1 Select menu page

► Navigation path: [SETUP] > [SYSTEM SETTINGS] > [RTC]

> Menu page displays the following informations:

Name	Description	Possible values
[YYYY]	Year	2000 ... 2100
[MM]	Month	1 ... 12
[DD]	Day	1 ... 31
[hh]	Hour	0 ... 23
[mm]	Minute	0 ... 59
[ss]	Second	0 ... 59

2 Change date and time of the real-time clock (RTC)

► Use [SELECT] / [BACKSEL] to select the required number field.

> Selected number field has focus (black frame).

► Use [▲] / [▼] to set the required value.

► Optional: Repeat step 2 to set the values of additional number fields.

3 Save changed values

► Press [SAVE] to confirm and save the changed values.

> Information bar displays acknowledge message.

► Press [EXIT] to return to the upper next menu level.

Setup: Set the display brightness

7405

The setup menu is always displayed with a 100 % brightness. The changed brightness will only become effective outside the setup menu.

So set the display brightness:

1 Select menu page

► Navigationspfad: [SETUP] > [SYSTEM SETTINGS] > [BACKLIGHT]

> Men u page displays the following informations:

Name	Description	Possible values	
[Brightness in %]	Relative brightness of the display (values at %)	0 ... 100	minimal brightness maximal brightness

2 Set brightness level of display

► Use [▲] / [▼] to set the required value.

► Press and keep pressed the [TEST] button to test the set value.

3 Save the new value

► Press [SAVE] to confirm and save the changed values.

> Information bar displays acknowledge message.

► Press [EXIT] to return to the upper next menu level.

Setup: Set the network services

11719

The secure network services Secure Shell (SSH) and Secure Copy (SCP) are always available. The user can therefore access the device via SSH and SCP if the network services Telnet and FTP are inactive.
(→ **Access the device with SSH**, → **Copy the files with SCP**).

To set the Telnet and FTP network services.

1 Select menu page

► Navigation path: [SETUP] > [SYSTEM SETTINGS] > [NETWORK SERVICES]

► Menu page displays the following informations:

Name	Description	Possible values	
[Telnet]	Enable / Disable the Telnet service of the device	☑	Telnet enabled
		☐	Telnet disabled
[FTP]	Enable / Disable the FTP service of the device	☑	FTP enabled
		☐	FTP disabled

2 Enable / Disable Telnet service

► Use [SELECT] / [BACKSEL] to select the [Telnet] check box.

► Press [ENTER] to enable / disable the the Telnet service.

> Information bar displays status of the service:
[running] = Telnet service is enabled
[not running] = Telnet service is disabled

3 Enable / Disabled FTP service

► Use [SELECT] / [BACKSEL] to select the [FTP] check box.

► Press [ENTER] to enable / disable the the FTP service.

> Information bar displays status of the service:
[running] = FTP service is enabled
[not running] = FTP service is disabled

> Press [EXIT] to return to the upper next menu level.

Setup: Set the password protection

11720

The menu page [SETUP] > [SYSTEM SETTINGS] > [PASSWORD SETTINGS] provides access to the following options:

- **Setup: Change the system password** (→ p. 47)
- **Setup: Enable / Disable password protection** (→ p. 48)

Setup: Change the system password

11722

In case of a password loss a recovery update has to be processed.

► Update the runtime system of the device (→ **Update the runtime system of the device** (→ p. 20))

► If necessary, restore the saved data (→ **Restore: Restore the system files** (→ p. 59))

Notes on operation of the on-screen keyboard: → **Operation: Enter a text using the on-screen keyboard** (→ p. 35)

To change the system password.

1 Select menu page

► Navigationspfad: [SETUP] > [SYSTEM SETTINGS] > [PASSWORD SETTINGS] > [Change System Password]

> Menu page displays on-screen keyboard.

2 Enter the old password

► Enter the old password.

► Use [CONFIRM] to confirm the input.

Gibt der Anwender das falsche Passwort ein, erscheint folgende Fehlermeldung: `PASSWORD INCORRECT. PLEASE TRY AGAIN.`

► Eingabe des Passwort wiederholen!

3 Enter new password

► Enter new password.

► Use [CONFIRM] to confirm the entered password.

► Re-enter the new password and use [CONFIRM] to confirm the entered password.

> Information bar displays acknowledge message: `PASSWORD CHANGED`.

> New password is applied.

If the operator enters 2 different passwords the following error message appears: `PASSWORDS DON'T MATCH. PLEASE TRY AGAIN!`

► Re-enter the passwords correctly!

► Press [EXIT] to return to the upper next menu level.

Setup: Enable / Disable password protection

11418

On delivery the password protection for the setup menu is deactivated.

If the password protection is activated after the first setup, the password set at the factory is valid: `pdm360ng`

► To ensure effective protection, change the default system password!

The programming software CODESYS can access the IEC project saved on the device even with activated password protection.

If the password protection is enabled the operator has restricted access to the functions of the setup menu (→ **Setup: Working with enabled password protection** (→ p. 37)).

To enable / disable the password protection of the device:

1 Select menu page

► Navigation path: [SETUP] > [SYSTEM SETTINGS] > [PASSWORD SETTINGS]

> Information bar displays status of the password protection:
[Setup Password is now enabled] = Password protection is enabled
[Setup Password is now disabled] = password protection is disabled

2 Set password protection

► Use [SELECT] / [BACKSEL] to select the [Enable Setup Password] / [Disable Setup Password] button.

► Use [ENTER] to press the button.

> Password protection is enabled / disabled.

> Infomation bar displays status of the password protection (→ step 1).

► Press [EXIT] to return to the upper next menu level.

The password protection does not become active before the next call of the setup menu.

► Use [EXIT] repeatedly to change to the startscreen of the setup mode.

► Use [ENTER] to enter the setup menu.

5.3.2 Setup: Display system information

7407

To display the system information:

1 Select menu page

► Navigation path: [SETUP] > [SYSTEM INFORMATION]

> Menu page displays the following information:

Name	Description
[Serial Number]	Serial number of the device
[SW Version]	Software version
[FW Version WD]	Firmware version of the watch-dog component
[FW Version IO]	Firmware version of the I/O interface
[FW Version Keyboard1]	Firmware version of the keyboard 1
[FW Version Keyboard2]	Firmware version of the keyboard 2
[FW Version Keyboard3]	Firmware version of the keyboard 3
[MAC Address]	MAC address of the Ethernet interface
[Supply Voltage]	Value of the supply voltage
[Internal Voltages]	Internal voltage values
[System Temperatures]	Temperatures in the system
[Memory Alloc Sitze/Used]	Memory size / memory used

5.3.3 Setup: More functions

7409

The submenu [SETUP] > [ACTIONS] provides access to the following functions:

Menu item	Function
[REBOOT]	→ **Setup: Reboot the device** (→ p. 51)
[CAN TEST]	→ **Setup: Test the CAN interfaces** (→ p. 51)
[SPLASH SCREEN]	→ **Setup: Define the splash screen** (→ p. 54)
[KEYBOARD TEST]	→ **Setup: Test input elements** (→ p. 55)
[BACKUP/RESTORE]	→ **Setup: Backup / Restore software components** (→ p. 56)
[USB REMOVE]	→ **Setup: Safely remove the USB device** (→ p. 60)
[FONTS]	→ **Setup: Manage fonts** (→ p. 61)
[FORMAT STORAGE]	→ **Setup: Format the internal storage** (→ p. 62)

Setup: Reboot the device

7410

To reboot the device:

1 **Select menu page**
   ► Navigation path: [SETUP] > [ACTIONS] > [REBOOT]
   > [Reboot] menu page appears.
2 **Reboot the device**
   ► Press [ENTER] to reboot the device.
   > Device reboots.

If a valid project is loaded, this project is started after a reboot.
If no valid project is saved, the setup splash menu appears again after rebooting.

Setup: Test the CAN interfaces

7411

To test the CAN interfaces of the device, process one of the following functiions:

- **CAN test: Transmit data** (→ p. 52)
- **CAN test: Receive data** (→ p. 53)

CAN test: Transmit data

9625

To test transmitting CAN messages:

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [CAN TEST]

> Menu page [CAN Bus Selection] appears.

2 Select CAN interface

► Use [SELECT] / [BACKSEL] to select the requested CAN interface and press [ENTER] to confirm.

> Menu page [CAN Test Selection] appears.

3 Set test message and sender ID

► Use [SELECT] / [BACKSEL] to select the [CAN Transmit Test, Bus x] button and press [ENTER] to confirm.

> Menu page displays the following information.

Name	Description	Possible values
[Tx Bytes (hex)]	CAN test message (8 bytes) Every byte can be set seperatly (hexadecimal number).	byte: 00 = 0 ... FF = 255
[CAN Tx-ID (hex)]	CAN-ID of the sender (hexadecimal number)	00000000 ... FFFFFFFF
[Tx Frame Counter]	Number of transmitted messages	

► Set the test CAN message to be sent in hex format in [Tx Bytes (hex)].

► Set the test CAN message to be sent in hex format in [Tx Bytes (hex)].

4 Transmit test message

► Press [START] to start the transmission.

> Device tries to send the test message on the selected CAN interface.

> The number of the CAN messages sent since the beginning of the transfer appears in [Tx Frame Counter].

If successful:

> Information bar displays confirmation message.

► Press [STOP] to finish the cyclic transmission of the test message.

If not successful:

> Information bar displays error message.

► Press [STOP] to finish the cyclic transmission of the test message.

► Check the CAN interface connection.

► Repeat the procedure.

If no baudrate of the CAN interface was set the test will be processed with 125 Kbits/s.

CAN test: Receive data

9627

To test receiving CAN messages:

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [CAN TEST]

> [CAN Bus Selection] menu page appears.

2 Select CAN interface

► Use [SELECT] / [BACKSEL] to select the requested CAN interface and press [ENTER] to confirm.

> [CAN Test Selection] menu page appears.

3 Receive CAN message

► Use [SELECT] / [BACKSEL] to select the [CAN Receive Test, Bus x] button and press [ENTER] to confirm.

> Menu page displays the following informations:

Name	Description	Possible values
[Rx Bytes (hex)]	Received CAN message (8 bytes); Each byte containes a hexadecimal value.	byte: 00 = 0 ... FF = 255
[CAN Rx-ID (hex)]	CAN-ID of the receiver (hexadecimal value)	00000000 ... FFFFFFFF
[Rx Frame Counter]	Number of received messages	

► Press [START] to start receiving messages on the selected CAN interface. (→ note)

> [Rx Bytes (hex)] displays the content of the received messages (hexadecimal value).

> [CAN Rx-Id (hex)] displays CAN ID related to the revceived message (hexadecimal value).

> [Rx Frame Counter] displays the number of messages received sicne the start of the transmission.

If successful:

> [Rx Frame Counter] displays value > 0.

► Press [STOP] to finish receiving CAN messages.

If not successful:

> [Rx Frame Counter] displays value 0.

► Pres [STOP] to finish receiving CAN messages.

► Check the CAN interface connection.

► Repeat the procedure.

If no baudrate of the CAN interface was set the test will be processed with 125 Kbits/s.

Setup: Define the splash screen

9629

Specifications for the image file, which is used as splash screen → **image properties** (→ p. 107)

To change the splash screen of the device:

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [SPLASH SCREEN]

2 Select source of image file

> Menu page displays the following options:

Name	Description
[Load from USB device]	Load the image file from a USB device
[Load from /tmp]	Load the image file from the directory `/tmp` of the device
[Load from ...]	Load the image file from a selectable directory

► Use [SELECT] / [BACKSEL] to select the requested menu item and press [ENTER] to confirm.

> Menu page displays the selected image source.

3 Set splash screen

► Select the requested image file.

> Selected image file has focus.

► Use [LOAD] to save the file in the device and to use the image as splash screen.

Specifications of the splash screen image

11643

The user can define a user-defined splash screen. The image file has the following restricitions:

Parameter	Restriction
File type	Bitmap (*.bmp) RLE compressed
File name	Naming convention = 8.3, only small characters
Resolution	800 x 480 Pixel
Colours	262.144 (18 Bit)
File size	≤ 256 kByte

Setup: Test input elements

13871

To test the function of the keys, the key LEDs and the navigation switch:

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [KEYBOARD TEST]

> [Keyboard Test] menu page appears.

2 Test function keys

► Press any function key and keep it pressed.

> The key-related area on the screen is displayed brighter.

> The key LED is off.

3 Test navigation switch

► Use [OK] or [◀] / [▶] to switch between the two input fields of the group [Navigation Switch Test].

> Selected field gets a black frame.

► Use [▲] / [▼] to increment / decrement the value displayed in the selected field.

Setup: Backup / Restore software components

7412

To backup data stored on the device as well as to restore data to the device, the following options are available:

Name	Description
[BACKUP SYSTEM INFORMATION]	Save the system information (→ **Backup: Save the system information** (→ p. 56))
[BACKUP SYSTEM]	Save the system software (→ **Backup: Save the system files** (→ p. 57))
[RESTORE SYSTEM]	Restore the system software (→ **Restore: Restore the system files** (→ p. 59))

Backup: Save the system information

9648

Contents of the file → **Setup: Display system information** (→ p. 49).

To save the system information:

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [BACKUP/RESTORE]

> [System Backup & Restore] menu page appears.

► Select [BACKUP SYSTEM INFORMATION] and press [ENTER] to confirm.

2 Select memory location

> Menu page displays the following options:

Name	Description
[Save to USB device]	Save data to a USB device
[Save to /tmp]	Save data to the device directory `/tmp`
[Save to ...]	Save data to a selectable directory of the device

► Use [SELECT] / [BACKSEL] to select the requested menu item and press [ENTER] to confirm.

> Menu page displays the content of the selected memory location.

3 Save the system information

► Select requested directory.

► Press [SAVE] to save the system information.

> Devices saves the file containing the system information in the selected directory.

Backup: Save the system files

9637

⚠ WARNING

If the backup and the restore device have different runtime versions, the restoration of a backup file can lead to an undefinable system behaviour of the device!

The runtime versions on the backup and restore devices must be identical.

► Ensure that the runtime systems on the backup and restore device are of the same version! (→ **Update the runtime system of the device** (→ p. 20))

To save the system files to a selectable location:

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [BACKUP/RESTORE]

> [System Backup & Restore] menu page appears.

► Select [BACKUP SYSTEM] and press [ENTER] to confirm.

2 Select software components

> Menu page displays the following options:

Name	Description
[Select All]	Enable all software components
[Backup Ethernet Settings]	Settings of the Ethernet network adapter
[Backup Backlight Settings]	Settings of the background illumination
[Backup Splash Screen]	Splash screen
[Backup Retain Data]	Retain data
[Backup User Filesystem (/home)]	User file system
[Backup Internal Mass Storage (/data)]	Mass storage contents

► Use [SELECT] / [BACKSEL] to select the required components and press [ENTER] zu enable / disable.
☐ = Exclude software component from backup file
☑ = Include software component in backup file

► Optional: Repeat procedure for additional software components.

Check [Select All] to include all software components in the backup file.

► Press [CONFIRM] to confirm the selection.

3 Select memory location for backup file

> Menu page displays the following options:

Name	Description
[Save to USB device]	Save data to a USB device
[Save to /tmp]	Save data to directory `/tmp`
[Save to ...]	Save data to a selectable directory

► Use [SELECT] / [BACKSEL] to select the requested menu item and press [ENTER] to confirm.

> Menu page displays content of the selected memory location.

4 Save backup file

► Optional: Select required target directory.

► Press [SAVE] to save the selected system files in a backup file (`.bup`).

> Device saves backup file in selected directory.

► If required,press [ABORT] to cancel the backup process.

Restore: Restore the system files

9636

⚠ WARNING

If the backup and the restore device have different runtime versions, the restoration of a backup file can lead to an undefinable system behaviour of the device!

The runtime versions on the backup and restore devices must be identical.

► Ensure that the runtime systems on the backup and restore device are of the same version! (→ **Update the runtime system of the device** (→ p. 20))

► Use the option "Auto Restore" for the automatic restoration of system files from a backup file (→ **Automatically restore system files** (→ p. 69)).

To restore the system files from a backup file (*.bup):

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [BACKUP/RESTORE]

> [System Backup & Restore] menu page appears.

► Select [RESTORE SYSTEM] and press [ENTER] to confirm.

2 Select backup file

► Menu page displays the following options:

Name	Description
[Restore from USB device]	Restore data from USB device
[Restore from /tmp]	Restore data from directory `/tmp`
[Restore from ...]	Restore data from a selectable directory

► Use [SELECT] / [BACKSEL] to select the memory location of the backup file and press [ENTER] to confirm.

> Menu page displays content of selected memory location.

► Select backup file.

► Press [RESTORE] to load the selected backup file.

3 Select software components

> Menu page displays list of software components included in the backup file.

Software components not included in the backup file are geyed out!

► Use [SELECT] / [BACKSEL] to select the software components to be restored and press [ENTER] to check / uncheck the option field:
☐ = Exclude software component from restoring process
☑ = Include software component in restoring process

► Optional: Repeat procedure for additional software components.

► Press [CONFIRM] to confirm the selection.

> Device restores the selected software components.

> Status bar displays progess of restoring process.

► If required, press [EXIT] to cancel the restoring process.

Setup: Safely remove the USB device

9631

NOTICE

Risk of data loss on the USB device!

► Unmount USB devices before removing them from the runtime system!

► Do not remove the USB device before unmounting has been confirmed!

To safely remove a USB device:

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [USB REMOVE]

2 Remove the USB device from runtime system

> Menu page displays directory `/media` containing all mounted USB devices (grey background).

► Use [▲] / [▼] to select the requested USB device.

► Press [REMOVE] to unmount the selected USB device from the file system.

> USB device is unmounted.

> Informaation bar displays confirmation message.

> Operator can safetly disconnect the USB device from the system.

Setup: Manage fonts

14372

To manage font files (True Type Fonts) on the device, the following options are available:

Name	Description
[LOAD FONT]	Install font files on the device (→ **Fonts: Load the font** (→ p. 61))
[REMOVE FONT]	Remove font files stored on the device (→ **Fonts: Remove the font** (→ p. 61))

Fonts: Load the font

13866

To install a font file (true type fonts) on the device:

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [FONTS] > [LOAD FONT]

2 Select source of the font file

> Menu page displays the following options:

Name	Description
[Load from USB device]	Load font file from USB device
[Load from /tmp]	Load font file from `/tmp`-directory
[Load from ...]	Load font file from a selectable directory

► Use [SELECT] / [BACKSEL] to select the requested menu item and press [ENTER] to confirm.

> Menu page displays selected source.

3 Install font file

► Select the requested font file.

> Selected font file has focus.

► Press [LOAD] to save the selected font file on the device.

> After rebooting the device the installed font file can be used in CODESYS projects.

► Optional: Repeat steps 2 and 3 to install additional font file on the device.

Fonts: Remove the font

14372

To remove font file installed on the device:

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [FONTS] > [REMOVE FONT]

> Men page displays installed font files saved in the directory `/home/fonts`.

2 Remove font files

► Use [▲] / [▼] to select the requested font file (.ttf).

► Press [REMOVE] to remove the selected font file from the device.

> Font file is removed from the device.

► Optional: Repeat step 2 to remove additional font files from the device.

Setup: Format the internal storage

9643

NOTICE

Risk of data loss!

> When this action is confirmed, all user data in the directory `/data` of the storage medium is irretrievably deleted!

► Save files that you will need to an external medium prior to formatting!

To format the user data of the device:

To format the device-internal storage:

1 Select menu page

► Navigation path: [SETUP] > [ACTIONS] > [FORMAT STORAGE]

> [Format internal Mass storage] menu page appears.

2 Speicher formatieren

► Press [ENTER] to start the formating process.

> The device-internal user data in directory `/data` is formated.

> Information bar displays a confirmation message.

5.4 PDF file viewer

Contents

11416

The setup menu uitlises a PDF file viewer. The operator can use it to display PDF files stored in the directories `/home/pdf` and `/media`.

To display a PDF file:

► Enter the setup mode (→ **Enter the setup mode** (→ p. 29))
► Select [PDF-Viewer] menu item.
► Use [ENTER] to change to the PDF file viewer.
► Choose one of the following options:

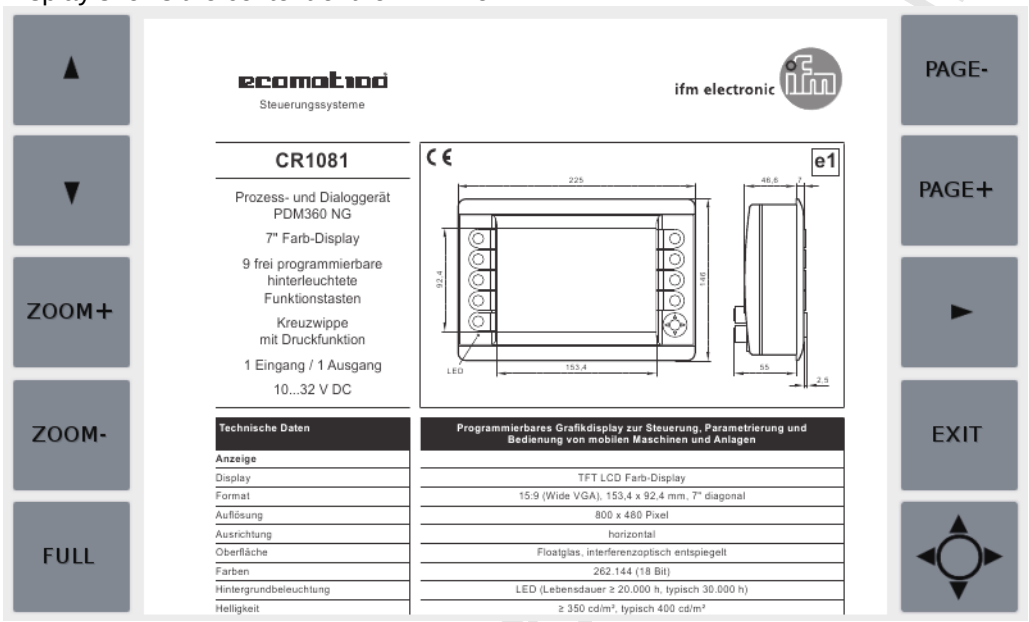
5.4.1 PDF: Navigating in the PDF file

11420

► Select requested PDF file and press [ENTER] to open the file (→ **Operation: Navigate in the file directory** (→ p. 34)).

> PDF file is loaded.

> Display shows the content of the PDF file:



► Use [▲] / [▼] to scroll up / down in the document.

► Use [ZOOM+] / [ZOOM-] to enlarge / reduce the document view.

► Use [PAGE+] / [PAGE-] to select the next / previous page of the PDF file.

► Use [▶] / [◀] to shift the document view to the right / left.

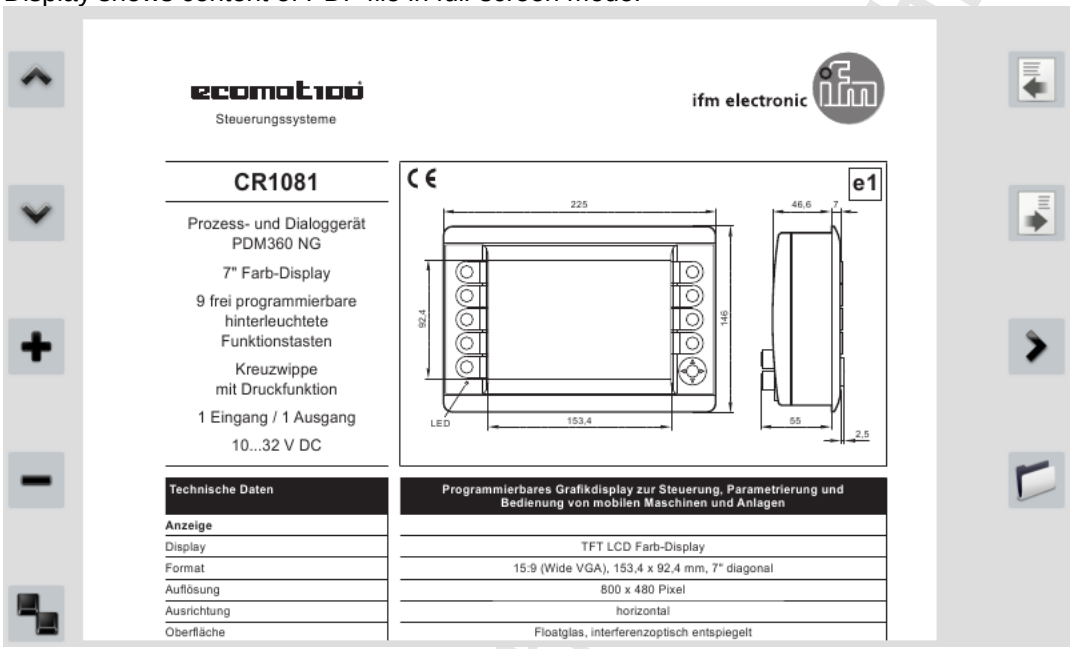
► Use [FULL] to change to the full-screen mode (→ **PDF-Viewer: Display PDF file in full-screen mode** (→ p. 65)).

► Use [EXIT] to change to the extended menu (→ **PDF-Viewer: Display extended menu** (→ p. 66)).

5.4.2 PDF-Viewer: Display PDF file in full-screen mode

11422

► Load PDF file (→ **PDF file viewer** (→ p. 63)).
► Use [FULL] to change to the full-screen mode.
> Display shows content of PDF file in full-screen mode:

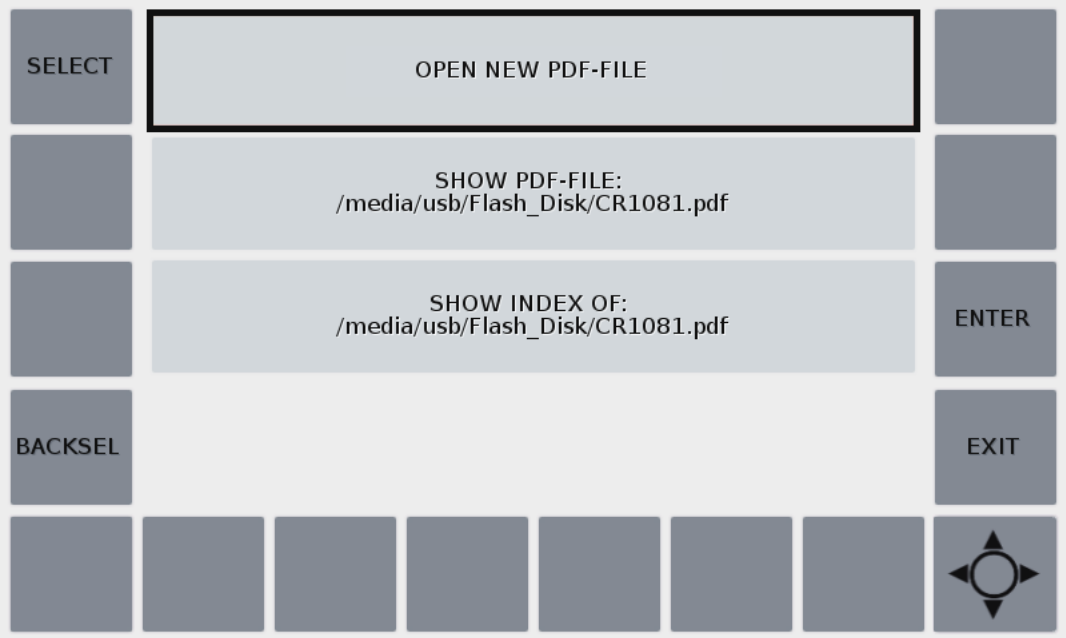


> Use [▲] / [▼] to scroll up / down in the document.
► Use [+] / [–] to enlarge / reduce the document view.
► Use / to select the next / previous page of the PDF file.
► Use [>] / [<] to shift the document view to the right / left.
► Use to return to the standard view (→ **PDF: Navigating in the PDF file** (→ p. 64)).
► Use [🗁] to change to the extended menu (→ **PDF-Viewer: Display extended menu** (→ p. 66)).

5.4.3 PDF-Viewer: Display extended menu

11424

► Start the PDF viewer (→ **PDF file viewer** (→ p. 63))
► Use [EXIT] to change to the extended menu.
> Menu page displays the extended menu:



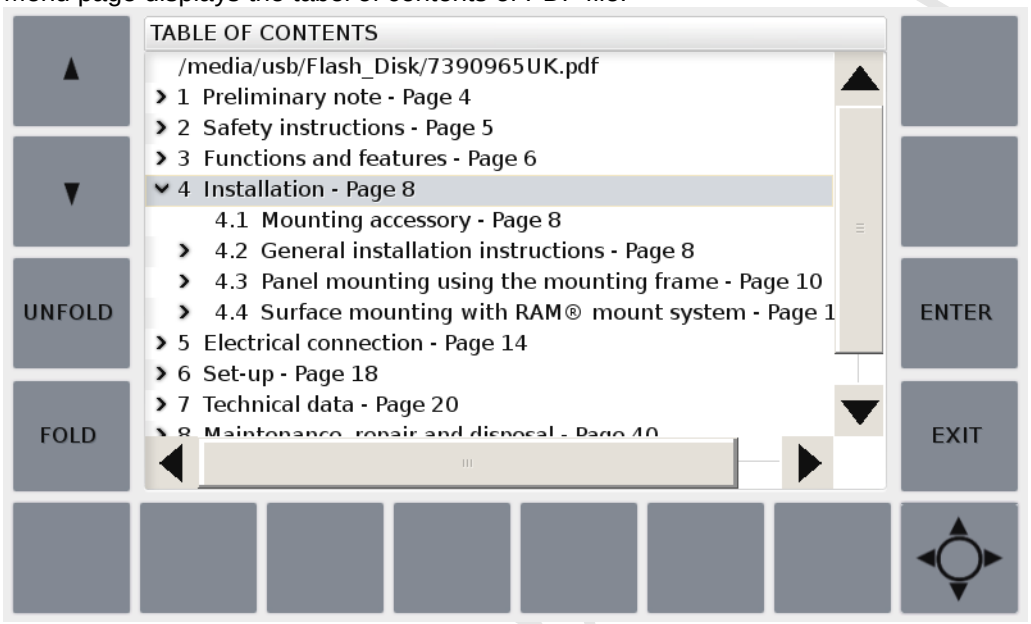
► Use [SELECT] / [BACKSEL] to select one of the following menu items and press [ENTER]:

- [OPEN NEW PDF FILE] Open new PDF file
  → **PDF file viewer** (→ p. 63)
- [SHOW PDF FILE: /Verzeichnis/Dateiname] Open the selected PDF file
  → **PDF: Navigating in the PDF file** (→ p. 64)
- [SHOW INDEX OF: /Verzeichnis/Dateiname] Open index of the selected PDF file
  → **PDF-Viewer: Display the tabel of content of the PDF file** (→ p. 67)

5.4.4 PDF-Viewer: Display the tabel of content of the PDF file

11431

► Display PDF file in full-screen mode (→ **PDF-Viewer: Display PDF file in full-screen mode** (→ p. 65)).
► Change to extended menu (→ **PDF-Viewer: Display extended menu** (→ p. 66)).
► Select [SHOW INDEX OF: /Verzeichnis/Dateiname] menu item and press [ENTER].
> Menu page displays the tabel of contents of PDF file:



► Use [▲] / [▼] to select an item in the table of contents.
► Use [UNFOLD] to to expand the view of the item.
► Use [FOLD] to hide the view of the item.
► Use [ENTER] to display the content of the selected item.
► Use [EXIT] to return to the extended menu (→ **PDF-Viewer: Display extended menu** (→ p. 66)).

5.5 Setup: Start the application

11438

► Enter the setup mode (→ **Enter the setup mode** (→ p. 29))
► Select [LOAD APPLICATION] button.
► Use [ENTER] to start the application.

If a valid application program is saved:
> The application starts.
> The LED flashes green at 2 Hz (→ **Status LED**).

If **no** valid application program is saved:
> A white blank screen appears.
> The LED lights green.
► Transfer a valid application to the device via CODESYS
► Reboot the device (supply voltage OFF > ON)

5.6 Automatically restore system files

14065

The function "Auto Restore" automatically identifies backup files saved in the root directory of a USB memory device (*.bup) and restores their contents. Several devices can be easily and time-savingly provided with the same software configuration (e.g. install a backup on controllers of the same type in several vehicles).

1 Create a backup file

► Enter the setup modus (→ **Enter the setup mode** (→ p. 29))

► Create backup of device configuration of the master device and store the backup file on an USB memory device (→ **Backup: Save the system files** (→ p. 57)).

The device only recognises backup files in the root directory of the USB memory device.

► Save the master backup file in the root directory of the USB memory device!

2 Automatically install the system files on the device

Only one backup file may be in the root directory of the USB memory device. If there are several backup files in the root directory, the automatic restore process will be stopped. An error message appears.

► Delete unneeded backup files in the root directory of the USB memory device!

► Connect the USB memory device with the master backup file with the target device

► Enter the setup mode (→ **Enter the setup mode** (→ p. 29))

► Select [SETUP] and press [ENTER] to change in the setup menu.

> The setup automatically recognises the structure and contents of the backup file.

> The menu screen shows the system components contained in the backup file.

► Press [START] to start the restore process.

> The selected system components are transferred to the device.

If successful:

> The menu screen shows the completion message.

3 Reboot the device

► Press [REBOOT] to reboot the device.

> Device reboots.

> If a valid project is loaded, this project is started after rebooting.
If no valid project is loaded, the device enters the setup mode after rebooting.

5.7 Exit setup, reboot device

7416

→ **Setup: Reboot the device** (→ p. 51)

6 First steps

Contents

15858

This chapter contains information about the first steps to program the CR1081/CR1085 with CODESYS.

6.1 Start CODESYS

20930

Prerequisites

> Software components are correctly installed (→ **Installation** (→ p. 15)).

Start CODESYS

► Double-click on symbol [CODESYS V3.5 SP8]

> CODESYS starts.

> CODESYS user interface appears.

6.2 Create CODESYS project

Contents

20938

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

ifm electronic provides a special profile for each model of the device class "Dialogue module PDM360 NG". The user can choose the suitable profile during project creation.

6.2.1 Create new project with CR1081/CR1085

17649

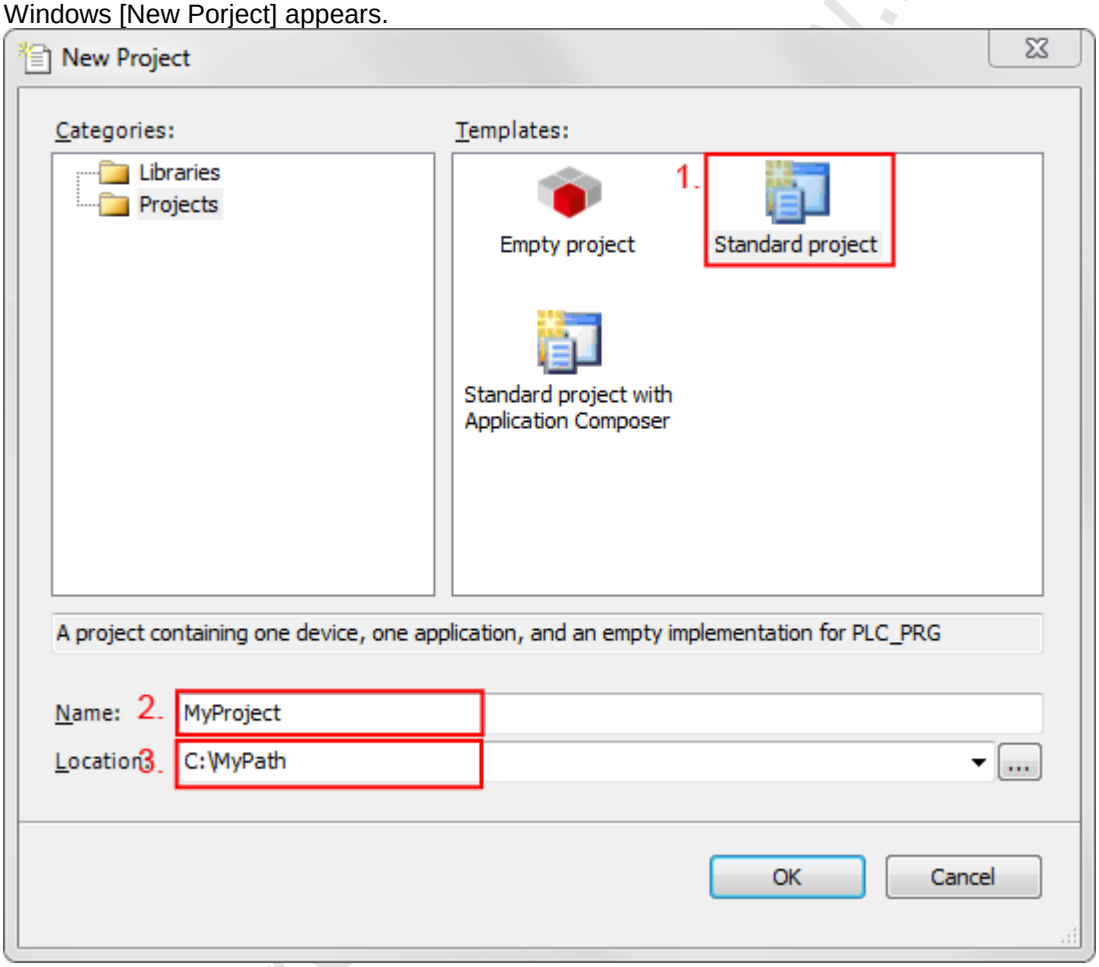
Requirements

► All required software components are correctly installed (→ **Installation** (→ p. 15)).

1 Create new CR1081/CR1085 project

► Select [File] > [New Project...].

> Windows [New Porject] appears.



► Set the following values:
1. [Templates]: Select [Standard project]
2. [Name]: Enter project name
3. [Location]: Select storage location for the project file.

► Press [OK] to confirm the entered values.

> Window [Standard Project] appears.

► Set the following values:
1. [Device]: Select required device.
2. [PLC_PRG in]: Select required programming language.

► Press [OK] to confirm the entered values.

> CODESYS creates a new CR1081/CR1085 project.

> Windows [Devices] shows a device tree of the project (→ **Overview: Project structure with CR1081/CR1085** (→ p. 76)).

2 Save the project

► Select [File] > [Save Project].

> CODESYS saves the project.

6.2.2 Add visualisation to the project

2270

To add a visualisation for the representation of the user interface of the CR1081/CR1085 to the project:

► In the device tree: Click on [Application].

► Select [Project] > [Add object] > [Visualization...].

> Window [Add Visualization] appears.

► Change the following settings:
1. [Name]: Enter designation for visualisation.
2. In the column [Active]: Activate checkbox for [VisuSymbols (System)].

► Click on [Add] to apply the changes.

> CODESYS adds visualisation elements to the device tree.

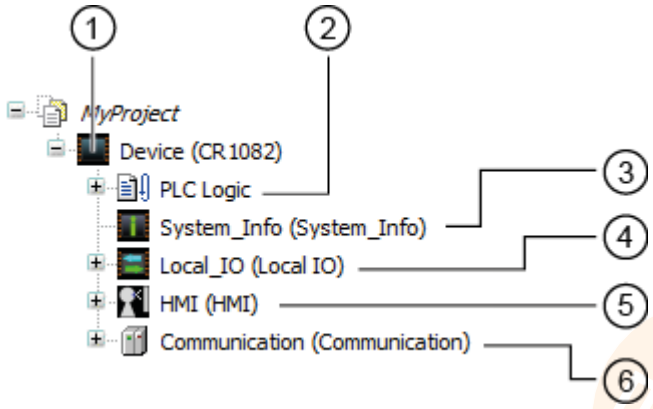
> Editor window shows the edit view of the visualisation.

► Save the project to apply changes.

6.2.3 Overview: Project structure with CR1081/CR1085

8573

A CODESYS project contains all components for configuration, management and programming of the CR1081/CR1085. All components of a project are shown in the window [Devices] in a hierarchic tree view. CODESYS projects with a CR1081/CR1085 have the following structure:



(1) [Device (CR1081/CR1085)] provides access to the settings of CR1081/CR1085.
→ **Configure PLC** (→ p. 82)

(2) [PLC Logic] provides access to the objects in the PLC applications.
→ **Objects of a PLC application** (→ p. 93)

(3) [System_Info] provides access to the device information.
→ **Display system information** (→ p. 141)

(4) [Local_IO] provides access to configuration options of the inputs and outputs
→ **Configure inputs and outputs** (→ p. 83)

(5) [HMI] provides access to the configuration options of the operating and display elements.
→ **Configure device keys** (→ p. 85)

(6) [Communication] provides access to the configuration options of the communication interfaces.
→ **Configure CAN interfaces** (→ p. 87)

6.3 Use CODESYS user manual

15221

This manual only describes the integration, configuration and the programming of the CR1081/CR1085 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

21784

Programming of the device-internal PLC is made via the Ethernet interface of the device (position of the connections: → Installaton 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 the connection in the network an experienced user or system administrator should set up the network addresses and do the configuration.

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 (CR1081/CR1085)]

> 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

21775

► Familiarise yourself with the following CODESYS functions!
  - Library manager
    → Online help > CODESYS Development System > Use 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 needs for the programming.

By means of the container library `ifmPDM360NG.library` the user can add all functions available for the device to the project.

To integrate a library into a project:

Prerequisites

> ifm package is correctly installed (→ **Install ifm package** (→ p. 18)).

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.

6.7 Access the Linux system of the device

21785

The user can access the Linux system of the device directly via the Ethernet interface. The following options are supported:

Option	Description	Example application
Telnet	Unencrypted access to the Linux command line	Putty (→ www.putty.org)
SSH	Encrypted access to the Linux command line	Putty (→ www.putty.org)
FTP	Unencrypted access to the Linux file system (copy files)	WinSCP (→ www.winscp.net)
SCP	Encrypted access to the Linux file system (copy files)	WinSCP (→ www.winscp.net)

The following login data is factory-preset:
User name/login: root
Password: pdm360ng

7 System configuration

Contents

11505

The device tree of the CODESYS project provides different possibilities to configure the properties of the device.

For configuration options in the CODESYS application: → **Configure system** (→ p. 100)

7.1 Configure PLC

22073

The configuration of the PLC of the CR1081/CR1085 is made via the "Generic device editor" of the CODESYS programming system.

► Familiarise yourself with the following CODESYS functions!

- Device editor
  → Online help > CODESYS Development System > Reference user interface > Objects > Object 'Device' and Generic device editor

The configuration of the PLC is made via the node [Device (CR1081/CR1085)] of the device tree.

To configure the PLC of the CR1081/CR1085:

► In the device tree: Double-click on [Device (CR1081/CR1085)]

> Editor window shows device editor of the PLC of the CR1081/CR1085.

7.2 Configure inputs and outputs

Contents

10378

On delivery the following default settings are preset:

- Operating mode of the inputs and outputs: Binary (positive-switching)
- Diagnostic function: Active
- Overload protection: active

7.2.1 Set operating mode of the inputs

10374

The input of the device can be operated in different operating modes. The user can set the operating mode of the input via the parameter [IN0000]. The device supports the following operating modes:

Parameter	Description	Possible values	
[IN0000]	Operating mode of the input IN0000	IN_DIGITAL_CSI	Digital input current sinking (positive-switching, low-side)
		IN_DIGITAL_CSO	Digital input current sourcing (negative-switching, highside)
		IN_VOLTAGE_10	Voltage input 0...10 V
		IN_VOLTAGE_32	Voltage input 0...32 V
		IN_VOLTAGE_RATIO	Voltage input ratiometric
		IN_CURRENT_CSI	Current input 0...20 mA

Technical details of the supported operating modes: →Installaton instructions.

To configure the operating mode of the input:

► In the device tree: Double-click on symbol [Local_IO] > [Inputs]
► In the editor window: Select tab [Parameter].
> Editor window shows table current operating mode.
► In line [Mode]: Select the requested operating mode from the list in the column [Wert].

7.2.2 Set operating mode of the outputs

10375

The operating mode of the output is fixed. The device supports the following operating modes:

Parameter	Description	Possible values	
[OUT0000]	Operating mode of the output OUT0000	OUT_DIGITAL_CSO	Digital output, current sinking (positive-switching, highside)

Technical details of the supported operating modes: →Installaton instructions.

7.3 Configure device keys

Contents

14626

7.3.1 Set the brightness of the key LEDs

11502

The user can set the brightness of the key LEDs via the following parameters:

Parameter	Description	Possible values
[LED bright]	Brightness of the key LEDs in normal mode (values in %)	0...100*
[LED bright night]	Brightness of the key LEDs in night mode (values in %)	0...15*...100

* Default setting

To set the brightness of the key LEDs for normal mode and night mode:

► In the device tree: Double-click on [Device] > [HMI] > [Key_Backlight_settings]
► Select tab [Parameter]
> Table shows the current parameter settings.
► Enter the desired value for [LED bright] and [LED bright night] in the column [Wert].

7.3.2 Set key assignment (key mapping)

7422

By means of key mapping certain keyboard functions can be assigned to the operating elements. This enables the user to access the input elements of the visualisation (on-screen keyboard, numpad) directly via the pushbuttons.

To assign certain keyboard functions to the operating elements:

► In the device tree: Double-click on [HMI] > [Keyboards] > [Key_Mapping]
► Select tab [Parameter].
> Detail window shows table with current key mapping.
► Assign the requested keyboard function to each operating element in the column [Wert].
► Save the project to apply changes.

Key mapping can only be used if the parameter [Activate standard keyboard handling] is activated in the visualisation manager.

► Activate standard keyboard operation (→ **Use visualisation manager** (→ p. 111)).
► Note symbol names of the operating elements (→ **Symbol names of the operating elements** (→ p. 97)).

One keyboard function can be simultaneously assigned to several operating elements.

7.4 Configure CAN interfaces

Contents

14645

► 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
- J1939 Manager

► Observe the notes about task configuration! (→ **Configure task processing** (→ p. 131))
► Add an individual CANbus device to the device tree for each CAN interface used in the application!

7.4.1 Add and configure CANbus

2271

► Familiarise yourself with the following CODESYS functions!
- CANbus settings
  → Online help > Fieldbus support > CAN-based fieldbuses > Tab "CANbus - General"
- CANbus-I/O representation
  → Online help > Fieldbus support > CAN-based fieldbuses > Tab " I/O representation"

► Add an individual CANbus device to the project tree for each CAN interface.

1 Add CAN interface

► In the device tree: Right-click on [CAN]
> Context menu appears.
► Select [Add Device…].
> Window [Add Device] appears.
► Set the following values:
1. [Vendor]: Select ifm electronic.
2. In the table: Select [ifmCANbus].
3. [Name]: Enter unambiguous name for the CAN interface.

► Select [Add Device] to add a CAN interface to the device tree.
> In the device tree: CAN interface appears as subelement of [CAN (CAN)].

2 Configure CAN interface

► In the device tree: Double-click on the added CAN node
> Editor window shows setting options of the CAN interface.
► Select the CAN ID of the CAN network in the field [Network].
► Select the transmission rate of the CAN network in the list [Baud rate (bit/s)].

Continue with:

- **RawCAN: Configure CANLayer 2** (→ p. 89)
- **CANopen: Configure CANopen Manager (master)** (→ p. 89)
- **CANopen: Configure CANopen Device (slave)** (→ p. 90)
- **J1939: Configure J1939 Manager** (→ p. 91)

7.4.2 RawCAN: Configure CANLayer 2

14646

No further actions are required to use a CAN interface as CANLayer 2 (RawCAN). The user can access the CAN interface directly in the application.

More information about programming of the RawCAN interface: → **Use RawCAN (CAN Layer 2)** (→ p. 129)

7.4.3 CANopen: Configure CANopen Manager (master)

11744

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

The configuration of the CANopen Manager is based on the CANopen Stack of the CODESYS 3.5 programming system.

Prerequisites

> Add and configure CAN interface (→ **Add and configure CANbus** (→ p. 88))

1 Add CANopen Manager

► In the device tree: Right-click on the added CAN node
> Context menu appears.
► Select [Add Device…] in the context menu.
> Dialogue window [Add Device] appears.
► Set the following values:
1. [Vendor]: [<All manufacturers>]
2. In table: Select [CiA CANopen] > [CiA CANopen Manager] > [CANopen Manager].
3. [Name]: Enter unambiguous name.

► Activate [Add Device].
> CODESYS adds CANopen Manager to the device tree.

2 Configure CANopen Manager

► In the device tree: Double-click on the added CANopen Manager
> Editor window shows configuration options.
► Configure CANopen Manager as requested.
► Save the project to apply changes.

More information about programming of the CANopen interface: → **Use CANopen** (→ p. 127)

7.4.4 CANopen: Configure CANopen Device (slave)

14644

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

The configuration of the CANopen Device is based on the CANopen Stack of the CODESYS 3.5 programming system.

Prerequisites

► Add and configure CAN interface (→ **Add and configure CANbus** (→ p. 88))

1 Add CANopen Device

► In the device tree: Right-click on the added CAN node
> Context menu appears.
► In the context menu: Select [Add Device…].
> Dialogue window [Add Device] appears.
► Set the following values:
1. [Vendor]: [<All manufacturers>]
2. In table: Select [CiA CANopen] > [CiA Local Device] > [CANopen Device].
3. [Name]: Enter unambiguous name.
► Activate [Add Device] to add a CANopen Device to the project.
> CODESYS adds CANopen Device to the device tree.

2 Configure CANopen Device

► In the device tree: Double-click on the added CANopen Device
> Editor window shows configuration options.
► Configure CANopen Device as requested.
► Save the project to apply changes.

More information about programming of the CANopen interface: → **Use CANopen** (→ p. 127)

7.4.5 J1939: Configure J1939 Manager

11743

The configuration of the J1939 Manager is based on the J1939 Stack of the CODESYS 3.5 programming system.

► Familiarise yourself with the following CODESYS functions:
  - J1939 Manager
    → Online help > Fieldbus support > CAN-based fieldbuses > J1939 > J1939-Manager

To configure a CAN interface as J1939 Manager:

Prerequisites

> Add and configure CAN interface (→ **Add and configure CANbus** (→ p. 88))

1 Add J1939 Manager

► In the device tree: Right-click on the added CAN node.
> Context menu appears.
► Select [Add Device…] in the context menu.
> Dialogue window [Add Device] appears.
► Set the following values:
1. [Vendor]: <All manufacturers>
2. In table: Select [SAE J1939] > [SAE J1939 Manager].
3. [Name]: Enter unambiguous name.

► Activate [Add Device].
> CODESYS adds J1939 Manager to the device tree.

2 Configure J1939 Manager

► In the device tree: Double-click on the added J1939 Manager
> Editor window shows configuration options.
► Configure J1939 Manager as requested.
► Save the project to apply changes.

8 Programming

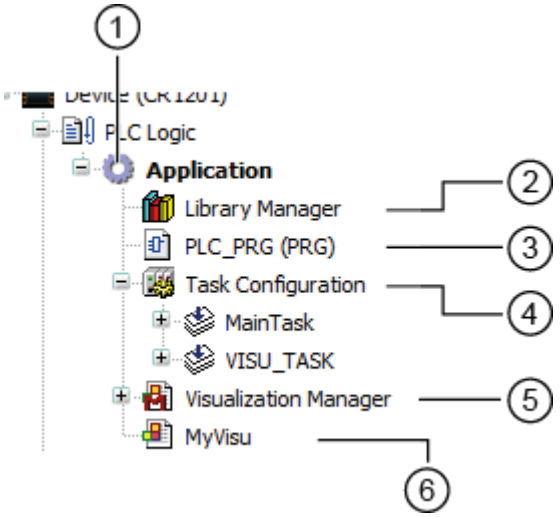
Contents

14603

8.1 Objects of a PLC application

3084

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



(1) [Application] is a container for objects of a PLC application

(2) [Library Manager] provides access to the standard and device-specific function libraries:
→ **Use ifm function libraries** (→ p. 99)

(3) [PLC_PRG(PRG)] provides access to the editor of the PLC application
→ **Create PLC application** (→ p. 94)

(4) [Task Configuration] provides access to the settings of the task processing:
→ **Configure task processing** (→ p. 131)

(5) [Visualization Manager]: Set properties of the visualisation:
→ **Use visualisation manager** (→ p. 111)

(6) [MyVisu]: Visualisation page:
→ **Create visualisation** (→ p. 110)

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

8.2 Create PLC application

14625

► Familiarise yourself with the following CODESYS functions!
  ▪ → Online help > CODESYS Development System > Create application

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 Notes

10376

Supported programming languages

21752

The following table shows which programming languages according to IEC 61131 are supported by the ifm function libraries:

Library	Function block language (FUP)	Sequential function chart (SFC)	Instruction list (IL)	Continuous function chart (CFC)	Ladder diagram (LD)	Structured text (ST)
ifmPDM360NGutil.library	X	X	X	X	X	X
ifmCANopenManager.library	X	X	X	X	X	X
ifmRawCAN.library	X	X	X	X	X	X
ifmEthernet.library	X	X	X	X	X	X
ifmFileUtil.library	X	X	X	X	X	X
ifmIOcommon.library	X	X	X	X	X	X
ifmSysInfo.library	X	X	X	X	X	X
ifmUSBstorageUtil.library	X	X	X	X	X	X

Legend:
X ... is supported
– ... is not supported

Available memory

21780

Memory area	Size
Total memory space*	RAM: 128 Mbytes Flash: 128 Mbytes
▪ Reserved for program code	8 Mbytes
Memory for retain data **	32 Kbytes
Memory for bitmap files (80 x 100pixels each, 24-bit colour depth)	45 Mbytes
Memory for fonts (font files)	
Memory space required by the system itself	75 Mbytes
Memory in the directory `/tmp`	Cannot be ensured
Data memory `/data`	1024 Mbytes

Legend:
* ... Size of the memory: Data sheet
** ... Retain data: Retain variables (→ **Supported variable types** (→ p. 96))

The device does not support an online change (= possibility to change the application code online without previously deleting the original application)

The data memory /data is NOT automatically initialised during the start process. To use the data memory:

► Integrate the data memory in the application into the file system of the device by means of the FB **InitNANDflashAsync** (→ p. 212).

The storage capacity of the device can be extended by an external USB memory device. The USB connectors of the device are not suitable for permanent operation in mobile applications.

► External USB memory device is only for momentary use (e.g. to back up files)!

Supported variable types

21762

► 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
  → Online help > CODESYS Development System > Exchange data in 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

CAN network variables are not supported!

Restrictions

21756

The device is designed as a display for representing process data. It is thus conditionally real-time capable. Computationally intensive processes can cause long cycle times.

► Avoid computationally intensive functions!
This includes the following functions:
- Controller
- Encoder
- Control of diesel engines
- Operation as CANopen master

► Monitor the cycle time during the runtime of the PLC program!
Cycle times in the range from 5 ... 10 ms are no problem.

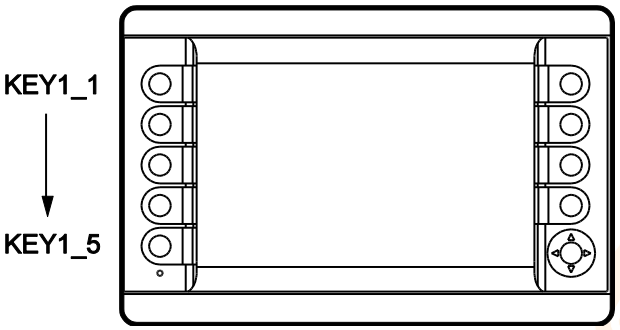
Symbol names of the operating elements

12652

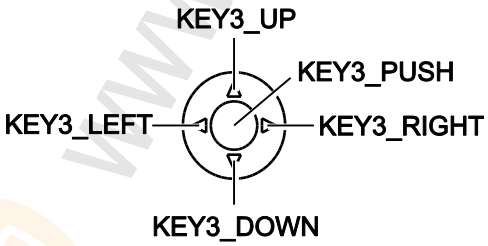
Certain symbol names are assigned to the operating elements of the device. By means of these symbol names the programmer can define certain actions and functions to be triggered upon actuation of the correspondent operating element (→ **Map functions to operating elements** (→ p. 112)).

The following figure shows the symbol names of the individual operating elements:

Function keys:



Navigation key:



Watchdog behaviour

11789

In this device a watchdog monitors the runtime of system processes, including the process of the CODESYS application. Each process has to call the watchdog monitor within a defined time intervall (watchdog timeout). If the maximum watchdog time is exceeded:

> all processes are stopped (reset)
> all outputs are switched off
> the screen goes black
> the status LED flashes red at 5 Hz

A process needs a certain time to proceed reset and initialisation routines (watchdog reset).

The following tabel shows the watchdog times for each process:

Prozess	Chain-ID	Watchdog-Timeout [ms]	Watchdog-Reset [ms]
ppc-plclinux	10	1 000	500
eplc main	20	10 000	500
eplc overlay	21	10 000	500
keyboard-service	40	2 000	500
io-coproc-service	50	2 000	500
wd-coproc-service	60	2 000	500

Eliminate the fault:

► Reboot the device via voltage on/off plus.

Operating sequence

7427

In principle, there are two options to create a project for display devices:

Sequence	Advantages	Disadvantages
Visualisation first, then the PLC program.	▪ In the program it is possible to cross-reference to the finished images. ▪ When the PLC program is tested the images already exist.	The PLC parameters and variables required in the images have not yet been defined.
PLC program first, then the visualisation	All parameters and variables are defined in the PLC program before they are referred to in the visualisations.	▪ The parameters from the images (image number, key, LED, etc.) must be found elsewhere. ▪ The PLC program can only be tested after creation of the visualisation.

In both cases we urgently recommend to design a precise structure of the visualisation and its contents **before** starting.

8.3 Use ifm function libraries

Contents

14624

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

Name	Description
ifmDevicePDM360NG.library	Data structures, enumeration types and global variables
ifmPDM360NGutil.library	Device-specific functions
ifmIOcommon.library	Functions for access to the inputs and outputs of the device
ifmCANopenManager.library	Functions for use of the CAN interfaces as CANopen Manager
ifmRawCAN.library	Functions for use of the CAN interfaces as CAN Layer 2
ifmFileUtil.library	File and help functions
ifmEthernet.library	Functions for the management of the IP settings of the Ethernet interface
ifmUSBstorageUtil.library	Functions for the management of USB memory devices

Detailed information about the ifm function libraries: → **ifm function libraries** (→ p. 144)

8.3.1 Configure system

14370

The following function elements are available to configure the device:

Name	Description	Reference
DisableTouchScreen*	Activate /deactivate touch functionality of the display	→ DisableTouchScreen (→ p. 210)
InitNANDflash	Initialise flash memory of the device	→ InitNANDflashAsync (→ p. 212)
LoadSplashScreen	Load individual start screen	→ LoadSplashScreen (→ p. 216)
SetBootupBacklight	Set the brightness of the display during start-up	→ SetBootupBacklight (→ p. 218)
SetKeyClick	Configure the sound generated upon actuation of a function key	→ SetKeyClick (→ p. 220)
BuzzerClick	Configure the sound generated by the device-internal buzzer	→ BuzzerClick (→ p. 222)

* ... only available for devices with touch screen

8.3.2 Configure Ethernet interface

11515

The following function elements are available to manage the settings of the Ethernet interface of the device:

Name	Description	Reference
GetIPsettings	Read IP settings of the Ethernet interface	→ GetIPsettings (→ p. 159)
SetIPsettings	Change IP settings of the Ethernet interface	→ SetIPsettings (→ p. 161)

The current IP settings of the Ethernet interface can be read in the online mode via the device tree (→ Display system information (→ p. 141)).

8.3.3 Control device

12322

The following function elements are available to manage the device:

Name	Description	Reference
BuzzerOn	Activate/deactivate device-internal buzzer with a specific frequency	→ BuzzerOn (→ p. 208)
LinuxSysCallAsync	Command transmitted to the Linux operating system of the device	→ LinuxSysCallAsync (→ p. 214)
SupplySwitch	Switch off the unit	→ SupplySwitch (→ p. 202)

8.3.4 Manage files

11507

The following function elements are available to manage (copy, delete) the files and directories:

Name	Description	Reference
Copy_PDM_To_USB	Copy files from the device to a USB memory device	→ Copy_PDM_To_USB (→ p. 165)
Copy_USB_To_PDM	Copy files from a USB memory device to the device	→ Copy_USB_To_PDM (→ p. 167)
DeleteFile	Delete the file from the device	→ DeleteFile (→ p. 169)
FileCopy	Copy files on the device	→ FileCopy (→ p. 171)
SyncFileMemory	Synchronise the contents of the FLASH memory and the RAM	→ SyncFileMemory (→ p. 173)

8.3.5 Manage CSV files

10377

The following function elements are available to manage (create, change, read) CSV files:

Name	Description	Reference
ReadCSV8Byte	Read the contents of a CSV file	→ ReadCSV8Byte (→ p. 175)
WriteCSV8Byte	Write the contents of a CSV file	→ WriteCSV8Byte (→ p. 177)
WriteCSV8ByteHeader	Write header section of a CSV file	→ WriteCSV8ByteHeader (→ p. 180)

8.3.6 Use help functions

11612

The user can use the following help functions:

Name	Description	Reference
GetMemoryInfoAsync	Display memory used of the device	→ GetMemoryInfoAsync (→ p. 190)
ByteArray_To_String	Convert array from bytes into a character string	→ ByteArray_To_String (→ p. 192)
Buffer_To_String	Convert array from bytes into an array of character strings	→ Buffer_To_String (→ p. 186)
Pack2Byte_To_Word	Convert 2 bytes into a word	→ Pack2Byte_To_Word (→ p. 194)
Pack4Byte_To_DW	Convert 4 bytes into a double word	→ Pack4Byte_To_DW (→ p. 195)
Word_To_2Byte	Convert word into 2 bytes	→ Word_To_2Byte (→ p. 197)
_8Byte_To_CSV	Convert array from 8 bytes into CSV format.	→ _8Byte_To_CSV (→ p. 198)
DW_To_4Byte	Convert DWORD into 4 bytes	→ DW_To_4Byte (→ p. 193)
RTC_To_String	Provide operating time of the device as STRING	→ RTC_To_String (→ p. 196)
GetFileSizeAsync	Display size of a file	→ GetFileSizeAsync (→ p. 188)
ReadParmSingleAsync	Read individual parameter set from a text file	→ ReadParmSingleAsync (→ p. 182)
WriteParmSingleAsync	Write individual parameter set into a text file	→ WriteParmSingleAsync (→ p. 184)

8.4 Use system flags

10373

System flags are factory-set global variables. They 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 and outputs

14623

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

Variable	Data type	Access	Description	Possible values	
IN0000.ValueAnalogue	UINT	r	Value of the analogue input IN0000	0 ... 65535	0 ... 65535
IN0000.ValueDigital	BOOL	r	Value of the digital input IN0000	FALSE	Input deactivated
				TRUE	Input activated
OUT0000.ValueDigital	BOOL	r/w	Value of the output OUT0000	FALSE	Output deactivated
				TRUE	Output activated

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

The valid value range of the input depends on the active operating mode of the input.
► Observe configuration of the inputs! → **Set operating mode of the inputs** (→ p. 84)

8.4.2 Read diagnostic data of the device

10372

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
uiLightSensor	UINT	r	Brightness detected by the photoelectric sensor (in %)	0 ... 100	Minimum brightness ... Maximum brightness

Legend:
r ... read only

8.4.3 Set the brightness of the display

20925

The user can use the following global variables to change the brightness of the display:

Name	Data type	Access	Description	Possible values	
uiBacklightBrightness	UINT	r/w	Set the brightness of the display (in %)	0 ... 65535	0 % ... 100 %

Legend:
r/w ... read and write

Simultaneous access to the brightness of the display via the global variable and the FB**SetBootupBacklight** (→ p. 218)is not permitted!

8.4.4 Control status LED

20939

The user can use the following global variable to control the status LED of the device:

Name	Data type	Access	Description	Possible values	
LED0.Value	BOOL	r/w	Activate/deactivate status LED of the device	FALSE	Status LED off
				TRUE	Status LED on
LED0.Colour_1	ENUM	r/w	LED colour for status ON	→ **LED_COLOUR (ENUM)** (→ p. 156)	
LED0.Colour_2	ENUM	r/w	LED colour for status OFF	→ **LED_COLOUR (ENUM)** (→ p. 156)	
LED0.Frequency	ENUM	r/w	Flashing frequency of the status LED	→ **LED_FLASH_FREQ (ENUM)** (→ p. 157)	

Legend:
r/w ... read and write

In case of an error the LED colour RED is set by the runtime system.
► Do not use the LED colour RED in the application!

8.4.5 Control buzzer (audio signalling device)

20940

The user can use the following global variables to change the properties of the device-internal buzzer (audio signalling device) and to control the buzzer:

Name	Data type	Access	Description	Possible values	
uiBuzzerFreq	UINT	r/w	Frequency of the buzzer sound (in Hz)	500 ... 5000	500 Hz ... 5000 Hz
xBuzzerON	BOOL	r/w	Activate or deactivate audio signalling device (buzzer)	FALSE	Buzzer off
				TRUE	Buzzer on

Legend:
r/w ... read and write

8.4.6 Control key LEDs

7415

The user can use the following global variables to control the LEDs of the operating elements of the device:

Name	Data type	Access	Description	Possible values	
xNightModeON	BOOL	r/w	Activate/deactivate night mode for lighting of the key LEDs	FALSE	Deactivate night mode
				TRUE	Activate night mode
LEDsKeyboard1.0	BOOL	r/w	Activate/deactivate LED of button KEY1_1 of the left key row	FALSE	LED off
				TRUE	LED on
LEDsKeyboard1.1	BOOL	r/w	Activate/deactivate LED of button KEY1_2 of the left key row	FALSE	LED off
				TRUE	LED on
LEDsKeyboard1.2	BOOL	r/w	Activate/deactivate LED of button KEY1_3 of the left key row	FALSE	LED off
				TRUE	LED on
LEDsKeyboard1.3	BOOL	r/w	Activate/deactivate LED of button KEY1_4 of the left key row	FALSE	LED off
				TRUE	LED on
LEDsKeyboard1.4	BOOL	r/w	Activate/deactivate LED of button KEY1_5 of the left key row	FALSE	LED off
				TRUE	LED on
LEDsKeyboard3.0	BOOL	r/w	Activate/deactivate LED of button KEY3_1 of the right key row	FALSE	LED off
				TRUE	LED on
LEDsKeyboard3.1	BOOL	r/w	Activate/deactivate LED of button KEY3_2 of the right key row	FALSE	LED off
				TRUE	LED on
LEDsKeyboard3.2	BOOL	r/w	Activate/deactivate LED of button KEY3_3 of the right key row	FALSE	LED off
				TRUE	LED on
LEDsKeyboard3.3	BOOL	r/w	Activate/deactivate LED of button KEY3_4 of the right key row	FALSE	LED off
				TRUE	LED on
LEDsKeyboard3.11	BOOL	r/w	Activate/deactivate LED of the Navigation key	FALSE	LED off
				TRUE	LED on

Legend:
r/w ... read and write

8.5 Use visualisations

Contents

14604

► Familiarise yourself with the following CODESYS functions!

- Visualisations
  → Online help > CODESYS Visualization

The device supports the following visualisation types:

- Target visualisation

8.5.1 Notes about visualisations

21786

Observe the following notes when creating visualisations.

Drawing area

8392

- The actual drawing area is equal to the resolution of the display.
- Virtual drawing area = 2560 x 1536 pixels
  (enlarges the actual drawing area)
- The left upper corner marks the home position (0,0) of the virtual and actual drawing area.
- All elements in and outside the virtual drawing area are calculated.

The images used in the project can be larger than the given drawing area. In this case only a selectable part of the image will be displayed.

image properties

21787

The images used in visualisations must have the following properties:

- **Colour depth:** 18 bits (recommended), 24 bits (maximum)
- **Formats**: BMP, BMP RLE compressed, JPG
- **Size:** Max. resolution of the display (→ installation instructions)

Images not corresponding to the size and colour requirements will be adapted accordingly during the processing of the application. Each time the image is opened, it must be resampled. This often leads to much longer times to change from one image to the other.

► Only use images in visualisations that meet the requirements on image and colour depth!

► Edit image files accordingly by means of an image processing software.

Use image collection

21788

► Familiarise yourself with the following CODESYS functions!

- Image collection
  → Online help > CODESYS Development System > Program application > Use image collection

To use own images in visualisations the user must first add the corresponding image files to an image collection in the application.

To add an image collection to a project:

► In the device tree: Mark node [Application].

► Select [Project] > [Add object] > [Image Pool...].

> Window [Add Image Pool] appears.

► Enter the name of the image collection and confirm with [Add].

> Image collection appears in the project tree as a subelement of the application.

Use optical and acoustic feedback

14300

All operating elements are backlit withLEDs. The LEDs can be controlledin the application program.

► Only switch those key LEDs in the application which can be sensibly actuated in the current screen.

For all keys currently without function:

► Switch off keys LEDs in the application.

Request to press a certain key:

► Programm the key LED so that it flashes.

Night design of the operating elements:

- The LEDs for alle operating elements are dimmable.
- For operating at night the LEDs have ONE adjustable basic brightness for all operating elements. So the operating elements are also backlit if the key LEDs were not switched on.

To set the brightness of the key LEDs: → **Set the brightness of the display** (→ p. 103)
To control the night mode of the key LEDs: → **Control key LEDs** (→ p. 105)
To control the device-internal audio signalling device: → **Control buzzer (audio signalling device)** (→ p. 104)

Texts

7428

- The smallest font size which is clearly visible on the device is 8 points.
- Permissible fonts (true type fonts):
  - Arial (standard)
  - Courier New
  - Times New Roman
  - ifm ISO symbols
  - Loadable fonts

Install all fonts required for use in the application on the PC as well as on the device!
(→ **Fonts: Load the font** (→ p. 61))

- Permissible font size:
  - Standard = 10 pixels
  - Max. = 72 pixels
  - Permissible nuances = 1 pixel
- Permissible font weights:
  - Normal (standard)
  - Italic
  - Bold
  - Bold cursive
- Permissible effects:
  - None (standard)
  - Underlined
  - Crossed out
- The following text scripts are accepted:
  - Western (standard)
  - Other scripts on request

8.5.2 Create visualisation

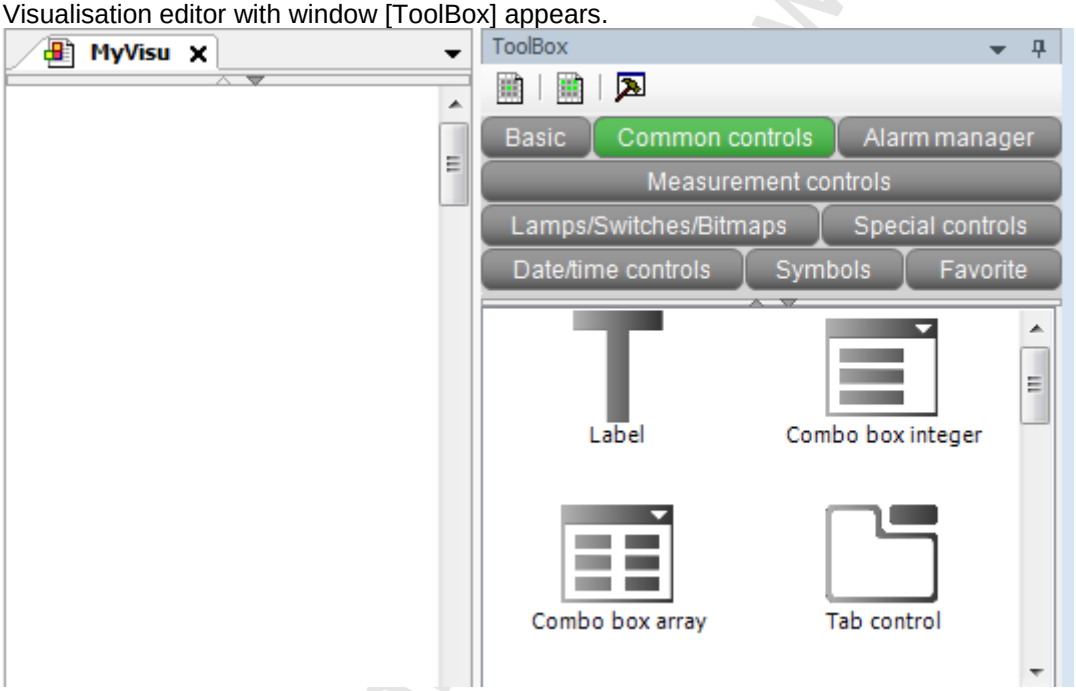
14940

► Familiarise yourself with the following CODESYS functions!
- Visualisation editor:
  → Online help > CODESYS Visualization > Visualisation editor
- Visualisation elements:
  → Online help > CODESYS Visualization > Visualisation elements

To create a visualisation for a PLC application:

► In the device tree: Double-click on [Visualization]

> Visualisation editor with window [ToolBox] appears.



► Create requested visualisation.

Observe the notes when creating the visualisation (→ **Notes about visualisations** (→ p. 107))!

► Save the project to apply changes.

8.5.3 Use visualisation manager

15818

► Familiarise yourself with the following CODESYS functions!
- Visualisation manager
  → Online help > CODESYS Visualization > Visualisation manager with clients > Visualisation manager

The visualisation manager contains the general settings of the visualisations:

1 Start visualisation manager

► In the device tree: Double-click on [Application] > [Visualization Manager]

> Editor window shows visualisation manager.

2 Change general settings

► Select tab [Settings].

> Editor window shows general settings of the visualisations.

► Set the parameters as requested.

If the key mapping function (→ **Set key assignment (key mapping)** (→ p. 86)) is to be used, the standard keyboard operation must be activated.

► Activate the checkbox [Activate standard keyboard handling] in the area [Additional settings].

► Save the project to apply changes.

Map functions to operating elements

20922

► Familiarise yourself with the following CODESYS functions!
- Keyboard configuration
  → Online help > CODESYS Visualization > Reference, User interface > Objects > Tab 'Visualization manager' - 'Default Hotkeys'

By using the CODESYS function [Default Hotkeys] the user can map certain function to the operating elements of the device (e.g. page switch, value switch). The defined keyboard functions are valid for all visualisations within the project.

To map a function to an operating element of the device:

1 Start visualization manager

► Start visualization manager.

> Editor window shows visualization manager.

2 Define keyboard functions

► In the editor windows: Select [Default Hotkeys] tab.

> Editor window shows the current configuration of the keyboard functions.

► Set the following values:
1. Column [Key]: Select required symbol name (→ **Symbol names of the operating elements** (→ p. 97)).
2. Column [Key down]: Define ON level
3. Column [Action type]: Select action type
4. Column [Action]: Select action.

3 Define additional keyboard functions

> Repeat step 2 to define additional keyboard functions.

Example: Signal actuation of the function key

20924

Goal: If the function key [KEY1_4] is pressed, the page "MyVisu" should be called.

1 Start visualisation manager

► Start visualisation manager.

> Editor window shows visualisation manager.

2 Define keyboard function

► In the editor window: Select tab [Default Hotkeys].

> Editor window shows current configuration of the keyboard functions.

► Set the following values:
1. Column [Key]: Select [KEY1_4]
2. Column [Key down]: Active checkbox
3. Column [Action type]: Select [Change shown Visualization]
4. Column [Action]: Activate checkbox [Assign] and select page "MyVisu"

8.5.4 Configure visualisation

21576

To change the properties of the created visualisation:

► In the device tree: Double-click on [Visualization Manager] > [TargetVisu]

> Editor window shows properties of the target visualisation

► Set the following values:
1. [Start Visualization]: Select requested visualisation.
2. [Update rate (ms)]: 150
3. [Scaling options]: Fixed
4. [Antialiased drawing]: active
5. [Default text input]: Select requested input device.

► Save the project to apply changes.

Observe the notes about the configuration of the visualisation task!
→ **Configure visualisation task** (→ p. 132)

8.6 Use mobile cameras

Contents

20928

The device supports the operation of mobile cameras.

Interface	Type of signal	Properties
Ethernet	Digital	▪ Up to 4 camera images representable at the same time ▪ Colour: 24 bits / greyscale: 8 bits ▪ Brightness of the camera image adjustable step by step (0...100: 0...100% ▪ Camera image and visualisation representable at the same time (overlaid) ▪ Failure of a camera is signalled ▪ Cameras can be activated / deactivated separately ▪ Rotation of the camera image possible by 90° / 180° / 270° ▪ Scaling of the camera image 0...400 % with unchanged proportion
Analogue video input (FBAS)	Analogue	▪ 2 analogue connections ▪ Automatic NTSC / PAL recognition ▪ 1 camera image representable at the same time ▪ Changing between 2 cameras possible ▪ Brightness of the camera image adjustable step by step (0...100%) ▪ Camera image and visualisation representable at the same time (overlaid) ▪ Failure of a camera is signalled ▪ Cameras can be activated / deactivated separately ▪ Rotation of the camera image possible by 90° / 180° / 270° ▪ Scaling of the camera image with unchanged proportion: 0...400 %

8.6.1 Supported cameras

SYS_OBJECTID>

	Analogue cameras	Ethernet cameras
Reference cameras (recommended)	ifm electronic: O2M200, O2M201, O2M202, O2M203	ifm electronic: O2M110, O2M113 (from firmware 5.1001)
Cameras from other manufacturers (tested)	▪ Motec: MC3100-3R ▪ Vision Techniques: VT70	--

► Only use recommended or tested cameras! The use of other cameras on the device is without warranty!

8.6.2 Activate extensions for mobile cameras

20923

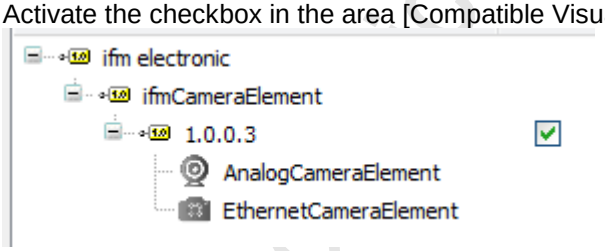
ifm electronic provides special visualisation extensions to add mobile cameras into a CODESYS project.

The library `ifmCameraElement.library` is part of the "CODESYS for ifm PDM360NG Products".

► Install CODESYS for ifm PDM360NG Products (→ **Install ifm package** (→ p. 18))

To activate the visualisation extensions:

► Open a project or create a new one.
► Select [Project] > [Project Settings...].
> Window [Project Settings] appears.
► Select [Visualization Profile].
► Activate the checkbox in the area [Compatible Visualization Extensions].



► Click [OK] to activate the extensions.
> In the tool library the following extensions for mobile cameras are available.

Symbol	Description	Reference
	▪ Ethernet camera element ▪ Placeholder for the image of an Ethernet camera	→ **Use Ethernet camera in the project** (→ p. 116)
	▪ Analogue camera element/item ▪ Placeholder for the image of an analogue camera	→ **Using analogue cameras in the project** (→ p. 120)

8.6.3 Use Ethernet camera in the project

14292

The image data of a connected Ethernet camera is transferred to a placeholder graphic on a visualisation page.

Add Ethernet camera image to the project

435

To add the image data of an Ethernet camera to a project:

Prerequisite

> The Ethernet camera is correctly connected to the device.

> Extension for mobile cameras is activated (→ **Activate extensions for mobile cameras** (→ p. 115))

1 Create visualisation

► Open or newly create visualisation (→ **Add visualisation to the project** (→ p. 75)).

2 Add a placeholder graphic for the camera image

► Select window [ToolBox].

► Select tab [Special controls].

► Drag symbol [Ethernet Camera element] into the visualisation via drag-and-drop.

> Visualisation shows the placeholder for the camera image (= rectangular visualisation element)

Placeholder graphics in frames are not supported!

The placeholder graphic of the camera image is automatically displayed on all visualisation pages of the project.

► To only display the placeholder graphic on certain pages: Create a copy of the placeholder graphic on the previous page and deactivate it by means of variable [Display image], if necessary (→ **Configure Ethernet camera image** (→ p. 117)).

Positioning the placeholder graphic outside the visible area of the visualisation page may cause visualisation problems during the runtime of the application.

► Always position the complete placeholder graphic in the visible area!

If the parameter [Invisible] is activated the other parameter settings are without effect.

► Move the placeholder graphic to the invisible area of the visualisation page to make the camera image invisible.

Configure Ethernet camera image

14078

To configure the Ethernet camera image:

1 Open configuration dialogue

► Open the visualisation page with placeholder graphic.

► Double-click on placeholder graphic

> Window [Properties] shows configuration options of the camera image.

2 Change position and size of the camera image

► Set the following parameters under [m_StaticPosition]:

Parameter	Description	Possible values
[X]	X coordinate of the top left corner of the camera image	E.g. 290
[Y]	Y coordinate of the top left corner of the camera image	E.g. 220
[Width]	Width of the camera image (in pixels)	Recommended (PAL): 360 Max.: 720
[Height]	Height of the camera image (in pixels)	Recommended (PAL): 288 Max.: 576

3 Set image properties

► Set the following parameters under [Bitmap configuration]:

Parameter	Description	Possible values	
[Static ID]	Assign an alternative placeholder image in the image pool		
[Scale type]	Select the type of image scaling	Isotropic	The image keeps its proportions.
		Anisotropic	The image adapts to the size of the button.
		Not scaled	The image keeps its original size, even if the size of the placeholder is changed.
[Transparent]	Activate/deactivate evaluation of the transparency colour	☐	Do not evaluate transparency colour
		☑	Evaluate transparency colour
[Transparent color]	Select transparency colour		
[Horizontal alignment]	Set horizontal orientation of the camera image	Left	Align to the left
		Centred	Align to the centre
		Right	Align to the right
[Vertical alignment]	Set vertical orientation of the camera image	Top	Align to the top
		Centred	Align to the centre
		Bottom	Align to the bottom

4 Select an alternative image

► Set the following parameters under [Bild-Variable]:

Parameter	Description	Possible values
[Bild-ID]	▪ Variable for the image from the image collection displayed if no camera image is available ▪ Data type: STRING	E.g. PLC_PRG.sImageNoCam

5 Select variables for scaling and shift

► Set the following parameters under [Move & scale variables]:

Parameter	Description	Possible values
[Absolute movement] > [X]	▪ Variable containing the new X coordinate of the top left corner of the camera image ▪ Data type: INT	E.g. PLC_PRG.iCamNewX
[Absolute movement] > [Y]	▪ Variable containing the new Y coordinate of the top left corner of the camera image ▪ Data type: INT	E.g. PLC_PRG.iCamNewY
[Change of width]	▪ Variable containing the new width of the camera image ▪ Data type: INT	E.g. PLC_PRG.iCamNewWidth
[Change of height]	▪ Variable containing the new height of the camera image ▪ Data type: INT	E.g. PLC_PRG.iCamNewHeight

6 Select state variables

► Set the following parameters under [State variables]:

Parameter	Description	Possible values	
[Invisible]	▪ Variable for the control of the visibility of the camera image ▪ Data type: BOOL	E.g. PLC_PRG.xCamVisible	
		FALSE	Camera image is not visible
		TRUE	Camera image is visible

7 Select general camera variables

► Set the following parameters under [Common camera variables]:

Parameter	Description	Possible values	
[Display image]	▪ Variable for the control of the display of the camera image ▪ Data type: BOOL	E.g. PLC_PRG.xCamEnable	
		FALSE	Deactivate the camera image
		TRUE	Activate the camera image
[Rotation angle]	▪ Variable for the clockwise rotation of the camera image ▪ Data type: INT	E.g. PLC_PRG.iRotAngle	
		0	No rotation
		90	90°
		180	180°
		170	270°
[Mirror]	▪ Variable for the mirroring of the camera image along its vertical centre axis ▪ Data type: BOOL	E.g. PLC_PRG.xCamMirror	
		FALSE	Do not mirror the camera image
		TRUE	Mirror the camera image
[Error flag]	▪ Variable signals if at least one of the following camera errors occurred: - Invalid angle of rotation - Invalid range - Invalid channel* - Invalid/wrong IP address** ▪ Data type of the variable: BOOL	E.g. PLC_PRG.xErrorFlag	
		FALSE	No error
		TRUE	Error occurred; CODESYS data logger shows error text

Legend:
* ... only for analogue cameras
** ... only for Ethernet cameras

8 Set camera configuration

► Set the following parameters under [Camera configuration]:

Parameter	Description	Possible values	
[Connection ID]	Select camera channel	[ConnToAnCam0]	Channel 0
		[ConnToAnCam1]	Channel 1
		[ConnToAnCam2]	Channel 2
		[ConnToAnCam3]	Channel 3

9 Select Ethernet camera variables

► Set the following parameters under [Ethernet camera variables]:

Parameter	Description	Possible values	
[IP address]	▪ Variable containing the IP address of the Ethernet camera ▪ Data type: STRING	E.g. PLC_PRG.sIPaddress	
[Greyscale mode]	▪ Variable activating/deactivating the scaling mode ▪ Data type: BOOL	E.g. PLC_PRG.xEnableGreyscale	
		FALSE	Greyscale mode deactivated
		TRUE	Greyscale mode active
[Link status]	▪ Variable signals if the camera is connected to the device ▪ Data type: BOOL	E.g. PLC_PRG.xConnStatus	
		FALSE	No error
		TRUE	Error occurred

8.6.4 Using analogue cameras in the project

20927

The image data of a connected analogue camera is transferred to a placeholder graphic on a visualisation page.

Display options for analogue cameras

21672

The following image resolutions are provided by analogue cameras depending on the colour coding system used:

- PAL: 720 x 576 (aspect ratio: 5:4)
- NTSC: 720 x 480 (aspect ratio: 3:2)

The CR1081/CR1085 supports the following display modes when analogue cameras are operated:

	Scaling mode (capture mode)	**1:1 mode (overlay mode)**
Description	The image can only be represented as a whole.	▪ A section of the image can be represented up to max. 100 % of the original image. ▪ The centre of the image section is identical with the centre of the original image.
Scalability	Any scaling of the image is possible.	The image cannot be scaled.
Behaviour if scaling is used	Any scaling of the aspect ratio is possible The image representation may be distorted.	The image is represented in the original aspect ratio.
Image repetition rate	Low image repetition rate **Reason: Long processing time!**	High image repetition rate

Example: Capture mode

9589

The original image can only be scaled in any manner as full-size image, also asymmetrically.

Original image:

Compressed image:

Streched image:

Example: Overlay mode

9560

The image cannot be scaled.
The centre of the image section is identical with the centre of the original image.

Original image:

Image section 1:

Image section 2:

Add analogue camera image into the project

22075

To add an analogue camera image to a project:

Prerequisite

> The analogue camera is correctly connected to the device.

> Extension for mobile cameras is activated (→ **Activate extensions for mobile cameras** (→ p. 115)).

1 Create visualisation

► Open or newly create visualisation (→ **Create visualisation** (→ p. 110)).

2 Add a placeholder graphic for the camera image

► Select window [ToolBox].

► Select tab [Special controls].

► Drag symbol [Analogue camera element] into the visualisation page via drag-and-drop.

> Visualisation shows the placeholder for the camera image (= rectangular visualisation element)

Placeholder graphics in frames are not supported!

The placeholder graphic of the camera image is automatically displayed on all visualisation pages of the project.

► To only display the placeholder graphic on certain pages: Create a copy of the placeholder graphic on the previous page and deactivate it by means of variable [Display image], if necessary (→ **Configure analogue camera image** (→ p. 124)).

Positioning the placeholder graphic outside the visible area of the visualisation page may cause visualisation problems during the runtime of the application.

► Always position the complete placeholder graphic in the visible area!

If the parameter [Invisible] is activated the other parameter settings are without effect.

► Move the placeholder graphic to the invisible area of the visualisation page to make the camera image invisible.

Configure analogue camera image

22076

To configure the analogue camera image:

1 Open configuration dialogue

► Open the visualization page with placeholder graphic.

► Double-click on placeholder graphic

> Window [Properties] shows configuration options of the camera image.

2 Change position and size of the camera image

► Set the following parameters under [m_StaticPosition]:

Parameter	Description	Possible values
[X]	X coordinate of the top left corner of the camera image	E.g. 290
[Y]	Y coordinate of the top left corner of the camera image	E.g. 220
[Width]	Width of the camera image (in pixels)	Recommended (PAL): 360 Max.: 720
[Height]	Height of the camera image (in pixels)	Recommended (PAL): 288 Max.: 576

3 Set image properties

► Set the following parameters under [Bitmap configuration]:

Parameter	Description	Possible values	
[Static ID]	Assign an alternative placeholder image in the image pool		
[Scale type]	Select the type of image scaling	Isotropic	The image keeps its proportions.
		Anisotropic	The image adapts to the size of the button.
		Not scaled	The image keeps its original size, even if the size of the placeholder is changed.
[Transparent]	Activate/deactivate evaluation of the transparency colour	☐	Do not evaluate transparency colour
		☑	Evaluate transparency colour
[Transparent color]	Select transparency colour		
[Horizontal alignment]	Set horizontal orientation of the camera image	Left	Align to the left
		Centred	Align to the centre
		Right	Align to the right
[Vertical alignment]	Set vertical orientation of the camera image	Top	Align to the top
		Centred	Align to the centre
		Bottom	Align to the bottom

4 Select an alternative image

► Set the following parameters under [Bild-Variable]:

Parameter	Description	Possible values
[Bild-ID]	▪ Variable for the image from the image collection displayed if no camera image is available ▪ Data type: STRING	E.g. PLC_PRG.sImageNoCam

5 Select variables for scaling and shift

► Set the following parameters under [Move & scale variables]:

Parameter	Description	Possible values
[Absolute movement] > [X]	▪ Variable containing the new X coordinate of the top left corner of the camera image ▪ Data type: INT	E.g. PLC_PRG.iCamNewX
[Absolute movement] > [Y]	▪ Variable containing the new Y coordinate of the top left corner of the camera image ▪ Data type: INT	E.g. PLC_PRG.iCamNewY
[Change of width]	▪ Variable containing the new width of the camera image ▪ Data type: INT	E.g. PLC_PRG.iCamNewWidth
[Change of height]	▪ Variable containing the new height of the camera image ▪ Data type: INT	E.g. PLC_PRG.iCamNewHeight

6 Select state variables

► Set the following parameters under [State variables]:

Parameter	Description	Possible values	
[Invisible]	▪ Variable for the control of the visibility of the camera image ▪ Data type: BOOL	E.g. PLC_PRG.xCamVisible	
		FALSE	Camera image is not visible
		TRUE	Camera image is visible

7 Select general camera variables

► Set the following parameters under [Common camera variables]:

Parameter	Description	Possible values	
[Display image]	▪ Variable for the control of the display of the camera image ▪ Data type: BOOL	E.g. PLC_PRG.xCamEnable	
		FALSE	Deactivate the camera image
		TRUE	Activate the camera image
[Rotation angle]	▪ Variable for the clockwise rotation of the camera image ▪ Data type: INT	E.g. PLC_PRG.iRotAngle	
		0	No rotation
		90	90°
		180	180°
		170	270°
[Mirror]	▪ Variable for the mirroring of the camera image along its vertical centre axis ▪ Data type: BOOL	E.g. PLC_PRG.xCamMirror	
		FALSE	Do not mirror the camera image
		TRUE	Mirror the camera image
[Error flag]	▪ Variable signals if at least one of the following camera errors occurred: - Invalid angle of rotation - Invalid range - Invalid channel* - Invalid/wrong IP address** ▪ Data type of the variable: BOOL	E.g. PLC_PRG.xErrorFlag	
		FALSE	No error
		TRUE	Error occurred; CODESYS data logger shows error text

Legend:
* ... only for analogue cameras
** ... only for Ethernet cameras

8 Set camera configuration

► Set the following parameters under [Camera configuration]:

Parameter	Description	Possible values	
[Connection ID]	Select camera channel	[ConnToAnCam0]	Channel 0

9 Select analogue camera variables

► Set the following parameters under [Analoge Kameravariablen]:

Parameter	Description	Possible values	
[Input channel]	▪ Variable containing the input channel of the requested analogue camera ▪ Data type: INT	E.g. PLC_PRG.iCamChan	
		0	Channel 1
		1	Channel 2
[Scaling mode]	▪ Variable activating/deactivating the scaling mode ▪ Data type: BOOL	E.g. PLC_PRG.xEnableScaleMode	
		FALSE	Scaling mode deactivated
		TRUE	Scaling mode active

8.7 Use CANopen

Contents

13902

- ► Observe the notes about task classification! (→ **Configure task processing** (→ p. 131))
- ► Observe the notes about CANopen! (→ System manual)

The following POUs are available to access a CAN interface configured for CANopen operation in an application.

Prerequisites

> Device configured as CANopen Manager (master) (→ **CANopen: Configure CANopen Manager (master)** (→ p. 89)).

8.7.1 CANopen: Send and receive SDO

7344

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

Name	Description	Reference
COP_SDOread	Read Service Data Object (SDO)	→ COP_SDOread (→ p. 148)
COP_SDOwrite	Write Service Data Object (SDO	→ COP_SDOwrite (→ p. 150)

8.7.2 CANopen: Network Management (NMT)

7343

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

Name	Description	Reference
COP_GetNodeState	Request state of one or several CANopen devices	→ COP_GetNodeState (→ p. 146)
COP_SendNMT	Send NMT control command to a CANopen device	→ COP_SendNMT (→ p. 152)

8.8 Use RawCAN (CAN Layer 2)

Contents

9726

The following POUs are available to access a CAN interface configured for CANopen operation in an application.

Prerequisites

> CAN interface is configured for operation as RawCAN (CAN Layer 2) (→ **Configure CAN interfaces** (→ p. 87)).

► Observe the notes about task configuration! (→ **Configure task processing** (→ p. 131))

8.8.1 RawCAN: Control CAN network nodes

21776

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

Name	Description	Reference
CAN_Enable	Activate CAN node	→ CAN_Enable (→ p. 225)
CAN_Recover	Reboot CAN node	→ CAN_Recover (→ p. 227)

8.8.2 RawCAN: Send and receive CAN messages

21777

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

Name	Description	Reference
CAN_Rx	Receive CAN message	→ CAN_Rx (→ p. 233)
CAN_RxMask	Receive CAN messages	→ CAN_RxMask (→ p. 235)
CAN_RxRange	Receive CAN messages	→ CAN_RxRange (→ p. 238)
CAN_Tx	Send CAN message	→ CAN_Tx (→ p. 240)

8.8.3 RawCAN: Request and send remote CAN messages

21778

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

Name	Description	Reference
CAN_RemoteRequest	Send request for a remote message	→ CAN_RemoteRequest (→ p. 229)
CAN_RemoteResponse	Reply to the request of a remote message	→ CAN_RemoteResponse (→ p. 231)

8.9 Configure task processing

Contents

11406

► Familiarise yourself with the following CODESYS functions!
  - Task configuration:
    → Online help > CODESYS Development System > Program application > Task configuration

The processing of the tasks is controlled by parameters.The user can set the parameters of the tasks separately.

CODESYS automatically creates the following tasks during project and visualisation creation:

Name	Description	Reference
MainTask	Task for the processing of the main program [PLC_PRG (PRG)]	→ **Configure main task** (→ p. 132)
VISU_TASK	Task for the processing of the visualisations	→ **Configure visualisation task** (→ p. 132)

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

► Create new task.

► Configure task properties:
1. [Type]: Cyclical
2. [Interval]: Requested cycle time

► Assign subprogram with POUs to the newly created task.

If the CAN buses are heavily utilised:

► Create an individual task for each CAN interface.

► Configure task properties:
1. [Priority]: High (< 5)
2. [Type]: Cyclical
3. [Interval]: requested cycle time (=transmission interval)

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

8.9.1 Configure main task

22074

The main task controls the execution of the standard program PLC_PRG. The programmer can assign additional subprograms to the main task.

To set the properties of the main task:

► In the device tree: Double-click on [MainTask]

> In the editor window: Tab [Configuration] shows current configuration of the visualisation task.

► Set the following values:
1. [Priority (0...31)]: Default setting or > Priority of the CAN tasks
2. [Type]: Cyclical
3. [Interval]: t#20ms

► Save the project to apply changes.

8.9.2 Configure visualisation task

3523

Each visualisation is executed separately from the program code in a separate task.

To set the properties of the visualisation task:

► In the device tree: Double-click on [VISU_TASK]

> In the editor window: Tab [VISU_TASK] > [Configuration] shows current configuration of the visualisation task.

► Set the following values:
1. [Priority (0...31)]: <16
2. [Type]: Cyclical
3. [Interval]: t#20ms

► Assign a priority that is as low as possible to the visualisation task ([VISU_TASK]) to avoid interruption of other tasks that are important for the core functions of the application.

► Execute the VISU_TASK in appropriate cyclic intervals to save the resources of the device-internal PLC and the fieldbus network.

► Save the project to apply changes.

9 Operation

Contents

7340

9.1 Transfer CODESYS project to the device

Contents

14059

- Familiarise yourself with the following CODESYS functions!
  - Transfer application to the device
    → Online help > CODESYS Development System > Transfer application to the PLC
  - Monitoring
    → Online help > CODESYS Development System > Application during the runtime > Monitoring of values
- Observe the notes about the states of the PLC application!
  → **Operating states of the PLC application** (→ p. 137)

9.1.1 Load application to CR1081/CR1085

12330

To load the created application to the device and store it non-volatilely:

Prerequisites:

> Connection between PC/laptop and CR1081/CR1085 established.
> Communication path has been set (→ **Set communication path of PLC** (→ p. 78)).
> Project has been tested.

1 Translate application

► In the project tree: Click on [Application].
► Select [Build] > [Build].
> CODESYS creates program code of the application.

2 Upload application to the device

► Select [Online] > [Login].
> CODESYS changes to the online mode.
> CODESYS loads active application to the device (download).
> Application on the device is in the STOP state.

3 Start application

► Select [Debug] > [Start].
> Application on the device changes to the RUN state.

4 Create boot application

► Select [Online] > [Create boot application].
> CODESYS stores the application non-volatilely on the device.

9.1.2 Delete application from the device

12418

To delete an application stored on the device:

1 Connect with the device

► In the device tree: Click on [Application].
► Select [Online] > [Login].
> CODESYS changes to the online mode.

2 Delete application

► In the device tree: Double-click on [Device (CR1081/CR1085)]
> Editor window shows device settings.
► Select tab [Applications].
► Click on [Liste aktualisieren].
> Editor window shows the applications stored on the device.
► Click on [Remove All] to delete all applications.
OR:
Select requested application and click on [Remove] to delete individual applications.
> CODESYS deletes the selected applications from the device.

9.2 Operating states of the PLC application

Contents

2688

The applications stored on the CR1081/CR1085 are executed independently from each other in separate tasks. An application can have the following operating states:

- **Unload**
  No application is stored on the device.
- **RUN**
  – The application is stored on the device.
  – The application is processed cyclically.
- **STOP**
  – The application is stored on the device.
  – The application is not processed.

9.2.1 Display operating mode of the PLC application

19910

To display the current operating status of the PLC application stored on the device:

> In the device tree: Symbol [Application] shows the current status

OR:

> In the online mode: CODESYS status bar shows the current state of the application.

9.2.2 Start PLC application

19929

To start the execution of the PLC application:

► In the device tree: Right-click on [Application] and select [Aktive Applikation setzen].
► Select [Online] > [Login].
> CODESYS changes to the online mode.
► Select [Debug] > [Starten].
> Application changes to the RUN state (→ **Operating states of the PLC application** (→ p. 137)).
► Optional: Repeat process for further applications.

9.2.3 Stop PLC application

19909

To stop the execution of the PLC application:

► In the device tree: Right-click on [Application] and select [Aktive Applikation setzen].
► Select [Online] > [Login].
> CODESYS changes to the online mode.
► Select [Debug] > [Stop].
> Application changes to the STOP state (→ **Operating states of the PLC application** (→ p. 137)).
► Optional: Repeat process for further applications.

9.3 Reset

Contents

18025

9.3.1 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. 140)
Reset (cold)	▪ application changes to the STOP state. ▪ All variables (VAR, VAR RETAIN) of the application are initialised.	→ **Reset application (cold)** (→ p. 140)
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. 140)

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

9.3.2 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.

9.3.3 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

9.3.4 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.

9.4 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 CR1081/CR1085.
► 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 Appendix

Contents

10354

10.1 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

10.2 ifm function libraries

Contents

7034

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

10.2.1 Bibliothek ifmCANopenManager.library

Contents

18443

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

COP_GetNodeState

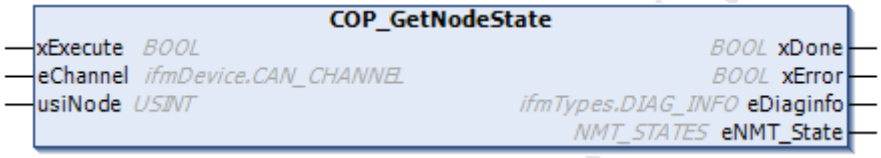
15956

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. 156)	
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. 154)	

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 bus user or bus user not available

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 parameter

YS_OBJECTID>

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. 156)	
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 parameter

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_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!

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 parameter

SYS_OBJECTID>

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. 156)	
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 parameter

7005

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:)	

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!

COP_SendNMT

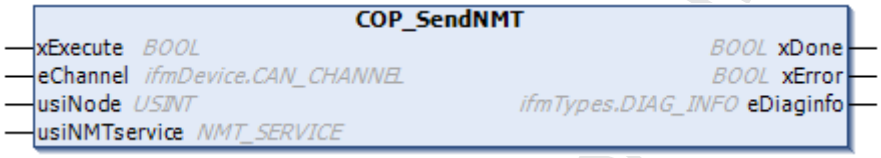
7006

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. 156)	
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. 154)	

Output parameter

7147

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:)	

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!

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

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

10.2.2 Bibliothek ifmDevicePDM360NG.library

Contents

10340

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

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

CAN_BAUDRATE (ENUM)

18519

Name	Description	Possible values		Data type	Value
CAN_BAUDRATE	Data transfer rate of the CAN interface	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

LED_COLOUR (ENUM)

14622

Name	Description	Possible values		Data type	Value
LED_COLOUR	Colour of the LED (RGB code)	BLACK (OFF)	Off	WORD	0x000000
		WHITE	White	WORD	0xFFFFFF
		RED	Red	WORD	0xFF0000
		GREEN	Green	WORD	0x00FF00
		BLUE	Blue	WORD	0x0000FF
		YELLOW	Yellow	WORD	0xFFFF00
		ORANGE	Orange	WORD	0xFF5A00
		MAGENTA	Magenta	WORD	0xFF00FF
		CYAN	Cyan	WORD	0x00FFFF

LED_FLASH_FREQ (ENUM)

21318

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	2 Hz	INT	3
		FRQ_5Hz	5 Hz	INT	4

IOchannel (GVL)

21327

Name	Description	Data type	Value
usiVBB30	Voltage supply, terminal 30	UINT	0
usiVBB15	Voltage supply, terminal 15	UINT	1

CANconstants (GVL)

20936

Name	Description	Data type	Value
usiNumberCANitf	Number of the CAN interfaces of the devices	UINT	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)** (→ p. 246))	ARRAY[0..8] OF **SysInfoStruct (STRUCT)** (→ p. 157)	

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

10.2.3 Bibliothek ifmEthernet.library

Contents

21213

The library contains program blocks (POU) and data structures for the management of the IP settings of the device.

GetIPsettings

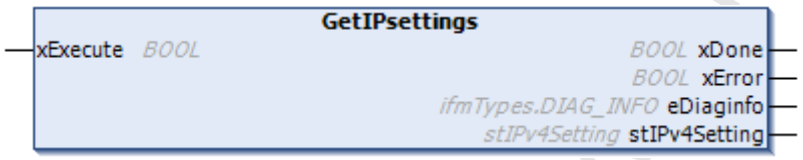
11473

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmEthernet.library

Symbol in CODESYS:



Description

21209

The FB reads the following IP settings of the device:

- IP address
- Network mask of the TCP/IP network as well as
- IP address of the gateway
- DHCP status

The FB provides the read values in a complex variable of the "stIPv4Setting" type.

Input parameter

21203

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

Output parameter

21208

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:)	
stIPv4Setting	stIPv4Setting	IPv4 settings of the device	→ **stIPv4Setting (STRUCT)** (→ p. 163)	

Diagnostic codes:

- STAT_DISABLE — FB/Function is disabled
- 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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_INVALID_CHANNEL — Error: Internal parameter setting fault
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

SetIPsettings

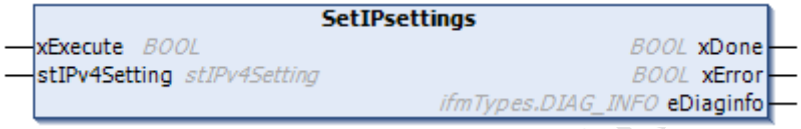
13779

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmEthernet.library

Symbol in CODESYS:



Description

21216

The FB changes the following IP settings of the device:

- IP address
- Subnet mask of the TCP/IP network
- IP address of the gateway
- DHCP status

The IP settings are transferred to the FB in a complex variable of the "stIPv4Setting" type.

The parameter DHCP has the highest priority. If DHCP = TRUE the values of the parameters IP address, subnet mask and gateway address are not evaluated.

After the call, the FB checks if DHCP is activated. If yes, the FB deactivates the DHCP client of the device and sets the required IP address.

► Enter the IP address using the following notation out the leading zeros when entering the IP address!
FALSE: `192.168.000.055`
CORRECT: `192.168.0.55`

Within a application only one instance of the FB may be active at a time.

► Call only one instance of the FB within a application!

Each call of the FB executes a write operation on the device-internal FLASH memory.

► Do not call FB cyclically!

Input parameter

21215

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
stIPv4Setting	stIPv4Setting	IPv4 settings of the device	→ **stIPv4Setting (STRUCT)** (→ p. 163)	

Output parameter

21217

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:)	

Diagnostic codes:

- STAT_DISABLE — FB/Function is disabled
- 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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_INVALID_CHANNEL — Error: Internal parameter setting fault
  ► 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!

stIPv4Setting (STRUCT)

21210

Designation	Data type	Description	Possible values	
sIpAddress	STRING (15)	IP address of the device (IPv4)	E.g. 192.168.1.10	
sSubnetMask	STRING (15)	Subnet mask	E.g. 255.255.255.0	
sDefaultGateway	STRING (15)	IP address of the network gateway	E.g. 192.168.1.1	
xDHCPStat	BOOL	Status of the DHCP client of the device	FALSE	DHCP client deactivated
			TRUE	DHCP client active

10.2.4 Bibliothek ifmFileUtil.library

Contents

21418

The library contains program blocks (POU), data structures and enumeration types for file operations.

Copy_PDM_To_USB

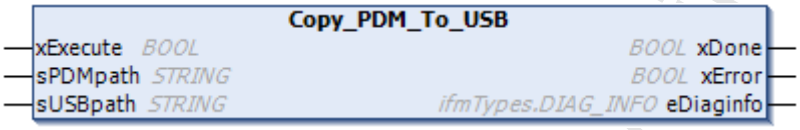
21645

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21646

The FB copies one or several files from a source directory on the device to a destination directory on an USB memory device.

The FB uses the Linux command "cp -rf <sPDMpath> <sUSBpath>".

Input parameter

21647

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
sPDMpath	STRING	Source directory / file on the device whose contents are to be copied	E.g. `/data/tmp`	
sUSBpath	STRING	Target directory on the USB memory device into which the files are to be copied ▪ Enter the directory as relative path to the mount point of the USB device. ▪ Example: `/backup` copies the data selected to the directory `/mnt/USB/backup`	E.g. `/backup`	

There are the following options for file selection:

- Individual files (e. g. /home/values/sample.csv)
- Several files by means of wildcards (e. g. /home/values/*.csv)
- All files of a directory (e. g. /home/values/)

The following combinations of source and target indications are valid:

- Values for source and target differ and both are folders
- Values for source and target differ and both are files
- Value for source is file and value for target is folder

The following combinations of source and target indications are invalid and cause an error message:

- No values for source and/or target
- Value for source is folder and value for target is file
- Values for source and/or target contain subsequent "/" (e. g. /home/values///file.csv)
- Values for source and target are identical (file and folder)

Output parameter

21648

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:)	

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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- 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!

Copy_USB_To_PDM

SYS_OBJECTID>

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:

Description

21652

The FB copies one or several files from a USB memory device to a destination directory on the device.

The FB uses the Linux command "cp -rf <sUSBpath> <sPDMpath>".

Input parameter

21653

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
sUSBpath	STRING	Source directory / file on the USB memory device ▪ Enter the directory as relative path to the mount point of the USB device. ▪ Example: `/backup` copies all data of the directory `/mnt/USB/backup`	E.g. `/data`	
sPDMpath	STRING	Target directory/file on the device	E.g. `/data/target`	

There are the following options for file selection:

- Individual files (e. g. /home/values/sample.csv)
- Several files by means of wildcards (e. g. /home/values/*.csv)
- All files of a directory (e. g. /home/values/)

The following combinations of source and target indications are valid:

- Values for source and target differ and both are folders
- Values for source and target differ and both are files
- Value for source is file and value for target is folder

The following combinations of source and target indications are invalid and cause an error message:

- No values for source and/or target
- Value for source is folder and value for target is file
- Values for source and/or target contain subsequent "/" (e. g. /home/values///file.csv)
- Values for source and target are identical (file and folder)

Output parameter

21654

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:)	

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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- 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!

DeleteFile

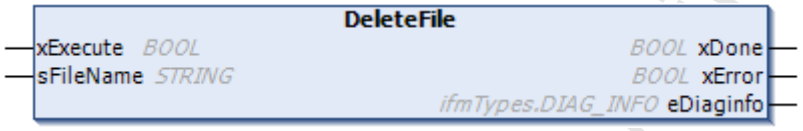
21657

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21658

The FB deletes a file from the FLASH memory of the device or the USB memory device.

Input parameter

21659

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
sFileName	STRING	Directory path and name of the file	E.g. '/home/project/data.txt'	

The following entries for "sFileName" are invalid and cause an error message:

- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

Output parameter

21660

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:)	

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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- 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!

FileCopy

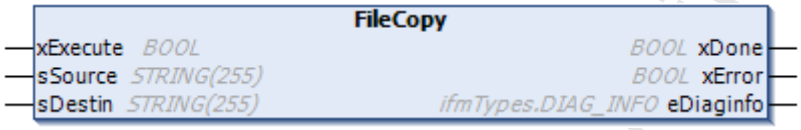
21661

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21662

The FB copies one or several files from a source directory on the device to a destination directory on the device.

The FB uses the Linux command "cp -rf <sSource> <sDestin>".

Input parameter

21663

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

sSource	STRING	Source directory / file	E.g.
			`/data/source`
			`/data/sourcefile.txt`
			`/data/*.txt`	

sDestin	STRING	Target directory/file on the device	E.g. `/data/destin`	

There are the following options for file selection:

- Individual files (e. g. /home/values/sample.csv)
- Several files by means of wildcards (e. g. /home/values/*.csv)
- All files of a directory (e. g. /home/values/)

The following combinations of source and target indications are valid:

- Values for source and target differ and both are folders
- Values for source and target differ and both are files
- Value for source is file and value for target is folder

The following combinations of source and target indications are invalid and cause an error message:

- No values for source and/or target
- Value for source is folder and value for target is file
- Values for source and/or target contain subsequent "/" (e. g. /home/values///file.csv)
- Values for source and target are identical (file and folder)

Output parameter

21664

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:)	

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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- 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!

SyncFileMemory

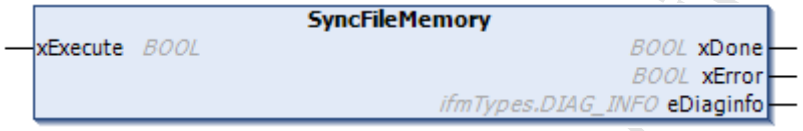
21667

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21668

The FB synchronises the NAND flash memory with the working memory (RAM) of the device.

► Carry out the FB as last operation before shutting down the device!

The FB uses the Linux command "sync".

Input parameter

21669

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

Output parameter

21670

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:)	

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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

ReadCSV8Byte

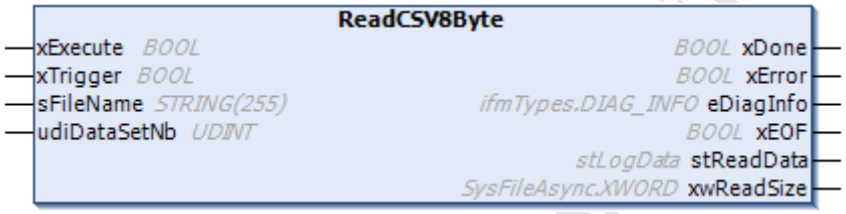
21526

Function block type: Function block (FB)

Behaviour model: EXECUTE with Trigger

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21527

The FB reads data sets of a CSV file and provides them. The FB assigns the value 0 to unused cells. The FB stores the time stamp of each data set.The CSV must have been created by means of the FB **WriteCSV8Byte** (→ p. 177).

Each data set has to have a size of 54 bytes.

Input parameter

21528

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
xTrigger	BOOL	Trigger action	FALSE ⇨ TRUE	FB reads the selected data set
			Other	No action
sFileName	STRING	Directory path and name of the file	E.g. '/home/project/data.txt'	
udiDataSetNb	UDINT	Number of the data set to be read from the file	0 ... 65535	

Output parameter

21529

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:)	
xEOF	BOOL	Indication of the file end	FALSE	File end not yet reached
			TRUE	File end reached
stReadData	stLogData	Data structure with all read data → **stLogData (STRUCT)** (→ p. 199)		
xwReadSize	XWORD	Size of the data set read last	54	Reading process successful
			Other	Reading process faulty

Diagnostic codes:

- STAT_INACTIVE — State: FB/Function is inactive.
- STAT_BUSY — State: FB/Function is currently executed.
- STAT_READY — State: File ready for triggered write / read accesses
- STAT_DONE — State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_FILE_SYSTEM — Error: File operation failed.
- ERR_NO_OBJECT — Error: File not available.

WriteCSV8Byte

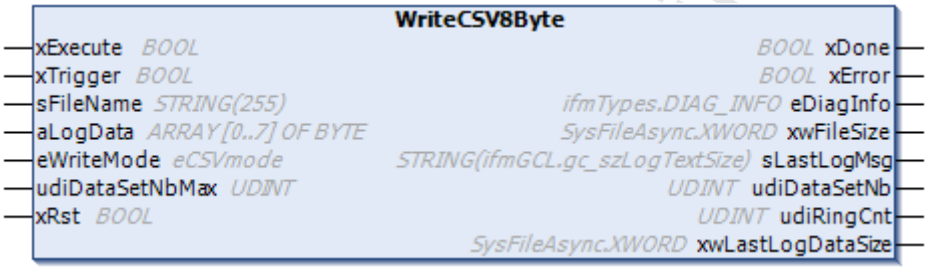
21550

Function block type: Function block (FB)

Behaviour model: EXECUTE with Trigger

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21551

The FB reads the data of an array of 8 bytes and writes them into a CSV file as a data set. The FB stores a time stamp in addition to each data set (date, time). After the data set has been written the FB automatically increments the pointer to the next array in the data structure.

Principle:

No. of the data set	Contents of the data set
1	Time stamp*, byte 0, byte 1, ..., Byte 7
2	Time stamp, byte 0, byte 1, ..., Byte 7
...	...
udiDataSetNbMax	Time stamp, byte 0, byte 1, ..., Byte 7

* ...Format of the time stamp DD.MM.YYYY HH:MM:SS

Example for a data set:

```
04.08.2016 19:59:55,0,15,245,15,251,15,0,8
```

The user can choose between the following write modes:

- **Linear**:
  The data sets are written linearly. Theoretically, the number of data sets is unlimited; it is only limited by the maximum file size predefined by the operating system. Existing data sets are not overwritten.

- **Ring**:
  The data is written into a ring buffer. The number of data sets is limited. After the last data set has been written the FB starts again at the first data set. Existing data sets are overwritten.

In the linear mode the bytes 0 to 53 are reserved for the CSV header. The CSV header can be added subsequently by means of the FB **WriteCSV8ByteHeader** (→ p. 180).

In the ring mode the CSV header must be written into the file before the writing of the data sets.

Input parameter

21552

Parameter	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE ⇨ TRUE	FB opens file
			TRUE ⇨ FALSE	▪ FB closes file ▪ All outputs are reset
xTrigger	BOOL	Trigger action	FALSE ⇨ TRUE	FB reads the selected data set
			Other	No action
sFileName	STRING	Directory path and name of the file	E.g. '/home/project/data.txt'	
aLogData	ARRAY [0...7] OF BYTE	Array with the data to be written into the CSV file		
eWriteMode	eCSVmode	Write mode → **eCSVmode (ENUM)** (→ p. 200)	LINEAR	Linear mode*
			RING	Ring mode
udiDataSetNbMax	UDINT	Maximum number of data sets in the ring mode	0 65535	
xRst	BOOL	Delete CSV file and pointer to the data set	FALSE ⇨ TRUE	Delete pointer and CSV file
			Other	No action

* ... preset value

The following entries for "sFileName" are invalid and cause an error message:

- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

Output parameter

21553

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:)	
xwFileSize	XWORD	Current file size of the file (in bytes)	0 ... 4294967295	
LastLogMsg	STRING	Data set written last as STRING	E.g. 06.12.2016 20:35:45,10,101,255,103,104,105,106,$n	
udiDataSetNb	UDINT	Number of the data set written into the file last	0 ... 65535	
udiRingCnt	UDINT	Counter for completed cycles of the ring buffer storage	0 ... 65535	

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.
- STAT_READY — State: File ready for triggered write / read accesses
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_FILE_SYSTEM — Error: File operation failed.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_NO_OBJECT — Error: File not available.

WriteCSV8ByteHeader

21562

Function block type: Function block (FB)

Behaviour model: EXECUTE with Trigger

Library: ifmFileUtil.library

Symbol in CODESYS:

Description

21563

The FB writes the header into a CSV file. The header is always written in the bytes 0 to 53 of the file.

Input parameter

21564

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
sFileName	STRING	Directory path and name of the file	E.g. '/home/project/data.txt'	
stHeader	stCSVHeader	Structure with header data → **stCSVHeader (STRUCT)** (→ p. 199)	Timestamp,R1C2,R1C3,...,R1C9*	

* ... preset value

The following entries for "sFileName" are invalid and cause an error message:

- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

Output parameter

21565

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:)	
LastLogMsg	STRING	Written header as STRING		
udiDataSetPosi	UDINT	Number of the data set written last	0 65535	

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_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_FILE_SYSTEM — Error: File operation failed.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_NO_OBJECT — Error: File not available.

ReadParmSingleAsync

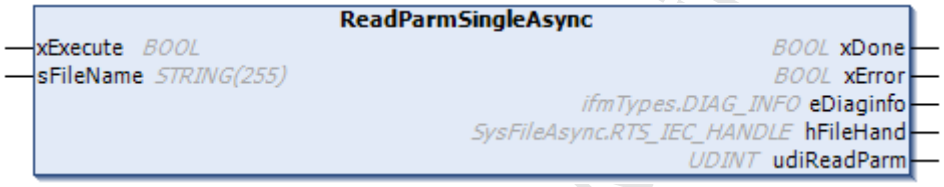
19010

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

12763

The FB reads a single data set of the UINT type of a text file and provides it.

The file must only contain this single data set. The value in the file must be stored as a 10-digit value and right-aligned.

Example:

Value	Saved in the file as
1	000000001
123	000000123
12345567890	1234567890

Input parameter

12766

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
sFileName	STRING	Directory path and name of the file	E.g. '/home/project/data.txt'	

Output parameter

12677

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:)	
hFileHand	RTS_IEC_HANDLE	File description of the runtime system	< 1	Error
			Other	No error
udiReadParm	UDINT	Parameter read from the text file	0 ... 4294967295	

▪

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_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_FILE_SYSTEM — Error: File operation failed.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_NO_OBJECT — Error: File not available.

WriteParmSingleAsync

11384

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:

Description

12732

The FB writes a single parameter of the UDINT type into a text file.
The parameter is stored in the file as a 10-digit value and right-aligned.
Example:

Value	Saved in the file as
1	0000000001
123	0000000123
1234567890	1234567890

Input parameter

12733

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
sFileName	STRING	Directory path and name of the file	E.g. '/home/project/data.txt'	
udiInData	UDINT	Parameter to be written to	0 ... 65535	

Output parameter

12733

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:)	
hFileHand	RTS_IEC_HANDLE	File description of the runtime system	< 1	Error
			Other	No error

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_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_FILE_SYSTEM — Error: File operation failed.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_NO_OBJECT — Error: File not available.

Buffer_To_String

SYS_OBJECTID>

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:

Description

21496

The FB reads a BYTE array, concatenates the values and provides the result in a STRING array.

- Size of the BYTE array: 500 bytes
- Size of the STRING array: 25 strings

If the FB finds the control character for line end/line break in the source data it writes the subsequent data into a new array line.

If the FB finds a 0 value or the control character for file end in the source data it terminates the conversion.

BUFFER_TO_STRING is a help function for other function blocks.

Input parameter

21497

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
abyInData	ARRAY [0...499] OF BYTE	Buffer storage containing the data to be read	Per byte: 0x00 ... 0xFF	

Output parameter

21498

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:)	
aTextLine	ARRAY [0..24] OF STRING	Array with converted text lines		
byNoOfTextLine	BYTE	Number of lines found in the array		

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.

GetFileSizeAsync

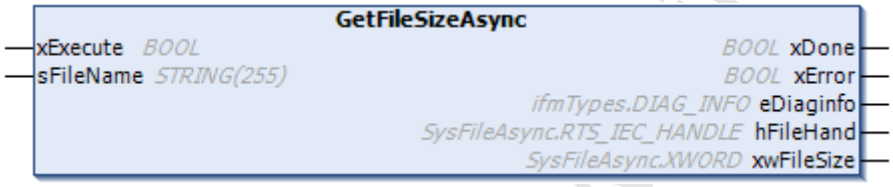
21700

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

3952

The FB determines the size of a file.

Input parameter

11620

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
sFileName	STRING	Directory path and name of the file	E.g. '/home/project/data.txt'	

The following entries for "sFileName" are invalid and cause an error message:

- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

Output parameter

2075

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:)	
hFileHand	RTS_IEC_HANDLE	File description of the runtime system	< 1	Error
			Other	No error
xwFileSize	XWORD	Current file size of the file (in bytes)	0 ... 4294967295	

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_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_FILE_SYSTEM — Error: File operation failed.
- ERR_NO_OBJECT — Error: File not available.

GetMemoryInfoAsync

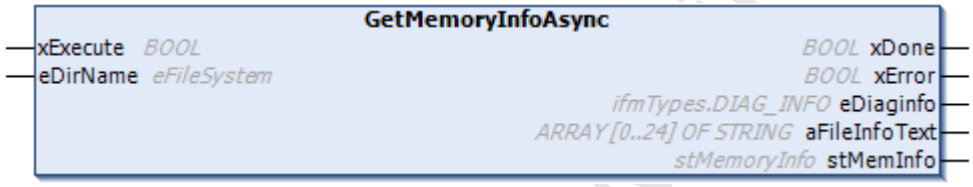
21520

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21521

The FB provides detailed information about the memory usage and the available memory of a directory (e. g. /data). The FB stores the information in a text file that has the same name as the value at "eDirName". The text files are stored in the following directories:

eDirName	Memory location
NAND_FLASH_1GB	/data/
Other	/home/cds-apps/PlcLogic/

Example:
eDirName: HOME
Memory location: /home/cds-apps/PlcLogic/home.txt

Input parameter

21522

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
eDirName	eFileSystem	Directory whose storage information is to be read	→ **eFileSystem (ENUM)** (→ p. 200)	

Output parameter

21523

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:)	
aFileInfoText	ARRAY [0..24] OF STRING	Array with storage information of the directory path		
stMemInfo	stMemoryInfo	Memory information read from the device	→ **stMemoryInfo (STRUCT)** (→ p. 199)	

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_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_FILE_SYSTEM — Error: File operation failed.
- ERR_NO_OBJECT — Error: File not available.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!

ByteArray_To_String

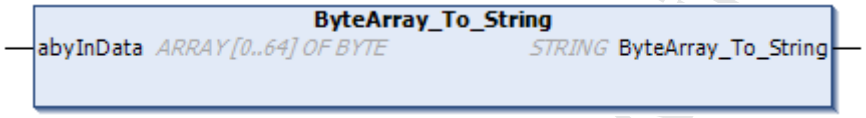
21578

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21579

The function reads a byte array, concatenates the values and provides them as a STRING.

Input parameter

21580

Parameter	Data type	Description	Possible values
abyInData	ARRAY [0...63] OF BYTE	Array with input data	Per byte: 0x00 ... 0xFF

Output parameter

21581

Parameter	Data type	Description	Possible values
ByteArray_To_String	STRING	Return value of the function	

DW_To_4Byte

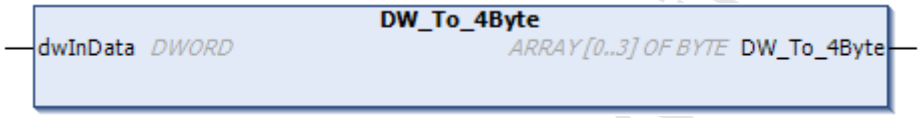
21584

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21585

The function converts a DWORD into an array of 4 bytes.
Example:
DWORD: 0xFFEEDDCC
ARRAY [0...3] OF BYTE: 0xFF | 0xEE | 0xDD | 0xCC

Input parameter

21586

Parameter	Data type	Description	Possible values
dwInData	DWORD	Data	E.g. 0xFFDDEECC

Output parameter

21587

Parameter	Data type	Description	Possible values
DW_To_4BytebyNoOfTextLine	ARRAY [0..3] OF BYTE	Return value of the function	E.g. Byte 0: CC Byte 1: DD Byte 2: EE Byte 3: FF

Pack2Byte_To_Word

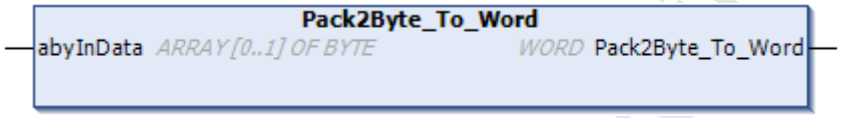
21591

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21592

The function converts an array of 2 bytes into a word.

Example:

aInData[0] = 0x00
aInData[1] = 0xAA

Pack2Byte_To_Word = 0xAA00

Input parameter

21593

Parameter	Data type	Description	Possible values
abyInData	ARRAY [0...1] OF BYTE	Array with input data	Per byte: 0x00 ... 0xFF

Output parameter

21595

Parameter	Data type	Description	Possible values
Pack2Byte_To_Word	WORD	Return value of the function	0x0000 ... 0xFFFF

Pack4Byte_To_DW

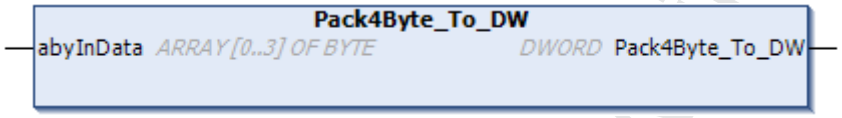
21614

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21615

The function converts an array of 4 bytes into a double word.

Example:

aInData[0] = 0x00
aInData[1] = 0x11
aInData[2] = 0xAA
aInData[3] = 0xFF

Pack4Byte = 0xFFAA1100

Input parameter

21616

Parameter	Data type	Description	Possible values
abyInData	ARRAY [0...3] OF BYTE	Array with input data	Per byte: 0x00 ... 0xFF

Output parameter

21617

Parameter	Data type	Description	Possible values
Pack4Byte_To_DW	DWORD	Return value of the function	0x00000000 ... 0xFFFFFFFF

RTC_To_String

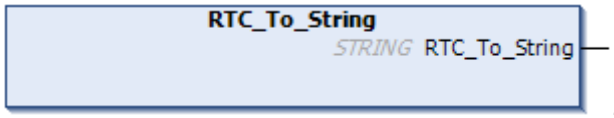
21621

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21622

The function reads the device-internal real-time clock (RTC) and provides the operating time in seconds since the last system start as a string.

Output parameter

21624

Parameter	Data type	Description	Possible values
RTC_To_String	STRING	Current time in the format `DD.MM.YYYY HH:MM:SS`	E.g. `02.08.2016 08:59:03`

Word_To_2Byte

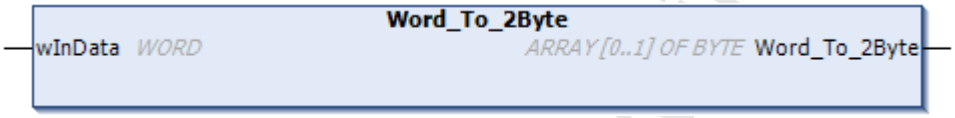
21634

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21635

The function converts a WORD into an array of 2 bytes.

Example:

wInData = 0xFFEE

Word_To_2Byte[0] = 0xEE
Word_To_2Byte[1] = 0xFF

Input parameter

21623

Parameter	Data type	Description	Possible values
wInData	WORD	Data	0x0000 ... 0xFFFF

Output parameter

21636

Parameter	Data type	Description	Possible values
Word_To_2Byte	ARRAY [0..1] OF BYTE	Return value of the function	Per byte: 0x00 ... 0xFF

_8Byte_To_CSV

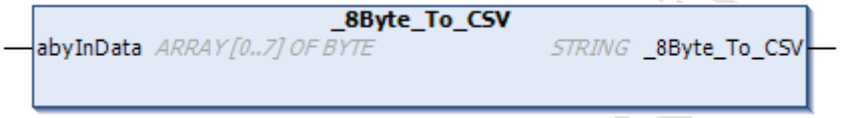
21639

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



Description

21640

The function converts the data of an array of 8 bytes into a CSV string and provides this string. One-digit and two-digit numbers are supplemented with leading zeros. The individual values are separated by a comma. The line end is indicated by a comma (,).

Example:

aInData[0] = 0x00
aInData[1] = 0x01
aInData[2] = 0x63
aInData[3] = 0x64
aInData[4] = 0xA0
aInData[5] = 0xEE
aInData[6] = 0xFF
aInData[7] = 0x11

8byte_To_CSV = 000,001,099,100,160,238,255,017,

Input parameter

21641

Parameter	Data type	Description	Possible values
abyInData	ARRAY [0..7] OF BYTE	Array with input data	Per byte: 0x00 ... 0xFF

Output parameter

21642

Parameter	Data type	Description	Possible values
_8Byte_To_CSV	STRING	CSV string; individual bytes are separated by a comma	E.g. 002,004,008,016,032,064,128,255

stCSVHeader (STRUCT)

21484

Designation	Data type	Description	Possible values
sColumn1	STRING (18)	Heading of column 1	DateAndTime*
sColumn2	STRING (3)	Heading of column 2	R1C2*
sColumn3	STRING (3)	Heading of column 3	R1C3*
sColumn4	STRING (3)	Heading of column 4	R1C4*
sColumn5	STRING (3)	Heading of column 5	R1C5*
sColumn6	STRING (3)	Heading of column 6	R1C6*
sColumn7	STRING (3)	Heading of column 7	R1C7*
sColumn8	STRING (3)	Heading of column 8	R1C8*
sColumn9	STRING (3)	Heading of column 9	R1C9*

* ... * preset value (R = Row, C = Column, R1C2 = Row 1, Column 2)

stMemoryInfo (STRUCT)

21486

Designation	Data type	Description	Possible values
udiTotalKB	UDINT	Total memory (in Kbytes)	
udiUsedKB	UDINT	Memory used (in Kbytes)	
udiAvailKB	UDINT	Free memory (in Kbytes)	
usiUsedPerc	UDINT	Percentage of the memory used referred to the entire memory (in %)	

stLogData (STRUCT)

21485

Designation	Data type	Description	Possible values
sTimeStamp	STRING	Time stamp (DD.MM.YYYY HH:MM:SS)	
aData	ARRAY [0...7] OF BYTE	CSV data set	
sRawData	STRING	sTimeStamp and aData as STRING; Values are separated by a comma	

eCSVmode (ENUM)

21487

Name	Description	Possible values	Data type	Value
eCSVmode	Write mode for CSV file	NONE	INT	0
		LINEAR	INT	1
		RING	INT	2

eFileSystem (ENUM)

22611

Name	Description	Possible values		Data type	Value
eFileSystem	Directory path for the integration of devices and files	ROOT	/	INT	0
		DEV	/dev	INT	1
		TMPFS	/tmpfs	INT	2
		HOME	/home	INT	3
		NAND_FLASH_1GB	/data	INT	4
		FONTS	/opt/qt-x.y.z/lib/fonts	INT	5
		USB1	/tmpfs/media/usb/<USB Name>	INT	6
		USB2	/tmpfs/media/usb/<USB Name>	INT	7
		USB3	/tmpfs/media/usb/<USB Name>	INT	8
		USB4	/tmpfs/media/usb/<USB Name>	INT	9

10.2.5 Bibliothek ifmIOcommon.library

Contents

21286

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

SupplySwitch

21288

Function block type: Function block (FB)

Behaviour model: ENABLE

Library: ifmIOcommon.library

Symbol in CODESYS:

Description

21289

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 simultaneously:

- Voltage at terminal 15 (VBB15) < minimal operating voltage
- Voltage at terminal 30 (VBB30) <= Power down threshold
- Voltage at terminal 30 (VBB30) >= Protected Overvoltage threshold

Separation of the VBBs is done before the next PLC cycle starts. Depending on the charging status of the internal capacitors it may take some time until the device switches off.

Input parameter

21290

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	FB is deactivated
			TRUE	FB is activated
xSwitchOff	BOOL	Deactivate latching switch of the device	FALSE	No action
			TRUE	Request deactivation of the latching switch

Output parameter

SYS_OBJECTID>

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_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- ERR_ACCESS — Error: FB/Funktion cannot access the required resource; Resource is blocked by another task.

INPUT_FILTER (ENUM)

14409

Name	Description	Possible values	
INPUT_FILTER	Valid filters for inputs of the FBs (in Hz)	FILTER_OFF	No filtering
		FILTER_120_HZ	120 Hz
		FILTER_47_HZ	47 Hz
		FILTER_22_HZ	22 Hz
		FILTER_10_HZ	10 Hz
		FILTER_5_0_HZ	5 Hz
		FILTER_2_5_HZ	2,5 Hz
		FILTER_1_2_HZ	1,2 Hz
		FILTER_0_7_HZ	0,7 Hz

INPUT_MODE (ENUM)

21349

Name	Description	Possible values	
INPUT_MODE	Operating mode of the inputs	OFF	Inactive; Outputs are not updated
		IN_DIGITAL_CSI_DIGITAL	digital current sinking voltage level measurement and evaluation.
		IN_DIGITAL_CSI	current sinking with analogue measurement and digital evaluation. No diagnosis.
		IN_DIGITAL_CSI_NAMUR	current sinking with analogue measurement and digital evaluation. Diagnosis capable by NAMUR.
		IN_DIGITAL_CSI_OSSD	current sinking for evaluation of OSSD compliant sensors with diagnosis.
		IN_DIGITAL_CSO	current sourcing with analogue measurement and digital evaluation. No diagnosis.
		IN_VOLTAGE_10	analogue current sinking voltage measurement 0-10 V.
		IN_VOLTAGE_32	current sinking voltage measurement 0-32 V.
		IN_VOLTAGE_RATIO	ratiometric current sinking voltage measurement.
		IN_VOLTAGE_RATIO_CSO	ratiometric current sourcing voltage measurement.
		IN_CURRENT_CSI	current sinking current measurement 0-20 mA.
		IN_FREQUENCY_CSI	current sinking frequency measurement.
		IN_FREQUENCY_CSO	current sourcing with frequency measurement.
		IN_RESISTOR	current sinking with resistance measurement.

Name	Description	Possible values	
		IN_PERIOD_RATIO_CSI	current sinking for absolute and ratiometcric period measurement.
		IN_PERIOD_RATIO_CSO	current sourcing for absolute and ratiometric period measurement.
		IN_INC_ENCODER_CSI	current sinking for incremental encoder evaluation. (‘A’ channel)
		IN_INC_ENCODER_CSO	current sourcing for incremental encoder evaluation. (‘A’ channel)
		IN_PHASE_CSI	current sinking for phase measurement (‘A’ channel).
		IN_PHASE_CSO	current sourcing for phase measurement (‘A’ channel).
		IN_COUNT_CSI	current sinking for counting edges of fast signals.
		IN_COUNT_CSO	current sourcing for counting edges of fast signals.
		IN_SYSTEM_VOLTAGE_CSI	current sinking for system voltage measurement.
		IN_BOARD_TEMPERATURE_CSI	current sinking for system temperature measurement. E.g. board temperature.

10.2.6 Bibliothek ifmPDM360NG.library

10341

The library is a container library. It contains all function libraries required for the programming of the device.

- **Bibliothek ifmCANopenManager.library** (→ p. 145)
- **Bibliothek ifmDevicePDM360NG.library** (→ p. 155)
- **Bibliothek ifmEthernet.library** (→ p. 158)
- **Bibliothek ifmFileUtil.library** (→ p. 164)
- **Bibliothek ifmIOcommon.library** (→ p. 201)
- **Bibliothek ifmPDM360NGutil.library** (→ p. 207)
- **Bibliothek ifmRawCAN.library** (→ p. 224)
- **Bibliothek ifmSysInfo.library** (→ p. 243)
- **Bibliothek ifmUSBstorageUtil.library** (→ p. 247)

10.2.7 Bibliothek ifmPDM360NGutil.library

Contents

21329

The library contains device-specific POUs for the programming of the device under CODESYS.

For complex data types (STRUCT), enumeration types (ENUM) and gloabal constants (GVL) used: → **Bibliothek ifmDevicePDM360NG.library** (→ p. 155)

BuzzerOn

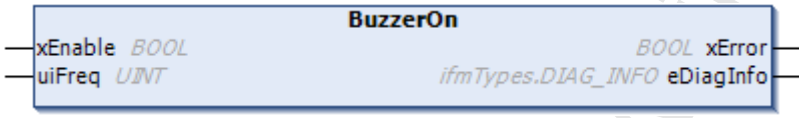
11533

Function block type: Function block (FB)

Behaviour model: ENABLE

Library: ifmPDM360NGutil.library

Symbol in CODESYS:



Description

21190

The FB activates/deactivates the device-internal buzzer with the configured properties.
Adjustable properties:

- Tone pitch of the buzzer

Within a application only one instance of the FB may be active at a time.
► Call only one instance of the FB within a application!

Configuration of tone pitch and tone duration: → **BuzzerClick** (→ p. 222)

Input parameter

21192

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	FB is deactivated
			TRUE	FB is activated
uiFreq	UINT	Tone pitch of the buzzer (in Hz)	500 ... 5000	500 Hz ... 5000 Hz

Output parameter

21193

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_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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- ERR_INVALID_FREQUENCY — Error: Invalid frequency value; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_IINVALID_CHANNEL — Error: No audio signalling device (buzzer) available in the device
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

DisableTouchScreen

21333

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmPDM360NGutil.library

Symbol in CODESYS:

Description

21330

The FB activates/deactivates the touch screen functionality of the display.

The POU is only available for the following devices: CR1082, CR1201.

The touch screen functionality is activated by default upon start-up of the device.

Input parameter

21342

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
xDisableTouch	BOOL	Activate/deactivate the touch screen functionality of the display	FALSE	Touch screen functionality activated
			TRUE	Touch screen functionality deactivated

Output parameter

21343

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:)	

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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_INVALID_CHANNEL — Error: Internal parameter setting fault
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

InitNANDflashAsync

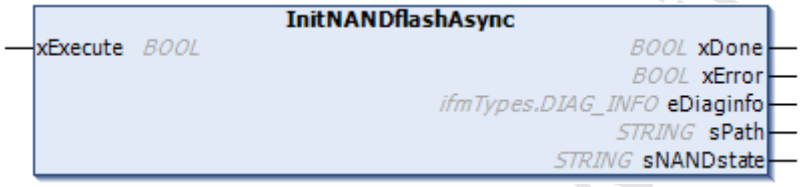
21420

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmPDM360NGutil.library

Symbol in CODESYS:



Description

21421

The FB initialises the device-internal NAND flash memory and adds it to the file system of the device under the path `/data`.

Only for devices with 1 GB NAND flash memory (→ installation instructions).

If the data memory `/data` is not to be used in the CODESYS project, no initialisation of the NAND memory is necessary.

The initialisation process of the NAND memory takes about 2 seconds. During that time the memory cannot be accessed.

► Do not access the NAND memory before the value TRUE is provided at FB output DONE.

The FB generates the following text files on the device during initialisation. The user can use the contents of the files for diagnostics:

- `/data/NANDmountLogs.txt`
  Contains status messages of the initialisation process
- `/data/NANDmountStat.txt`
  Contains the final initialisation status of the memory

Input parameter

21422

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

Output parameter

21423

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:)	
sPath	STRING	Absolute directory path of the NAND flash memory after initialisation	\data	Directory path
			No NAND Flash found!	Error occurred
sNANDstate	STRING	Status of the NAND flash memory	ubi1.datafs on /data type ubifs (rw,snyc,realtime)	NAND successfully initialised
			No NAND Flash found!	Error occurred

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_FILE_SYSTEM — Error: NAND memory cannot be initialised.
  ► Check whether the device has a NAND memory.
  OR:
  Error: The initialisation status could not be read from the file NANDmountStat.txt
  ► Check whether the following message is contained in the file:
  "ubi1:datafs on /data type ubifs (rw,sync,realtime)$N"
  OR:
  Error: File NANDmountStat.txt cannot be read.
  ► Check whether the file can be accessed.
- ERR_NO_OBJECT — Error: The initialisation status could not be read from the file NANDmountStat.txt
  ► Check whether the file exists.

LinuxSysCallAsync

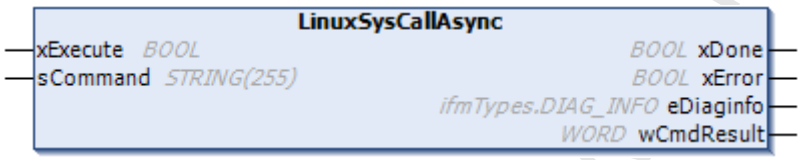
20935

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmPDM360NGutil.library

Symbol in CODESYS:



Description

21464

The FB transmits a Linux command with parameters (max. 255 characters) to the operating system of the device and provides the return value of the command. The transmitted command is executed asynchronously in the background.

Input parameter

21465

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
sCommand	STRING	Linux-Befehl inkl. Parameter (max. 255 characters)		

Output parameter

21466

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:)	
wCmdResult	WORD	Return value of the Linux command	Depending on the command	

Diagnostic code:

- 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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- 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!

LoadSplashScreen

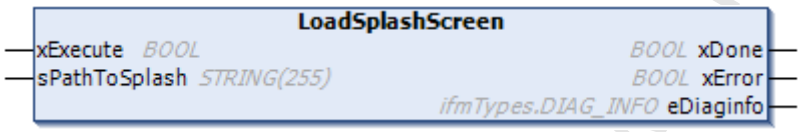
20934

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmPDM360NGutil.library

Symbol in CODESYS:



Description

14533

The FB loads an image file in the flash memory of the device. The image file will be used as splash screen after the next reboot of the device.

Within a application only one instance of the FB may be active at a time.

► Call only one instance of the FB within a application!

Each call of the FB executes a write operation on the flash memory of the device.

► Do not call the function block cyclically in the program code!

► Observe the characteristics of the splash screen → **image properties** (→ p. 107)

Input parameter

21110

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
sPathToSplash	STRING	Absolute directory path to the image file	E.g. \data\screenpic.bmp	

Output parameter

SYS_OBJECTID>

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:)	

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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- 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_INVALID_CHANNEL — Error: Internal parameter setting fault
  ► Contact the ifm Service Center!
- ERR_FILE_SYSTEM — Error: Image file and/or directory path not available
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

SetBootupBacklight

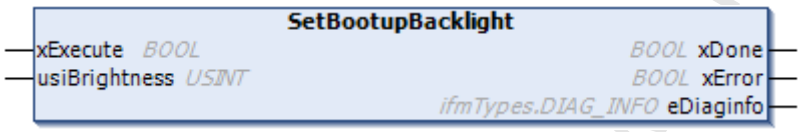
20933

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmPDM360NGutil.library

Symbol in CODESYS:



Description

21428

The FB changes the brightness of the display for the duration of the start-up of the device. The new value is applied after the next reboot of the device.

Each call of the FB executes a write operation on the flash memory of the device.

► Do not call the function block cyclically in the program code!

Input parameter

21429

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
usiBrightness	UINT	Brightness of the display (in %)	0 ... 10 ... 100	0 % ... 10 %* ... 100 %

* ... preset value

Output parameter

21430

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:)	

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_INVALID_MODULE — Error: Internal parameter setting fault
  ► Contact the ifm Service Center!
- ERR_INVALID_CHANNEL — Error: Internal parameter setting fault
  ► Contact the ifm Service Center!
- 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_INVALID_FREQUENCY — Error: Internal parameter setting fault
  ► Contact the ifm Service Center!
- ERR_ACCESS — Error: FB/Funktion cannot access the required resource; Resource is blocked by another task.
- ERR_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

SetKeyClick

20931

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmPDM360NGutil.library

Symbol in CODESYS:

Description

21471

The FB activates/deactivates the click sound function of the operating system. The click sound function confirms the actuation of a function key of the device by means of a sound.

The FB configures the properties of the click sound.

Adjustable properties:

- Tone pitch
- Duration of the sound

Input parameter

21472

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
uiFreq	UINT	Frequency of the buzzer sound (in Hz)	500 ... 5000	500 Hz* ... 5000 Hz
tDuration	TIME	Duration of the buzzer sound (in ms)	1 ... 10	1 millisecond* ... 10 milliseconds
xOperMode	BOOL	Control key click function	FALSE	Deactivate function
			TRUE	Activate function

* ... preset value

Output parameter

21473

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:)	

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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- ERR_INVALID_VALUE — Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR_INVALID_FREQUENCY — Error: Invalid frequency value; Function call has been stopped.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_IINVALID_CHANNEL — Error: No audio signalling device (buzzer) available in the device
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

BuzzerClick

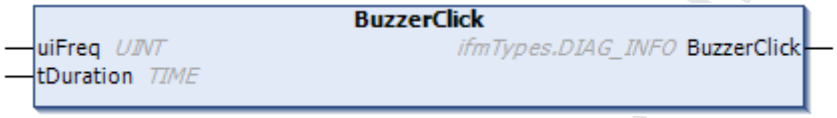
20932

Function block type: Function (FUN)

Behaviour model: --

Library: ifmPDM360NGutil.library

Symbol in CODESYS:



Description

18517

The function generates a sound via the device-internal buzzer.
Adjustable properties:

- Tone pitch
- Duration of the sound

Input parameter

19014

Parameter	Data type	Description	Possible values	
uiFreq	UINT	Frequency of the buzzer sound (in Hz)	500 ... 5000	500 Hz* ... 5000 Hz
tDuration	TIME	Duration of the buzzer sound (in ms)	1 ... 10	1 millisecond* ... 10 milliseconds

* preset value

Output parameter

7984

Parameter	Data type	Description	Possible values
BuzzerClick	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)

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_INVALID_FREQUENCY — Error: Invalid frequency value; Function call has been stopped.
- ERR_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- ERR_INVALID_VALUE — Error: Invalid time value; Function call has been stopped.
- ERR_INVALID_CHANNEL — Error: Internal parameter setting fault
  ► Contact the ifm Service Center!
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

10.2.8 Bibliothek ifmRawCAN.library

Contents

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.

CAN_Enable

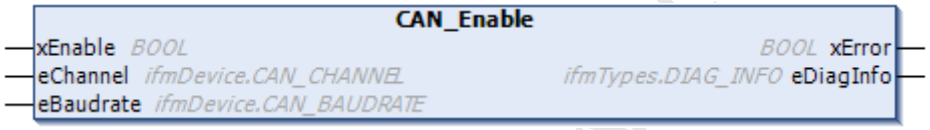
8709

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. 156)	
eBaudrate	CAN_ BAUD RATE	Baud rate of the CAN channel	→ **CAN_BAUDRATE (ENUM)** (→ p. 156)	

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!

CAN_Recover

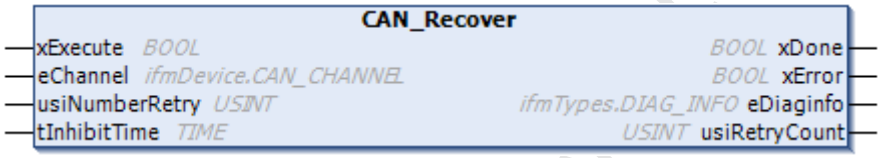
11765

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. 156)	
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!

CAN_RemoteRequest

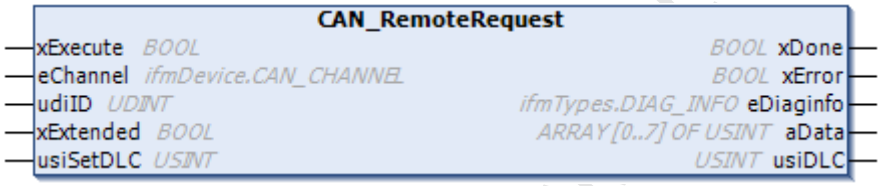
10884

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. 156)	
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.

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. 156)	
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!

CAN_Rx

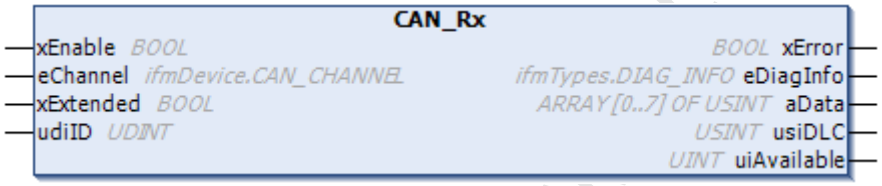
6939

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. 156)	
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!

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. 233)

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. 156)	
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!

CAN_RxRange

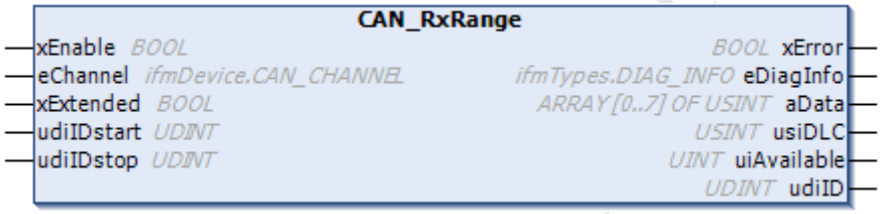
11731

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. 233)

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. 156)	
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!

CAN_Tx

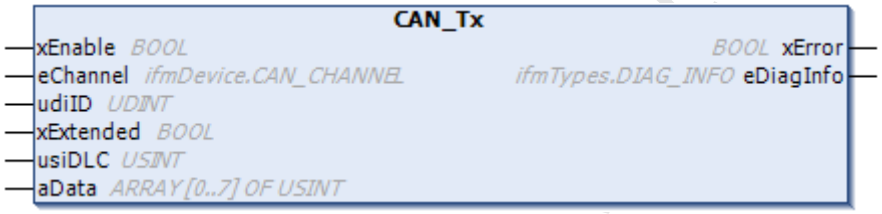
2269

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 parameter

SYS_OBJECTID>

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. 156)	
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!

CAN_Info (GVL)

12281

Name	Description	Data type	Possible values	
eBusState	Status of the CAN interface to CiA 11898	→ CAN_BUS_STATE (STRUCT) (→ p. 242)	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

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

10.2.9 Bibliothek ifmSysInfo.library

Contents

21218

The library contains function block (FB) and data structures (STRUCT, ENUM) for the provision of the device information.

GetInfo

20929

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmSysInfo.library

Symbol in CODESYS:

Description

21219

The FB reads the following information of the device:

- Firmware version of the device (FW device)
- Firmware version of key row (FW keyboard 1)
- Firmware version of key row (FW keyboard 2)
- Firmware version of key row (FW keyboard 3)
- Firmware version of the watchdog (FW watchdog)
- Firmware version of the I/O driver (FW IO driver)
- Hardware revision (HW revision)
- Serial number of the device (SerialNumber)
- Manufacturing date (Manufacturer Date)

The FB writes the read values in the global variable → **aSysInfoList (GVL)** (→ p. 246).

Input parameter

21203

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

Output parameter

21221

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:)	

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_NOT_SUPPORTED — Error: Invalid function calls; Function is not supported.
- ERR_INTERNAL — Error: Internal system error
  ► Contact the ifm Service Center!
- ERR_INVALID_VALUE — Error: At least one information type to be read is not supported by the device
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!

aSysInfoList (GVL)

21470

Name	Description	Data type	Value
FIRMWARE_DEVICE	Firmware version of the device	→ SYS_INFO (STRUCT) (→ p. 246)	0*
FIRMWARE_KEYBOARD_1	Firmware version of key row 1	→ SYS_INFO (STRUCT) (→ p. 246)	0*
FIRMWARE_KEYBOARD_2	Firmware version of key row 2	→ SYS_INFO (STRUCT) (→ p. 246)	0*
FIRMWARE_KEYBOARD_3	Firmware version of key row 3	→ SYS_INFO (STRUCT) (→ p. 246)	0*
FIRMWARE_WD	Firmware version of the watchdog	→ SYS_INFO (STRUCT) (→ p. 246)	0*
FIRMWARE_IO	Firmware version of the I/O driver	→ SYS_INFO (STRUCT) (→ p. 246)	0*
HW_REVISION_HW_REL	Hardware revision	→ SYS_INFO (STRUCT) (→ p. 246)	0*
DEVICE_SERIAL_NUM	Serial number of the device	→ SYS_INFO (STRUCT) (→ p. 246)	0*
MANUFACTURER_DATE	Manufacturing date	→ SYS_INFO (STRUCT) (→ p. 246)	0*

* ... Initialisation value

SYS_INFO (STRUCT)

21469

Designation	Data type	Description	Possible values
sName	STRING (32)	Name of the system component	E.g. firmware version
sValue	STRING (255)	Value of the system component	E.g. 3.1

10.2.10 Bibliothek ifmUSBstorageUtil.library

Contents

21402

The library contains function blocks (FB) for the control and management of USB memory devices.

USBstorageHandler

21403

Function block type: Function block (FB)

Behaviour model: ENABLE with Busy Extension

Library: ifmUSBstorageUtil.library

Symbol in CODESYS:

Description

21404

The FB manages the USB device connected to the device. The FB carries out the following functions:

- Integrate USB device automatically into the file system of the device (mount)
- Provide path to the USB device in the file system of the device
- Remove USB device from the file system of the device upon command of the user (unmount)
- Signal insertion and removal of the USB device

Input parameter

21405

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	FB is deactivated
			TRUE	FB is activated
xRemoveDevice	BOOL	Remove USB device from the file system (unmount)	FALSE ⇨ TRUE	USB is removed
			Other	No action

Output parameter

21406

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
usiNumberUSBdevice	USINT	Number of the connected USB memory device	Usually `0` or `1`	
sAccessPath	STRING	Absolute directory path of the USB storage device	E.g. `/mnt/usb`	
xDevicePlugged	BOOL	Signals the insertion of a USB device within the current PLC cycle	FALSE	No USB device inserted
			TRUE	USB device inserted
xDeviceUnplugged	BOOL	Signals the removal of a USB device within the current PLC cycle	FALSE	USB device not removed
			TRUE	USB device has been removed

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.
- STAT_PREPARING — State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- ERR_UNDEFINED — Error: Unknown error
  ► Contact the ifm Service Center!
- ERR_NOT_IMPLEMENTED — Error: Function not implemented in the runtime system

11 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

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.

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

Boot loader

On delivery **ecomat*mobile*** controllers only contain the boot loader.
The boot loader is a start program that allows to reload the runtime system and the application program on the device.
The boot loader contains basic routines...
• for communication between hardware modules,
• for reloading the operating system.
The boot loader is the first software module to be saved 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.

Category (Cat.)

Classification of the safety-related parts of a control system in respect of their resistance to →faults and their subsequent behaviour in the fault condition. This safety is achieved by the structural arrangement of the parts, fault detection and/or by their reliability (→ ISO 13849).

CCF

Common **C**ause **F**ailure
Failures of different items, resulting from a common event, where these failures are not consequences of each other.

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

CRC

CRC = **C**yclic **R**edundancy **C**heck.
CRC is a method of information technology to determine a test value for data, to detect faults during the transmission or duplication of data. Prior to the transmission of a block of data, a CRC value is calculated. After the end of the transaction the CRC value is calculated again at the target location. Then, these two test values are compared.

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

Data type	min. value	max. value	size in the memory
ULINT	0	18 446 744 073 709 551 615	64 Bit = 8 Bytes
STRING			number of char. + 1

DC

Direct **C**urrent

DC

Diagnostic **C**overage
Diagnostic coverage is the measure of the effectiveness of →diagnostics as the ratio between the failure rate of detected dangerous failures and the failure rate of total dangerous failures:

Formula: DC = failure rate detected dangerous failures / total dangerous failures

Designation	Range
none	DC < 60 %
low	60 % ≤ DC < 90 %
medium	90 % ≤ DC < 99 %
high	99 % ≤ DC

Table: Diagnostic coverage DC

An accuracy of 5 % is assumed for the limit values shown in the table.
Diagnostic coverage can be determined for the whole safety-related system or for only parts of the safety-related system.

Demand rate rd

The demand rate r_d is the frequency of demands to a safety reaction of an →SRP/CS per time unit.

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.

Diagnostic coverage

→ **DC** (→ p. 252)

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.

Diversity

In technology diversity is a strategy to increase failure safety.
The systems are designed →redundantly, however different implementations are used intentionally and not any individual systems of the same design. It is assumed that systems of the same performance, however of different implementation, are sensitive or insensitive to different interference and will therefore not fail simultaneously.
The actual implementation may vary according to the application and the requested safety:

- use of components of several manufacturers,
- use of different protocols to control devices,
- use of totally different technologies, for example an electrical and a pneumatic controller,
- use of different measuring methods (current, voltage),
- two channels with reverse value progression:
  channel A: 0...100 %
  channel B: 100...0 %

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

Failure

Failure is the termination of the ability of an item to perform a required function.
After a failure, the item has a →fault. Failure is an event, fault is a state.
The concept as defined does not apply to items consisting of software only.

Failure, dangerous

A dangerous failure has the potential to put the →SRP/SC in a hazardous or fail-to-function

state. Whether or not the potential is realized can depend on the channel architecture of the system; in redundant systems a dangerous hardware failure is less likely to lead to the overall dangerous or fail-to-function state.

Failure, systematic

A systematic failure is a failure related in a deterministic way (= not coincidental) to a certain cause. The systematic failure can only be eliminated by a modification of the design or of the manufacturing process, operational procedures, documentation or other relevant factors.

Corrective maintenance without modification of the system will usually not eliminate the failure cause.

Fault

A fault is the state of an item characterized by the inability to perform the requested function, excluding the inability during preventive maintenance or other planned actions, or due to lack of external resources.
A fault is often the result of a failure of the item itself, but may exist without prior failure.
In →ISO 13849-1 "fault" means "random fault".

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).

First fault occurrence time

Time until the first failure of a safety element. Within a specified period of time the runtime system checks the controller via internal monitoring and test routines.
This "test cycle time" must be smaller than the statistical first fault occurrence time for the application.

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.

FMEA

FMEA = **F**ailure **M**ode and **E**ffects **A**nalysis. Method of reliability engineering, to find potential weak points. Within the framework of quality or security management, the FMEA is used preventively to prevent faults and increase the technical reliability.

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.

Functional safety

Part of the overall safety referred to the →EUC and the EUC control system which depends on the correct functioning of the electric or electronic safety-related system, safety-related systems of other technologies and external devices for risk reduction.

H

Harm

Physical injury or damage to health.

Hazard

Hazard is the potential source of harm.
A distinction is made between the source of

the hazard, e.g.:
- mechanical hazard,
- electrical hazard,

or the nature of the potential harm, e.g.:
- electric shock hazard,
- cutting hazard,
- toxic hazard.

The hazard envisaged in this definition is either permanently present during the intended use of the machine, e.g.:
- motion of hazardous moving elements,
- electric arc during a welding phase,
- unhealthy posture,
- noise emission,
- high temperature,

or the hazard may appear unexpectedly, e.g.:
- explosion,
- crushing hazard as a consequence of an unintended/unexpected start-up,
- ejection as a consequence of a breakage,
- fall as a consequence of acceleration/deceleration.

Heartbeat

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

HFT

HFT = hardware fault tolerance is an indicator value in the standard IEC 61508 to describe systems with a safety-related function.

- An HFT = N indicates that N + 1 hardware fault, poor distribution, may result in the loss of the safety function.
- The higher the value for HFT, the better suitability of the device for high safety requirements.

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 61508

Standard: Functional safety of electrical/electronic/programmable electronic safety-related systems

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 13849

Standard: Safety of machinery, safety-related parts of control systems
- Part 1: General design principles

Part 2: Validation

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.

Life, mean

Mean **T**ime **T**o **d**angerous **F**ailure = the expectation of the mean time to dangerous failure.

Designation	Range
low	3 years ≤ $MTTF_d$ < 10 years
medium	10 years ≤ $MTTF_d$ < 30 years
high	30 years ≤ $MTTF_d$ ≤ 100 years

Table: Mean time of each channel to the dangerous failure $MTTF_d$

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.

Mission time TM

Mission time T_M is the period of time covering the intended use of an SRP/CS.

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. 255)

Monitoring

→Safety function which ensures that a →protective measure is initiated:
• if the ability of a component or an element to perform its function is diminished.
• if the process conditions are changed in such a way that the resulting →risk increases.

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

MTBF

Mean **T**ime **B**etween **F**ailures (MTBF)
Is the expected value of the operating time between two consecutive failures of items that are maintained.
For items that are NOT maintained the mean life →MTTF is the expected value (mean value) of the distribution of lives.

MTTF

Mean **T**ime **T**o **F**ailure (MTTF) or: mean life.

MTTFd

Mean **T**ime **T**o **d**angerous **F**ailure = the expectation of the mean time to dangerous failure.

Designation	Range
low	3 years ≤ $MTTF_d$ < 10 years
medium	10 years ≤ $MTTF_d$ < 30 years
high	30 years ≤ $MTTF_d$ ≤ 100 years

Table: Mean time of each channel to the dangerous failure $MTTF_d$

Muting

Muting is the temporary automatic suspension of a →safety function(s) by the →SRP/CS.
Example: The safety light curtain is bridged, if the closing tools have reached a finger-proof distance to each other. The operator can now approach the machine without any danger and guide the workpiece.

N

NMT

NMT = **N**etwork **M**anagemen**t** = (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.

Operational

Operating state of a CANopen participant. In this mode →SDOs, →NMT commands and →PDOs can be transferred.

OSSD

OSSD = **O**utput **S**ignal **S**witching **D**evice
= output signal of a safety switching device, e.g. SafetySwitch, AS-i safety monitor.

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.
The PDU is an item of the →CAN protocol →SAE J1939. PDU indicates a part of the destination or source address.

Performance Level

Performance **L**evel
According to →ISO 13849-1, a specification (PL a...e) of safety-related parts of control systems to perform a →safety function under foreseeable conditions.
→ chapter **Performance level PL**
→ chapter **Erforderlichen PL (=PLr) mittels Risikograf herleiten**

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 = PDU format (PF) + PDU source (PS)
The parameter group number is an item of the →CAN protocol →SAE J1939. PGN collects the address parts PF and PS.

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).

PL

Performance **L**evel
According to →ISO 13849-1, a specification (PL a...e) of safety-related parts of control systems to perform a →safety function under

foreseeable conditions.
→ chapter **Performance level PL**
→ chapter **Erforderlichen PL (=PLr) mittels Risikograf herleiten**

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.

PLr

Using the "required performance level" PL_r the risk reduction for each →safety function according to →ISO 13849 is achieved.
For each selected safety function to be carried out by a →SRP/CS, a PL_r shall be determined and documented. The determination of the PL_r is the result of the →risk assessment and refers to the amount of the risk reduction.

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.

Process safety time

The max. time it may take between the occurrence of a fault and the establishment of the safe state in the application without having to assume a danger for people.
The safety time of the application program and the possible delay and response times due to switching elements have to be considered.
The resulting total time must be smaller than the **process safety time** (→ p. 259) of the application.

Programming language, safety-related

Only the following programming languages shall be used for safety-related applications:

- Limited variability language (LVL).
  In →CODESYS these are LD (ladder diagram) and FBD (function block diagram).
- Full variability language (FVL) with using of coding rules.
  These include e.g. C, C++, Assembler. In CODESYS it is ST (structured text).

Protective measure

Measure intended to achieve →risk reduction, e.g.:
• fault-excluding design,
• safeguarding measures (guards),
• complementary protective measures (user information),
• personal protective equipment (helmet, protective goggles).

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.

Redundant

Redundancy is the presence of more than the necessary means so that a function unit performs a requested function or that data can represent information.

Several kinds of redundancy are distinguished:

- Functional redundancy aims at designing safety-related systems in multiple ways in parallel so that in the event of a →failure of one component the others ensure the task.
- In addition it is tried to separate redundant systems from each other with regard to space. Thus the →risk that they are affected by a common interference is minimised.
- Finally, components from different manufacturers are sometimes used to avoid that a systematic fault causes all redundant systems to fail (→diverse redundancy).

The software of redundant systems should differ in the following aspects:

- specification (different teams),
- specification language,
- programming (different teams),
- programming language,
- compiler.

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.

Reset, manual

The manual reset is an internal function within the →SRP/CS used to restore manually one or more →safety functions before re-starting a machine.

Residual risk

→Risk remaining after →protective measures have been taken. The residual risk has to be clearly warned against in operating instructions and on the machine.

Risk

Combination of the probability of occurrence of →harm and the severity of that harm.

Risk analysis

Combination of ...

- the specification of the limits of the machine (intended use, time limits),
- →hazard identification (intervention of people, operating status of the machine, foreseeable misuse) and
- the risk estimation (degree of injury, extent of damage, frequency and duration of the risk, probability of occurrence, possibility of avoiding the hazard or limiting the →harm).

Risk assessment

Overall process comprising →risk analysis and →risk evaluation.
According to Machinery Directive 2006/42/EU the following applies: "The manufacturer of machinery or his authorised representative must ensure that a risk assessment is carried out in order to determine the health and safety requirements which apply to the machinery. The machinery must then be designed and constructed taking into account the results of the risk assessment." (→ Appendix 1, General principles)

Risk evaluation

Judgement, on the basis of the →risk analysis, of whether risk reduction objectives have been achieved.

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.

Runtime system

Basic program in the device, establishes the connection between the hardware of the device and the application program.

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

Safety function

Function of the machine whose →failure can result in an immediate increase of the →risk(s). The designer of such a machine therefore has to:
• safely prevent a failure of the safety function,
• reliably detect a failure of the safety function in time,
• bring the machine / plant into a safe →state in time in the event of a failure of the safety function.

Safety-standard types

The safety standards in the field of machines are structured as below:
Type-A standards (basic safety standards)
Type-B standards (generic safety standards)
Type-C standards (machine safety standards)

Type-A standards (basic safety standards) give basic concepts, principles for design and general aspects that can be applied to machinery. Examples:
• Terminology, methodology (ISO 12100),
• Technical principles (ISO 12100),
• Risik assessment (ISO 12100), ...

Type-B standards (generic safety standards) deal with one safety aspect or one type of safeguard that can be used across a wide range of machinery. The implementation of the machine function must be effected during risk assessment, specification and development. The manufacturer of the machine is responsible for the correct assessment.

- Type-B1 standards on particular safety aspects. Examples:
  • Safety distances (ISO 13857),
  • Arm / hand speeds (ISO 13855),
  • Safet-related parts of control systems (→ISO 13849),
  • Temperatures, noise, ...
- Type-B2 standards on safeguards. Examples:
  • Emergency stop circuits ((ISO 13850),
  • Two-hand controls,
  • Interlocking devices or electro-sensitive protective equipment (IEC 61496), ...

Type-C standards (machine safety standards, product standards) deal with detailed safety requirements on a certain machine or a group of machines. Often the risk assessment for the whole machine or partial functions is already documented there. Therefore it should be followed for the development of a machine.

Examples:
- Refuse collection vehicles (EN 1501),
- Mobile elevating work platforms (EN 280), ...

SCT

In CANsafety / CANopenSafety the **S**afeguard **C**ycle **T**ime (SCT) monitors the correct function of the periodic transmission (data refresh) of the →SRDOs. The data must have been repeated within the set time to be valid. Otherwise the receiving controller signals a serious error and passes into the safe →state (→ chapter **Safety state**).

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.

SIL

According to →IEC 62061 the safety-integrity level SIL is a classification (SIL CL 1...4) of the safety integrity of the →safety functions. It is used for the evaluation of electrical / electronic / programmable electronic (E/E/EP) systems with regard to the reliability of safety functions. The safety-related design principles that have to be adhered to so that the →risk of a malfunction can be minimised result from the required level.

Slave

Passive participant on the bus, only replies on request of the →master. Slaves have a clearly defined and unique →address in the bus.

SRDO

In CANsafety / CANopenSafety safe data is exchanged via SRDOs (**S**afety-**R**elated **D**ata **O**bjects). An SRDO always consists of two →CAN messages with different →identifiers:
- message 1 contains the original user data,
- message 2 contains the same data which are inverted bit by bit.

SRP/CS

Safety-**R**elated **P**art of a **C**ontrol **S**ystem
Part of a control system that responds to safety input signals and generates safety output signals. The combined safety-related parts of a control system start at the point where the safety input signals are initiated (including, for example, the actuating cam and the roller of the position switch) and end at the output of the power control elements (including, for example, the main contacts of a contactor).

SRVT

The SRVT (**S**afety-**R**elated **O**bject **V**alidation **T**ime) ensures with CANsafety / CANopenSafety that the time between the SRDO-message pairs is adhered to.

Only if the redundant, inverted message has been transmitted after the original message within the SRVT set are the transmitted data valid. Otherwise the receiving controller signals a serious error and will pass into the safe →state (→ chapter **Safety state**).

State, safe

The state of a machine is said to be safe when there is no more →hazard formed by it. This is usually the case if all possible dangerous movements are switched off and cannot start again unexpectedly.

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.

Test rate rt

The test rate r_t is the frequency of the automatic tests to detect →faults in an →SRP/CS in time.

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).

Uptime, mean

Mean **T**ime **B**etween **F**ailures (MTBF)
Is the expected value of the operating time between two consecutive failures of items that are maintained.
For items that are NOT maintained the mean life →MTTF is the expected value (mean value) of the distribution of lives.

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.

wo

WO = write only
Unidirectional data transmission: Data can only be changed and not read.

12 Index

D

E

F

G

H

I

J

K

L

S

T

U

W

13 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