



# COMBIVERT Generation 6

PROGRAMMING MANUAL | Fieldbus systems – V 2.9







# 1 Preface

The described hardware and software are developments of the KEB Automation KG. The enclosed documents correspond to conditions valid at printing. Misprint, mistakes and technical changes reserved.

## 1.1 Signal words and symbols

Certain operations can cause hazards during the installation, operation or thereafter. There are safety informations in the documentation in front of these operations. Security signs are located on the device or on the machine. A warning contains signal words which are explained in the following table:

	➤ Dangerous situation, which will cause death or serious injury in case of non-observance of this safety instruction.
	➤ Dangerous situation, which may cause death or serious injury in case of non-observance of this safety instruction.
	➤ Dangerous situation, which may cause minor injury in case of non-observance of this safety instruction.
	➤ Situation, which can cause damage to property in case of non-observance.

### RESTRICTION

Is used when certain conditions must meet the validity of statements or the result is limited to a certain validity range.



- Is used when the result will be better, more economic or trouble-free by following these procedures.

## 1.2 More symbols

- ▶ This arrow starts an action step.
- / - Enumerations are marked with dots or indents.
- => Cross reference to another chapter or another page.

	Note to further documentation. <a href="http://www.keb.de">Document search on www.keb.de</a>	
---	---	---



### 1.3 Laws and guidelines

KEB Automation KG confirms with the CE mark and the EU declaration of conformity, that our device complies with the essential safety requirements.

The EU declaration of conformity can be downloaded on demand via our website. Further information is provided in the chapter "Certification".

### 1.4 Warranty

The warranty and liability on design, material or workmanship defects for the acquired device is given in the general conditions of sale.

	Here you will find our general sales conditions. <a href="#"><u>AGB</u></a>	
---	--	---

Further agreements or specifications require a written confirmation.

### 1.5 Support

Through multiple applications not every imaginable case has been taken into account. If you require further information or if problems occur which are not treated detailed in the documentation, you can request the necessary information via the local KEB Automation KG agency.

**The use of our units in the target products is beyond of our control and therefore exclusively the responsibility of the customer.**

The information contained in the technical documentation, as well as any user-specific advice in spoken and written and through tests, are made to best of our knowledge and information about the intended use. However, they are considered for information only without responsibility and changes are expressly reserved, in particular due to technical changes. This also applies to any violation of industrial property rights of a third-party. Selection of our units in view of their suitability for the intended use must be done generally by the user.

**Tests can only be carried out by the customer within the scope of the intended end use of the product (application). They must be repeated, even if only parts of hardware, software or the unit adjustment are modified.**

### 1.6 Copyright

The customer may program manual for use as well as further documents or parts from it for internal purposes. Copyrights are with KEB Automation KG and remain valid in its entirety.

This KEB product or parts thereof may contain foreign software, incl. free and/or open source software. If applicable, the license terms of this software are contained in the instructions for use. The instructions for use are already available, can be downloaded from the KEB website or can be requested from the respective KEB contact person.

Other wordmarks and/or logos are trademarks (™) or registered trademarks (®) of their respective owners.

## Content

<b>1</b>	<b>Preface .....</b>	<b>3</b>
1.1	Signal words and symbols.....	3
1.2	More symbols .....	3
1.3	Laws and guidelines .....	4
1.4	Warranty.....	4
1.5	Support .....	4
1.6	Copyright .....	4
<b>2</b>	<b>Basic Safety Instructions .....</b>	<b>9</b>
2.1	Target group .....	9
2.2	Validity of this manual .....	9
2.3	Electrical connection .....	10
2.4	Start-up and operation .....	10
<b>3</b>	<b>Product Description .....</b>	<b>11</b>
3.1	Functional overview .....	11
3.2	Used terms and abbreviations.....	11
3.3	Trademarks.....	13
<b>4</b>	<b>KEB operating tools .....</b>	<b>14</b>
4.1	COMBIVIS studio 6 .....	14
4.2	COMBIVIS 6 .....	14
4.3	KEB Ftp Application .....	14
<b>5</b>	<b>Instructions for the start-up of the fieldbus interfaces .....</b>	<b>15</b>
5.1	Introducing explanation about KEB process data communication.....	15
5.1.1	Definitions .....	15
5.1.2	Basic structure of process data mapping.....	15
5.2	Identify KEB communication interfaces .....	17
5.2.1	Diagnostic interface .....	17
5.2.2	Multi-Realtime-Ethernet module on the real-time Ethernet interface.....	18
5.2.3	Fieldbus-specific handling of return values for parameter requests .....	19
5.3	Adjust fieldbus parameter.....	22
5.3.1	Selection of the fieldbus system via fb68.....	22
5.3.2	Selection of the process data size via fb60 .....	23
5.3.3	Selection of the configuration of the active fieldbus system .....	24
5.3.4	Selection of the node address of the fieldbus system .....	25
5.3.5	Selection of the transmission speed (CANopen) .....	26
5.4	Restart the device .....	27
5.5	Check the fieldbus status .....	29
5.5.1	Checking MAC addresses and IP configuration .....	29
5.5.2	Checking the fieldbus STATUS LED (NET ST) .....	33
5.5.3	Checking the exception state.....	35
5.5.4	Check the fieldbus state and possible fieldbus error codes .....	36
5.6	Checking the process data mapping .....	41
5.6.1	Mapping of the process data.....	41
5.6.2	Additional parameters for process data mapping for CANopen .....	43
5.6.3	Additional parameters for process data mapping for POWERLINK .....	43
5.6.4	Fieldbus wizard .....	44

5.7	Synchronization to the fieldbus master .....	45
5.7.1	Checking the synchronization .....	45
5.7.2	Synchronous transfer of process data .....	47
5.8	Fieldbus watchdog .....	48
<b>6</b>	<b>CANopen Interface .....</b>	<b>49</b>
6.1	Basics of the CAN BUS .....	49
6.2	CAN Functions .....	49
6.3	Sync Identifier .....	50
6.4	Broadcast objects .....	50
6.5	Communication objects .....	50
6.6	Request/Response-Identifier (SDO) .....	51
6.7	Out/In-Identifier (PDO) .....	52
6.8	CANopen process data mapping .....	53
6.9	CANopen Bootup-Sequence .....	57
6.10	Node guarding .....	59
6.11	Life guarding .....	59
6.12	Emergency object .....	60
6.13	Heartbeat .....	63
6.14	CAN-TelegramTypes .....	64
6.14.1	SDO-Telegrams .....	64
6.14.2	SDO(rx)-Telegram .....	65
6.14.3	SDO(tx)-Telegram .....	65
6.14.4	RPDO1...4 telegram .....	67
6.14.5	TPDO1...4 telegram .....	68
6.15	CAN bit timing .....	69
6.16	CAN diagnosis .....	69
<b>7</b>	<b>EtherCAT interface .....</b>	<b>70</b>
7.1	Hardware and software version .....	71
7.2	Sync Manager .....	72
7.2.1	Sync Manager Parameters .....	73
7.3	EtherCAT diagnosis and timing (control type K and P) .....	75
7.3.1	Diagnostic cells of the EtherCAT core (hardware) .....	75
7.3.2	Time measurement EtherCAT Frame <=> Sync pulse .....	75
7.3.3	Application error counter .....	76
7.3.4	EtherCAT diagnosis assistant .....	76
7.4	EtherCAT diagnosis (control type A) .....	77
7.5	Ethernet over EtherCAT (EoE) .....	78
7.5.1	Start-up of Ethernet over EtherCAT .....	78
7.5.2	COMBIVIS functions via Ethernet over EtherCAT .....	79
7.6	EtherCAT Hot-Connect .....	80
<b>8</b>	<b>VARAN interface .....</b>	<b>81</b>
8.1	Electronic nameplate .....	81
8.2	Dual Port Memory .....	81
8.3	Available commands .....	81
8.4	DPM mapping .....	82
8.5	Parameterization data (asynchronous objects) .....	84
8.6	Process data (isochronous objects) .....	84

8.6.1	To activate the process data objects in the device, it is necessary to set the mapping of the process data by using the parameters defined in chapter 5.6.1 Mapping of the process data .	84
8.7	<b>VARAN diagnosis</b>	84
<b>9</b>	<b>PROFINET interface</b>	<b>85</b>
9.1	<b>PROFINET certificate</b>	85
9.2	<b>PROFINET device description (GSDML)</b>	85
9.3	<b>PROFINET functions</b>	85
9.3.1	Cyclic data exchange (process data communication)	85
9.3.2	Acyclic data exchange (parameter - channel) to PROFIdrive	85
9.3.3	Additional diagnostic channel via standard Ethernet communication	88
9.4	<b>PROFINET diagnosis</b>	89
9.4.1	PROFINET MAC addresses	89
9.4.2	PROFINET device name	90
<b>10</b>	<b>POWERLINK interface</b>	<b>91</b>
10.1	<b>Basics of the POWERLINK BUS</b>	91
10.2	<b>POWERLINK functions</b>	92
10.2.1	Variable process data offset	92
10.2.2	Mapping version	92
10.3	<b>POWERLINK diagnosis</b>	93
10.3.1	POWERLINK MAC address	93
10.3.2	POWERLINK IP configuration	94
10.4	<b>Configuration of the KEB POWERLINK CN</b>	95
<b>11</b>	<b>EtherNet/IP™ interface</b>	<b>96</b>
11.1	<b>Data transmission</b>	96
11.1.1	Explicit messaging (parameterization channel)	96
11.1.2	Implicit Messaging (process data communication)	97
11.2	<b>EtherNet/IP™ objects</b>	101
11.2.1	Identity Object (class code: 0x01)	101
11.2.2	Assembly object (class code: 0x04)	101
11.2.3	TCP/IP Interface Object (class code: 0xF5)	102
11.2.4	Ethernet link object (class code: 0xF6)	102
11.2.5	KEB inverter object (class code: 0x64)	104
11.3	<b>Status codes</b>	104
<b>12</b>	<b>Modbus/TCP interface</b>	<b>107</b>
12.1	<b>Address range</b>	108
12.2	<b>Supported functions</b>	109
12.2.1	Function 3: Read Multiple Registers	109
12.2.2	Function 16: Write Multiple Registers	110
12.3	<b>Error handling (exception handling)</b>	111
12.3.1	Error messages	111
12.3.2	Error codes	112
12.4	<b>Modbus/TCP specific parameters</b>	112
12.4.1	Configure IP addressing of the device (fb114)	112
12.4.2	Addressing the subindex via the parameter channel (fb115)	113

<b>13 Annex .....</b>	<b>114</b>
13.1 Mapping parameters and Sync manager parameters .....	114
13.2 Fieldbus parameters .....	117
13.3 Solutions for KEB specific fieldbus error codes .....	120
13.4 Revision history .....	121

## Figures

Figure 1: Timing of the LED flashing pattern .....	35
Figure 2: Fieldbus watchdog .....	48
Figure 3: COMBIVIS 6 Property-Editor .....	53
Figure 4: CANopen Bootstrap-Sequence.....	57
Figure 5: Sync Manager Parameter Calc and Copy Time .....	74
Figure 6: EtherCAT diagnosis assistant.....	76
Figure 7: Client / Server Model .....	107
Figure 8: Address range.....	108
Figure 9: Exception handling.....	111

## Tables

Table 3-1: Used terms and abbreviations .....	12
Table 5-1: Conversion of the KEB return value 0 .....	19
Table 5-2: Conversion of the KEB return value 1 .....	19
Table 5-3: Conversion of the KEB return value 2 .....	20
Table 5-4: Conversion of the KEB return value 3 .....	20
Table 5-5: Conversion of the KEB return value 4 .....	20
Table 5-6: Conversion of the KEB return value 5 .....	20
Table 5-7: Conversion of the KEB return value 6 .....	20
Table 5-8: Conversion of the KEB return value 7 .....	20
Table 5-9: Conversion of the KEB return value 8 .....	20
Table 5-10: Conversion of the KEB return value 9 .....	21
Table 5-11: Conversion of the KEB return value 10 .....	21
Table 5-12: Conversion of the KEB return value 11 .....	21
Table 5-13: Conversion of the KEB return value 12 .....	21
Table 5-14: Conversion of the KEB return value 13 .....	21
Table 5-15: Conversion of the KEB return value 14 .....	21
Table 5-16: non-resettable fb parameters.....	28
Table 5-17: Differences in the support of MAC / IP parameters .....	29
Table 7-1: Differences between the mailbox sizes of the control types .....	70



## 2 Basic Safety Instructions

The COMBIVERT is constructed and built according to the state of the art and the recognized safety regulations. Nevertheless, their use may create dangers to life and limb of the user or third parties or damage to the machine and other material property.

The following safety instructions have been created by the manufacturer for the area of electric drive technology. They can be supplemented by local, country- or application-specific safety instructions. This list is not exhaustive. Non-observance of the safety instructions by the customer, user or other third party leads to the loss of all claims against the manufacturer.

### NOTICE

#### Hazards and risks through ignorance!

- Read all parts of the instructions for use!
- Observe the safety and warning instructions!
- If anything is unclear, please contact KEB!

### 2.1 Target group

This manual is intended exclusively for electrical personnel. Electrical personnel for the purpose of this manual must have the following qualifications:

- Knowledge and understanding of the safety instructions.
- Skills for installation and assembly.
- Start-up and operation of the product.
- Understanding about the function in the used machine.
- Recognition of dangers and risks of electrical drive technology.
- Knowledge about DIN IEC 60364-5-54.
- Knowledge about national accident prevention regulations (e.g. [DGUV regulation 3](#)).
- When using the operating software COMBIVIS, basic knowledge of the Windows operating system is required.

### 2.2 Validity of this manual

This programming manual is part of the instructions for use.

The programming manual

- describes the parameterization
  - of the Multi-Real-Time Ethernet module at F6A and S6A
  - of the fieldbus interface at F6K/P, S6K/P
- contains only supplementary safety instructions.
- is only valid in connection with the installation manual and programming manual of the respective device.

### 2.3 Electrical connection

#### **DANGER**

#### **Electrical voltage at terminals and in the device !**

##### **Danger to life due to electric shock !**

- For any work on the unit switch off the supply voltage and secure it against switching on.
- Connect or disconnect all pluggable terminals in a deenergised state only.
- Wait until all drives have stopped in order that no regenerative energy can be generated.
- Await capacitor discharge time (5 minutes) if necessary, measure DC voltage at the terminals.
- Never bridge upstream protective devices (also not for test purposes).
- For operation, install all necessary covers and protective devices.

For a trouble-free and safe operation, please pay attention to the following instructions:

- The electrical installation shall be carried out in accordance with the relevant requirements.
- Cable cross-sections and fuses must be dimensioned according to the design of the machine manufacturer. Specified minimum / maximum values may not be fallen below /exceeded.
- With existing or newly wired circuits the person installing the units or machines must ensure the EN requirements are met.
- For drive controllers that are not isolated from the supply circuit (in accordance with [EN 61800-5-1](#)) all control lines must be included in other protective measures (e.g. double insulation or shielded, earthed and insulated).
- When using components without isolated inputs/outputs, it is necessary that equipotential bonding exists between the components to be connected (e.g. by the equipotential line). Disregard can cause destruction of the components by equalizing currents.

### 2.4 Start-up and operation

The drive controller must not be started until it is determined that the installation complies with the machine directive; Account is to be taken of [EN 60204-1](#).

#### **WARNING**

#### **Software protection and programming!**

##### **Hazards caused by unintentional behavior of the drive!**


- Check especially during initial start-up or replacement of the drive converter if the parameterization is compatible to the application.
- Securing a unit solely with software-supported functions is not sufficient. It is imperative to install external protective measures (e.g. limit switch) that are independent of the drive converter.
- Secure motors against automatic restart.

### 3 Product Description

#### 3.1 Functional overview

- Operation of the fieldbus interface
- Details of the EtherCAT interface
  - EtherCAT with Multi-Real-Time Ethernet module
  - EtherCAT without Multi-Real-Time Ethernet module
  - Ethernet over EtherCAT (EoE)
- Details of the CANopen interface
- Details of the VARAN interface
- Details of the PROFINET interface
- Details of the POWERLINK interface
- Details of the EtherNet/IP™ interface

#### 3.2 Used terms and abbreviations

Term	Description
Adapter	EtherNet/IP™ slave during process data communication
AS	Automation Studio (B&R development environment for POWERLINK)
Assemblies	Assemblies of supported communication channels for EtherNet/IP™
CAN / CANopen	Fieldbus system
CiA	CAN in Automation association
CiA DS301	CAL based communication profile
CiA DS402	CAN unit profile for drives
CIP	Common Industrial Protocol
Client	Modbus/TCP name for the (fieldbus) master
CN	Controlled Node (POWERLINK Slave)
COMBIVERT	KEB drive converter
COMBIVIS	KEB start-up and parameterization software
Controller	PROFINET master
Device	PROFINET slave
DIN	German Institute for Standardization
Recipient (consumer)	EtherNet/IP™ master during SDO communication
EMC	Electromagnetic compatibility
EN	European standard
EoE	Ethernet over EtherCAT
ESI file	EtherCAT Slave Information file
EtherCAT	Real-time Ethernet bus system; EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany. It is marked by the following logo: 
Ethernet	Real-time bus system - defines protocols, connectors, cable types
EtherNet/IP™	Real-time Ethernet bus system of the company Rockwell Automation

Term	Description
Explicit Messaging	Exchange of data via request/response-oriented functions
Fieldbus	(Real-time capable) communication protocol at field level
FSoE	Functional safety via EtherCAT
GSDML	PROFINET device description file
HSP5	Fast, serial protocol
IEC	International standard
IEEE	Institute of Electrical and Electronics Engineers
Implicit Messaging	Exchange of data via an I/O connection
(Fieldbus) master	Fieldbus node responsible for controlling the processes
MN	Managing Node (POWERLINK Master)
Modbus/TCP	Modbus protocol in which the data is transmitted via TCP/IP
Modulation	Means in drive technology that the power semiconductors are controlled
MRTE	Multi-Realtime-Ethernet
MTTF	Mean service life to failure
ODVA	Open DeviceNet Vendors Association
PD-In	Process input data (from perspective of the control)
PD-Out	Process output data (from perspective of the control)
PDO	Process data object
POWERLINK	Real-time Ethernet bus system of the company B&R
PROFINET	Industrial Ethernet standard of the PROFIBUS user organization e.V.
Process data mapping	Process data mapping (mapping of the configured process data)
Request	Request (in connection with SDO communication)
Response	Response (in connection with SDO communication)
Scanner	EtherNet/IP™ master during process data communication
SDO	Service Data Object
Transmitter (producer)	EtherNet/IP™ Slave during SDO communication
Server	Modbus/TCP name for the (fieldbus) slave
(Fieldbus) slave	Fieldbus device that processes the individual subtasks
SYNC / SYNC-Interrupt	Signal in the fieldbus communication for synchronization of the participants
VARAN	Real-time Ethernet bus system of the company SIGMATEK
XDD	POWERLINK device description file

Table 3-1: Used terms and abbreviations

### 3.3 Trademarks

**CANopen®** is registered trademark of CAN in AUTOMATION - International Users and Manufacturers Group e.V.

**EtherCAT®** is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

**Safety over EtherCAT®** is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

**EtherNET/IP** is a trademark of ODVA, Inc.

**ETHERNET POWERLINK** is a registered trademark of Bernecker + Rainer Industrie Elektronik Ges.m.b.H.

**PROFINET®** is a registered trademark of PROFIBUS user organisation e.V.

**VARAN-Bus** is a registered trademark of Sigmatek GmbH &Co. KG.

**Modbus/TCP** is an open de facto standard for industrial communication.

## **4 KEB operating tools**

### **4.1 COMBIVIS studio 6**

COMBIVIS studio 6, the KEB software tool based on CODESYS IEC 61131.

Assistant-guided component selection, fieldbus configuration, drive parameterization, and IEC 61131-3 project generation.

### **4.2 COMBIVIS 6**

License-free operating and analysis program for KEB COMBIVERT.

### **4.3 KEB Ftp Application**

This tool runs on Windows XP or higher with the .NET Framework. It uses a COM port or the Windows IP stack to transfer files to and from KEB devices.

## 5 Instructions for the start-up of the fieldbus interfaces

The following chapter contains a short, general manual for the first start-up of the KEB fieldbus interfaces as well as a description and explanation of the KEB process data communication. More detailed information about the individual fieldbus systems can be found in the following chapters.



- 
- The topic of synchronisation is not described in this document.  
Information about the synchronization with the fieldbus master can be found in the [Programming Manual | Control Application / Compact / Pro](#).
- 

### 5.1 Introducing explanation about KEB process data communication

#### 5.1.1 Definitions

To describe the KEB process data communication, the following chapters use a combination of terms from the CANopen DS301 nomenclature and the KEB nomenclature.

The KEB pr parameters (communication profile objects) are named according to the DS301 nomenclature. The process data communication of all fieldbus systems is mapped via these parameters. The following terms are used:

- Process data object (PDO), single object which can be read or written via the process data communication.
- Receive PDO (RPDO), PDO received from the inverter
- Transmit PDO (TPDO), PDO transmitted by the inverter
- Mapping, the mapping of the PDOs to the process data communication
- Communication parameters, CANopen specific parameters for setting the process data communication (=> chapter 6.8 CANopen process data mapping)

In addition, KEB-specific nomenclature is also used in this document for the description of process data communication. This includes the following terms:

- Process input data (PD-In), PDO received from the control (TPDO)
- Process output data (PD-Out), PDO sent by the control (RPDO)

#### 5.1.2 Basic structure of process data mapping

KEB devices support up to four process data mappings for process output data and process input data. The mapping of the PDOs is based on the CANopen DS301 standard.

The available number of process data mappings, the length of these mappings and the maximum number of PDOs per mapping is depending on the used fieldbus system.

Further information can be found in the chapter 5.6 Checking the process data mapping and in the fieldbus-specific chapters of this manual.

##### 5.1.2.1 CANopen

CANopen uses four process data mappings. Each mapping supports up to 8 process data objects. The maximum length of a CANopen process data mapping is 8 bytes.

##### 5.1.2.2 EtherCAT

By default, EtherCAT supports up to 8 EtherCAT and 8 FSoE PDOs in one mapping. A maximum of 32 bytes are available for EtherCAT. For FSoE, the number of bytes is specified by the mapping to be selected. The PDOs are divided KEB internally to the first and second process data mapping. The EtherCAT objects are always mapped to the first process data mapping. The FSoE objects are always mapped to the second process data mapping.

The maximum EtherCAT process data mapping is variable. Both the length and the maximum number of PDOs can be changed via parameter [fb60: process data size selection](#). Depending on the set size of the EtherCAT process data mapping the minimum possible EtherCAT cycle time changes.

The maximum size of the FSoE process data mapping cannot be changed.

### 5.1.2.3 VARAN, PROFINET, POWERLINK, EtherNet/IP and Modbus/TCP

VARAN, PROFINET, POWERLINK, EtherNet/IP and Modbus/TCP use the first process data mapping. Up to 8 PDOs are supported. The maximum length of a process data mapping is 32 bytes.



## 5.2 Identify KEB communication interfaces

KEB devices offer the following communication interfaces:

- One serial diagnostic interface for the connection with diagnostic tools (e.g. COMBIVIS) or an operator.
- One CANopen fieldbus interface
- One real-time Ethernet interface

Depending on the control board version, EtherCAT (control type P), EtherCAT or VARAN (control type K) or EtherCAT, PROFINET, POWERLINK and EtherNet/IP (control type A) are available on the real-time Ethernet interface (control type A).

The diagnostic interface is always available parallel to the fieldbus interface. It can be changed between CAN interface and real-time Ethernet interface.

### NOTICE

- The different versions of the real-time Ethernet interface at control type K are based on different hardware. The selection of the real-time Ethernet protocol must therefore already be observed when ordering.
- Parallel operation of CAN and real-time Ethernet interface is not possible.
- Parallel access (reading or writing) to the object directory by the diagnostic interface and the acyclic fieldbus channel may result in errors in the acyclic fieldbus channel.
- In the device variant "F6 PRO" (device part numbers xxF6Pxx-xxxx), the CAN interface is not available for all hardware revisions. Notes on the support of the interface can be found in the associated installation manual in chapter "Fieldbus interfaces".

### 5.2.1 Diagnostic interface

#### 5.2.1.1 Diagnostic interface on devices of control type A and K

In order to use the diagnostic interface with KEB diagnostic tools (e.g. COMBIVIS) it is necessary to configure them before. The diagnostic interface can be configured via the following objects:

Index	Id-Text	Name	Function
0x2B0D	<a href="#">fb13</a>	<a href="#">DIN66019 node id</a>	Node address of the unit
0x2B0E	<a href="#">fb14</a>	<a href="#">DIN66019 baudrate</a>	Baud rate



- The diagnostic interface is not made for permanent operation of the device.
- Node address 1 and baud rate 38400 bit/s are set as default.
- For more information about the diagnostic interface, please refer to the [Programming manual | Control Application / Compact / Pro](#).

### 5.2.1.2 Diagnostic interface on devices of control type P

In order to use the diagnostic interface with KEB diagnostic tools (e.g. COMBIVIS) it is necessary to configure them before. The diagnostic interface can be configured via the following objects:

Index	Subindex	Id-Text	Name	Function
0x2B0D	0	<a href="#">fb13</a>	<a href="#">DIN66019 drive node ID</a>	Node address of the inverter
0x2B0E	0	<a href="#">fb14</a>	<a href="#">DIN66019 drive baudrate</a>	Baud rate of the inverter
0x2B0F	1	<a href="#">fb15.1</a>	<a href="#">application node ID</a>	Node address of the internal application
0x2B0F	2	<a href="#">fb15.2</a>	<a href="#">debugger node ID</a>	Node address of the internal debugger
0x2B10	0	<a href="#">fb16</a>	<a href="#">fieldbus node injection</a>	Node address where fieldbus telegrams are sent

Several object directories are available on the devices of control type P, which can be read out via the diagnostic interface.

The object directory of the inverter is addressed via the node address and baud rate set in parameters [fb13](#) and [fb14](#). (as well as on devices of control type A and K)

Communication to other object directories located on the device can be set via the sub-parameters under [fb15](#). Additional object directories are application and customer specific. For further information, please contact your sales partner.



- The diagnostic interface is not made for permanent operation of the device.
- Node address 1 and baud rate 38400 bit/s are set as default for communication with the inverter.
- For more information about the diagnostic interface, please refer to the [Programming manual | Control Application / Compact / Pro](#).

### 5.2.2 Multi-Realtime-Ethernet module on the real-time Ethernet interface

The fieldbus systems of the real-time Ethernet interface are implemented at the devices of control type A by using a Multi-Realtime Ethernet module (MRTE Module). The functions of the control board are independent of this MRTE module.

Parameter [fb80](#) is used to display information about the Multi-Realtime Ethernet module in the COMBIVIS interface.

Parameter [fb80](#) is an invisible parameter and is not displayed in the COMBIVIS interface with the default settings. To display the invisible parameters in COMBIVIS the KEB parameterization must be changed under Tools -> Options.

Index	Id-Text	Name	SubIdx	Name-SubIdx	Function
0x2B50	fb80	MRTE module	0		Number of sub-indices
			1	MRTE module support	Support of the MRTE module (MRTE module required / MRTE module not required)
			2	MRTE module state	Statue of the MRTE module (ready / not found)
			3	MRTE module version	Version of the MRTE module (NetX50 / NetX51 / not found)

Cards of control type A can also be operated without the MRTE module if no Ethernet based fieldbus system is used.

In order to avoid an error message on a device without MRTE module it is necessary to set parameter fb80 Sub-Index 1 MRTE module support from "MRTE module required" to "MRTE module not required".

### 5.2.3 Fieldbus-specific handling of return values for parameter requests

Feedback to parameter requests contain a return value.

For requests via the diagnostic interface, this is a KEB specific value. The possible values are described in the [Programming Manual | Control Application / Compact / Pro](#).

For SDO requests via the fieldbus interface the KEB specific return value is converted to a fieldbus specific value.

An exception is the fieldbus system VARAN, via which the return values are not transmitted and the fieldbus system Modbus/TCP, whose return values are described in chapter 12.3. Error handling (exception handling).



- There is no 1 to 1 conversion between KEB return values and fieldbus specific return values. Therefore, for some fieldbus return values it is not possible to infer to the KEB return value.
- e.g.: The EtherCAT return value 08000021h is assigned to the KEB return values 4, 6, 8 and 14.

The following tables describe this conversion:

#### KEB return value 0: OK

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
00000000h No error	00000000h No error	FFFFh No error	00000000h No error	0000h Success

Table 5-1: Conversion of the KEB return value 0

#### KEB return value 1: Device not ready

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
05040000h SDO protocol timeout	08000020h Device not ready	11h Device not ready	05040000h SDO protocol timeout	0002h Resources not available

Table 5-2: Conversion of the KEB return value 1

**KEB return value 2: Address or password invalid**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
06020000h Object does not exist	06010010h Password invalid	00h Invalid parameter number	06010000h Object access not supported	000Ch Object status conflict

Table 5-3: Conversion of the KEB return value 2

**KEB return value 3: Data invalid**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
06090030h Value range exceeded	06090030h Data invalid	17h Data invalid	06090030h Value range exceeded	0020h Invalid parameter

Table 5-4: Conversion of the KEB return value 3

**KEB return value 4: Parameter write protected**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
08000021h No transfer due to local settings	06010002h Parameter write protected	01h Parameter write protected	06010002h Parameter write protected	000Eh Non-modifiable attribute

Table 5-5: Conversion of the KEB return value 4

**KEB return value 5: BCC error**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
06040047h Internal incompatibility	06010000h Operation not possible	65h Manufacturer-specific 0	06060000h No access due to hardware error	001Fh Manufacturer-specific

Table 5-6: Conversion of the KEB return value 5

**KEB return value 6: Unit busy**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
08000021h No transfer due to local settings	08000022h Unit busy	14h Value inadmissible	08000022h Unit busy	0002h Resources not available

Table 5-7: Conversion of the KEB return value 6

**KEB return value 7: Service not available**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
06040047h Internal incompatibility	05040001h Service not available	66h Manufacturer-specific 1	06040047h Internal incompatibility	001Fh Manufacturer-specific

Table 5-8: Conversion of the KEB return value 7

**KEB return value 8: Password invalid**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
08000021h No transfer due to local settings	06010010h Password invalid	67h Manufacturer-specific 2	08000021h No transfer (local settings)	000Ch Object status conflict

Table 5-9: Conversion of the KEB return value 8

**KEB return value 9: Telegram frame error**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
06060000h	08000020h No data transfer	68h	06060000h	001Fh

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
No access due to hardware error		Manufacturer-specific 3	No access due to hardware error	Manufacturer-specific

Table 5-10: Conversion of the KEB return value 9

**KEB return value 10: Transmission error**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
06060000h No access due to hardware error	06010000h Operation not possible	69h Manufacturer-specific 4	06060000h No access due to hardware error	001Fh Manufacturer-specific

Table 5-11: Conversion of the KEB return value 10

**KEB return value 11: Subindex invalid**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
06090011h Subindex invalid	06090011h Subindex invalid	03h Subindex invalid	06090011h Subindex invalid	0009h Invalid attribute data

Table 5-12: Conversion of the KEB return value 11

**KEB return value 12: Language invalid**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
06040047h Internal incompatibility	06090012h Language invalid	6Ah Manufacturer-specific 5	06040047h Internal incompatibility	001Fh Manufacturer-specific

Table 5-13: Conversion of the KEB return value 12

**KEB return value 13: Address invalid**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
06020000h Object does not exist	06020000h Invalid address	00h Invalid parameter number	06020000h Object does not exist	0016h Object does not exist

Table 5-14: Conversion of the KEB return value 13

**KEB return value 14: Operation not possible**

EtherCAT	CANopen	PROFINET	POWERLINK	EtherNet/IP
08000021h No transfer due to local settings	06010000h Operation not possible	6Bh Manufacturer-specific 6	06040043h Parameter incompatibility	001Fh Manufacturer-specific

Table 5-15: Conversion of the KEB return value 14

## 5.3 Adjust fieldbus parameter

### 5.3.1 Selection of the fieldbus system via fb68

To set the fieldbus it is necessary to establish a connection between the device and the KEB operating tool COMBIVIS.

For this purpose, the diagnostic interface EoE (=> Chapter Ethernet over EtherCAT (EoE)) or the standard Ethernet channel can be used via the PROFINET ports on the devices of control type A (=> Chapter 9.3.3 Additional diagnostic channel via standard Ethernet communication).



- EtherCAT is set as default value for a newly delivered device. When using EtherCAT, it is not necessary to readjust the fieldbus type.

The selection of the fieldbus interface is controlled via the following object:

Index	Id-Text	Name	Function
0x2B44	fb68	fieldbus selection	Selection of the used fieldbus system

fb68	fieldbus selection		0x2B44
Value	Name	Notice	
0	EtherCAT	Fieldbus communication occurs via EtherCAT (Default)	
1	CANopen	Fieldbus communication occurs via CANopen	
2	VARAN	Fieldbus communication occurs via VARAN Requires VARAN hardware version of control type K, not possible with control type A or control type P	
3	PROFINET	Fieldbus communication via PROFINET Requires hardware version of control type A, not possible with control type K or control type P	
4	POWERLINK	Fieldbus communication via POWERLINK Requires hardware version of control type A, not possible with control type K or control type P	
5	Reserved	-	
6	Ether-Net/IPTM	Fieldbus communication occurs via EtherNet/IPTM Requires hardware version of control type A, not possible with control type K or control type P	
7	Modbus/TCP	Fieldbus communication occurs via Modbus/TCP Requires special version of control type A, not possible with standard version of control type A, control type K or control type P If you are interested, please contact your sales partner	

A newly selected fieldbus system only becomes active after a restart.

Control boards of control type A require time to switch between two Ethernet based fieldbus systems after restarting the device. (=> Chapter 5.4 Restart the device.

The error code "ERROR fieldbus type changed" in ru01 indicates that parameter fb68 has changed since the last restart.

Changing the active fieldbus system resets the process data image. (Chapter 5.6 Checking the process data mapping)

### 5.3.2 Selection of the process data size via fb60

After setting the required fieldbus system via [fb68 fieldbus selection](#) the required maximum process data size can be set via parameter [fb60 process data size selection](#).

Parameter [fb60 process data size selection](#) influences the length and the maximum number of process data objects of the first process data mapping. Additionally, [fb60](#) influences the minimum adjustable fieldbus cycle time.

The possible process data variables differ depending on the selected control type (A, K or P). Currently, different process data sizes are only supported for the EtherCAT fieldbus system.

If the active fieldbus system is changed via [fb68 fieldbus selection](#), [fb60 process data size selection](#) is reset to the default value.

For an optimal time behaviour of the software it is recommended to choose the process data size selected by [fb60](#) as small as possible.

Index	Id-Text	Name	Function
0x2B3C	<a href="#">fb60</a>	<a href="#">process data size selection</a>	Selection of the size of the first PD mapping

<a href="#">fb60</a>	<a href="#">process data size selection</a>			0x2B3C
Bit	Function	Value	Plaintext	Notice
0...7	mode	0	8 objects, 32 byte, min 400µs cycle time	Default, active for all fieldbuses
		1	8 objects, 32 byte, min 250µs cycle time	Control type P, EtherCAT
		2	-	Reserved for 125µs
		3	16 objects, 32 byte, min 500µs cycle time	Only EtherCAT
		4	32 objects, 64 byte, min 1000µs cycle time	Control type A, P, EtherCAT
		4	24 objects, 64 byte, min 1000µs cycle time	Control type K, EtherCAT

If a process data size is selected via [fb60 process data size selection](#), this influences the configuration of the active fieldbus system, which is visible via [fb67](#). Information on [fb67](#) can be found in the following chapter.

A change of the process data size becomes active only after a restart of the software. The error code "ERROR fieldbus type changed" indicates in [ru01](#) whether the parameter [fb60](#) has changed since the last restart.

[fb60](#) can also be changed if an active mapping is already set. However, the size of the mapping (number of PDOs and length) must be permissible for the newly selected process data size. Changes in [fb60](#) only become active after a restart.

By writing on [fb60](#), the minimum SYNC interval set via [is22 Basic Tp](#) can be influenced. The value of [is22](#) is not changed by writing on [fb60](#).

SYNC intervals adjustable on KEB devices correspond to the minimum SYNC interval adjustable via [is22](#) or a full multiple thereof. A minimum value of the SYNC interval is preset by [fb60](#), which cannot be fallen below independently of [is22](#). However, the cycle time that can actually be set still depends on [is22](#).

More information about [is22](#) and the minimum SYNC interval can be found in the [Programming Manual | Control Application / Compact / Pro](#) and in chapter 5.7 Synchronization to the fieldbus master in this manual.

### 5.3.3 Selection of the configuration of the active fieldbus system

After setting the required process data size via [fb60 process data size selection](#), the configuration of the selected fieldbus system can be checked via parameter [fb67 fieldbus configuration](#).

Bits [1](#) and [3](#) of [fb67](#) can be changed by the user. Other bits are used to display the current configuration and cannot be influenced.

Index	Id-Text	Name	Function
0x2B43	<a href="#">fb67</a>	<a href="#">fieldbus configuration</a>	Display of the fieldbus configuration

<a href="#">fb67</a>	<a href="#">fieldbus configuration</a>			0x2B43
Bit	Function	Value	Plaintext	Notice
0	Synchronous mode	0	No SyncMode support	No synchronous mode possible
		1	SyncMode support	Synchronous mode possible
1	Enable exemplary PD mapping at startup	0	No Exemplary PDMapp	No change to the mapping
		2	Exemplary PdMapp	A sample mapping is loaded during start-up
2	Enable application specific alarm messages	0	No ApplAlarm	Standard alarm messages
		4	ApplAlarm	Not yet supported
3	Disable non-volatile mapping parameter	0	Non volatile Mapp	Mapping is retained on reset
		8	Volatile Mapp	Mapping is deleted by reset
4	PDO / SDO multiplexing	0	No PDO / SDO multiplexing	PDOs, SDOs in the same Mid IRQ
		16	PDO / SDO multiplexing	One Mid IRQ PDOs, in the next SDOs
5	PDO multiplexing	0	No PDO multiplexing	All PDOs in the same Mid IRQ
		32	Multiplexing per 16 PDOs	16 PDOs per Mid IRQ

[Bit 0](#) indicates whether a synchronous mode is supported for the selected fieldbus system settings.

[Bits 1](#) and [3](#) influence the process data mapping after a restart. If [bit 1](#) is set the KEB default process data mapping is loaded after each restart. ([aa16](#) to [aa23](#) or [aa16](#) and [aa17](#) for CANopen). If [bit 3](#) is set, the set mapping is deleted after a restart. The effect of [bit 1](#) overwrites the effect of [bit 3](#). If both bits are set, the device starts with the KEB default process data mapping.



### 5.3.3.1 Further information on the KEB default process data mapping can be found in chapter 5.6 Checking the process data mapping

Bit 2 is not yet supported.

Bit 4 indicates whether process data and acyclic data are processed in the same interrupt or in several interrupts in succession. The separate processing of process data and acyclic data avoids internal runtime problems.

Bit 5 indicates whether all process data are processed in the same interrupt or several interrupts in succession. If this bit is active, the time required by the KEB device to process the process data completely increases significantly. The separate processing of the process data avoids internal runtime problems.

## 5.3.4 Selection of the node address of the fieldbus system

### 5.3.4.1 Selection of the CANopen node address

Each drive converter gets a unique CAN address, the CAN node ID.

Index	Id-Text	Name	Function
0x2B40	fb64	CAN node ID	Setting the CANopen node address value



➤ After delivery, all KEB drive controllers have the CAN node address "1". If several KEB drive converters are to be networked via CAN, these drive converters must be given different CAN addresses. (=> Chapter 6 CANopen )



➤ On the devices of control type A and P it is possible to influence the CAN node address via the rotary coding switches located on the device. (See next chapter)

### 5.3.4.2 Selection of the node address via the rotary coding switches (control type A and P)

For devices of control type A or P, the node address can be set via the rotary coding switches at the device or via parameter fb101.

Index	Id-Text	Name	Function
0x2B64	fb100	node ID switch value	Indicates the read node address value of the rotary coding switches.
0x2B65	fb101	adjusted node ID value	Parameter for the alternative setting of the node address.
0x2B66	fb102	effective node ID	Active node address value in the device. Corresponds to fb100 as long as fb101 = 0. Otherwise equivalent to fb101. Updated when the device is restarted.

The effective node address value (fb102) is updated only when the device is restarted. Without a restart, changes in fb100, fb101 have no effect.

The effective node address value (fb102) is accepted in fb64 CAN node ID when the device is restarted, if it corresponds to a value valid for CAN (1 - 127) and CANopen is the active fieldbus system.

The effective node address value (fb102) affects the base IP configuration used by PROFINET, POWERLINK, EtherNet/IP, Modbus/TCP and the Ethernet channel. (=> Chapter 5.5.1.3 )

The effective node address value (fb102) affects the storage of the PROFINET device name. fb101 is furthermore coupled to the writing of the PROFINET device name. (=> Chapter 9.4.2 PROFINET device name)

### 5.3.5 Selection of the transmission speed (CANopen)

For the basic configuration of CANopen, the transmission speed must be selected by parameter fb66.

The bit-timing abides by the specifications of the working committee Physical-Layer of CiA.

What kind of transmission rates are possible depends on the line length, the sum of the deceleration times and the bit-timing and must be cleared up for each individual case.

Index	Id-Text	Name	Function
0x2B42	fb66	CAN baud rate	Setting the transmission speed

fb66	CAN baud rate		0x2B42
Value	Name	Meaning	
1	-	20kBit	
2	-	25kBit	
3	-	50kBit	
4	-	100kBit	
5	-	125kBit	
6	-	250kBit	
7	-	500kBit	
8	-	1MBit	

## 5.4 Restart the device

A restart must be carried out in order to activate the parameters changed in chapter 5.3 Adjust fieldbus parameter.

To carry out a restart, either parameter **co09** "reset ctrl" can be set or the device is restarted via the 24V supply voltage.



- Restarting the software interrupts communication to the device for a short period.

Index	Id-Text	Name	Function
0x2509	<b>co09</b>	<b>reset ctrl</b>	Trigger a software restart by writing the value 1. (Writing of value 0 has no effect)

The device can be reset to factory setting by restarting the software. This requires additional settings in parameter **co08**.

Index	Id-Text	Name	Function
0x2508	<b>co08</b>	<b>reset options</b>	Software restart options

<b>co08</b>	<b>reset options</b>		<b>0x2508</b>
Value	Name	Notice	
0	no options	Parameter values are retained during software restart (default)	
1	default after every reset (DER)	The default settings are loaded after each software restart	
2	default after next reset (DNR)	The default settings are only loaded after the next software restart	
3	DER + DNR	The default settings are loaded after each software restart	

To ensure that after resetting to factory settings the communication to the KEB device can be restarted, some fieldbus parameters are not affected by the reset.

Parameter	Name	Control board version
fb13	DIN66019 nodeID	x6A, x6K, x6P
fb64	CAN node ID	x6A, x6K, x6P
fb66	CAN baud rate	x6A, x6K, x6P
fb68	fieldbus selection	x6A, x6K, x6P
fb71	fieldbus options	x6P
fb80	MRTE module	x6P
fb101	adjusted node ID value	x6A
fb103	FB MAC Address (Base)	x6A
fb104	PNET MAC Address (Port1)	x6A
fb105	PNET MAC Address (Port2)	x6A
fb106	MAC Address EthChannel	x6P
fb108	Ethernet over fieldbus IP configuration	x6A (PROFINET and Ether-Net/IP)
fb109	Basic IP configuration	x6A, x6P
fb113	EtherNet/IP Configuration	x6A
fb114	Modbus/TCP Configuration	x6A

Table 5-16: non-resettable fb parameters

## NOTICE

- After a restart, the new selected Ethernet-based fieldbus stack is copied on the devices of control type A. While copying is in progress, the fieldbus STATUS LED (NET ST) and the control STATUS Led (DEV ST) will flash yellow. Further information on the LED flashing patterns can be found in chapter 5.5.2 Checking the fieldbus STATUS LED (NET ST)
- The selected fieldbus system can only be used after copying has been completed.

## 5.5 Check the fieldbus status

To determine whether the fieldbus system is in the required state, the parameters presented in this chapter can be checked. Such a check is recommended after the active fieldbus system has been changed and when fieldbus communication is started up for the first time.

### 5.5.1 Checking MAC addresses and IP configuration

MAC addresses and IP configurations are required for the Ethernet based fieldbus systems PROFINET, POWERLINK and EtherNet/IP as well as the EtherCAT functionality EoE. These are indicated via the parameters **fb103** to **fb106** as well as **fb108** and **fb109**.

Depending on the used control type and the active fieldbus system, only a subset of these parameters is supported. The following table gives an overview of these subsets.

Control type	Fieldbus system	<b>fb103</b> : FB MAC Address (Base)	<b>fb104</b> : PNet MAC Address (Port 1)	<b>fb105</b> : PNet MAC Address (Port 2) / MAC Address (EoE Channel)	<b>fb106</b> : MAC Address (EthChannel)	<b>fb108</b> : Ethernet over fieldbus IP configuration	<b>fb109</b> : Basic IP configuration
S6A / F6A	EtherCAT	-	-	-	x	x	-
	PROFINET	x	x	x	x	x	x
	POWERLINK	x	-	-	-	-	x
	EtherNet/IP	x	-	-	x	x	x
	Modbus/TCP	x	-	-	-	x	x
S6K / F6K	EtherCAT	-	-	-	x	x	-
F6P	EtherCAT	-	-	x	-	x	-
	Ethernet	-	-	-	x	-	x

Table 5-17: Differences in the support of MAC / IP parameters

#### 5.5.1.1 Checking the MAC addresses

The MAC addresses **fb103** - **fb105** on the devices of control type A are set to a fixed value in the production process by a protected mechanism.

The MAC address **fb106** on the devices of control type A is a virtual MAC address which displays the EoE MAC address, the MAC address of the PROFINET Ethernet channel and the MAC address of the EtherNet/IP Ethernet channel.

The virtual MAC address is set on the devices of control type A to a default value intended for PROFINET. To the value of **fb103: FB MAC Address (Base)** + 3. An EoE capable EtherCAT master can overwrite this value. The value written by the EtherCAT master is only stored on the device until the next restart.

On the devices of control type K, only **fb106** is available as virtual MAC address of the EoE channel. This can also be written by an EoE-capable EtherCAT master. The value is retained until the next software restart.

On the devices of control type P, the Ethernet MAC address is always displayed in **fb106**. This is set to a fixed value in the production process by a protected mechanism.

The virtual EoE MAC address is displayed in parameter **fb105**, with the name **fb105: MAC Address (EoE Channel)** on devices of control type P. The EoE MAC address behaves as on the devices of control type A and K.

Further information on the use of MAC addresses and the virtual MAC address can be found in the corresponding chapters of the respective fieldbus systems.

- => Chapter 7.5.1 Start-up of Ethernet over EtherCAT
- => Chapter 9.4.1 PROFINET MAC addresses
- => Chapter 10.3.1 POWERLINK MAC address

Index	Id-Text	Name	Function
0x2B67	fb103	Fb MAC Address (Base)	MAC address for Ethernet based fieldbus systems
0x2B68	fb104	PNet MAC Address (Port1)	PROFINET MAC Address for Port1
0x2B69	fb105	PNet MAC Address (Port2) / MAC Address (EoE Channel)	PROFINET MAC Address for Port2 (A) / volatile EoE MAC address (P)
0x2B6A	fb106	MAC Address (EthChannel)	MAC address for all Ethernet channels (EoE (A, K), PROFINET Ethernet channel (A), Ethernet (P))

#### 5.5.1.2 Checking the IP configuration for the fieldbus Ethernet channel

The IP configuration for the fieldbus Ethernet channel is displayed via parameter **fb108 Ethernet over fieldbus IP configuration**. (EoE, PROFINET Ethernet channel, EtherNet/IP Ethernet diagnostic channel, Modbus/TCP diagnostic channel)

Index	Id-Text	Name	SubIdx	Name-SubIdx	Function
0x2B6C	fb108	Ethernet over fieldbus IP con- figuration	0		Number of sub-indices
			1	IP address	IP Address
			2	subnet mask	Subnet mask
			3	gateway address	Gateway address

##### 5.5.1.2.1 fb108: Ethernet over fieldbus IP configuration at EtherCAT

If EoE is active, parameter **fb108** is set by the EtherCAT master. Changes to **fb108** via COMBIVIS do not affect the active configuration.

Further information about EoE can be found in chapter 7.5 Ethernet over EtherCAT (EoE).

##### 5.5.1.2.2 fb108: Ethernet over fieldbus IP configuration at PROFINET

If PROFINET is active, parameter **fb108** displays the IP configuration for the parallel Ethernet channel. The default value of the IP configuration for the parallel Ethernet channel corresponds to the default value of the PROFINET basic IP configuration (**fb109**) increased by 100. (**fb109**: 192.168.0.100 leads to **fb108**: 192.168.0.200)

Parameter **fb108** can be changed by the PROFINET host and by the procedures described in this chapter. Changes are stored permanently.

If PROFINET is active, parameter **fb108** can change its value when the device is restarted. This depends on the effective node address value (**fb102**).

- If the effective node address value is 254, **fb108** is reset to the default value.
- If the effective node address value is 241 to 243, **fb108** is set to 0.
- For other values of the effective node address, **fb108** is not changed. The value set before the restart is retained.

Further information on the effective node address value (**fb102**) can be found in chapter 5.3.4 Selection of the node address.

If PROFINET is active, parameter **fb108** can change its value by writing to **fb109**. This depends on the value in **fb108**.

- If **fb108** corresponds to the default value (192.168.0.200; 255.255.255.0; 0), then **fb108** is set to **fb109** when writing to **fb109**, with an offset of 100 on the IP address.

- If **fb108** does not correspond to the default value, **fb108** is not affected by writing to **fb109**.



- Example:
- **fb108** corresponds to the default value. (192.168.0.200; 255.255.255.0; 0)
- **fb109** is written by the PROFINET host to (192.168.100.5; 255.255.252.0; 0).
- **fb108** will automatically become (192.168.100.105; 255.255.252.0; 0).
- With a new write access to **fb109**, **fb108** does not change, because **fb108** does not correspond to the default value.

When accessing via the COMBIVIS parameters, **fb108** is not updated until the gateway of **fb109** is written.

More information on the parallel Ethernet channel for PROFINET can be found in chapter 9.3.3 Additional diagnostic channel via standard Ethernet communication

#### 5.5.1.2.3 fb108: Ethernet over fieldbus IP configuration at EtherNet/IP

If PROFINET is active, parameter **fb108** displays the IP configuration for the Ethernet diagnostic channel.

The value of the IP configuration for the EtherNet/IP diagnostic channel corresponds to the value of the EtherNet/IP basic IP configuration(**fb109**) increased by a subnet. (e.g.: **fb109**: 192.168.0.100 leads to **fb108**: 192.168.1.100)

#### 5.5.1.2.4 fb108: Ethernet over fieldbus IP configuration at Modbus/TCP

If Modbus/TCP is active, parameter **fb108** indicates the IP configuration of the Ethernet diagnostic channel. It can only be changed via COMBIVIS.

### 5.5.1.3 Checking the Basic IP Configuration

The basic IP configuration for the Ethernet-based fieldbus systems PROFINET, POWERLINK, EtherNet/IP and Modbus/TCP is displayed on the devices of control type A via parameter [fb109](#).

On the devices of the control type P the IP configuration for the PROFINET and the Ethernet channel is displayed via parameter [fb109](#).

Parameter [fb109](#) is not available on devices of control type K.

Index	Id-Text	Name	SubIdx	Name-SubIdx	Function
0x2B6D	<a href="#">fb109</a>	<a href="#">Basic IP configuration</a>	0		Number of sub-indices
			1	IP address	IP Address
			2	subnet mask	Subnet mask
			3	gateway address	Gateway address

The default value corresponds to the basic IP configuration ([fb109](#)):

IP address 192.168.0.100  
Subnet mask: 255.255.255.0  
Gateway: 0.0.0.0

The value of the basic IP configuration is changed on the devices of control type A and P depending on the effective node address value ([fb102](#)) after restarting the device.

Further information on the effective node address value ([fb102](#)) can be found in chapter 5.3.4 Selection of the node address.

How the IP configuration is affected depends on the fieldbus system selected in [fb68 fieldbus selection](#).

Fieldbus system		Effective node address value ( <a href="#">fb102</a> )	IP configuration	Changeable via <a href="#">fb109</a>
PROFINET		0 – 240, 244 – 253, 255	Is not affected	Yes
		241 - 243	0 0 0	Yes
		254	192.168.0.100 255.255.255.0 0	No
POWERLINK		1 – 239, 241 - 254	192.168.100.Node address value 255.255.255.0 192.168.100.254	No, except for the gateway
Fieldbus system	IP configuration method ( <a href="#">fb113</a> )	Effective node address value ( <a href="#">fb102</a> )	IP configuration	Changeable via <a href="#">fb109</a>
EtherNet/IP	Bootp or DHCP(default)	0	Is not affected	No, value is specified by the fieldbus system
	Static	1 - 255	192.168.0.Node address value 255.255.255.0 0	No



	IP configuration method (fb114)	Effective node address value (fb102)	IP configuration	Changeable via fb109
Modbus/TCP	Bootp or DHCP(default)	0	Is not affected	No, value is specified by the fieldbus system
	Static	1 - 255	192.168.0.Node address value 255.255.255.0 0	No

It is possible to change the basic IP configuration via the COMBIVIS interface. Depending on the active fieldbus system, further steps must be taken to activate the basic IP configuration in the device:

- For PROFINET, POWERLINK, EtherNet/IP and Modbus/TCP a restart is required to accept the IP configuration. For POWERLINK only the gateway can be changed via the COMBIVIS interface.
- The Ethernet channel on the devices of control type P automatically adopts the new IP configuration.



- The node address values 0, 240 and 255 for a POWERLINK device are invalid values. The IP configuration is set to 192.168.100.0 in this case. Connection with the device via POWERLINK is then **not** possible.
- Modbus/TCP only shows the current IP configuration in fb109 if the IP configuration "static" is selected.

## 5.5.2 Checking the fieldbus STATUS LED (NET ST)

The status of the fieldbus can be determined with the STATUS LED (NET ST). It is a two-colour LED with the basic colours green and red. If no error has occurred, the LED should display the flashing pattern "Operational".

A detailed description of the status LEDs can be found in the [Programming Manual | Control Application / Compact / Pro](#).

### 5.5.2.1 Light patterns of the different fieldbus systems

Fieldbus system	Flashing	Color	State
EtherCAT	Continuous flashing	Green	Pre-Operational
	Flickering	Green	Boot
	Single lightning	Green	Safe Operational
	On	Green	Operational
	Continuous flashing	Red	Error in the configuration
	Flickering	Red	Error during boot
	Single lightning	Red	Error during status change
	Double lightning	Red	Watchdog error
CANopen	Continuous flashing	Green	Pre-Operational
	Single lightning	Green	Stopped
	On	Green	Operational
	Flickering	Green	Error in the initialization
	Continuous flashing, double flash	Green, red	Error in Pre-Operational
	Single flash, double flash	Green, red	Error: CANopen stopped
	Single flash, on	Red, green	Error in Operational

## Check the fieldbus status

Fieldbus system	Flashing	Color	State
VARAN	Continuous flashing	Red	Initialization
	On	Green	Operational
PROFINET	Continuous flashing	Green	PROFINET LED flashing mode
POWERLINK	Single lightning	Green	Pre-Operational 1
	Double lightning	Green	Pre-Operational 2
	Triple lightning	Green	Ready for Operational
	On	Green	Operational
	Continuous flashing	Green	Stopped
	Flickering	Green	Ethernet mode
	On	Red	Error
EtherNet/IP	Continuous flashing	Green	No connection
	Continuous flashing	Red	Timeout
	On	Green	connected
	On	Red	Double IP
	Continuous flashing	Green, red	Self test
Modbus/TCP	Continuous flashing	Green	No connection
	On	Green	connected
All fieldbus systems	Off	-	Fieldbus system not active

### 5.5.2.2 Description of the LED light pattern

Flashing	Description
Off	Sensor is constantly off
On	Sensor is constantly on
Continuous flashing	Cyclic change with 200ms on/200ms off
Single lightning	Cyclic change with 200ms on/1000ms off
Double lightning	Cyclic change with 200ms on/200ms off/200ms on/1000ms off
Triple lightning	Cyclic change with 200ms on/200ms off/200ms on/200ms off/200ms on/1000ms off
Flickering	Cyclic change with 50ms on/50ms off

### 5.5.2.3 Timing of the flashing patterns

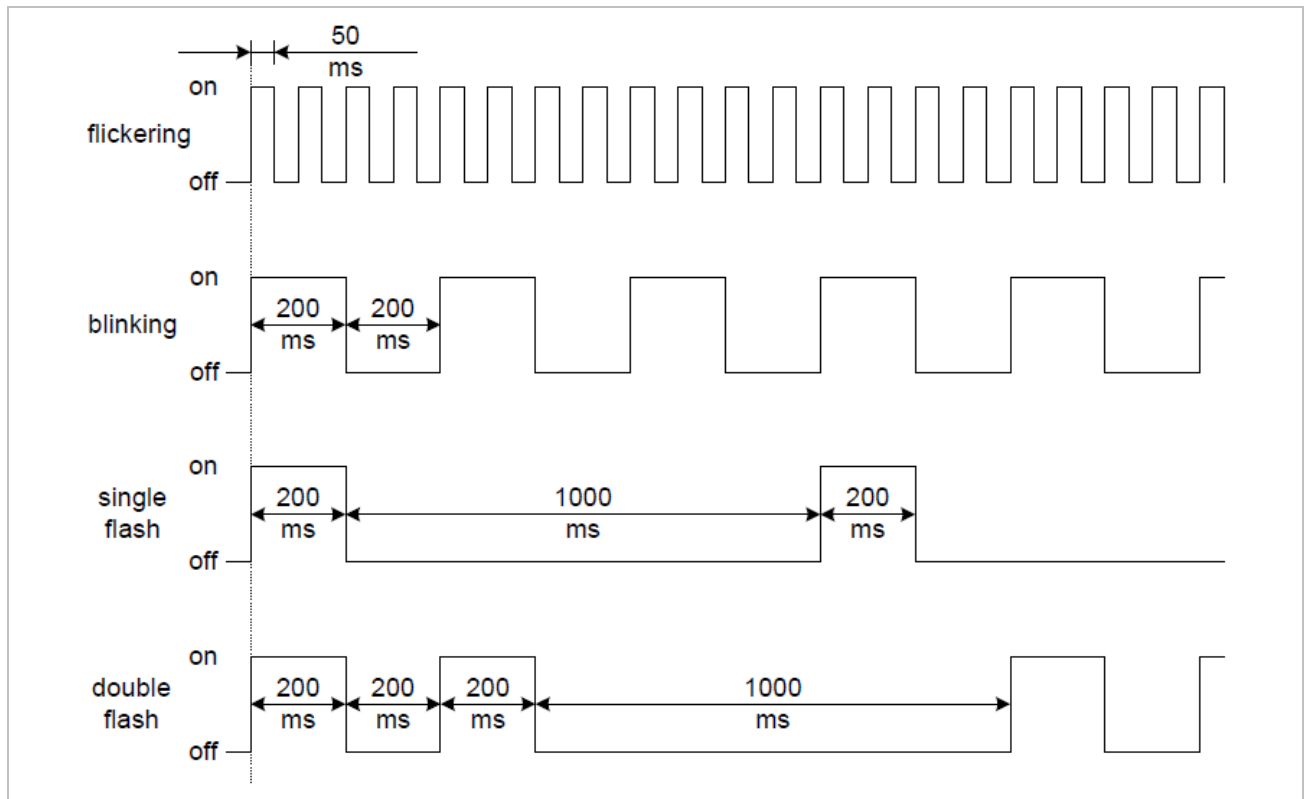


Figure 1: Timing of the LED flashing pattern

### 5.5.3 Checking the exception state

Occurred errors can be determined by reading out of the exception state and by reading out of the error code via the diagnostic interface.

Index	Id-Text	Name	Function
0x2C01	ru01	exception state	Display of an occurred error as plaintext
0x2101	st01	error code	Display of an occurred error as error code

The following error codes can occur in case of errors in the fieldbus module:

ru01	Error text	Description	st01
24	ERROR fieldbus memory	Incorrect drive software configuration	8: Fault
58	ERROR fieldbus watchdog	Timeout of the fieldbus watchdog	8: Fault
124	General Fieldbus Error	General Fieldbus Error	8: Fault
125	ERROR fieldbus type changed	Fieldbus system changed	8: Fault

"General Fieldbus Error" summarizes all errors that may occur during initialization and execution of the selected fieldbus system. The actual occurred error is listed in parameter [fb91](#) described in chapter 5.5.4

Check the fieldbus state and possible fieldbus error codes.

Error "ERROR fieldbus type changed" is triggered when changing the fieldbus system via [fb68](#). The error code indicates that a restart is necessary to activate the newly selected fieldbus system. Further information can be found in chapter 5.3.1 Selection of the fieldbus [system](#) via [fb68](#).

A more detailed description of parameter [ru01](#) and a list of all error codes can be found in the [Programming Manual | Control Application / Compact / Pro](#).

#### 5.5.4 Check the fieldbus state and possible fieldbus error codes

Using the diagnostic interface, both the status ([fb90](#)) and possible error codes ([fb91](#)) can be determined from the fieldbus parameters. Parameters [fb90](#), [fb91](#) indicate the supported fieldbus systems on the respective control board.

Index	SubIdx	Id-Text	Name	Function
0x2B5A	0	<a href="#">fb90</a>	<a href="#">fieldbus state (number)</a>	Number of available fieldbus systems
0x2B5A	1	<a href="#">fb90</a>	<a href="#">EtherCAT fieldbus state</a>	Status display of the EtherCAT fieldbus system (register AL status)
0x2B5A	2	<a href="#">fb90</a>	<a href="#">CANopen fieldbus state</a>	Status display of the CANopen fieldbus system (Register CO_RunState)
0x2B5A	3	<a href="#">fb90</a>	<a href="#">VARAN fieldbus state</a>	Status display of the VARAN fieldbus system (only control type K)
0x2B5A	3	<a href="#">fb90</a>	<a href="#">PROFINET fieldbus state</a>	Status display of the PROFINET state machine (only control type A)
0x2B5A	4	<a href="#">fb90</a>	<a href="#">POWERLINK fieldbus state</a>	Status display of the POWERLINK state machine (only control type A)
0x2B5A	5	<a href="#">fb90</a>	<a href="#">EtherNet/IP fieldbus state</a>	Status display of the EtherNet/IP state machine (only control type A)
0x2B5A	6	<a href="#">fb90</a>	<a href="#">ModbusTCP fieldbus state</a>	Status display of the Modbus/TCP state machine (only control type A)
0x2B5B	0	<a href="#">fb91</a>	<a href="#">fieldbus error code (number)</a>	Number of available fieldbus systems
0x2B5B	1	<a href="#">fb91</a>	<a href="#">EtherCAT fieldbus error code</a>	Error code of the EtherCAT fieldbus system (Register AL Status Code)
0x2B5B	2	<a href="#">fb91</a>	<a href="#">CANopen fieldbus error code</a>	Error-Code of the CANopen fieldbus system (Register EmergencyErrField)
0x2B5B	3	<a href="#">fb91</a>	<a href="#">VARAN fieldbus error code</a>	Error code of the VARAN fieldbus system (only control type K)
0x2B5B	3	<a href="#">fb91</a>	<a href="#">PROFINET fieldbus error code</a>	Error code of the PROFINET fieldbus system (only control type A)
0x2B5B	4	<a href="#">fb91</a>	<a href="#">POWERLINK fieldbus error code</a>	Error code of the POWERLINK fieldbus system (only control type A)
0x2B5B	5	<a href="#">fb91</a>	<a href="#">Ethernet/IP fieldbus error code</a>	Error code of the EtherNet/IP fieldbus system (only control type A)
0x2B5B	6	<a href="#">fb91</a>	<a href="#">ModbusTCP fieldbus error code</a>	Error code of the EtherNet/IP fieldbus system (only control type A)

## 5.5.4.1 fb90: fieldbus state

fb90 displays the state of all fieldbus systems which are supported by the used hardware. The state of all inactive fieldbus systems is „0xFFFF: Fieldbus inactive“.



- The displayed values for PROFINET are KEB-defined values which do not correspond to the PROFINET communication profile.
- Modbus/TCP currently does not support plain texts.

Fieldbus system	Value	Plaintext
EtherCAT	0x0001	ECAT_ALSTATUS_INIT
	0x0002	ECAT_ALSTATUS_PREOP
	0x0003	ECAT_ALSTATUS_BOOTSTRAP
	0x0004	ECAT_ALSTATUS_SAFEOP
	0x0008	ECAT_ALSTATUS_OPERATIONAL
	0x0010	ECAT_ALSTATUS_ERROR
CANopen	0x0000	CAN301_STATE_INITIALISATION
	0x0004	CAN301_STATE_STOPPED
	0x0005	CAN301_STATE_OPERATIONAL
	0x007F	CAN301_STATE_PREOPERATIONAL
VARAN	0x0000	VARAN_STATE_INIT
	0x0001	VARAN_STATE_RUNNING
PROFINET	0x0000	PROFINET_STATE_OFF
	0x0001	PROFINET_STATE_INIT_DONE
	0x0002	PROFINET_STATE_COMM_READY
	0x0004	PROFINET_STATE_COMM_RUN
	0x0008	PROFINET_STATE_SYNC_ACTIVE
POWERLINK	0x0000	EPG_DS301_NMT_GS_OFF
	0x0019	EPG_DS301_NMT_GS_INITIALISING
	0x001C	EPG_DS301_NMT_CS_NOT_ACTIVE
	0x001D	EPG_DS301_NMT_CS_PREOP
	0x001E	EPG_DS301_NMT_CS_BASIC_ETH
	0x0029	EPG_DS301_NMT_GS_RESET_APPL
	0x0039	EPG_DS301_NMT_GS_RESET_COMM
	0x004D	EPG_DS301_NMT_CS_STOPPED
	0x005D	EPG_DS301_NMT_CS_PREOP2
	0x006D	EPG_DS301_NMT_CS_READYTOOP
	0x0079	EPG_DS301_NMT_GS_RESET_CONFIG
	0x00FD	EPG_DS301_NMT_CS_OPERATIONAL
EtherNet/IP	0x0100	ENIPSLV_STATE_SETMACADDR
	0x0200	ENIPSLV_STATE_SETCONFIG
	0x0300	ENIPSLV_STATE_REGAP
	0x0400	ENIPSLV_STATE_CHANNELINIT
	0x0A00	ENIPSLV_STATE_RUN_IDLE

#### 5.5.4.2 fb91: fieldbus error code

Error codes for the supported fieldbus systems are displayed in **fb91**. The state of all inactive fieldbus systems is „0xFFFF: Fieldbus inactive“. If no error has occurred, the state is normally "0x0000".

Fieldbus-specific error codes are also displayed in **fb91**. Only the KEB specific error codes are displayed here. Information on the fieldbus-specific error codes can be found in the documentation of the respective fieldbus system.

Further information and solutions for some of the KEB specific error codes listed here can be found in the appendix under 12.3 Solutions for KEB specific fieldbus error codes .



- The errors listed in **fb91** have nothing to do with the error codes in **ru01**. **fb91** describes only errors which are related to bus systems.
- If an error occurs during the initialization process (INIT\_ERR), the error code: General fieldbus error is additionally displayed in **ru01**.
- If an error code is displayed in **fb91**, which is neither listed here nor in the specification of the respective fieldbus system, please contact your KEB contact person.
- Modbus/TCP currently does not support plain texts.

Fieldbus system	Value	Plaintext
Error codes for all fieldbus systems	0xFE70	INIT_ERR_PD
	0xFED4	INIT_ERR_HSP5_NR_PD
	0xFF9C	INIT_ERR_HSP5_PD
	0xFFCE	INIT_ERR_SHM
	0xFFE2	INIT_ERR_NETX_BASE
	0xFFEC	INIT_ERR_I2C_MUTEX
	0xFFF5	INIT_ERR_FB_SLV_OR_DRV
	0xFFF6	INIT_ERR_I2C
	0xFFF7	INIT_ERR_LED_TMR
	0xFFF8	INIT_ERR_LED_DRV
	0xFFF9	INIT_ERR_BASE_DRV_NOT_ONLINE
	0xFFFA	INIT_ERR_MSG_QUEUE
	0xFFFB	INIT_ERR_FB_TYPE_INVALID
	0xFFFF	Fieldbus inactive
EtherCAT	0x0000	ECAT_AL_STATUS_CODE_NO_ERROR
	0x0001	ECAT_AL_STATUS_CODE_UNSPECIFIED_ERROR
	0x0011	ECAT_AL_STATUS_CODE_INVALID_REQUESTED_STATE_CHANGE
	0x0012	ECAT_AL_STATUS_CODE_UNKNOWN_REQUESTED_STATE
	0x0013	ECAT_AL_STATUS_CODE_BOOTSTRAP_NOT_SUPPORTED
	0x0014	ECAT_AL_STATUS_CODE_NO_VALID_FIRMWARE
	0x0015	ECAT_AL_STATUS_CODE_INVALID_MAILBOX_CONFIGURATION_BOOTSTRAP
	0x0016	ECAT_AL_STATUS_CODE_INVALID_MAILBOX_CONFIGURATION_PREOP
	0x0017	ECAT_AL_STATUS_CODE_INVALID_SYNC_MANAGER_CONFIGURATION
	0x0018	ECAT_AL_STATUS_CODE_NO_VALID_INPUTS_AVAILABLE
	0x0019	ECAT_AL_STATUS_CODE_NO_VALID_OUTPUTS
	0x001A	ECAT_AL_STATUS_CODE_SYNCHRONIZATION_ERROR
	0x001B	ECAT_AL_STATUS_CODE_SYNC_MANAGER_WATCHDOG
	0x001C	ECAT_AL_STATUS_CODE_SYNCTYPES_NOT_COMPATIBLE
	0x001D	ECAT_AL_STATUS_CODE_INVALID_OUTPUT_CONFIGURATION
	0x001E	ECAT_AL_STATUS_CODE_INVALID_INPUT_CONFIGURATION
	0x001F	ECAT_AL_STATUS_CODE_INVALID_WD_CONFIGURATION

Fieldbus system	Value	Plaintext
	0x0020	ECAT_AL_STATUS_CODE_SLAVE_NEEDS_COLD_START
	0x0021	ECAT_AL_STATUS_CODE_SLAVE_NEEDS_INIT
	0x0022	ECAT_AL_STATUS_CODE_SLAVE_NEEDS_PREOP
	0x0023	ECAT_AL_STATUS_CODE_SLAVE_NEEDS_SAFEOP
	0x0024	ECAT_AL_STATUS_CODE_INVALID_INPUT_MAPPING
	0x0025	ECAT_AL_STATUS_CODE_INVALID_OUTPUT_MAPPING
	0x0026	ECAT_AL_STATUS_CODE_INCONSISTENT_SETTINGS
	0x002C	ECAT_AL_STATUS_CODE_FATAL_SYNC_ERROR
	0x0030	ECAT_AL_STATUS_CODE_DC_INVALID_SYNC_CONFIG
	0x0031	ECAT_AL_STATUS_CODE_DC_INVALID_LATCH_CONFIG
	0x0032	ECAT_AL_STATUS_CODE_DC_PLL_SYNC_ERROR
	0x0033	ECAT_AL_STATUS_CODE_DC_SYNC_IO_ERROR
	0x0034	ECAT_AL_STATUS_CODE_DC_SYNC_MISSED
	0x0035	ECAT_AL_STATUS_CODE_INVALID_SYNC_CYCLE_TIME
	0x0036	ECAT_AL_STATUS_CODE_DC_SYNC0_CYCLE_TIME
	0x0037	ECAT_AL_STATUS_CODE_DC_SYNC1_CYCLE_TIME
	0xFE0A	ECAT_SLAVE_FLAGS_INIT_ERROR
	0xFE0B	ECAT_NO_STACK_INIT_ERROR
	0xFE0C	ECAT_SSI_INIT_ERROR
	0xFFD8	INIT_ERR_EOE
	0xFFFC	ECAT_EOE_INIT_ERROR
	0xFFFD	ECAT_MBX_INIT_ERROR
	0xFFFE	ECAT_NODEID_INIT_ERROR
CANopen	0x0000	CAN301_ERROR_NO_ERROR
	0x0010	CAN301_ERROR_GENERIC_ERROR
	0x0020	CAN301_ERROR_CURRENT
	0x0030	CAN301_ERROR_VOLTAGE
	0x0040	CAN301_ERROR_TEMPERATURE
	0x0050	CAN301_ERROR_DEVICE_HARDWARE
	0x0060	CAN301_ERROR_DEVICE_SOFTWARE
	0x0070	CAN301_ERROR_ADDITIONAL_MODULES
	0x0080	CAN301_ERROR_MONITORING
	0x0090	CAN301_ERROR_EXTERNAL
	0x00F0	CAN301_ERROR_ADDITIONAL_FUNCTIONS
	0x00FF	CAN301_ERROR_DEVICE_SPECIFIC
	0xFFCC	CAN_ERROR_HB_TIMEOUT
	0xFFCD	CAN_ERROR_NG_TIMEOUT
	0xFFFE	INIT_ERR_CAN_DRV
VARAN	0x0000	VARAN_NO_ERROR
	0xFFFE	VARAN_FPGA_INIT_ERROR
PROFINET	0x0000	PROFINET_NO_ERROR
	0x0001	PROFINET_ERROR_INIT
	0xFD12	PROFINET_PD_CFG_FLAGS_ERROR
	0xFD44	PROFINET_PD_CFG_ERROR
	0xFE0A	PROFINET_SLAVE_FLAGS_INIT_ERROR
	0xFE0B	PROFINET_NO_STACK_INIT_ERROR
	0xFE0C	PROFINET_MSG_QUEUE_INIT_ERROR
	0xFF38	PROFINET_MAC_ADDR_ERROR
	0xFFFC	PROFINET_MAC_ADDR_INV
	0xFFFE	PROFINET_CLIENT_INV

Fieldbus system	Value	Plaintext
POWERLINK	0x0000	EPDG_DS301_ERR_NO_ERR
	0xFC0E	EPL_EPDG_MAP_INIT_ERROR
	0xFD44	EPL_PD_SIZE_ERROR
	0xFE0A	EPL_SLAVE_FLAGS_INIT_ERROR
	0xFE0B	EPL_NO_STACK_INIT_ERROR
	0xFE0C	EPL_MSG_QUEUE_INIT_ERROR
EtherNet/IP	0xFE0A	ENIP_SLAVE_FLAGS_INIT_ERROR
	0xFE0B	ENIP_NO_STACK_INIT_ERROR
	0xFE0C	ENIP_MSG_INIT_ERROR
	0xFF38	ENIP_MAC_ADDR_INIT_ERROR
	0xFFFE	ENIP_CLIENT_INV



## 5.6 Checking the process data mapping

### 5.6.1 Mapping of the process data

The mapping of the process data for all fieldbuses occurs CANopen-conform via the mapping objects [0x1600 - 0x1603](#) (process output data) and [0x1A00 - 0x1A03](#) (process input data).

A KEB mapping object can contain up to 8 parameters and up to 32 bytes by default. A KEB mapping object can contain a maximum of 32 parameters and 64 bytes. To enable smaller SYNC intervals, a KEB mapping object can also contain only up to 8 objects and 16 bytes. For more information, see chapter 5.3.2 Selection of the process data size via fb60.

EtherCAT supports one mapping object each for unsafe process output data ([0x1600](#)) and unsafe process input data ([0x1A00](#)) as well as one mapping object each for safe (FSoE) process output data ([0x1601](#)) and safe process input data ([0x1A01](#)).

CANopen supports all KEB mapping objects. However, a CANopen mapping object can only contain up to 8 bytes.

VARAN, PROFINET, POWERLINK, EtherNet/IP and Modbus/TCP support only one mapping object for process output data ([0x1600](#)) and one mapping object for process input data ([0x1A00](#)).

Mapping parameters cannot be changed as long as the mapping is active. The mapping is active as long as the subindex 0 of the respective mapping object corresponds to a value unequal zero.

To set the process data mapping it is recommended to use the COMBIVIS Fieldbus Wizard. (=> Chapter 5.6.4 Fieldbus wizard)

In order to avoid invalid process data settings, the mapping of the process data is deleted when the fieldbus system is changed and the process data are deactivated. The following special cases must be observed:

- When changing to EtherCAT on the devices of control type A, the default parameters (user parameters: [aa16 – aa23](#)) are loaded into the first mapping object and activated.

For PROFINET, the device must be restarted again after the required process data has been set. (=> Chapter 5.4 Preface).

For Modbus/TCP the number of registers to be set depends on the set mapping. More information in chapter 12.1 Address range.

Further information on selecting the fieldbus system can be found in chapter 5.3.1 Selection of the fieldbus [system](#) via [fb68](#).

Index	SubIdx	Id-Text	Name	Function
0x1600	0	-	1st receive PDO mapping (process output data)	Number of the mapped objects
	1...8			Mapping of the process data object
0x1601	0	-	2nd receive PDO mapping (only CANopen/FSOE)	Number of the mapped objects
	1...8			Mapping of the process data object
0x1602	0	-	3rd receive PDO mapping (only CANopen)	Number of the mapped objects
	1...8			Mapping of the process data object
0x1603	0	-	4th receive PDO mapping (only CANopen)	Number of the mapped objects
	1...8			Mapping of the process data object

Index	SubIdx	Id-Text	Name	Function
0x1A00	0	-	1st transmit PDO mapping (process input data)	Number of the mapped objects
	1...8			Mapping of the process data object
0x1A01	0	-	2nd transmit PDO mapping (only CANopen/FSOE)	Number of the mapped objects
	1...8			Mapping of the process data object
0x1A02	0	-	3rd transmit PDO mapping (only CANopen)	Number of the mapped objects
	1...8			Mapping of the process data object
0x1A03	0	-	4th transmit PDO mapping (only CANopen)	Number of the mapped objects
	1...8			Mapping of the process data object

The description of the mapped objects occurs in the following format:

Bit	Meaning
0...7	Object length in bits (8, 16 or 32)
8...15	Subindex of the mapped object
16...31	Index of the mapped object

## 5.6.1.1 The KEB default process data mapping

The KEB default process data mapping differs depending on the control type of the used device and the active fieldbus system.

The KEB default process data mapping is an empty, inactive process data mapping for devices of the control type K and P.

On the devices of the control type A for EtherCAT, PROFINET, EtherNet/IP and Modbus/TCP the KEB default process data mapping contains the user parameters [aa16](#) to [aa23](#). All 8 user parameters are active.

The default mapping is not activated if CANopen is the active fieldbus system on the devices of control type A.

The KEB default process data mapping is an empty, inactive process data mapping if POWERLINK is the active fieldbus system on the devices of control type A.

Index	SubIdx	Value	Function
0x1600	0	8	Number of mapped process output objects = 8
0x1600	1	0x29100020	Index = 0x2910, Subindex = 0x00, Length = 0x20 (32bit)
0x1600	2	0x29110020	Index = 0x2911, Subindex = 0x00, Length = 0x20 (32bit)

Index	Subidx	Value	Function
0x1600	3	0x29120020	Index = 0x2912, Subindex = 0x00, Length = 0x20 (32bit)
0x1600	4	0x29130020	Index = 0x2913, Subindex = 0x00, Length = 0x20 (32bit)
0x1600	5	0x29140020	Index = 0x2914, Subindex = 0x00, Length = 0x20 (32bit)
0x1600	6	0x29150020	Index = 0x2915, Subindex = 0x00, Length = 0x20 (32bit)
0x1600	7	0x29160020	Index = 0x2916, Subindex = 0x00, Length = 0x20 (32bit)
0x1600	8	0x29170020	Index = 0x2917, Subindex = 0x00, Length = 0x20 (32bit)
0x1A00	0	8	Number of mapped process output objects = 8
0x1A00	1	0x29100020	Index = 0x2910, Subindex = 0x00, Length = 0x20 (32bit)
0x1A00	2	0x29110020	Index = 0x2911, Subindex = 0x00, Length = 0x20 (32bit)
0x1A00	3	0x29120020	Index = 0x2912, Subindex = 0x00, Length = 0x20 (32bit)
0x1A00	4	0x29130020	Index = 0x2913, Subindex = 0x00, Length = 0x20 (32bit)
0x1A00	5	0x29140020	Index = 0x2914, Subindex = 0x00, Length = 0x20 (32bit)
0x1A00	6	0x29150020	Index = 0x2915, Subindex = 0x00, Length = 0x20 (32bit)
0x1A00	7	0x29160020	Index = 0x2916, Subindex = 0x00, Length = 0x20 (32bit)
0x1A00	8	0x29170020	Index = 0x2917, Subindex = 0x00, Length = 0x20 (32bit)

### 5.6.2 Additional parameters for process data mapping for CANopen

The following communication parameters are only used by CANopen. More detailed information about these parameters can be found in Chapter 6 CANopen Interface .



- The communication parameters **0x1400** and **0x1800** specified in the EtherCAT and POWERLINK specification are not displayed in COMBIVIS due to overlap with the CANopen communication parameters.

Index	Subidx	Id-Text	Name	Function
0x1400 - 0x1403	0	-	RPDO communication parameter	Number of elements of the structure
	1		cob-ID	CAN-ID, control information
	2		transmission type	Process data transmission type

Index	Subidx	Id-Text	Name	Function
0x1800 - 0x1803	0	-	TPDO communication parameter	Number of elements of the structure
	1		cob-ID	CAN-ID, control information
	2		transmission type	Process data transmission type
	3		inhibit time	Min. distance between CAN telegrams
	4		reserved	-
	5		event time	Timer for asynchronous transmission

### 5.6.3 Additional parameters for process data mapping for POWERLINK

Unlike other bus systems, the offset for process data in POWERLINK is variable. The process data offset is specified in bits and is displayed via parameters **fb111** and **fb112**.

By default, the process data offset of a parameter corresponds to the sum of the length of all preceding parameters.

A process data offset set for POWERLINK is not visible in the COMBIVIS Wizard.

Further information on the process data offset can be found in chapter 10.2.1 Variable process data offset.

Index	Id-Text	Name	Function
0x2B6F	fb111	POWERLINK RPDO offset	Array with 8 elements, display of the freely selectable process data offset for process output data
0x2B70	fb112	POWERLINK TPDO offset	Array with 8 elements, display of the freely selectable process data offset for process input data

### 5.6.4 Fieldbus wizard

The COMBIVIS 6 fieldbus wizard provides a graphical representation of the process data mapping. It is possible to set the process data mapping via the fieldbus wizard, but the specifications of the active fieldbus system must be observed.

The following boundary conditions are automatically considered by the fieldbus wizard:

- Data type (the complete value of the mapping parameter is calculated depending on the data type).
- Properties (depending on the object properties (RO,RW, mapping allowed), the mapping is allowed or disabled)

The wizard displays how many PDOs are available, how many parameters can be used per PDO and how many bytes are available per PDO.

It is expected that from version 6.7.0 it will also be possible to set the required process data size via the wizard.

The wizard can also be used to load examples of process data mapping.

Depending on the used fieldbus system, the device description files corresponding to the fieldbus system can also be exported. The following files can be created by the wizard:

- ESI files for EtherCAT
- EDS files for CANopen
- XDD files for POWERLINK

For VARAN, PROFINET, EtherNet/IP and Modbus/TCP no description files can be exported.

Further information on the fieldbus wizard can be found in the COMBIVIS manual.

## 5.7 Synchronization to the fieldbus master

To ensure that the process data process every fieldbus cycle, KEB devices can be synchronized to the fieldbus cycle using an internal PLL. It is essential that a SYNC interrupt is made available by the fieldbus master. Synchronization to the fieldbus cycle occurs automatically as soon as a SYNC interrupt is available.

For devices of control type A and K, it must be ensured in synchronous operation that the time of reception and processing of the process data do not overlap. (=> Chapter 5.7.2 Synchronous transfer of process data)

### 5.7.1 Checking the synchronization

Using the diagnostic interface, the synchronization between fieldbus master and KEB device can be checked.

#### 5.7.1.1 Synchronization interval and measured synchronization interval

Parameter **fb10 sync interval** is automatically set to the synchronization interval of the fieldbus master after power-on, if such a value is provided by the fieldbus system. This is valid for EtherCAT, PROFINET and POWERLINK.

If such a value is not available for the fieldbus system, **fb10** is set once after power-on to the measured synchronization interval (**fb19**). This is valid for CANopen and VARAN.

EtherNet/IP and Modbus/TCP do not support synchronous application, therefore neither **fb10** nor **fb19** is set.

The KEB device synchronizes to the synchronization interval displayed in **fb10**. **fb10** can also be adjusted later by the user via COMBIVIS.

Index	Id-Text	Name	Function
0x2B0A	fb10	sync interval	Activation of the synchronous operating mode Specification for the cycle time
0x2B13	fb19	measured sync interval	Time difference between 2 StartOfCycle Interrupts. Measured cycle time of the active fieldbus system

Independent of **fb10** the distance between two synchronization signals is measured in **fb19**. If **fb10** and **fb19** agree, the device is synchronous to the fieldbus cycle and bit 8 (synchronous) is set in: **st00: statusword**.

The synchronization interval can also be read out via the Sync Manager parameters for EtherCAT. (=> Chapter 7.2.1 Sync Manager Parameters)

KEB devices cannot support any SYNC intervals. The supported SYNC intervals depend on the minimum SYNC interval that can be set in **is22**. Only whole multiples of this interval up to a limit of 16ms are supported.

If a wrong SYNC interval is specified or measured, the device internally synchronizes to the next appropriate SYNC interval value. This value is then also displayed in **fb10**.



- Depending on the used fieldbus system, the minimum cycle time may vary.
- The minimum permissible cycle time for the respective fieldbus system is specified in the corresponding chapter of this manual. If there are no specifications for the minimum cycle time, all values that can be set by [is22](#) are supported.

Wrong SYNC interval values are acknowledged with an error code by using EtherCAT and the device does not synchronize. To signal this error, the error code [ECAT\\_AL\\_STATUS\\_CODE\\_INVALID\\_SYNC\\_CYCLE\\_TIME](#) is displayed in parameter [fb91](#).

More information about [is22](#) and the minimum SYNC interval can be found in the [Programming Manual | Control Application / Compact / Pro](#).

#### 5.7.1.2 Interpolation time period

Additionally to the display in [fb10](#), the synchronization interval is displayed in the pr parameter group under parameter [interpolation time period](#). The synchronization interval is divided into a value ([interpolation time value](#)) and an exponent ([interpolation time index](#)).

Writing to [interpolation time period](#) affects the value in [fb10](#) and vice versa.

Index	Name	Subindex	Function
0x60C2	<a href="#">interpolation time period</a>	<a href="#">interpolation time value</a>	Setting of the value of the cycle time
		<a href="#">interpolation time index</a>	Specification of the exponent of the cycle time

#### 5.7.1.3 Synchronization PLL and synchronous bit

Parameters [fb11: set sync level](#) and [fb12: KP sync PLL](#) can influence the synchronous bit in [st00: statusword](#) and the synchronization PLL.

Only if the synchronous bit in [st00: statusword](#) is set, the application is in synchronous mode and the process data are processed accordingly. If the synchronous bit is not set, the process data are processed in the asynchronous mode. (=> Chapter 5.7.2 Synchronous transfer of process data)

Further information on the synchronization PLL and the statusword can be found in the [Programming Manual | Control Application / Compact / Pro](#).

Index	Id-Text	Name	Function
0x2B0B	<a href="#">fb11</a>	<a href="#">set sync level</a>	Tolerance value between measured and set SYNC interval
0x2B0C	<a href="#">fb12</a>	<a href="#">KP sync PLL</a>	KP value of the controller of the SYNC-PLL

#### 5.7.1.4 Checking of the process data exchange

Different parameters are available for checking the successful process data exchange between fieldbus master and KEB device for the individual device types. These parameters are fieldbus-specific and are described in the diagnostic chapters of the individual fieldbus systems, with the exception of the fieldbus-independent [fb31](#).

The synchronization intervals wherein no process data were received are counted in [fb31](#). [fb31](#) is reset with each restart of the device.

During the initialization phase of most fieldbus systems, synchronization interrupts are often already sent, although process data communication has not yet started. After initialization it is common that the value of **fb31** is higher than 0. It is significant for synchronous communication that **fb31** does not increase during operation

Index	Id-Text	Name	Function
0x2B1F	<b>fb31</b>	no PDO data per sync cnt	Number of synchronization intervals wherein no process data was received

### 5.7.2 Synchronous transfer of process data

In synchronous operation, the processing of the process data is depending on the SYNC interrupt of the active fieldbus system.

In the synchronous operating mode, the process data is read from the device and written to the bus only once per fieldbus cycle. During a synchronous fieldbus cycle, the process data are updated once in the object directory and activated in the control.

If CANopen is the active fieldbus system, the transmission and reception of process data additionally depends on the subparameter **transmission type** of parameters **1st - 4th RPDO communication parameter** and **1st - 4th TPDO communication parameter** in the pr group. (=> Chapter 6.8 CANopen process data mapping)

In asynchronous operation, the process data are processed as fast as possible. The processing time of the process data is then only dependent on the set minimum SYNC interval (**is22**).

Asynchronous operation is supported by all fieldbus systems except POWERLINK.

#### NOTICE

**The correct behaviour of the inverter can only be guaranteed in synchronous operation if the sending of the PD telegrams by the master and the access to the PD telegrams by the KEB device do not occur in the same time window!**

- Due to the temporal inaccuracy during transmission and receiving the process data, an overlapping of the time windows can cause that no or several process data per cycle are received / transmitted.
- The time window for the access of the KEB device to the PD telegram is between 0 and 150us after the SYNC interrupt.
- After accessing the PD telegram, the data is first stored temporarily. The actual transfer of the values from the PD telegram occurs at the end of each SYNC cycle.
- The time window for accessing the PD telegram is displayed graphically in the COMBIVIS fieldbus wizard for EtherCAT on the devices of control type K and P. (=> Chapter 7.3.4 EtherCAT diagnosis **assistant**)

## 5.8 Fieldbus watchdog

The fieldbus watchdog function can be used to switch the drive to a defined state in case of communication loss.

The function can be parameterized via the following objects:

Index	Id-Text	Name	Function
0x2A15	pn21	fieldbus watchdog time	Time in milliseconds until the watchdog is triggered (0 = off)
0x2A16	pn22	E.fb watchdog stop mode	Error response

If the fieldbus watchdog is active, a process data telegram must be received within the watchdog time specified in pn21, otherwise the selected event in parameter pn22 is triggered.

Receiving a process data telegram resets the fieldbus watchdog.

The fieldbus watchdog is activated with the first received process data telegram if a time unequal 0 is set in pn21.

Further information on the fieldbus watchdog can be found in the [Programming manual | Control Application / Compact / Pro](#).

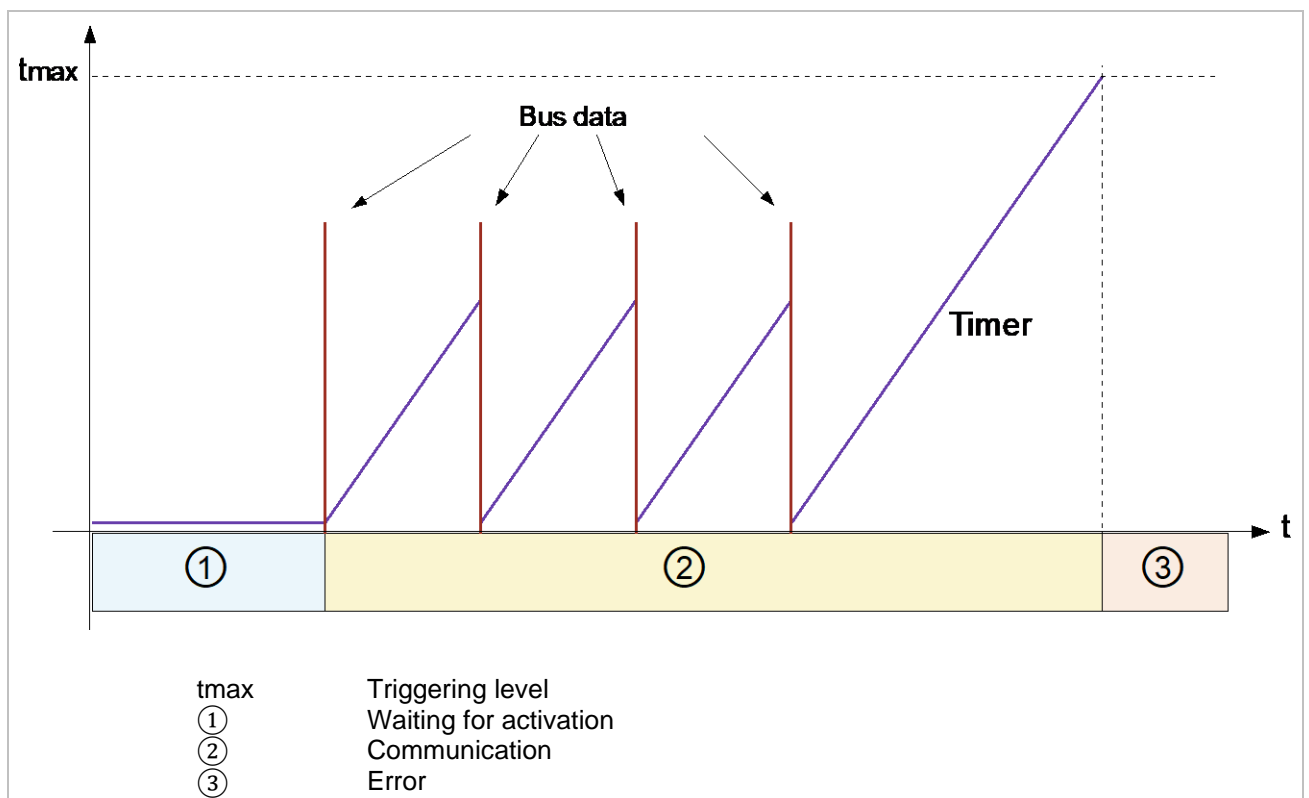


Figure 2: Fieldbus watchdog



## 6 CANopen Interface

### 6.1 Basics of the CAN BUS

Here we like to introduce the system of the CAN (Controller-Area-Network) BUS and also explain some terms that are frequently used in the following.



- On KEB devices the CANopen specification is the basis for all bus functionalities. The object dictionary as well as the mapping functionality are CANopen conform. The basis for the implementation of other bus systems is always CANopen.

The CAN is a multi master system, i.e. each user has access to the BUS and can send telegrams.

In order to avoid invalid conditions during simultaneous access of two users, the CAN-BUS knows a so-called arbitration phase, which defines the telegram beginning.

In case of access conflicts all users recognize during this arbitration, who sends the lowest telegram number (identifier).

Then this user can continue to send his telegram completely, without having to start from the beginning again.

Now all other (willing-to-send) users pass over into the receiving status and abort their telegram for the time being.

Thus it is specified that lower telegram numbers automatically have priority over higher numbers.

CAN telegrams can contain a maximum of 8 byte user data.

The term logical CAN master used in the following, refers to the CAN user, who is responsible for the control of the entire CAN system.

Even if there are physically only masters at CAN, in most applications there will be one or several users who exercise control.

In this combination the KEB frequency inverter is considered as recipient of orders (logical slave).

### 6.2 CAN Functions

The CAN protocol is uniformly standardized for the data backup layer.

This protocol is completely processed by a CAN master.

Furthermore, the CAN in Automation association (CiA) has passed a standard for the higher protocol layer that was named CAN Application Layer (CAL).

Based on this standard the „CAL-based Communication Profile“ (CiA,DS301) was published in September 1995. This standard is the basis for all CANopen device profiles.

In this standard, a certain subset of the CAL standards is selected. The communication profile defines, among other things, a minimum capability device.

That is the minimum required functionality, which a CANopen node must make available. The present CAN interface connection realizes such a minimum capability device.

### 6.3 Sync Identifier

cob-ID sync message		Var	0x1005
Bit	Value	Function	
0...10	128	Standard SYNC Identifier (non-adjustable)	
11...28	0	0	
29	0	F6 supports only value 0: 11-Bit CAN-ID	
30	0	F6 does not generate a SYNC message	
31	0	reserved	

### 6.4 Broadcast objects

Address	Broadcast object
0dec. (000h)	NMT
128dec. (080h)	SYNC

### 6.5 Communication objects

from address	to address	Communication object
129 (081h)	255 (0FFh)	EMCY
385 (181h)	511 (1FFh)	TPDO1
513 (201h)	639 (27Fh)	RPDO1
641 (281h)	767 (2FFh)	TPDO2
769 (301h)	895 (37Fh)	RPDO2
897 (381h)	1023 (3FFh)	TPDO3
1025 (401h)	1151 (47Fh)	RPDO3
1153 (481h)	1279 (4FFh)	TPDO4
1281 (501h)	1407 (57Fh)	RPDO4
1409 (581h)	1535 (5FFh)	SDO(tx)
1537 (601h)	1663 (67Fh)	SDO(rx)
1793 (701h)	1919 (77Fh)	NMT Error Control

## 6.6 Request/Response-Identifier (SDO)

Over the request-identifier any CAN node can request the reading or writing of a parameter value.

The response-identifier is reserved for the appropriate response of the frequency inverter.

The mechanism of request and response is also referred to as acknowledged service.

The CANopen communication profile combines these functions under the term Service Data Object (SDO):

$\text{SDO(rx)} = \text{Request-Identifier} = 1536 + \text{CAN Node ID} = 600\text{h} + \text{CAN Node ID}$

$\text{SDO(tx)} = \text{Response-Identifier} = 1408 + \text{CAN Node ID} = 580\text{h} + \text{CAN Node ID}$

The CAN Node ID at the request is the Node ID of the receiver, the CAN Node ID at the response is the Node ID of the transmitter.

With an SDO - read access, the length of the addressed parameter and the parameter value are returned.

With an SDO - write access always 4 bytes are transferred as parameter value. For parameters with smaller length the remaining bytes are set to 0. In these cases, pay attention to the position of the parameter value within the telegram.

Example:

A read/write request is sent to the bus node with the CAN node ID = 30: used identifier (Request-Identifier) = 1566(dec) = 61E (hex)

The bus node with the CAN Node ID = 30 answers with a read/write confirmation: used identifier (Response Identifier) = 1438(dec) = 59E (hex)



- Basically, the function of the SDO is completely sufficient, to control the KEB F6 drive converter via CAN.  
Each parameter value in the inverter can be changed or inquired herewith.

The SDO channel is continuous operated independent of the process data transfer.

### 6.7 Out/In-Identifier (PDO)

The CAN master can give the frequency inverter unaddressed and unconfirmed data via the Out identifier.

The identification Out is based on the data direction from master to slave.

The frequency inverter transfers new data unaddressed and unconfirmed to the CAN master via the In identifier.

This function is called Process-Data-Object (PDO) by the communication profile. The processing of process data occurs with the minimum cycle time of 500 µs per activated PDO. Four process data objects (PDOs) with two object parts Out/In are available at COMBI-VERT F6.

They are addressed as follows:

PDO1(rx)	=	Out-Identifier	=	200h + CAN Node ID
PDO1(tx)	=	IN-Identifier	=	180h + CAN Node ID
PDO2(rx)	=	Out-Identifier	=	300h + CAN Node ID
PDO2(tx)	=	IN-Identifier	=	280h + CAN Node ID
PDO3(rx)	=	Out-Identifier	=	400h + CAN Node ID
PDO3(tx)	=	IN-Identifier	=	380h + CAN Node ID
PDO4(rx)	=	Out-Identifier	=	500h + CAN Node ID
PDO4(tx)	=	IN-Identifier	=	480h + CAN Node ID

The four process data objects (PDO) can be assigned each with 8 bytes per direction. That means: up to eight objects with a total of 8 bytes can be transferred per PDO.

## 6.8 CANopen process data mapping

Address	Parameter		
0x1400	1st RPDO communication parameter	0x1800	1st TPDO communication parameter
0x1401	2nd RPDO communication parameter	0x1801	2nd TPDO communication parameter
0x1402	3rd RPDO communication parameter	0x1802	3rd TPDO communication parameter
0x1403	4th RPDO communication parameter	0x1803	4th TPDO communication parameter
0x1600	1st transmit PDO mapping	0x1A00	1st transmit PDO mapping
0x1601	2nd transmit PDO mapping	0x1A01	2nd transmit PDO mapping
0x1602	3rd transmit PDO mapping	0x1A02	3rd transmit PDO mapping
0x1603	4th transmit PDO mapping	0x1A03	4th transmit PDO mapping

The four process data objects (PDOs) can be assigned each with 8 bytes per direction via the mapping parameters.

The following combinations of objects with a certain data size are possible:

2x Long	1x Long, 2x Word	1x Long, 1x Word, 2x Byte
4x Word	3x Word, 2x Byte	2x Word, 4x Byte
8x Byte		

Both the data size of a parameter and the property "Available for process data" can be checked in the "Property Editor" in COMBIVIS 6.

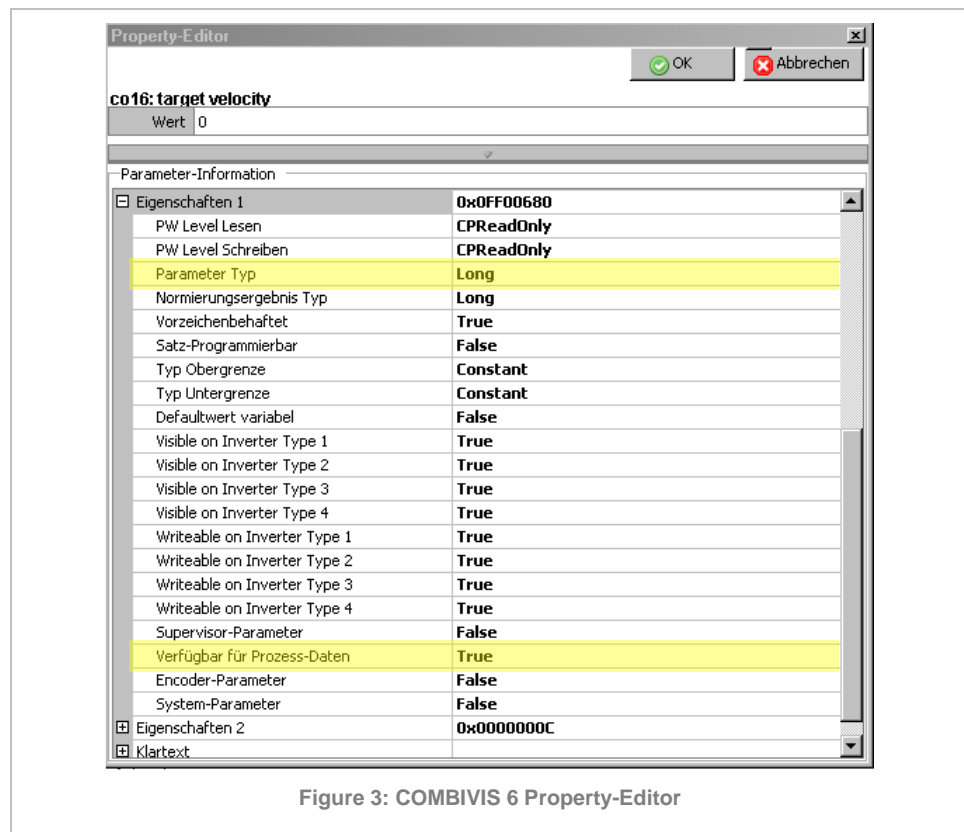


Figure 3: COMBIVIS 6 Property-Editor

The definition of the target for the data in the PDO(rx) telegrams respectively the source for the data in the PDO(tx) telegrams completely abides by the regulations of the CANopen communication profile.

Here a complex structured object (parameter) defines the PDO-mapping for each data direction.

Another object per data direction defines the communication definition (Communication parameter).

1st RPDO communication parameter	Struct	Default value:	00000200h + Node_Id	0x1400
2nd RPDO communication parameter			00000300h + Node_Id	0x1401
3rd RPDO communication parameter			00000400h + Node_Id	0x1402
4th RPDO communication parameter			00000500h + Node_Id	0x1403
Name cob-ID		Long		Subindex 1
Bit	Value	Function	Notice	
0...10		11-bit CAN-ID for the base frame of the CAN Bit 0...10 read-only	<p>Indicates to which identifier the RPDO for the transfer of the process output data is transmitted.</p> <p>Additionally there are control information for the RPDO contained in the highest bits. A changed value takes effect immediately and is stored non-volatile.</p> <p>When process data processing is switched on (bit 31 from "1" to "0"), process output data processing is switched on.</p> <p>Bit 0...28 indicate the RPDO identifier plus Node_Id and cannot be changed. During writing these bits are ignored.</p>	
11...28	0	29-bit-CAN-ID is not supported		
29	0	0: 11-Bit CAN-ID 1: 29-Bit CAN-ID		
30	0	Remote Frame on the appropriate identifier is responded		
	1	Remote frame is not answered		
31	0	The processing of the process output data is activated.		
	1	The processing of the process output data is deactivated.		
Name transmission type		Byte		Subindex 2
Value		Function	Notice	
0...240		The currently active process data out will be send to the controller after every value SYNC command (identifier = 128dec, data length = 0) is received.	<p>Defines, when and how this object is transmitted on the CAN bus.</p> <p>A changed value takes effect immediately and is stored non-volatile.</p> <p>! KEB inverters support this function only limited. Process data are no longer interpolated correctly with values unequal to 1, 254 and 255.</p>	
241...253		reserved		
254		(asynchronous, manufacturer-specific) The process output data are transferred to the FI control as soon as at least one byte has changed.		
255		(asynchronous, profile-specific) The process output data are transferred to the FI control in the fastest possible grid (minimum cycle time).		

1st receive PDO mapping 2nd receive PDO mapping 3rd receive PDO mapping 4th receive PDO mapping	Struct	0x1600 0x1601 0x1602 0x1603
Subindex 0: number (Byte)		
Value	Function	
0...8	Indicates the number of the mapped objects.	
Subindex 1...8 (Long)		
Bit	Function	Meaning
0...7	Object length	Byte: 0x08 Word: 0x10 Long: 0x20
8...15	Subindex	Subindex
16...31	Index	Parameter address (Bit 16...23 = Lowbyte/ Bit 24...31 = Highbyte)

Defines an object mapping. The index, subindex and the object length are specified in bits.	
Writing of subindex 1...8 requires that the count (subindex 0) is set to 0.	

1st TPDO communication parameter 2nd TPDO communication parameter 3rd TPDO communication parameter 4th TPDO communication parameter	Struct	Default value:	00000180h + Node_Id 00000280h + Node_Id 00000380h + Node_Id 00000480h + Node_Id	0x1800 0x1801 0x1802 0x1803
Name	cob-ID	Long	Subindex 1	
Bit	Value	Function	Notice	
0...10		11-bit CAN-ID for the base frame of the CAN Bit 0...10 read-only	Indicates on which identifier the TPDO for the transfer of the process input data is transferred. Additionally there are control information for the TPDO contained in the highest bits. A changed value takes effect immediately and is stored non-volatile. When process data processing is switched on (bit 31 from "1" to "0"), process output data processing is switched on. Bit 0...28 indicate the TPDO identifier plus Node_Id and cannot be changed. During writing these bits are ignored.	
11...28	0	29-bit-CAN-ID is not supported		
29	0	0: 11-Bit CAN-ID 1: 29-Bit CAN-ID		
30	0	Remote Frame on the appropriate identifier is responded		
	1	Remote frame is not answered		
31	0	The processing of the process input data is activated.		
	1	The processing of the process input data is switched off.		

1st TPDO communication parameter	Struct	Default value:	00000180h + Node_Id	0x1800
2nd TPDO communication parameter			00000280h + Node_Id	0x1801
3rd TPDO communication parameter			00000380h + Node_Id	0x1802
4th TPDO communication parameter			00000480h + Node_Id	0x1803
Name transmission type Byte Subindex 2				
Value	Function			Notice
0	(Synchronous, acyclic) At every receipt of a SYNC a TPDO telegram is transmitted on CAN.			Defines, when and how this object is transmitted on the CAN bus. A changed value becomes immediately active.
1...240	(Synchronous, cyclic) Number of SYNC telegrams before a TPDO telegram is sent on CAN. Value = 1 means that in the OPERATIONAL state a TPDO telegram is transmitted to CAN immediately after the receipt of a SYNC telegram. For all values 1...240 applies, the SYNC telegram triggers the transmission of the respective TPDO.			
252, 253	A TPDO telegram is only transmitted after a remote request (remote telegram enabled) on the TPDO identifier.			
254, 255	(event driven) A TPDO telegram is transmitted as soon as at least one byte has changed.			
Name inhibit time Word Subindex 3				
Value	Function		Notice	
0...65535	Value * 0.1 ms (e.g. 100 = 10ms)	Describes the minimal temporal distance between two CAN telegrams on this identifier.		

1st transmit PDO mapping	Struct	0x1A00
2nd transmit PDO mapping		0x1A01
3rd transmit PDO mapping		0x1A02
4th transmit PDO mapping		0x1A03
Subindex 0: number (Byte)		
Value	Function	
0...8	Indicates the number of the mapped objects.	
Subindex 1...8 (Long)		
Bit	Function	Meaning
0...7	Object length	Byte: 0x08 Word: 0x10 Long: 0x20
8...15	Subindex	Subindex
16...31	Index	Parameter address (Bit 16...23 = Lowbyte/ Bit 24...31 = Highbyte)

Defines an object mapping. The index, subindex and the object length are specified in bits.	
Writing of subindex 1...8 requires that the count (subindex 0) is set to 0.	



## 6.9 CANopen Bootup-Sequence

After the initialization phase the KEB CAN control changes automatically into status Pre-Operational.

In this status the communication by SDO(rx) and SDO(tx) with the services domain download (parameter write) and domain upload (parameter read) is already activated.

The process data communication is still inactive in this status.

It is released by the NMT command `Start_Remote_Node()` (=> picture below).

The target of this start sequence is the operating condition operational.

In this status the communication is completely activated.

With the NMT protocol certain CAN nodes are addressed by the CAN Node ID (=> Chapter 6.2 CAN Functions).

The F6 CANopen control board realizes the following transitions drawn with solid line in the following figure:

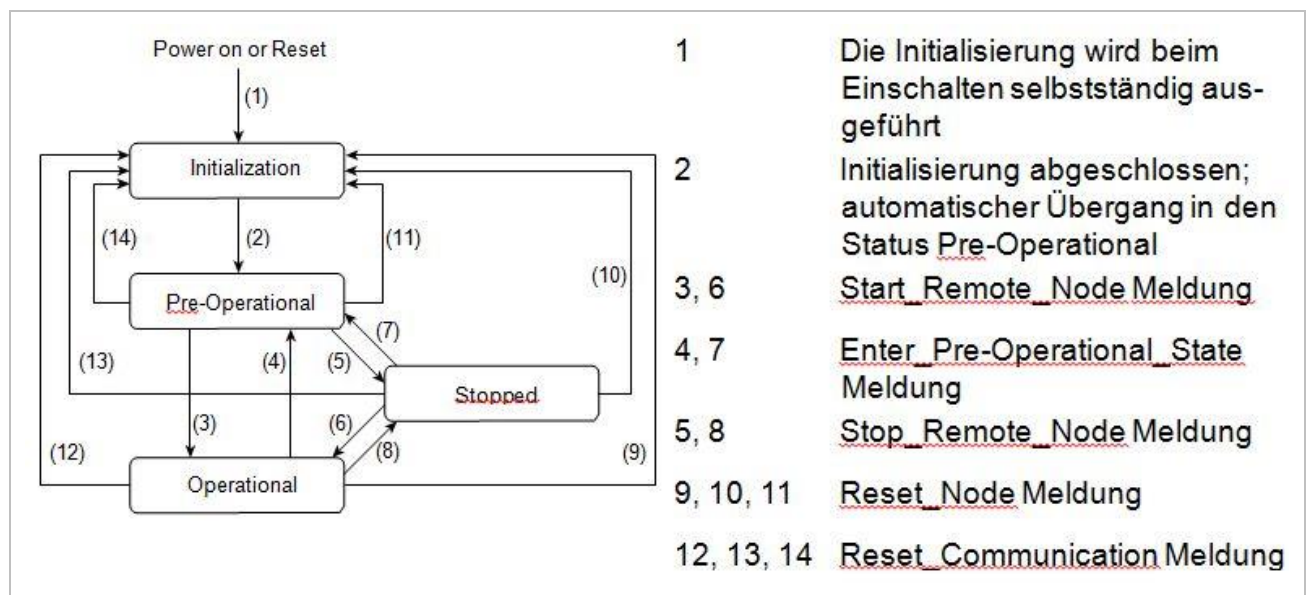


Figure 4: CANopen Bootup-Sequence

### Bootup message

The F6 CAN control board releases a bootup message, if the initialization phase is completed after power on. It is a telegram to identifier = 1792 + Node\_Id with the data length = 1 and the value = 0.

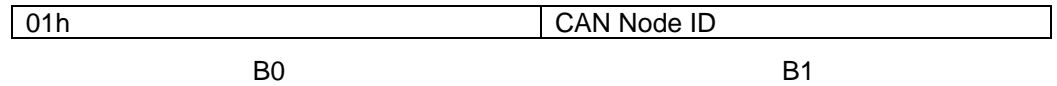
The transitions between the individual states are carried out by the following telegrams.

Node\_Id = 0 means: all NMT slaves are addressed.

Node\_Id = CANnode ID means: only 1 drive converter is addressed.

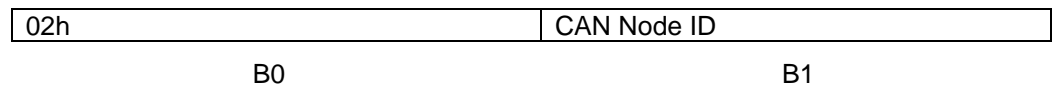
### Transition 3, 6 Start\_Remote\_Node()

CAN-Telegram: Identifier = 0



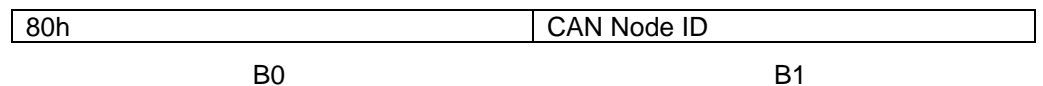
### Transition 5, 8 Stop\_Remote\_Node()

CAN-Telegram: Identifier = 0



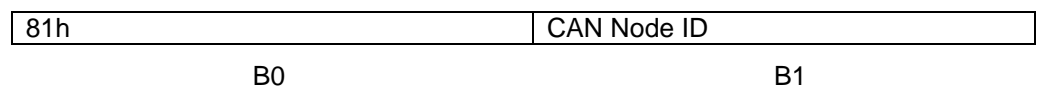
### Transition 4, 7 Enter\_Pre-Operational\_State ()

CAN-Telegram: Identifier = 0



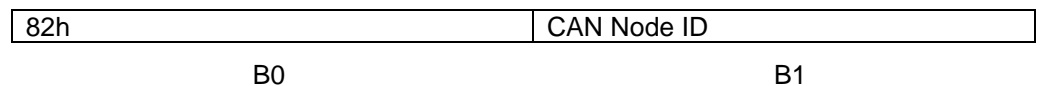
### Transition 9, 10, 11 Reset\_Node()

CAN-Telegram: Identifier = 0



### Transition 12, 13, 14 Reset\_Communication ()

CAN-Telegram: Identifier = 0



## 6.10 Node guarding

Node guarding belongs to the network management functionality (NMT) of the CAN node and describes a protocol whereby a CAN node can request the current status of any node.

Monitoring of the master only in conjunction with life guarding.

The F6 CAN control board supports the node guarding.

The node guarding request is deposited on the node guarding identifier by a remote frame.

The response arrives as data telegram with 1 byte data on the same identifier. The data byte contains the node status plus one toggle bit (MSBit).

The toggle bit is inverted from message to message.

Each node has its special node guarding identifier:

Node guarding identifier = 1792 + Node-Id

Value of the node status	Meaning
1	DISCONNECTED
2	CONNECTING
3	PREPARING
4	PREPARED (STOPPED)
5	OPERATIONAL
127dec.	PRE_OPERATIONAL

## 6.11 Life guarding

The F6 inverter supports life guarding.

It is a monitoring of the cyclic node guarding of the CAN master.

That means: life guarding should only be activated during cyclic node guarding.

The life guarding operates completely detached from all other monitoring functions.

It is activated by the product of the two parameter values guard time and life time factor.

The product specifies the life guarding timeout time.

Shows the product = 0, then the life guarding is not activated.

With activated life guarding the node guarding monitoring starts as soon as the first node guard request is received.

The function that is executed upon occurrence of the life guarding timeout can be adjusted via parameter 0x1029 error behaviour.

guard time		100Ch
Value		
0	Life guarding switched off	Defines the monitoring time for the life guarding together with the life time factor
1...65535	Time in ms	

life time factor		100Dh
Value		
0	Life guarding switched off	Defines the monitoring time for the life guarding together with the life time factor
1...255	Factor for the guard time	

error behaviour		Array	1029h
Subindex 1 (Byte)			
Value	Function		
0	Change into NMT status "Pre-Operational" (only at actual status "Operational")		
1	no change into NMT status		
2	Change into NMT status "Stopped"		

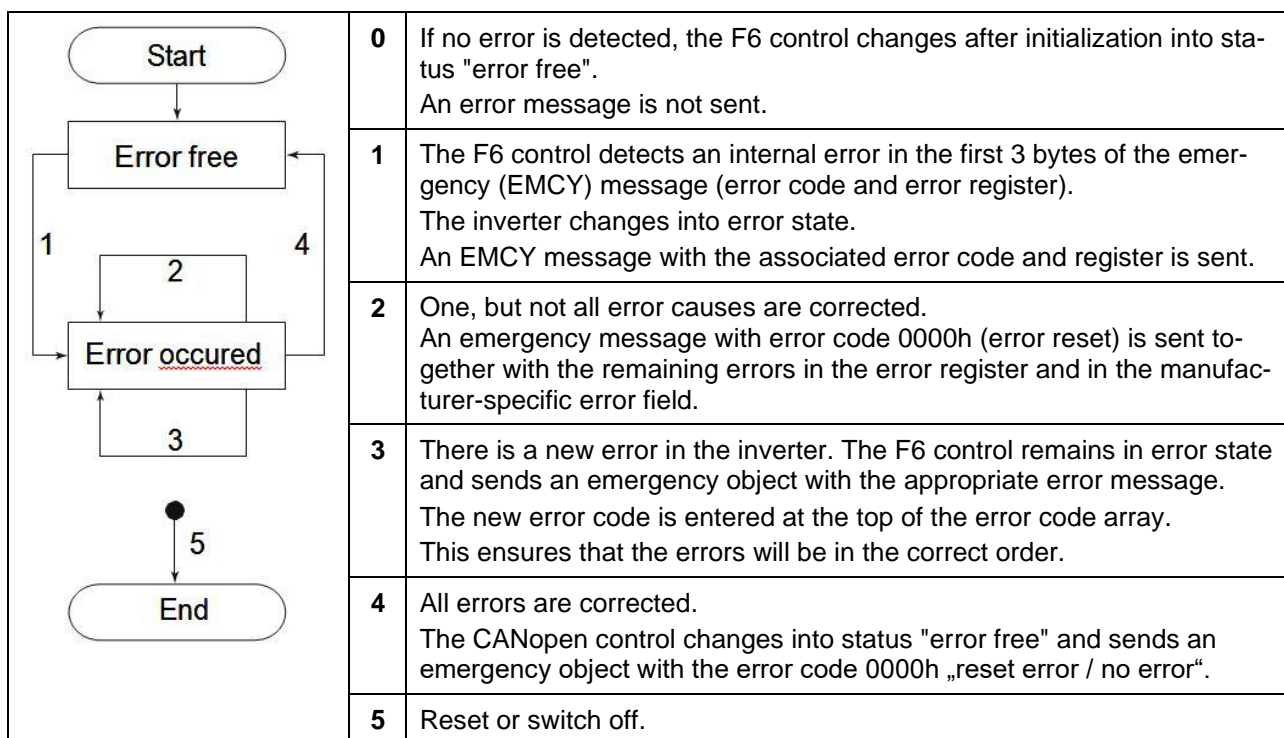
## 6.12 Emergency object

The CANopen communication profile DS301 defines a mechanism, after which the nodes signal independently, if the important events incidents.

This emergency message supports also the F6 CAN control.

If the status has changed an emergency message (EMCY) is send to identifier 128dec. + Node\_Id.

This means, also the transition from error state to normal operating conditions is announced by an EMCY message.



The content of the error telegram is only in part firmly set by the profile. The content of the F6 emergency message is as follows:  
Identifier = 128 + Node\_Id

B0		B1	B2	B3	B4	B5	B6	B7
Error code			Error register	Error field				
LB	HB	<a href="#">=&gt; Programming Manual   Control Application / Compact / Pro</a>	last error code	last but one error code	...	...		oldest error code
<a href="#">=&gt; Programming Manual   Control Application / Compact / Pro</a>								

All errors are stored in the error field defined by the profile.

This field contains a maximum of five entries in the F6 CAN control.

The first entry always contains the last error that occurred.

The error code in the emergency object is not completely identical with parameters 0x2101 / 0x603F error code. There are some CAN specific exceptions:

ru01	st01	CAN Error-Code
8: reset E. overload	1000h	FF01h
11: reset E. overheat pmod.	4210h	FF02h
13: reset E. overheat intern	4110h	FF03h
15: reset E. motorprotection	1000h	FF04h
17: reset ERROR drive overheat	4310h	FF05h
58: ERROR fieldbus watchdog	1000h	8100h

General meaning of the error code in the emergency object:

CAN Error-Code	Explanation
0000h	No error
1000h	General error
2200h	Current inside the CANopen device“ => Overcurrent power unit (ERROR overcurrent PU)
2300h	„Current, CANopen device output side“ => Overcurrent analog (ERROR overcurrent analog)
3210h	Overvoltage
3220h	Undervoltage
4210h	Overtemperature power semiconductors (heat sink)
4110h	Overtemperature interior
4310h	Temperature sensor in the motor (e.g. PTC or KTY) has triggered
8100h	General communication error
	KEB specific error (reset required)
FF01h	Reset overload error
FF02h	Reset power unit overtemperature
FF03h	Reset internal overtemperature
FF04h	Reset error motor protection
FF05h	Reset error motor overtemperature

### 6.13 Heartbeat

Index	Name	Function
0x1017	producer heartbeat time	Time between two transmitted heartbeat telegrams in ms
0x1016	consumer heartbeat time	Maximum time between two received heartbeat telegrams in ms and Node-Id of the producer
0x1029	error behaviour	Behaviour at triggering the heartbeat monitoring

The heartbeat protocol provides monitoring of the CAN bus without knowledge of the heartbeat producer via the connected user.

If the protection function responds, the behaviour can be defined with parameter 0x1029 [error behaviour](#).

<a href="#">error behaviour</a>		Array	1029h
Subindex 1 (Byte)			
Value	Function		
0	Change into NMT status "Pre-Operational" (only at actual status "Operational")		
1	no change into NMT status		
2	Change into NMT status "Stopped"		

#### Producers heartbeat

A heartbeat producer sends cyclically a heartbeat message.

The time when the message is sent, is adjustable (parameter 0x1017 producer heartbeat time).

The heartbeat protocol starts as soon as the producer heartbeat time is set.

If the producer heartbeat time already has a value unequal 0 when the device is switched on, the transmission of the heartbeat protocol starts during transition from initialization to Pre-Operational.

In this case, the bootup message is considered before the heartbeat message.

<a href="#">producer heartbeat time</a>		1017h
Value	Function	
0	Switches off the producer heartbeat.	
1...65536	Setting the producer heartbeat time: Time between two heartbeat telegrams in ms	

#### Consumers heartbeat

One or more heartbeat consumer receive the message.

The consumers must define a max. time in which a heartbeat message has to be receive.

If no message is received within this time, an adjustable behavior is started.

Heartbeat monitoring starts when the first heartbeat signal is received.

Consumer heartbeat time		Array	1016h
Subindex 1 (Long)			
Bit	Function	Meaning	
0...15	Setting of the monitoring time in ms.	If no heartbeat telegram is received during the monitoring time, the function set in parameter 0x1029 error behaviour is executed. Activation of the monitoring occurs with receipt of the first heartbeat telegram. A monitoring time of 0 ms switches off the function.	
16...23	Node-Id in the range of 0...255	The Node-ID of the heartbeat producer must be set here. A node address of 0 switches off the function.	
24...31	reserved		

**NOTICE**

- It is not possible to activate the guarding protocol and the heartbeat protocol simultaneously on one unit. The guarding protocol only works when the consumers heartbeat time is 0.

## 6.14 CAN-TelegramTypes

### 6.14.1 SDO-Telegrams

The logical CAN master can request (read) or change (write) the value of a parameter via the SDO telegrams.

In the communication profile a write-service is referred to as domain download and a read service as domain upload.

The KEB CAN interface connection supports only the short form of these two services. Only one telegram can be exchanged for the service request and another for the service acknowledgement between logical CAN master and F6 control.

The addressing of the parameter is done via unsigned 16 bit index and unsigned 8 bit subindex.

The manufacturer-specific parameters of the inverter are in the index range 2000h to 5EFFh.



- The CAN index is identical with the parameter address.

The subindex serves as additional addressing for complex parameters. The following applies:

Subindex	Type	Access to
0	Variable	Parameter value
	Array / Struct.	Subindex 0 (number)
1...n	Variable	not possible
	Field/structure	Subindex 1...n; Multiple selection not possible



### 6.14.2 SDO(rx)-Telegram

#### Initiate domain download request (write request of the master)

Bit	7...0	7...0	15...8	7...0	7...0	15...8	23...16	31...24
	0010nn1s	Index		Sub in- dex	Data			
		LB	HB		LSB			MSB
Byte	0	1	2	3	4	5	6	7

- nn: Only valid if s = 1  
Contains the number of bytes of the data field, that contains no data.
- s: 0 = no display of data length in nn.  
1 = see description nn
- Data: Data to be transmitted. The LS byte is transmitted first.

#### Initiate domain upload request (read request of the master)

Bit	7...0	7...0	15...8	7...0	7...0	15...8	23...16	31...24
	01000000	Index		Sub in- dex	reserved			
		LB	HB		LSB			MSB
Byte	0	1	2	3	4	5	6	7

### 6.14.3 SDO(tx)-Telegram

#### Initiate Domain Download Response (write confirmation from the inverter)

This response is transmitted when the requested write service could be executed error-free.

Bit	7...0	7...0	15...8	7...0	7...0	15...8	23...16	31...24
	01100000	Index		Sub in- dex	reserved			
		LB	HB		LSB			MSB
Byte	0	1	2	3	4	5	6	7

**Initiate domain upload response (read confirmation from the FI)**

This response is transmitted when the requested read service could be executed error-free.

Bit	7...0	7...0	15...8	7...0	7...0	15...8	23...16	31...24
	0100nn1s	Index		Sub in- dex	Data			
		LB	HB		LSB			MSB
Byte	0	1	2	3	4	5	6	7

nn: Only valid, if s = 1.  
Contains the number of bytes of the data field, that contains no data.

s: 0 = no display of data length in nn.  
1 = see description nn

Data: Data to be transmitted. The LS byte is transmitted first.

**Abort domain transfer (error response from the FI)**

This response is transmitted if the requested write or read service could not be executed. In this case an error description is returned.

Bit	7...0	7...0	15...8	7...0	7...0	15...8	7...0	7...0
	10000000	Index		Sub in- dex	Additional code		Error code	Error class
		LB	HB		LB	HB		
Byte	0	1	2	3	4	5	6	7

Error class	Error code	Additional code	Meaning
00	00	0000h	OK, no error
05	04	0001h	Service not supported
06	01	0000h	Invalid operation
06	01	0002h	Attempt to write to a read-only parameter
06	01	0010h	Invalid password
06	02	0000h	Invalid address
06	09	0011h	Subindex does not exist
06	09	0012h	Invalid language identifier
06	09	0030h	Invalid value for this parameter
08	00	0020h	Data can not be transferred or stored
08	00	0022h	Unit busy

#### 6.14.4 RPDO1...4 telegram

With this telegram the logical CAN master transfers new process output data to the inverter.

Up to 8 bytes of data can be transferred with each PDO in each data direction.

For configuration of the assignment => Chapter 6.7 Out/In-Identifier (PDO).

Byte	Process data							
	0	1	2	3	4	5	6	7

#### Example:

Mapping:

Index	Subidx	Value	Meaning	Function	Object
0x1600	0	2		Number of the mapped objects in Out PDO	
0x1600	1	2500 00 10h	Index = 0x2500 Subindex = 0 Length = Word	First object mapped in the process output data	co00 controlword
0x1600	2	2510 00 20h	Index = 0x2510 Subindex = 0 Length = Long	Second object mapped in the process output data	co16 Target Velocity
0x1600	3	0	not assigned	PDO contains no other objects	
0x1600	4	0			
0x1600	5	0			
0x1600	6	0			
0x1600	7	0			
0x1600	8	0			

Telegram:

Bit	7...0	15...8	7...0	15...8	23...16	31...24	7...0	15...8
	controlword		target velocity			free		
	LB	HB	LB			HB	LB	HB
Byte	0	1	2	3	4	5	6	7

### 6.14.5 TPDO1...4 telegram

With this telegram the inverter control announces process input data to the (logical) CAN master.

Up to 8 bytes of data can be transferred with each PDO in each data direction.

For configuration of the assignment => Chapter 6.7 Out/In-Identifier (PDO)

Byte	Process data							
	0	1	2	3	4	5	6	7

#### Example:

Mapping:

Index	Subidx	Value	Meaning	Function	Object
0x1A00	0	3		Number of the mapped objects in Out PDO	
0x1A00	1	2100 00 10h	Index = 0x2100 Subindex = 0 Length = Word	First object mapped in the process input data	st00 statusword
0x1A00	2	2C01 00 08h	Index = 0x2C01 Subindex = 0 Length = Byte	Second object mapped in the process input data	ru01 exception state
0x1A00	3	2120 00 20h	Index = 0x2120 Subindex = 0 Length = Long	Third object mapped in the process input data	st32 velocity ac- tual value
0x1A00	4	0	not assigned	PDO contains no other objects	
0x1A00	5	0			
0x1A00	6	0			
0x1A00	7	0			
0x1A00	8	0			

Telegram:

Bit	7...0	15...8	7...0	7...0	15...8	23...16	31...24	7...0
	statusword		exception state	velocity actual value				free
Byte	LB 0	HB 1	2	LB 3	4	5	HB 6	7

## 6.15 CAN bit timing

Regarding the adjusted bit timing the KEB CAN control adhere to the specifications of the CiA standard. The nominal bit timing is as follows:

Area for each segment: Bit time = 8  $T_q$  to 25  $T_q$

Bit time		
SYNC (1 $T_q$ )	TSEG1 (4...16 $T_q$ )	TSEG2 (2...8 $T_q$ )

Samplepoint



➤ SYNC: Only the edges from recessive to dominant are used for the synchronization.

Bit rate	Timequantum ( $t_q$ )	TSEG1	TSEG2	SJW
20 Kbit/s	4.975 $\mu$ s	6 $T_q$	3 $T_q$	1 $T_q$
25 Kbit/s	3.95 $\mu$ s	6 $T_q$	3 $T_q$	1 $T_q$
50 Kbit/s	1.975 $\mu$ s	6 $T_q$	3 $T_q$	1 $T_q$
100 Kbit/s	995 ns	6 $T_q$	3 $T_q$	1 $T_q$
125 Kbit/s	750 ns	6 $T_q$	3 $T_q$	1 $T_q$
250 Kbit/s	350 ns	6 $T_q$	3 $T_q$	1 $T_q$
500 Kbit/s	150 ns	6 $T_q$	3 $T_q$	1 $T_q$
1000 Kbit/s	50 ns	6 $T_q$	3 $T_q$	1 $T_q$

## 6.16 CAN diagnosis

General diagnostic parameters for fieldbuses and the flashing pattern of the fieldbus LED can be used for the diagnosis of the CANopen interface.

Index	Id-Text	Name	SubIdx	Function
0x2B1F	fb31	no PDO data per sync cnt	0	Number of synchronization intervals wherein no new process data was received
0x2B5A	fb90	fieldbus state (number)	0	
		CANopen fieldbus state	2	Value of the register CO_RunState
0x2B5B	fb91	fieldbus error code (number)	0	
		CANopen fieldbus error code	2	Value of the register EmergencyErrField
0x2B45	fb69	lost messages	0	Increased when received data is too long for the receive data buffer. Corresponds to the number of lost data in bits.

Further information about the general diagnostic parameters can be found in chapter 5.5.4 Check the fieldbus state and possible fieldbus error codes.

## 7 EtherCAT interface

For the successful start-up of a KEB-x6 EtherCAT slave, the master has two different possibilities to determine the device description according to the EtherCAT specification.

On the one hand there is a ESI file (EtherCAT slave information file). This is an XML-based device description file according to the EtherCAT standard. This file can be created amongst others with the COMBIVIS process data wizard for KEB x6 slaves. (=> Chapter 5.6.4 Fieldbus wizard)

Furthermore, the device description can be read from the contents of the EtherCAT EEPROM, the so-called SII (Slave Information Interface). This is made possible by a special protocol which uses special registers in the EtherCAT slave. It is even possible to change the EtherCAT EEPROM (SII) from the EtherCAT master by way.

Most EtherCAT masters use the ESI files for the configuration. However, in order to be able to create an assignment between ESI files and the slaves active on the BUS during so-called 'scanning' of the EtherCAT bus, for example, EtherCAT masters also read the values for VendorId and ProductCode from the slave via the SII protocol. Then the EtherCAT master can read all further configurations from the assigned ESI file.

KEB devices of different control types can not be operated with the same ESI file, because e.g. the mailbox sizes of the individual control types differ. (MBoxIn and MBoxOut are always identical).

ESI mailbox details	Control type A	Control type K	Control type P
MinSize	128	54	96
MaxSize	128	256	1024
DefaultSize	128	256	1024

Table 7-1: Differences between the mailbox sizes of the control types

The following EtherCAT specific objects are available for KEB devices.

## 7.1 Hardware and software version

Parameters [0x1009 Hardware Version](#) and [0x100A Software Version](#) are used for the versioning of the KEB devices. They are placed under the profile objects in the [pr](#) parameter group. Parameters [0x1009](#) and [0x100A](#) can currently only be read out via an EtherCAT master.

Parameter [0x1009 Hardware Version](#) is always set to value 1.

The main version of the currently used software is displayed in parameter [0x100A Software Version](#). This corresponds to the upper half of parameter [de16 : ctrl software version](#).

Index	Name	Function
0x1009	<a href="#">Hardware version</a>	No function, always 1
0x100A	<a href="#">Software version</a>	Main version of the software

## 7.2 Sync Manager

Parameter [sync manager com type](#) specifies the communication type of the used SyncManager. Each subindex represents a SyncManager.

A distinction is made between the following communication types:

- 1: Mailbox receive (master to slave)
- 2: Mailbox send (slave to master)
- 3: Processdata output (master to slave)
- 4: Processdata input (slave to master)

Index	Name	SubIdx	Function	Value
0x1C00	sync manager com type	0	Number of Sync Manager channels	4
		1	Communication type of SyncManager 0	1
		2	Communication type of SyncManager 1	2
		3	Communication type of SyncManager 2	3
		4	Communication type of SyncManager 3	4

Parameters [sync manager 2 PDO assign](#) and [sync manager 3 PDO assign](#) refer via indices to the mapping objects to which SyncManager 2 or SyncManager 3 refers.

Index	Name	SubIdx	Value	Function
0x1C12	sync manager 2 PDO assign	0	1...2	Number of available receive PDOs
		1	0x1600	Index of parameter <a href="#">1st receive PDO mapping</a>
		2	0x1601	Index of parameter <a href="#">2nd receive PDO mapping</a>
0x1C13	sync manager 3 PDO assign	0	1...2	Number of available transmit PDOs
		1	0x1A00	Index of parameter <a href="#">1st transmit PDO mapping</a>
		2	0x1A01	Index of parameter <a href="#">2nd transmit PDO mapping</a>



### 7.2.1 Sync Manager Parameters

The Sync Manager parameters specify information about the available and current state of the synchronization. There is one sync manager parameter object for the process output data (0x1C32 [Output sync manager para](#)) and one object for the process input data (0x1C33 [Input sync manager para](#)).

The Sync Manager parameters cannot be written via COMBIVIS. They only serve as display parameters for the current synchronization status.

The Sync Manager parameters cannot be used as process data. The only exceptions are the subobjects 0x1C32:20 [Sync Error](#) and 0x1C33:20 [Sync Error](#).

KEB devices support the following subobjects of the Sync Manager parameters:

Index	Name	Subindex	Name	Function
0x1C32	<a href="#">Output sync manager para</a>	1	Sync mode	Current synchronization mode
		2	Cycle Time	Current cycle time
		4	Sync modes supported	Supported synchronization modes
		5	MinCycleTime	Minimum supported cycle time ( <a href="#">is22</a> )
		6	Calc and Copy Time	Time measurement PdOut
		11	SM Event Missed	Error counter: missing SYNC events
		12	Cycle Time Too Small	Error counter: cycle time too small
		32	Sync Error	Error during synchronization

Index	Name	Subindex	Name	Function
0x1C33	<a href="#">Input sync manager para</a>	1	Sync mode	Same value as 0x1C32:01
		2	Cycle Time	Same value as 0x1C32:02
		4	Sync modes supported	Same value as 0x1C32:04
		5	MinCycleTime	Same value as 0x1C32:05
		6	Calc and Copy Time	Time reserve Pd-In
		11	SM Event Missed	Same value as 0x1C32:0B
		12	Cycle Time Too Small	Same value as 0x1C32:0C
		32	Sync Error	Same value as 0x1C32:20

The subobject [\[1\]Sync mode](#) is automatically set to Sync 0 synchronous if [\[2\]Cycle Time](#) is set to a valid value unequal 0. Restarting or writing an invalid cycle time resets [\[1\]Sync mode](#) to FreeRun.

The subobject [\[2\]Cycle Time](#) is automatically set by the EtherCAT Master. Valid values for [\[2\]Cycle Time](#) are integer multiples of [\[5\]Minimum Cycle Time](#). The value of [\[2\]Cycle Time](#) is used in synchronous mode for the synchronization of the KEB device. (=> Chapter 5.7.1 Checking the synchronization)

The subobject [\[4\]Sync modes supported](#) indicates all supported synchronization modes. For KEB devices the modes FreeRun, Sync and DC Sync 0 are supported.

Subobject [\[5\]Minimum Cycle Time](#) is updated when parameter [is22 Basic Tp](#) is written.

The subobject [\[6\]Calc and Copy Time](#) of the Output Sync Manager parameter displays the time difference between the SYNCN event and the earliest possible time when the process output data (master -> slave) become active in the KEB slave.

The subobject [\[6\]Calc and Copy Time](#) of the Input Sync Manager parameters displays the time reserve between the locking of the process input data (slave -> master) and the minimum cycle time.

The subobject [\[11\]SM-Event Missed](#) is an error counter. It is increased if an expected SYNC event does not receive the KEB slave.

The subobject [\[12\]Cycle Time Too Small](#) is an error counter. It is increased if a cycle time specified by the master is lower than [\[5\]Minimum Cycle Time](#).

The subobject [\[32\]Sync Error](#) indicates whether a synchronization error has occurred. A synchronization error occurs if the SYNC event fails for two cycles.

The subobjects [\[6\]Calc and Copy Time](#), [\[11\]SM-Event Missed](#) and [\[32\]Sync Error](#) are only used in synchronous mode.

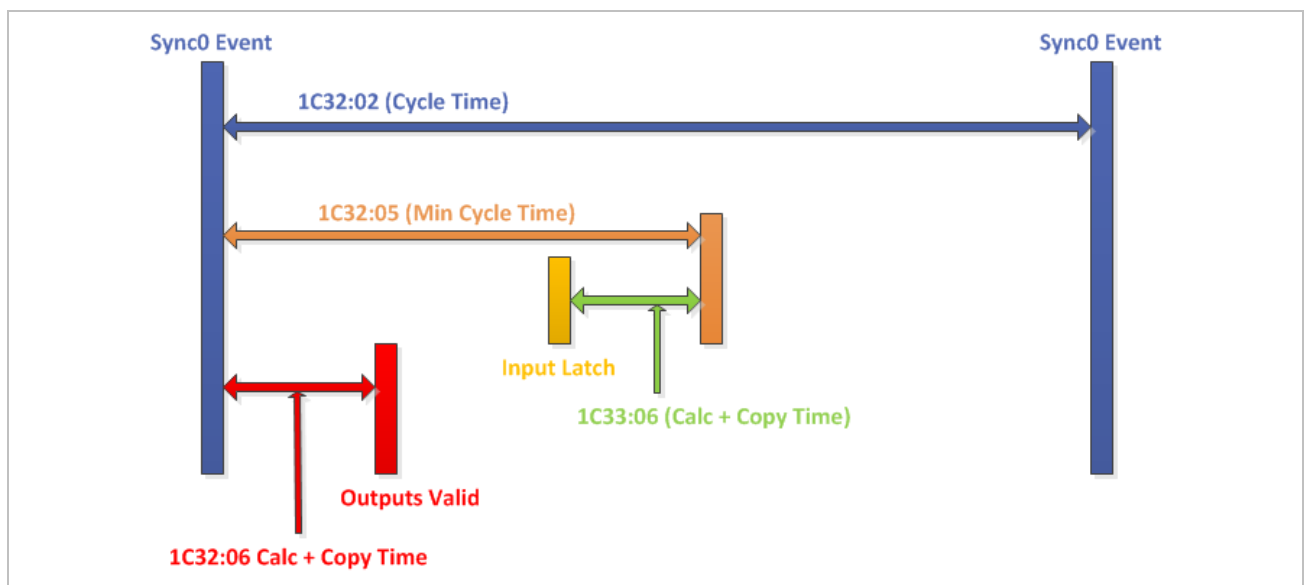


Figure 5: Sync Manager Parameter Calc and Copy Time

### 7.3 EtherCAT diagnosis and timing (control type K and P)

In order to facilitate the diagnosis of communication faults on the EtherCAT bus, the inverters of generation 6 offer a range of error counters and measured values for control types K and P. This includes:

Index	Id-Text	Name	Subldx	Function
0x2B5A	fb90	fieldbus state (number)	0	
		EtherCAT fieldbus state	1	Value of register AL status (register address 130h)
0x2B5B	fb91	fieldbus error code (number)	0	
		EtherCAT fieldbus error code	1	Value of register AL status code (register address 134h)

#### 7.3.1 Diagnostic cells of the EtherCAT core (hardware)

The following objects of the EtherCAT core are available:

Index	Id-Text	Name	Function
0x2B14	fb20	ETC invalid frame count P0	Counts the invalid RX frames at port 0
0x2B15	fb21	ETC RX error count P0	Counts the RX errors at port 0
0x2B16	fb22	ETC invalid frame count P1	Counts the invalid RX frames at port 1
0x2B17	fb23	ETC RX error count P1	Counts the RX errors at port 1
0x2B18	fb24	ETC forwarded RX error count P0	Counts the faulty transferred frames at port 0
0x2B19	fb25	ETC forwarded RX error count P1	Counts the faulty transferred frames at port 1
0x2B1A	fb26	ETC processing unit error count	Error counter of the processing unit

These objects represent direct images of the error counters integrated in the FPGA. A more detailed description of the functionality can be found in the instruction manual of the EtherCAT IP Core.

#### 7.3.2 Time measurement EtherCAT Frame <=> Sync pulse

In applications where drive control and motion control are operated synchronously, faults caused by non-synchronous data processing in the drive control or the control application are difficult to diagnose.

These problems are often caused by the fact that the processing of the EtherCAT frame and the access of the drive control to the data will be overlap.

By using the following two objects it is possible to check the temporal position of the frame processing relative to the sync pulse. This check can also be executed via the fieldbus wizard (=> chapter 7.3.4 **EtherCAT diagnosis assistant**).

Index	Id-Text	Name	Function
0x2B1B	fb27	ETC min. sync delay	Minimum time between EtherCAT frame and Sync pulse [us]
0x2B1C	fb28	ETC max. sync delay	Maximum time between EtherCAT frame and Sync pulse [us]

## 7.3.3 Application error counter

The following error counters are available on the application level:

Index	Id-Text	Name	Function
0x2B1D	fb29	ETC no frame per sync cnt	Number of synchronization intervals wherein no EtherCAT frame was received
0x2B1E	fb30	ETC multiple frames per sync cnt	Number of synchronization intervals wherein several EtherCAT frames were received
0x2B1F	fb31	no PDO data per sync cnt	Number of synchronization intervals wherein no new process data was received

## 7.3.4 EtherCAT diagnosis assistant

The temporal position of the frame processing relative to the sync pulse can also be diagnosed graphically with COMBIVIS. However, the graphic diagnosis requires that not more than one telegram is sent per sync pulse or that the device has already reached the synchronous state.

For safe synchronous operation, the time position of the frame processing must not overlap the range of the incoming process data telegrams. A shift of the process data telegrams can be achieved by setting a "User Time Shift" from the master or from the slave.

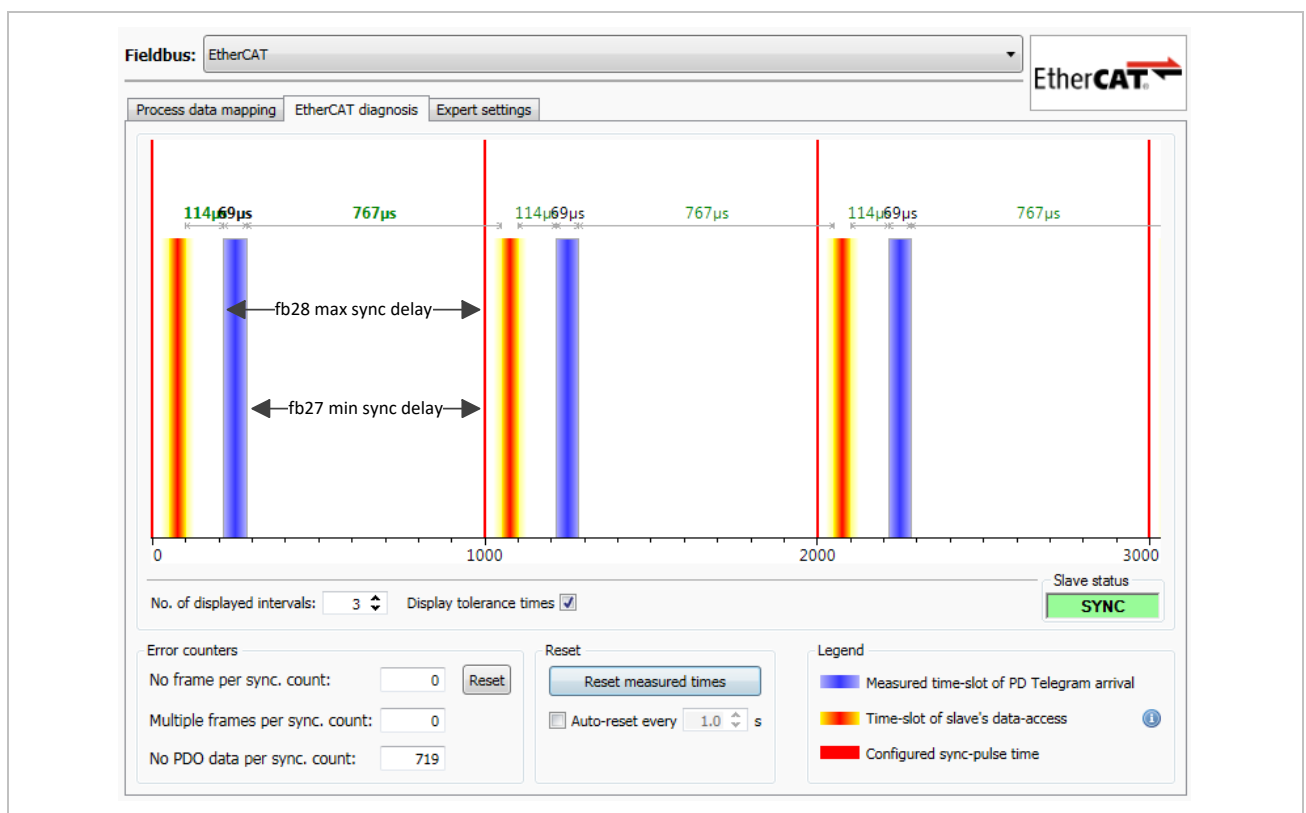


Figure 6: EtherCAT diagnosis assistant

## 7.4 EtherCAT diagnosis (control type A)

No additional parameters for the EtherCAT diagnosis are available for control type A. The general diagnostic parameters and the fieldbus LEDs should be used here. Only the following parameters can be used for diagnosis:

Index	Id-Text	Name	SubIdx	Function
0x2B1F	fb31	no PDO data per sync cnt	0	Number of synchronization intervals wherein no new process data was received
0x2B5A	fb90	fieldbus state (number)	0	
		EtherCAT fieldbus state	1	Value of register AL status (register address 130h)
0x2B5B	fb91	fieldbus error code (number)	0	
		EtherCAT fieldbus error code	1	Value of register AL status code (register address 134h)

A more detailed description of the above mentioned parameters and the flashing pattern of the fieldbus LED can be found in chapter 5.5 Check the fieldbus status.

## 7.5 Ethernet over EtherCAT (EoE)

EoE enables Ethernet frames to be fed into an EtherCAT network via an EoE-capable master. This allows participants of such a network to send and receive Ethernet frames.

From software version 2.4 KEB devices of control type A and control type K support the function Ethernet over EtherCAT (EoE). KEB devices of control type P also support EoE.

### 7.5.1 Start-up of Ethernet over EtherCAT

The function EoE is activated by selecting EtherCAT as active fieldbus system via parameter [fb68 fieldbus selection](#).

The default value for parameter [fb68](#) is EtherCAT.

A restart is required if the active fieldbus system is changed during runtime.

An EoE-capable master is required for EoE. The master serves as interface between the Ethernet network and the EtherCAT network.

EoE uses a digital MAC address which is specified by the master. This MAC address can be read out via COMBIVIS via parameter [fb106 MAC Address \(EthChannel\)](#) (A, K) or [fb105 MAC Address \(EoE Channel\)](#) (P) after it has been specified by the EtherCAT master.

#### NOTICE

- On the devices of control type A parameter [fb106 MAC Address \(EthChannel\)](#) is also used for the PROFINET Ethernet channel. Therefore, [fb106](#) is written with a default value when the device is started. This default value has **no** influence on the functions of the EoE channel.

#### NOTICE

- On the devices of control type P, the virtual MAC address is displayed via parameter [fb105 MAC Address \(EoE Channel\)](#). The Ethernet MAC address is always displayed in [fb106](#).

The IP configuration is also preset by the master. Parameter [fb108 Ethernet over fieldbus IP configuration](#) can be used to read out the IP configuration set by the master.

The preset parameters by the master are only stored volatile.



- With the setting of the IP configuration it must be observed that the transmitter of the Ethernet telegrams (e.g. PC with COMBIVIS) and the KEB inverter are **not** in the same subnet.
- Several configuration steps are required to configure a COMBIVIS PC and a KEB control for a functional EoE network. These are described in a FAQ. The FAQ can be found under the name "SetupEoEcommunication" on the KEB website.

Further information about parameter [fb68](#) can be found in chapter 5.3.1 Selection of the fieldbus [system](#) via [fb68](#) and information about the virtual MAC address and the IP configuration can be found in chapter 5.5. Check the fieldbus status.

## 7.5.2 COMBIVIS functions via Ethernet over EtherCAT

By EoE it is possible to operate COMBIVIS functionalities (parameterization, KEB FTP, etc.) via the fieldbus interface.

When accessing the file system via EoE it is not necessary to set parameter [de10 operator cfg data](#) SubIdx 9 [com mode](#) to 1.

Further information about [de10](#) can be found in the [Programming Manual | Control Application / Compact / Pro](#).

### 7.5.2.1 Functional limitations of Ethernet over EtherCAT

COMBIVIS functionalities which are operated via the fieldbus interface are not as performant as COMBIVIS functionalities which are operated via a serial interface.

Depending on the load of the device, the number of active connections and the number of nodes between EoE capable master and COMBIVIS user, the performance may vary.

If the COMBIVIS user waits too long for a response due to poor performance, COMBIVIS timeout monitoring is activated and the communication is terminated with an error message.

With the help of the following steps it is possible to improve the performance of the EoE channel in order to avoid interruption of the communication:

- A direct connection between the EoE capable master and the COMBIVIS user should be implemented.
- Not more than one COMBIVIS instance should be active at the same time.
- No files should be downloaded or uploaded via KEB FTP when the device is in active operation (active modulation, active encoder, ...).
- No other COMBIVIS functionality should be used when downloading or uploading files via KEB FTP.

If the steps described above are not feasible / not sufficient, it is also possible to increase the timeout time in COMBIVIS to avoid an interruption of the communication.

The timeout time can be increased under [Tools – Options – KEB Parameterization – Communication](#). Further information on the timeout monitoring can be taken from the COMBIVIS manual.

### 7.6 EtherCAT Hot-Connect

KEB devices of control types A, K and P support the function EtherCAT Hot-Connect via the station alias.

EtherCAT Hot-Connect via a DIP switch is not supported.

EtherCAT Fast Hot Connect is not supported.

EtherCAT Hot Connect is primarily a functionality of the EtherCAT master. For information about functionalities and limitations of EtherCAT Hot-Connect please contact the manufacturer of your EtherCAT master.



## 8 VARAN interface

### 8.1 Electronic nameplate

For unique identification of the bus user, the electronic nameplate is already stored in the SPI flash.

Parameter [de11](#) contains the VARAN license number.

Index	Id-Text	Name	Function
0x200B	<a href="#">de11</a>	<a href="#">VARAN licence number</a>	Unique VARAN licence number

Index	Name	Sub index	Name	Value	Function
0x1018	identity object	<a href="#">1</a>	vendor ID	26	Manufacturer identification
		<a href="#">2</a>	product code	1157	Device identification

### 8.2 Dual Port Memory

The entire bus is treated like a 4GB memory, a defined memory area is assigned to each client.

This allows the control CPU to access the clients with simple memory-write and memory-read commands.

This defined memory area addresses an area in a Dual Port Memory (DPM) which can also be accessed by the application layer of the control board firmware.

### 8.3 Available commands

Memory Read	Reads data from the memory of a bus participant. The command contains the start address and the number of bytes to read. Then the client responds with the requested data.
Memory Write	Writes data from the memory of a bus participant. The command contains the start address and the data to be written. The client sends an confirmation
Global Write	All bus participants are addressed simultaneously. This command is used to global reset of the bus participants and for transmission of SYNC

## 8.4 DPM mapping

The following shows the mapping of the DPM. Separated areas for the isochronous Objects (PDO) and the asynchronous Objects (SDO) are available. The byte order for all data objects is "least significant (LS) byte first".

Address dec.	hex	Length in Byte	Description	Access (*1)
0	0	32	Configurable Input-PDO Data (F6 -> VARAN control)	ro
32	0x20	32	Configurable Output-PDO Data (VARAN control -> F6)	rw
64	0x40	4	SDO Request Data	rw
68	0x44	2	SDO Request Index	rw
70	0x46	1	SDO Request Subindex (Format CANopen DS301)	rw
71	0x47	1	SDO Request Cmd/MsgID (*2)	rw
72	0x48	4	SDO Response Data	ro
76	0x4C	1	SDO Response ErrorCode (*3)	ro
77	0x4D	1	SDO Response Cmd/MsgID (*2)	ro

(\*1) ro – read only => read only access  
rw – read/write => read/write access

(\*2) Definition Cmd/MsgID

This byte is not identical to the command byte of the data link layer

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Command ID				Init Bit	Message ID		

Bit	Name	Function
0...2	Message ID	Telegram ID = Counter from 0 to 7 The response has the same ID as the request
3	Init Bit	The initialization bit is set to 1 in the initialization command with all message ID bits. It must be set once after each power-on. By way the inverter changes from state "init" into state "operational".
4...7	Command ID	valid values: 1: read 2: write 3 initialisation (Init bit and Message ID bits are also set)

(\*3) SDO Response ErrorCode

Value	Meaning
0	OK, no error
1	Device not ready
2	Invalid address or password
3	Invalid data
4	Parameter write protected
5	BCC error
6	Device busy
7	Service not supported
8	Invalid password
9	Telegram frame error
10	Transmission error
11	Invalid subindex
12	Invalid language
13	Invalid index
14	Invalid operation

## 8.5 Parameterization data (asynchronous objects)

Communication via parameterization data occurs as described in chapter 8.4 DPM mapping.

It is not necessary that the COMBIVERT F6 has synchronized to the fieldbus cycle.

## 8.6 Process data (isochronous objects)

There are 32 bytes of process data available in both directions.

The process data can be accessed via the addresses described in chapter 8.4 DPM mapping.

### 8.6.1 To activate the process data objects in the device, it is necessary to set the mapping of the process data by using the parameters defined in chapter 5.6.1 Mapping of the process data.

In addition the application layer of the F6 must be synchronized to the VARAN cycle.

## 8.7 VARAN diagnosis

The general diagnostic parameters and the fieldbus LED can be used for the diagnosis of the VARAN interface. The following parameters can be used for diagnosis:

Index	Id-Text	Name	SubIdx	Function
0x2B1F	fb31	no PDO data per sync cnt	0	Number of synchronization intervals wherein no new process data was received
0x2B5A	fb90	fieldbus state (number)	0	
		VARAN fieldbus state	2	Status display of the VARAN fieldbus system
0x2B5B	fb91	fieldbus error code (number)	0	
		VARAN fieldbus error code	2	Error code of the VARAN fieldbus system

A more detailed description of the above mentioned parameters and the flashing pattern of the fieldbus LED can be found in chapter 5.5 Check the fieldbus status.

## 9 PROFINET interface

### 9.1 PROFINET certificate

The KEB COMBIVERT S6A device has been successfully certified for PROFINET Conformance Class C. => [Certificat](#)

### 9.2 PROFINET device description (GSDML)

A device description file for KEB COMBIVERT S6A and F6A according to GSDML standard is available in the download area of the KEB homepage.

### 9.3 PROFINET functions

In the following only the essential information about PROFINET functionalities are listed, which are required for the operation of a KEB x6 device with PROFINET. A basic understanding of how the PROFINET bus system works is required for this chapter.

Information about the PROFINET bus system can be obtained from the PROFIBUS User Organization e.V. (PNO).

#### 9.3.1 Cyclic data exchange (process data communication)

KEB COMBIVERT S6A can receive and process up to 32-byte process output data per cycle from the controller. The device can also provide up to 32 bytes of process input data per cycle to the master. This applies to both the non-synchronous operating mode (PROFINET IO-RT) and the synchronous operating mode (PROFINET IO-IRT). The minimum cycle time is 1ms.

#### 9.3.2 Acyclic data exchange (parameter - channel) to PROFIdrive

KEB COMBIVERT S6A and F6A support the coding of the acyclic services according to PROFIdrive profile also called Base Mode Parameter Access. However, this only applies to the transport of the data, not to its content or coding. This means, the PROFINET interface connection does not support parameters according to the PROFIdrive profile, but only the transport mechanism.

The acyclic parameter request according to Profidrive provides a mechanism via a list of up to 39 parameters can be written or read by means of two PROFINET record accesses. The basic procedure at PROFINET level is as follows:

1. Write-Record-Request of the master with index = B02Eh, net data = parameter request (see chapter 9.3.2.1 Parameter request)
2. Write response of the slave, no net data
3. Read-record-request of the master with index = B02Eh, no net data
4. Read-response of the slave, net data = parameter response (see chapter 9.3.2.2 Parameter response)

For the revision the network byte order (MSByte first) is valid.

## 9.3.2.1 Parameter request

Parameter request:			
Block definition	Byte n	Byte n+1	n
Request Header	Request Reference	Request ID	0
	Axis-No.	No. of parameters = n	2
1st parameter address	Attribute	No. of elements	4
	Parameter Number (PNU)		6
	Subindex		8
:	:	:	:
n th parameter address	Attribute	No. of elements	4 + 6 x (n-1)
	Parameter number (PNU)		
	Subindex		
1st parameter value(s)	Format	No. of values	4 + 6 x n
	Values		
:	:	:	:
n th parameter value(s)			

The [request reference](#) is a sequential number for distinction of different requests. (Values: 0x01 ... 0xFF)

**Request IDs:**

Value	Meaning
0x01	Request parameter (read)
0x02	Change parameter (write)

The parameter values are only contained in a request with request ID = 2.

The [Axis-No.](#) always corresponds to value 0, since PROFINET represents a 1:1 gateway for the x6 devices.

Only a maximum of 39 parameters can be read / written via one request. Therefore, a minimum of 1 and a maximum of 39 can be specified for the [No. of parameters](#).

**Attribute:**

Value	Meaning
0x10	Value
0x20	Description (not supported here)
0x30	Text (not supported here)

The [No. of elements](#) as well as the [No. of values](#) always correspond to value 1.

The required object for writing / reading is addressed via [Parameter number \(PNU\)](#) and [Subindex](#). The addressing corresponds to the addressing in the KEB object directory. The [Parameter number \(PNU\)](#) corresponds to the KEB parameter address and [Subindex](#) to the KEB subindex.

**Format:**

Value	Meaning
1	Boolean
2	Integer8
3	Integer16
4	<b>Integer32</b>
5	Unsigned8
6	Unsigned16
7	Unsigned32
0x41	Byte
0x42	Word
0x43	DWord
0x44	Error

When reading all KEB parameters on the x6 the format = 4 is returned. Accordingly, the PROFINET master can also specify the format = 4 when writing all parameters.

## 9.3.2.2 Parameter response

Parameter response:			
Block definition	Byte n	Byte n+1	n
Response Header	Request Reference mirrored	Response ID	0
	Axis-No. mirrored	No. of parameters = n	2
1st parameter value(s)	Format	No. of values	4 + 6 x n
	Values		
:	:	:	:
n th parameter value(s)			

**Response IDs:**

Value	Meaning
0x01	Request parameter ok (read)
0x02	Change parameter ok (write)
0x81	Request parameter with error (read)
0x82	Change parameter with error (write)

If an error occurred during read access to a parameter, the **Response ID** = 0x81 is set. Furthermore, the **format** = 0x44 (Error) is set for this parameter value and the error value(**Value**) is set as a 16-bit value as follows:

Value	Description
0x11	Timeout when accessing the parameter or drive converter busy
0x14	Drive converter busy
0x00	Invalid parameter address or password
0x17	Data invalid
0x65	Error in internal communication (BCC error)
0x66	Error in internal communication (invalid service or invalid operation)
0x67	Invalid password
0x68	Error in internal communication (invalid telegram)
0x69	Error in internal communication (parity error)
0x03	Invalid parameter set (subindex)
0x6B	Error in internal communication (invalid operation)

The parameter values are only contained in the response if **Response ID** = 1 or 0x81 or 0x82. Error-free accesses are listed in a negative write response (**Response-ID** = 0x82) with **Format** = 0x40 (Zero) and **No of Values** = 0.

#### 9.3.2.3 Error codes of the acyclic communication

The error code returned for acyclic communication is a 32-bit value composed of four components: ErrorCode, ErrorDecode, ErrorCode1, ErrorCode2. Currently, the application of the x6-PROFINET device does not generate its own error codes. All values != 0 are always generated by the used PROFINET stack. The meaning of these error codes can be found in the PROFINET specification.

#### 9.3.3 Additional diagnostic channel via standard Ethernet communication

All Ethernet communication which is not assigned to the PROFINET communication is transferred to the standard Ethernet channel of the KEB COMBIVERT S6A. Thus it is possible

Parameter to read/write (KEB COMBIVIS via Ethernet)

to download or upload files from the S6A internal file system (KEB FTP via Ethernet)

The exact specifications of the two options are shown in the following table:

Identification	Ethernet communication layer	Parameter	Protocol on the user data
KEB COMBIVIS via Ethernet	IPv4/UDP	UDP-Port = 8000	KEB DIN66019II
KEB FTP via Ethernet	IPv4/UDP	UDP-Port = 8002	KEB FTP



- The same restrictions apply to the standard Ethernet channel for PROFINET as to the Ethernet over EtherCAT channel. (=> Chapter 7.4. EtherCAT diagnosis (control type A)).



## 9.4 PROFINET diagnosis

See chapter 5 in this document.

### 9.4.1 PROFINET MAC addresses

PROFINET has a special position with regard to the demand for MAC addresses among the Ethernet-based fieldbuses.

For PROFINET operation, a PROFINET device with 2 ports requires three MAC addresses. One for the interface(IF), one for Port1 and another for Port2. If you also want to use the standard Ethernet channel via the PROFINET ports, an additional MAC address and an additional IP configuration are required for KEB-x6.

The MAC addresses **fb103** to **fb105** are assigned to each device in the production process by a protected mechanism. The MAC address **fb106** is reserved in the production process for each KEB device. This is a virtual MAC address which is set to the value of **fb103** + 3 when the device is started.

Index	Id-Text	Name	Function
0x2B67	<b>fb103</b>	<b>MAC Address (Base)</b>	The basis MAC address. Required for all Ethernet based fieldbus systems. (except EtherCAT)
0x2B68	<b>fb104</b>	<b>MAC Address (Port1)</b>	Only required for PROFINET. Own MAC address for Port1
0x2B69	<b>fb105</b>	<b>MAC Address (Port2)</b>	Only required for PROFINET. Own MAC address for Port2
0x2B6A	<b>fb106</b>	<b>MAC Address (EthChannel)</b>	Virtual MAC address for the Ethernet channel. Required for EoE, Ethernet over PROFINET and the Ethernet channel on the cards of control type P.

### 9.4.2 PROFINET device name

The PROFINET device name is used to simplify the identification of the device for the user.

The KEB default value for the PROFINET device name is `kebx6-NodeId`. `x6` stands for device type (`S6` or `F6`). `NodeId` corresponds to the value of the effective node address value (`fb102`).

If the node address value is smaller than `240`, the device name is reset to the default value described above when the device is restarted.

If the effective node address value is `241`, `242` or `254`, the device name is completely deleted when the device is restarted.

For all other values the PROFINET device name is not changed. The device name can be changed via a PROFINET controller (e.g. the TIA portal) or via COMBIVIS.

When the device name is set by a PROFINET controller, the default node address value (`fb101`) is automatically set to 240 to avoid resetting the device name.

When setting the device name by a PROFINET controller, the *remanent* attribute must be observed. Only if the *remanent* attribute is set to TRUE, the device name is also stored non-volatile.

The following rules apply for setting the PROFINET device name:

- Name parts are separated by dots
- The total length is between 1 and 240 characters
- A name part has a total length of 1 to 63 characters
- Name parts consist exclusively of lowercase letters, numbers and the hyphen
- Neither the first nor the last character of a name part is a hyphen



- The PROFINET device name is updated in the COMBIVIS view only after a restart of the device. Internally, however, the new name is active.
- If PROFINET is not the active fieldbus system, the PROFINET device name is also not initialized. The displayed default value is `kebx6-1`.

Further information on the node address value can be found in chapter 5.3.4 Selection of the node address of the fieldbus system.

Index	Id-Text	Name	Function
0x2B6B	<code>fb107</code>	<code>PROFINET NameOfStation</code>	Array for the display of the PROFINET device name

## 10 POWERLINK interface

### 10.1 Basics of the POWERLINK BUS

POWERLINK is an open real-time Ethernet protocol.

The master is referred to as Managing Node (MN) in a POWERLINK network. A slave is called Controlled Node (CN).

POWERLINK networks operate according to the so-called "polling" approach. The MN controls access to the network. CNs only send on request of the MN.

A POWERLINK cycle is divided into an isochronous phase and an asynchronous phase. In the isochronous phase, process data (PDOs) are transmitted cyclically. The asynchronous phase allows the exchange of non-cyclic data (SDOs).

The request of the MN to a CN is referred to in the isochronous phase as Poll Request (PReq) and in the asynchronous phase as Start of Asynchronous (SoA). The response of a CN is called Poll Response (PRes) in the isochronous phase and Asynchronous Send (ASnd) in the asynchronous phase.

The following characteristics apply to KEB CNs:

- A network can contain up to 253 CNs.
- The minimum cycle time is 400µs.
- The maximum process data size is 32 bytes per cycle for process input and process output data.
- A maximum of 8 bytes of data can be transmitted asynchronously per cycle and client.
- The process data image is dynamic and is stored **volatile**.
- Hot plugging is supported.
- Multiplexing, direct cross-communication and Poll Response chaining are **not** supported.
- The KEB CN can **not** be used as POWERLINK router.

Further information about the POWERLINK bus system can be found on the Ethernet POWERLINK Standardization Group website. (=> [Ethernet Powerlink - EPSG | Ethernet Powerlink](#)).

## 10.2 POWERLINK functions

The following functions are POWERLINK specific functions which are only available on devices of control type A.

If a bus system other than POWERLINK is active, these functions are not available and the parameters described in this chapter have no effect.

### 10.2.1 Variable process data offset

The offset of the individual process data objects can be freely selected in a POWERLINK network. The process data offset is specified in bits.

KEB CNs only support whole multiples of bytes as values for the process data offset. Other values are rejected.

The mapping must be deactivated to change the process data offset.

By default, the process data offset of a parameter corresponds to the sum of the length of all preceding parameters. When downloading a complete list, the process data offset must be adjusted subsequently if it does not correspond to this default.

Information about the fieldbus-independent mapping parameters can be found in chapter 5.6. Checking the process data mapping .

Index	Id-Text	Name	Function
0x2B6F	fb111	POWERLINK RPDO offset	Array with 8 elements, enables freely selectable process data offset for receive PDO mapping
0x2B70	fb112	POWERLINK TPDO offset	Array with 8 elements, enables freely selectable process data offset for transmit PDO mapping

### NOTICE

- Freely selected offsets can lead to gaps and/or overlaps between the individual process data objects.

### 10.2.2 Mapping version

The mapping version can be used to uniquely identify different process data mappings. Each mapping is assigned to a mapping version. CNs reject incoming process data with a wrong mapping version. The mapping version is updated by activating the process data mapping.

Index	SubIdx	Name	Function
0x1400	0	PDO_RxCommParam	
	1	Node ID (Source of Data)	No function (cross-traffic is not supported)
	2	Mapping version	Version number, incoming process data with a different version number are discarded

Index	SubIdx	Name	Function
0x1800	0	PDO_TxCommParam	
	1	Node ID (Source of Data)	Not used by CNs (always 0)
	2	Mapping version	Version number, is sent with process data



- The POWERLINK objects 0x1400 and 0x1800 can not be read out via the diagnostic interface. The pr parameters [RPDO/TPDO communication parameter](#) which can be read out via the diagnostic interface are CANopen parameters.

### 10.3 POWERLINK diagnosis

The general diagnostic parameters and the fieldbus LED are available for POWERLINK diagnosis.

Index	Id-Text	Name	SubIdx	Function
0x2B1F	<a href="#">fb31</a>	<a href="#">no PDO data per sync cnt</a>	0	Number of synchronization intervals wherein no process data was received
0x2B5A	<a href="#">fb90</a>	<a href="#">fieldbus state (number)</a>	0	Number of supported bus systems
		<a href="#">POWERLINK fieldbus state</a>	4	Value of the register NmtState (EPSP DS301 state machine)
0x2B5B	<a href="#">fb91</a>	<a href="#">fieldbus error code (number)</a>	0	Number of supported bus systems
		<a href="#">POWERLINK fieldbus error code</a>	4	Value of the register EplErrorCode (EPSP DS301 error code)

A more detailed description of the above mentioned parameters and the flashing pattern of the fieldbus LED can be found in chapter 5.5

#### 10.3.1 POWERLINK MAC address

The MAC address supported by POWERLINK is displayed in parameter [fb103](#). The MAC addresses displayed in [fb104](#) to [fb106](#) are not used.

Index	Id-Text	Name	Function
0x2B67	<a href="#">fb103</a>	<a href="#">MAC Address (Base)</a>	Basis MAC address.

### 10.3.2 POWERLINK IP configuration

The POWERLINK specification prescribes a fixed IP configuration. The IP address corresponds to 192.168.100.Node ID. The subnet mask corresponds to 255.255.255.0. The default value of the gateway address corresponds to 192.168.100.254.

The value of the NodeID, also called node address value in this document, can be read out via parameter [fb102](#). The values 0, 240 and 255 are invalid for a POWERLINK CN.

The IP configuration is updated at each restart depending on the node address. The node address is displayed in parameter [fb102](#).

The IP configuration can not be changed by the user, with the exception of the gateway address. Values that do not correspond to the configuration specified by [fb102](#) are rejected. The user can specify any gateway address, which is stored non-volatile.

Further information about the IP configuration and the node address can be found in chapter 5.5 Check the fieldbus status.

## 10.4 Configuration of the KEB POWERLINK CN

A tool for configuring the POWERLINK master is required to configure a POWERLINK CN. (e.g.: the B&R Tool Automation Studio (AS))

A device description file (XDD) is required to import a KEB CN into AS. The XDD of the KEB CN can be generated via the COMBIVIS fieldbus wizard. (=> Chapter 5.6.4 Fieldbus wizard)



- In contrast to other device description files (e.g. EtherCAT ESI file) the XDD file generated by COMBIVIS does not contain the preset mapping.
- The mapping must be set in the POWERLINK Master configuration tool after the XDD file has been imported.

Not all parameters of the KEB CN are displayed in the AS interface. Parameters that are not writable ([AccessType = ro](#)) and cannot be used as process data ([PDOMapping = no](#)) are not displayed in the AS interface.

A complete overview of all parameters supported by the CN is provided by the device description file or the interface of the KEB configuration tool COMBIVIS.

## 11 EtherNet/IP™ interface

EtherNet/IP™ is a communication protocol which is mainly used in industrial applications. It enables the exchange of information between devices. EtherNet/IP™ is based on standardized Ethernet and TCP/IP technology. It is used to transport communication packets based on CIP (Common Industrial Protocol). The EtherNet/IP™ protocol and the CIP are managed by the ODVA (Open DeviceNet Vendors Association).

### 11.1 Data transmission

EtherNet/IP™ uses a producer/consumer model to exchange time-critical information. The transmitter (producer) provides information for the recipient (consumer). This procedure enables the producer to provide the data to several consumers without a need to transmit the data multiple times.

EtherNet/IP™ uses the standard of the IEEE 802.3 specification. In order to achieve the best possible deterministic behavior, it is recommended to use commercial switch technology with 100 Mbps bandwidth and full duplex cables.

#### 11.1.1 Explicit messaging (parameterization channel)

The "explicit messaging" enables the access of the master (scanner) to read or write to each parameter in the KEB drive converter. This function can also be used to execute special management functions which are referred to as services. The available services are:

Service code	Service name	Service description
0x01	Get_Attributes_All	Read all attributes of an object. (not supported by all classes)
0x0E	Get_Attribute_Single	Read one attribute of an object.
0x10	Set_Attribute_Single	Write one attribute of an object.
0x05	Reset	Reset an object. (Only supported by objects of class 0x01).

With each message, the user must define the type of service as well as the class, instance, and the attribute of the object. All KEB objects are compliant with the CiA DS301 object specification. Each parameter is defined by an index and a sub-index.

##### 11.1.1.1 Class

The class determines the type of the object to be accessed. In this case, the value 0x64 or 100 decimal describes the parameter objects.

##### 11.1.1.2 Instance

The instance represents the index and the address of the parameter in the inverter. Further information about the parameter index can be found in the application description.

##### 11.1.1.3 Attribute

The attribute corresponds to the sub-index of the parameter with an offset of 0x64 or 100 decimal.



#### 11.1.1.4 Example READ

As an example, the system time is read by the inverter here. The parameter is [ru53: system time](#) with the index 0x2C35 or 11317 decimal. The sub-index is 0x00 or 0 decimal plus an offset of 0x64 or 100 decimal. `get_attribute_single` is used as service.

Service: 0x0E (Get\_Attribute\_Single)

Class: 0x64

Instance: 0x2C35

Attribute: 0x64

#### 11.1.1.5 Example WRITE

As an example for the write access, parameter [vl05 velocity max](#) is written with value 1000. The index of parameter is 0x2305 or 8965 decimal. The sub index corresponds to 0x00 or 0 decimal plus the offset mentioned above. `set_attribute_single` is used as service.

Service: 0x10 (Set\_Attribute\_Single)

Class: 0x64

Instance: 0x2305

Attribute: 0x64

Data: 0x03E8 or 1000 decimal

### 11.1.2 Implicit Messaging (process data communication)

"Implicit Messaging" enables a dedicated communication between producers and consumers. Application-specific data can be exchanged via this communication path. This is also called I/O connection.

This document uses the terms process output data (PD-Out) and process input data (PD-In) to describe the direction of the process data. PD-Out and PD-In describe the process data from the perspective of the control.

PD-Out refers to the data which are transmitted from the control (scanner) to the inverter (adapter).

PD-In refers to the data which are transmitted from the inverter (adapter) to the control (scanner).

The inverter supports different combinations (assemblies) of the process data for the communication. The combinations 100 and 101 are special, manufacturer-specific combinations of KEB for controlling the inverter based on the CiA402 application profile. The combinations 20 and 70 are used for the "basic speed control", which is based on the CIP provided by ODVA.

## 11.1.2.1 Assembly parameters

The different "assemblies" that are available on KEB devices are described below. The selection of the active "Assembly", as well as the selection of the IP configuration method, is made via parameter [fb113 EtherNet/IP Configuration](#).

Further information on the IP configuration can be found in chapter 5.5.1 Checking MAC addresses and IP configuration.

fb113	EtherNet/IP Configuration				0x2B71
Bit	Function	Value	Plaintext	Description	
0...1	IP configuration method	0	Static	IP configuration is static	
		1	Bootp	IP configuration is set via Bootp	
		2	DHCP	IP configuration is set via DHCP	
3	Assembly Instances	0	Basic Speed Control 20/70	KEB specific	
		4	KEB Control 100/101	CIP Basic Speed Control	

## 11.1.2.1.1 KEB Control 100/101

Parameters [co00 controlword](#) and [st00 statusword](#) are required for the KEB specific assemblies 100 and 101. The bits of the [controlword](#) and [statusword](#) parameters are shown below. A more detailed description can be found in the [Programming Manual | Control Application / Compact / Pro](#).

co00	controlword				0x2500
Bit	Function	Value	Plaintext	Description	
x	None	0	empty controlword	Waiting for control command	
0	Switch on	1	switch on	State change to switch on in state machine	
1	Enable Voltage	2	enable voltage	State change to enable voltage in state machine	
2	No Quick Stop	4	no quick stop	Value of 0 activates quickstop	
3	Enable Operation	8	enable operation	State change to enable operation to move the motor.	
4	Operation Mode Specific	16	op. mode spec. 4	Definition depends upon the operating modes	
5	Operation Mode Specific	32	op. mode spec. 5	Definition depends upon the operating modes	
6	Operation Mode Specific	64	op. mode spec. 6	Definition depends upon the operating modes	
7	Fault Reset	128	fault reset	Activates a reset to clear a drive error	
8	Stop	256	halt	Stop is not supported in most operating modes	
9	Operation Mode Specific	512	op. mode spec. 9	Definition depends upon the operating modes	
10	Reserved	1024	Reserved	Reserved	
11	Manufacturer Specific	2048	manufacturer spec. 11	Manufacturer specific use	

co00	controlword				0x2500
Bit	Function	Value	Plaintext	Description	
12	Manufacturer Specific	4096	manufacturer spec. 12	Manufacturer specific use	
13	Manufacturer Specific	8192	manufacturer spec. 13	Manufacturer specific use	
14	Manufacturer Specific	16384	manufacturer spec. 14	Manufacturer specific use	
15	Brake Control	32768	brake ctrl 15	Activates brake functions depending on co21 brake control mode settings	

st00	statusword				0x2100
Bit	Function	Value	Plaintext	Description	
x	None	0	empty statusword	Waiting for status	
0	State Machine	1	ready to switch on	Display of state in State Machine	
1	State Machine	2	switched on	Display of state in State Machine	
2	State Machine	4	operation enabled	Display of state in State Machine	
3	Fault	8	fault	1 = fault	
4	State Machine	16	voltage enabled	1 = Operating voltage in power circuit is OK	
5	Quick Stop	32	no quick stop	1 = quick stop not active / 0 = quick stop active	
6	State Machine	64	switch on disabled	Display of state in State Machine	
7	Warning	128	warning	1 = There is a warning status	
8	Synchronous	256	synchron	Indicates drive synchronous to fieldbus	
9	Remote	512	remote	1 = Drive is controlled via bus	
10	Target Reached	1024	target reached	1 = Target position or speed reached	
11	Internal Limit	2048	internal limit active	1 = A limit has been reached	
12	Mode Specific	4096	op. mode spec. 12	Setpoint acknowledge in pp-mode	
13	Mode Specific	8192	op. mode spec. 13	Following error for positioning	
14	Manufacturer Specific	16384	manufacturer spec. 14	Manufacturer specific use	
15	Manufacturer Specific	32768	manufacturer spec. 15	Braking state for brake control	

## 11.1.2.1.2 Basic Speed Control 20/70

The process data assembly for the "CIP Basic Speed Control" is displayed below. This is a simple configuration of the application to set start and stop commands and the reference speed.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
20	0						Fault Reset		Run Fwd
	1								
	2	Speed Reference (Low Byte)							
	3	Speed Reference (High Byte)							
70	0						Running1		Faulted
	1								
	2	Speed Actual (Low Byte)							
	3	Speed Actual (High Byte)							

## 11.2 EtherNet/IP™ objects

### 11.2.1 Identity Object (class code: 0x01)

The Identity Object contains parameters that allow identification and provide general information about the device. These parameters can be used for electronic coding, general status displays and for determining the devices on the network.

#### 11.2.1.1 Class attribute

Class	Instance	Attribute	Name	Type	Access
1	0	1	Revision	UINT	RO
1	0	2	Max Instance	UINT	RO
1	0	6	Max. Class Attribute	UINT	RO
1	0	7	Max. Instance Attribute	UINT	RO

#### 11.2.1.2 Instance Attribute

Class	Instance	Attribute	Name	Type	Access
1	1	1	Vendor ID	UINT	RO
1	1	2	Device Type	UINT	RO
1	1	3	Product Code	UINT	RO
1	1	4	Revision	STRUCT of	RO
			Major	USINT	RO
			Minor	USINT	RO
1	1	5	State	WORD	RO
1	1	6	Serial Number	UDINT	RO
1	1	7	Product Name	SHORT STRING	RO
1	1	8	State	USINT	RO
1	1	9	Configuration	UINT	RO

### 11.2.2 Assembly object (class code: 0x04)

The assembly object contains parameters which connect the attributes of several objects. This allows data to be sent or received from or to each of these objects via a single connection.

Assembly objects can be used to connect input or output data. KEB process output data (PD-Out) are sent from a controlling scanner, while process input data (PD-In) are output from the drive converter to the control (scanner) or network.

#### 11.2.2.1 Class attribute

Class	Instance	Attribute	Name	Type	Access
4	0	1	Revision	UINT	RO

### 11.2.2.2 Instance Attribute

Class	Instance	Attribute	Name	Type	Access
4	100	4	Size in Bytes	UINT	RO
4	100	3	KEB Data	ARRAY of BYTE	RW
4	100	4	Size in Bytes	UINT	RO
4	20	3	Basic Speed Control Data	ARRAY of BYTE	RW
4	20	4	Size in Bytes	UINT	RO
4	70	3	Basic Speed Control Data	ARRAY of BYTE	RW
4	70	4	Size in Bytes	UINT	RO

### 11.2.3 TCP/IP Interface Object (class code: 0xF5)

The TCP/IP interface object provides a mechanism to configure the network interface of the device. The physical connection contains parameters such as the IP address or the network mask.

#### 11.2.3.1 Class attribute

Class	Instance	Attribute	Name	Type	Access
F5	0	1	Revision	UINT	RO
F5	0	2	Max Instance	UINT	RO

#### 11.2.3.2 Instance Attribute

Class	Instance	Attribute	Name	Type	Access
F5	1	1	State	DWORD	RO
F5	1	2	Configuration Capability	DWORD	RO
F5	1	3	Configuration Control	DWORD	RW
F5	1	4	Physical Link	STRUCT of	
			Path Size	UINT	RO
			Path	EPATH	RO
F5	1	5	Interface Configuration	STRUCT of	
			IP Address	UDINT	RW
			Network Mask	UDINT	RW
			Gateway Address	UDINT	RW
			Name Server	UDINT	RW
			Name Server 2	UDINT	RW
			Domain Name	STRING	RW
F5	1	6	Host Name	STRING	RW

### 11.2.4 Ethernet link object (class code: 0xF6)

The Ethernet link object contains connection-specific status information for the IEEE 802.3 interface. The interface supports two instances for the two connected ports.

## 11.2.4.1 Class attribute

Class	Instance	Attribute	Name	Type	Access
F6	0	1	Revision	UINT	RO
F6	0	2	Max Instance	UINT	RO
F6	0	3	Number of Instances	UINT	RO

## 11.2.4.2 Instance Attribute

Class	Instance	Attribute	Name	Type	Access
F6	1,2	1	Interface Speed	UDINT	RO
F6	1,2	2	Interface Flags	DWORD	RO
F6	1,2	3	Physical Address	ARRAY of USINT	RO
F6	1,2	4	Interface Counters	STRUCT of	
			IN Octets	UDINT	RO
			IN Ucast Packets	UDINT	RO
			IN Non-Ucast Packets	UDINT	RO
			IN Discards	UDINT	RO
			IN Errors	UDINT	RO
			IN Unknown Protocols	UDINT	RO
			OUT Octets	UDINT	RO
			OUT Ucast Packets	UDINT	RO
			OUT Non-Ucast Packets	UDINT	RO
			OUT Discards	UDINT	RO
			OUT Errors	UDINT	RO
F6	1,2	5	Media Counters	STRUCT of	
			Alignment Errors	UDINT	RO
			FCS Errors	UDINT	RO
			Single Collisions	UDINT	RO
			Multiple Collisions	UDINT	RO
			SQE Test Errors	UDINT	RO
			Deferred Transmissions	UDINT	RO
			Late Collisions	UDINT	RO
			Excessive Collisions	UDINT	RO
			MAC Transmit Errors	UDINT	RO
			Carrier Sense Errors	UDINT	RO
			Frame Too Long	UDINT	RO
			MAC Receive Errors	UDINT	RO
F6	1,2	6	Interface Control	STRUCT of	
			Control Bits	WORD	RO
			Forced Interface Speed	UINT	RO
F6	1,2	7	Interface Type	USINT	RO
F6	1,2	8	Interface State	USINT	RO

Class	Instance	Attribute	Name	Type	Access
F6	1,2	9	Admin State	USINT	RW
F6	1,2	10	Interface Label	SHORT STRING	RW
F6	1,2	11	Interface Capability	STRUCT of	
			Capability Bits	WORD	RO
			Speed/Duplex Options	STRUCT of	
			Array Count	USINT	RO
			Speed/Duplex Array	ARRAY of STRUCT of	
			Interface Speed	UINT	RO
			Interface Duplex Mode	USINT	RO

### 11.2.5 KEB inverter object (class code: 0x64)

The KEB inverter object enables access to the manufacturer-specific parameters of the inverter. This class enables the writing and reading of objects. Services Get\_Attribute\_Single and Set\_Attribute\_Single can be used to access these parameters.

#### 11.2.5.1 Class attribute

There are no class attributes for this class.

#### 11.2.5.2 Instance Attribute

Parameter access to the inverter parameters is described as example in 11.1.1 Explicit messaging (parameterization channel). A list with all index values of the inverter parameters can be found in the [Programming Manual | Control Application / Compact / Pro](#).

Class	Instance	Attribute	Name	Type	Access
0x64	Index	Sub Index	Parameter Name	Type	RO/RW

## 11.3 Status codes

The following table lists the general status messages which can be contained in a response message. The "extended code field" is available for the future description of general status messages.

Status codes	Name of the status	Description
0x00, 0	Success	The service was successfully executed by the object.
0x01, 1	Connection failure	The connection to the device has failed.
0x02, 2	Resource unavailable	The required resources for the requested service are not available.
0x03, 3	Invalid parameter value	The parameter value of the request is not supported.
0x04, 4	Path segment error	An error occurred during the evaluation of the communication path.
0x05, 5	Path destination unknown	The communication path references an unknown object class, instance or unknown attribute.
0x06, 6	Partial transfer	Only a part of the data could be transferred.
0x07, 7	Connection lost	The connection to the device has been lost.



Status codes	Name of the status	Description
0x08, 8	Service not supported	The requested service is not supported by the selected object.
0x09, 9	Invalid attribute value	The selected attribute is not supported by the object.
0x0A, 10	Attribute list error	An attribute in the Get_Attribute_List or Set_Attribute_List has a non-zero status.
0x0B, 11	Already in requested state	The object is already in the requested state.
0x0C, 12	Object state conflict	The object can not execute the requested service in the current state.
0x0D, 13	Object already exists	The requested creation of the object has failed, since the object already exists.
0x0E, 14	Attribute not settable	The selected attribute cannot be changed.
0x0F, 15	Privilege violation	Authorization check failed
0x10, 16	Device state conflict	The current state of the device prohibits the requested service.
0x11, 17	Reply data too large	The response memory is too small for the data to be sent.
0x12, 18	Fragmentation of primitive	The selected service attempts to perform an operation that fragments a primitive data type.
0x13, 19	Not enough data	The data provided by the service are not enough.
0x14, 20	Attribute not supported	The selected attribute is not supported.
0x15, 21	Too much data	The service has provided too much data.
0x16, 22	Object does not exist	The selected object does not exist.
0x17, 23	Service fragmentation	Service fragmentation is not active.
0x18, 24	No stored attribute data	The attribute of the selected object was not stored.
0x19, 25	Store operation failure	The attribute of the selected object could not be stored.
0x1A, 26	Routing failure, request packet too large	The selected service request packet was too large to be sent over the network. The routing service has aborted the transmission.
0x1B, 27	Routing failure, response packet too large	The selected service response packet was too large to be sent over the network. The routing service has aborted the transmission.
0x1C, 28	Missing attribute list entry data	The selected service does not have a suitable attribute to perform the selected behavior.
0x1D, 29	Invalid attribute value list	The selected service returns the list of invalid attributes with associated status information.
0x1E, 30	Embedded service error	An embedded service has triggered an error.
0x1F, 31	Vendor specific error	A vendor-specific error has occurred. This error occurs if no other general error code can be assigned to the behavior.
0x20, 32	Invalid parameter	A parameter belonging to the request is invalid. The parameter does not meet the requirements of the specification or application.
0x21, 33	Write-once already written	An attempt was made to write-once already written.
0x22, 34	Invalid reply	An invalid reply was received. The received service code is non-compliant with the sent service code or the reply is less than the minimum expected size.
0x23, 35	Buffer overflow	The received message is larger than the corresponding memory. The message has been rejected.
0x24, 36	Invalid message format	The format of the received message is not supported.
0x25, 37	Key failure in path	The key segment contained as the first segment in the path does not correspond to the target module.

Status codes	Name of the status	Description
0x26, 38	Path size invalid	The size of the received routing path is too small or too much routing data is included.
0x27, 39	Unexpected attribute	An attempt has been made to set an attribute that cannot be set at this time.
0x28, 40	Invalid Member ID	The received Member ID does not exist.
0x29, 41	Member not settable	A request to modify a non-modifiable member was received.
0x2A-0xCF, 42-207	Reserved	Reserved for future extensions.
0xD0-0xFF, 208-255	Reserved	Reserved for object class-specific error codes.

## 12 Modbus/TCP interface

The Modbus protocol is a communication protocol based on the client/server architecture. KEB devices support only the Modbus variant Modbus/TCP, where the Modbus data are transmitted via TCP/IP. Modbus TCP is part of the IEC 61158 standard.

In the following chapter KEB device with Modbus/TCP support are synonymously called server. Applications that access the KEB device via Modbus/TCP are called clients.



- Only one **Special version** of the KEB devices of control type A supports Modbus/TCP. For further information, please contact your sales partner.

Class 0 implementation is supported. Clients can read parameters from the server and write them to the server. The functions **3: Read Multiple Registers** and **16: Write Multiple Registers** are supported. All requested Modbus/TCP transactions are carried out with 16-bit registers.

Each Modbus message is divided into user data and a prefixed identifier (**Function Code**) that defines the function of the message.

A transaction in the client/server model is shown below.

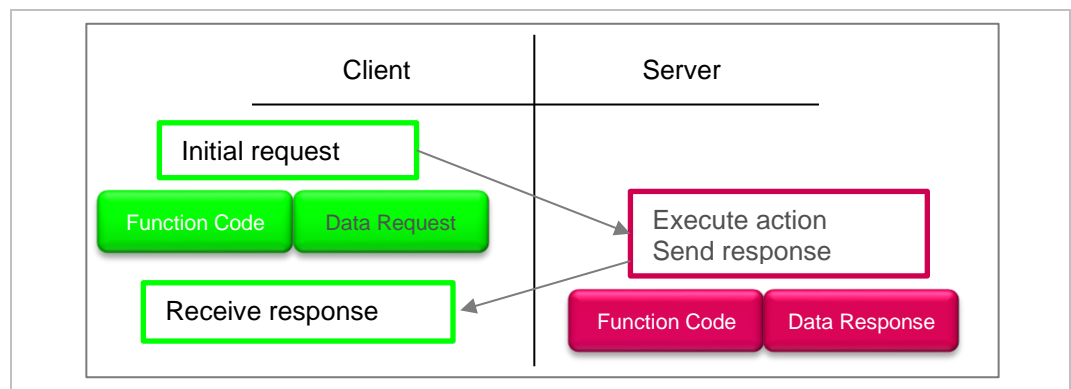


Figure 7: Client / Server Model

As long as Modbus/TCP is the active fieldbus system, the current state of the Modbus/TCP connection is displayed via the fieldbus LED of the device. Further information on the light pattern in chapter 5.5.2 Checking the fieldbus STATUS LED (NET ST).



- If the client does not send a request to the server for more than 30 seconds, the established connection is rejected.

## 12.1 Address range

The server is divided into two address ranges. One for process data communication and one for parameter access to individual parameters.

- 0x0000 → 0x1FFF: Process data mapping
- 0x2000 → 0xFFFF: Parameter channel access

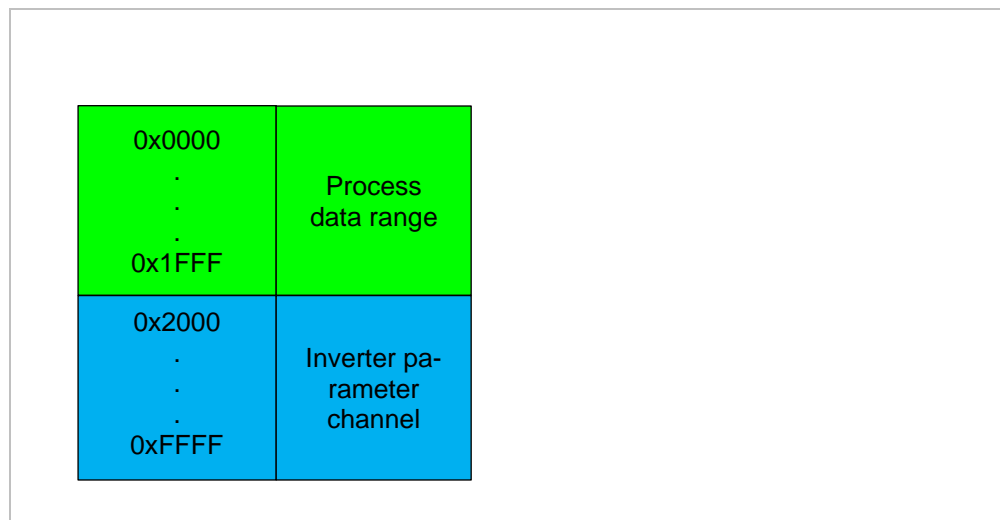


Figure 8: Address range

The process data mapping is a continuous memory area that starts at address 0. All data of the process data mapping are addressed in one access.

The number of registers available for mapping is determined by the length of the mapping. One register is available for each 16 bit process data mapping.

The parameter channel allows read or write access to individual parameters. The parameter channel has a maximum size of 2 registers on KEB devices. Modbus registers are 16 bit long by default.



- When accessing different data types, it is important to pay attention to the different length of these parameters. KEB parameters can be 1, 2 or 4 bytes long. To avoid data loss, the right size for setting the registers must be considered. This applies in particular to the access of parameter types "signed" and "unsigned".

## 12.2 Supported functions

### 12.2.1 Function 3: Read Multiple Registers

This function is used to read registers. The start address of the first register and the number of registers are sent as a request.

#### Request read process data

Description	Size	Data
Function Code	1 byte	0x03
Address	2 bytes	0x0000 → 0x1FFF
Register number	2 bytes	0x0001 → Mapping length

#### Response read process data

Description	Size	Data
Function Code	1 byte	0x03
Size of the register values in bytes	1 byte	2 * Register number
Register values	2 bytes * Register number	Read out values

#### Request read parameter channel

Description	Size	Data
Function Code	1 byte	0x03
Address	2 bytes	0x2000 → 0xFFFF
Register number	2 bytes	0x0001 or 0x0002

#### Response read parameter channel

Description	Size	Data
Function Code	1 byte	0x03
Size of the register values in bytes	1 byte	2 * Register number
Register values	2 bytes * Register number	Read out values

### 12.2.2 Function 16: Write Multiple Registers

This function is used to write registers. The start address of the first register, the number of registers, the number of bytes (2 \* number of registers) and the values to be written are sent as a request.

#### Request write process data

Description	Size	Data
Function Code	1 byte	0x10
Address	2 bytes	0x0000 → 0x1FFF
Register number	2 bytes	0x0001 → Mapping length
Size of the values in byte	1 byte	2 * Register number
Values to be written	2 bytes * Register number	Values to be written

#### Response write process data

Description	Size	Data
Function Code	1 byte	0x10
Address	2 bytes	0x0000 → 0x1FFF
Register number	2 bytes	0x0001 → Mapping length

#### Request write parameter channel

Description	Size	Data
Function Code	1 byte	0x10
Address	2 bytes	0x2000 → 0xFFFF
Register number	2 bytes	0x0001 or 0x0002
Size of the values in byte	1 byte	2 * Register number
Values to be written	2 bytes * Register number	Values to be written

#### Request write parameter channel

Description	Size	Data
Function Code	1 byte	0x10
Address	2 bytes	0x2000 → 0xFFFF
Register number	2 bytes	0x0001 or 0x0002

### 12.3 Error handling (exception handling)

In case of a faulty transaction, the server sends an error message as response to the client. An error message is always 2 bytes long. The most significant bit of the message is set to 1.

The following client/server model shows the error handling with Modbus/TCP.

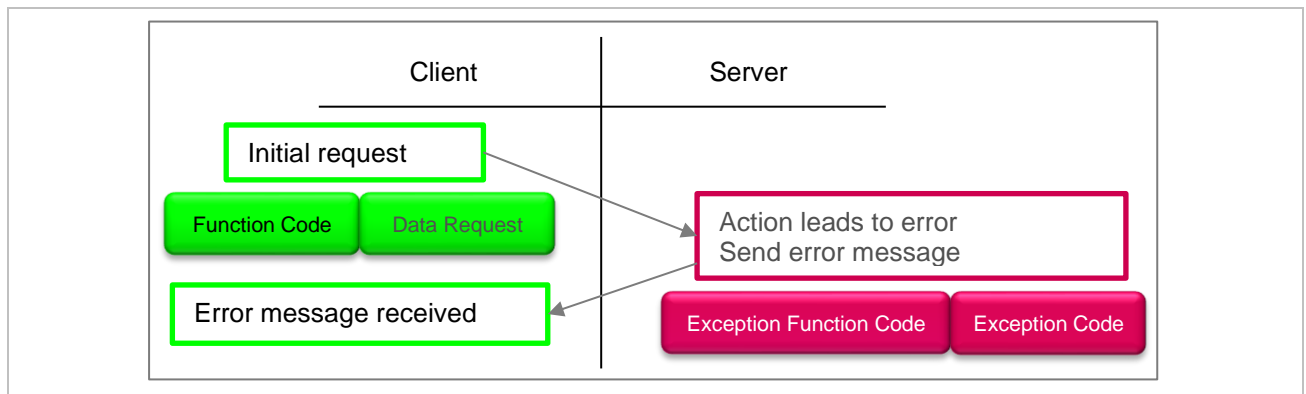


Figure 9: Exception handling

#### 12.3.1 Error messages

Each error message consists of an identifier (**Exception Function Code**) and an error code. The identifier of the error message corresponds to the identifier of the request with the most significant bit set to 1.

##### Function 3 error message

Description	Size	Data
Exception Function Code	1 byte	0x83
Error code	1 byte	0x01

##### Function 16 error message

Description	Size	Data
Exception Function Code	1 byte	0x90
Error code	1 byte	0x01

### 12.3.2 Error codes

Modbus error codes		
code	Name	Description
0x01	ILLEGAL FUNCTION	The received function code is not supported by the server. It is possible that the function code will be supported in a future version of the software.
0x02	ILLEGAL DATA ADDRESS	The received data address or the combination of address and number of registers is not supported by the server. Possibly the length of the registers exceeds the permissible address range or the length of the registers does not match the data type of the addressed parameter.
0x03	ILLEGAL DATA VALUE	The received value is not permissible for the addressed parameter.
0x04	SLAVE DEVICE FAILURE	A critical error occurred on the server while processing the request. The device cannot correct the error on its own.

## 12.4 Modbus/TCP specific parameters

### 12.4.1 Configure IP addressing of the device (fb114)

To reach the KEB device via TCP/IP IP address, subnet mask and gateway must be configured.

The default IP configuration is variable and can be set via parameter **fb114**. Similar to Ethernet/IP, the IP configuration can be specified statically, via Bootp or via DHCP.

Further information on the IP configuration can be found in chapter 5.5.1 Checking MAC addresses and IP configuration

fb114	ModbusTCP Configuration			0x2B72
Bit	Function	Value	Text	Description
0...1	IP configuration method	0	Static	IP configuration is static
		1	Bootp	IP configuration via bootp
		2	DHCP	IP configuration via DHCP



### 12.4.2 Addressing the subindex via the parameter channel (fb115)

The Modbus parameter channel allows access to individual parameters of the server. However, the addressing specified by Modbus does not support subindices.

To be able to access also the subindexes of parameters, KEB devices provide the parameter **fb115**. If required, the subindex of the parameter to be addressed can be entered here.

The value entered in **fb115** is deleted by a restart of the device.

Index	SubIndex	Id-Text	Name	Function
0x2B73	0	<b>fb115</b>	ModbusTCP SubIndex	Subindex for access via the parameter channel



- If a parameter is addressed that does not support the subindex set in **fb115**, an error message is returned. When accessing parameters, always pay attention to the set value in **fb115**.

## 13 Annex

### 13.1 Mapping parameters and Sync manager parameters

Abbreviations	RO	ReadOnly		
	nPD	not available for ProcessDataCommunication		
	CAN	CANopen Type	V	Variable
			ST	Structure
			A	Array
	HW	Control board version	A	S6A / F6A
			K	S6K / F6K
			P	S6P / F6P

Index	SubIdx	CAN	Type	Name	Upper limit	Lower limit	Mult.	Div	Unit	nPD	RO	HW
1009h	0	V	STRING	Hardware version	1	1	1	1	---	X	X	A,K,P
100Ah	0	V	STRING	Software version	4294967295	0	1	1	---	X	X	A,K,P
1400h	0	ST	UINT8	1st RPDO communication parameter	2	2	1	1	---	X	X	A,K,P
	1		UINT32	cob-ID	4294967295	1	1	1	---	X	---	
	2		UINT8	transmission type	255	0	1	1	---	X	---	
1401h	0	ST	UINT8	2nd RPDO communication parameter	2	2	1	1	---	X	X	A,K,P
	1		UINT32	cob-ID	4294967295	1	1	1	---	X	---	
	2		UINT8	transmission type	255	0	1	1	---	X	---	
1402h	0	ST	UINT8	3rd RPDO communication parameter	2	2	1	1	---	X	X	A,K,P
	1		UINT32	cob-ID	4294967295	1	1	1	---	X	---	
	2		UINT8	transmission type	255	0	1	1	---	X	---	
1403h	0	ST	UINT8	4th RPDO communication parameter	2	2	1	1	---	X	X	A,K,P
	1		UINT32	cob-ID	4294967295	1	1	1	---	X	---	
	2		UINT8	transmission type	255	0	1	1	---	X	---	
1600h	0	ST	UINT8	1st receive PDO mapping	8	0	1	1	---	X	---	A,K,P
	1...8		UINT32		4294967295	0	1	1	---	X	---	
1601h	0	ST	UINT8	2nd receive PDO mapping	8	0	1	1	---	X	---	A,K,P
	1...8		UINT32		4294967295	0	1	1	---	X	---	
1602h	0	ST	UINT8	3rd receive PDO mapping	8	0	1	1	---	X	---	A,K,P
	1...8		UINT32		4294967295	0	1	1	---	X	---	
1603h	0	ST	UINT8	4th receive PDO mapping	8	0	1	1	---	X	---	A,K,P
	1...8		UINT32		4294967295	0	1	1	---	X	---	
1800h	0	ST	UINT8	1st TPDO communication parameter	5	5	1	1	---	X	X	A,K,P
	1		UINT32	cob-ID	4294967295	1	1	1	---	X	---	
	2		UINT8	transmission type	255	0	1	1	---	X	---	
	3		UINT16	inhibit time	65535	0	1	1	ms	X	---	
	4		UINT8	reserved	0	0	1	1	---	X	X	

Index	SubIdx	CAN	Type	Name	Upper limit	Lower limit	Mult.	Div	Unit	nPD	RO	HW
1800h	5	ST	UINT16	event time	65535	0	1	1	---	X	---	A,K,P
1801h	0	ST	UINT8	2nd TPDO communication parameter	5	5	1	1	---	X	X	A,K,P
	1		UINT32	cob-ID	4294967295	1	1	1	---	X	---	
	2		UINT8	transmission type	255	0	1	1	---	X	---	
	3		UINT16	inhibit time	65535	0	1	1	ms	X	---	
	4		UINT8	reserved	0	0	1	1	---	X	X	
	5		UINT16	event time	65535	0	1	1	---	X	---	
1802h	0	ST	UINT8	3rd TPDO communication parameter	5	5	1	1	---	X	X	A,K,P
	1		UINT32	cob-ID	4294967295	1	1	1	---	X	---	
	2		UINT8	transmission type	255	0	1	1	---	X	---	
	3		UINT16	inhibit time	65535	0	1	1	ms	X	---	
	4		UINT8	reserved	0	0	1	1	---	X	X	
	5		UINT16	event time	65535	0	1	1	---	X	---	
1803h	0	ST	UINT8	4th TPDO communication parameter	5	5	1	1	---	X	X	A,K,P
	1		UINT32	cob-ID	4294967295	1	1	1	---	X	---	
	2		UINT8	transmission type	255	0	1	1	---	X	---	
	3		UINT16	inhibit time	65535	0	1	1	ms	X	---	
	4		UINT8	reserved	0	0	1	1	---	X	X	
	5		UINT16	event time	65535	0	1	1	---	X	---	
1A00h	0	ST	UINT8	1st transmit PDO mapping	8	0	1	1	---	X	---	A,K,P
	1...8		UINT32		4294967295	0	1	1	---	X	---	
1A01h	0	ST	UINT8	2nd transmit PDO mapping	8	0	1	1	---	X	---	A,K,P
	1...8		UINT32		4294967295	0	1	1	---	X	---	
1A02h	0	ST	UINT8	3rd transmit PDO mapping	8	0	1	1	---	X	---	A,K,P
	1...8		UINT32		4294967295	0	1	1	---	X	---	
1A03h	0	ST	UINT8	4th transmit PDO mapping	8	0	1	1	---	X	---	A,K,P
	1...8		UINT32		4294967295	0	1	1	---	X	---	
1C00h	0	A	UINT8	Sync Manager Communication Type	4	4	1	1	---	X	X	A,K,P
	1...4				4	0						
1C12h	0	A	UINT16	sync manager 2 PDO assign	1	1	1	1	---	X	---	K
	1				1600h	1600h						
1C12h	0	A	UINT16	sync manager 2 PDO assign	2	2	1	1	---	X	---	A,P
	1...2				1601h	1600h						
1C13h	0	A	UINT16	sync manager 3 PDO assign	1	1	1	1	---	X	---	K

## Mapping parameters and Sync manager parameters

Index	SubIdx	CAN	Type	Name	Upper limit	Lower limit	Mult.	Div	Unit	nPD	RO	HW
1C13h	1	A	UINT16	sync manager 3 PDO assign	1A00h	1A00h	1	1	---	X	---	K
1C13h	0	A	UINT16	sync manager 3 PDO assign	2	2	1	1	---	X	---	A,P
	1...2				1A01h	1A00h						
1C32h	0	ST	UINT8	Output sync manager para	32	32	1	1	---	X	X	A,K,P
	1		UINT16	Sync mode	3	0	1	1	---	X	X	
	2		UINT32	Cycle Time	16000000	0	1	1000	us	X	X	
	4		UINT16	Sync modes supported	65535	0	1	1	---	X	X	
	5		UINT32	Minimum Cycle Time	16000000	0	1	1000	us	X	X	
	6		UINT32	Calc and Copy Time	4294967295	0	1	1000	us	X	X	
	11		UINT16	SM Event Missed	65535	0	1	1	---	X	X	
	12		UINT16	Cycle Time Too Small	65535	0	1	1	---	X	X	
	32		UINT8	Sync Error	1	0	1	1	---	---	X	
1C33h	0	ST	UINT8	Input sync manager para	32	32	1	1	---	X	X	A,K,P
	1		UINT16	Sync mode	3	0	1	1	---	X	X	
	2		UINT32	Cycle Time	16000000	0	1	1000	us	X	X	
	4		UINT16	Sync modes supported	65535	0	1	1	---	X	X	
	5		UINT32	Minimum Cycle Time	16000000	0	1	1000	us	X	X	
	6		UINT32	Calc and Copy Time	4294967295	0	1	1000	us	X	X	
	11		UINT16	SM Event Missed	65535	0	1	1	---	X	X	
	12		UINT16	Cycle Time Too Small	65535	0	1	1	---	X	X	
	32		UINT8	Sync Error	1	0	1	1	---	---	X	

## 13.2 Fieldbus parameters

Abbreviations	RO	ReadOnly		
	nPD	not available for ProcessDataCommunication		
	CAN	CANopen Type	V	Variable
			ST	Structure
			A	Array
	HW	Control board version	A	S6A / F6A
			K	S6K / F6K
			P	S6P / F6P

For structures, the name of the structure is entered in the line of subindex 0 (number).  
HW indicates the control types of the devices which support the specified parameter.

Index	SubIdx	CAN	Type	Name	Upper limit	Lower limit	Mult.	Div	Unit	nPD	RO	HW
2B0Ah	0	V	UINT16	sync interval	16000	0	1	1	µs	X	---	A,K,P
2B0Bh	0	V	UINT16	Set sync level	1000	0	1	10	µs	X	---	A,K,P
2B0Ch	0	V	UINT16	KP sync PLL	256	0	1	1	---	X	---	A,K,P
2B0Dh	0	V	UINT8	DIN66019 node id	238	1	1	1	---	X	---	A,K
2B0D	0	V	UINT8	drive node ID	238	1	1	1	---	X	---	P
2B0Eh	0	V	UINT8	DIN66019 baud rate	12	0	1	1	---	X	---	A,K,P
2B0Fh	0	ST	UINT8	node IDs	2	2	1	1	---	X	X	P
	1		UINT8	application node ID	255	0	1	1	---	X	---	P
	2		UINT8	debugger node ID	255	0	1	1	---	X	---	P
2B10h	0	V	UINT8	Fieldbus node injection	255	1	1	1	---	X	---	P
2B13h	0	V	UINT16	measured sync interval	65535	0	2	3	µs	---	X	A
2B13h	0	V	UINT16	measured sync interval	65535	0	4	5	µs	---	X	K
2B13h	0	V	UINT16	measured sync interval	65535	0	64	75	µs	---	X	P
2B14h	0	V	UINT8	ETC invalid frame count P0	255	0	1	1	---	X	X	K,P
2B15h	0	V	UINT8	ETC RX error count P0	255	0	1	1	---	X	X	K,P
2B16h	0	V	UINT8	ETC invalid frame count P1	255	0	1	1	---	X	X	K,P
2B17h	0	V	UINT8	ETC RX error count P1	255	0	1	1	---	X	X	K,P
2B18h	0	V	UINT8	ETC for. RX error count P0	255	0	1	1	---	X	X	K,P
2B19h	0	V	UINT8	ETC for. RX error count P1	255	0	1	1	---	X	X	K,P
2B1Ah	0	V	UINT8	ETC processing unit error count	255	0	1	1	---	X	X	K,P
2B1Bh	0	V	UINT16	ETC min. sync delay	32000	0	8	25	µs	X	---	K,P
2B1Ch	0	V	UINT16	ETC max. sync delay	32000	0	8	25	µs	X	---	K,P
2B1Dh	0	V	UINT16	ETC no frame per sync cnt	32000	0	1	1	---	X	---	K,P
2B1Eh	0	V	UINT16	ETC mult. frames per sync cnt	32000	0	1	1	---	X	---	K,P
2B1Fh	0	V	UINT16	no PDO data per sync cnt	32000	0	1	1	---	X	---	A,K,P
2B20h	0	V	UINT8	LED 'DEV ST' blink status	1	0	1	1	---	---	---	A,K,P
2B25h	0	V	INT16	ETC PD access min. sync delay	3200	0	8	25	µs	X	---	K,P
2B26h	0	V	INT16	ETC PD access max. sync delay	3200	0	8	25	µs	X	---	K,P

## Fieldbus parameters

Index	SubIdx	CAN	Type	Name	Upper limit	Lower limit	Mult.	Div	Unit	nPD	RO	HW
2B3Ch	0	V	UINT8	process data size selection	4	0	1	1	---	X	---	A,K,P
2B40h	0	V	UINT8	CAN node ID	127	1	1	1	---	X	---	A,K,P
2B42h	0	V	UINT8	CAN baud rate	8	1	1	1	---	X	---	A,K,P
2B43h	0	V	UINT16	fieldbus configuration	15	0	1	1	---	X	---	A,K,P
2B44h	0	V	UINT8	fieldbus selection	6	0	1	1	---	X	---	A
2B44h	0	V	UINT8	fieldbus selection	2	0	1	1	---	X	---	K
2B44h	0	V	UINT8	fieldbus selection	1	0	1	1	---	X	---	P
2B45h	0	V	UINT16	CAN lost messages	65535	0	1	1	---	---	X	A,K
2B46h	0	ST	UINT8	CAN options	7	7	1	1	---	X	X	A,K
	1		UINT32	CAN option code	4294967295	0	1	1	---	X	---	A,K
	2		UINT16	Tx PDO base ID	65535	0	1	1	---	X	---	A,K
	3		UINT16	Rx PDO base ID	65535	0	1	1	---	X	---	A,K
	4		UINT16	Tx PDO1 cycle time	10000	10	1	1	---	X	---	A,K
	5		UINT16	Tx PDO2 cycle time	10000	10	1	1	---	X	---	A,K
	6		UINT16	Tx PDO3 cycle time	10000	10	1	1	---	X	---	A,K
	7		UINT16	Tx PDO4 cycle time	10000	10	1	1	---	X	---	A,K
2B47h	0	V	UINT32	fieldbus options	2	1	1	1	---	---	---	P
2B48h	0	V	UINT32	change cnt	4294967295	0	1	1	---	---	---	P
2B50h	0	ST	UINT8	MRTE module	3	3	1	1	---	X	X	A
	1		UINT8	MRTE module support	1	0	1	1	---	X	---	A
	2		UINT8	MRTE module state	255	0	1	1	---	X	X	A
	3		UINT16	MRTE module version	255	0	1	1	---	X	X	A
2B5Ah	0	ST	UINT8	fieldbus state	6	6	1	1	---	X	X	A,K,P
	1		UINT16	EtherCAT fieldbus state	65535	0	1	1	---	X	X	A,K,P
	2		UINT16	CANopen fieldbus state	65535	0	1	1	---	X	X	A,K,P
	3		UINT16	VARAN fieldbus state	65535	0	1	1	---	X	X	K
	3		UINT16	PROFINET fieldbus state	65535	0	1	1	---	X	X	A
	4		UINT16	POWERLINK fieldbus state	65535	0	1	1	---	X	X	A
	5		UINT16	EtherNet/IP fieldbus state	65535	0	1	1	---	X	X	A
	6		UINT16	ModbusTCP fieldbus state	65535	0	1	1	---	X	X	A
2B5Bh	0	ST	UINT8	fieldbus error code	6	6	1	1	---	X	X	A,K,P
	1		UINT16	EtherCAT fieldbus error code	65535	0	1	1	---	X	X	A,K,P
	2		UINT16	CANopen fieldbus error code	65535	0	1	1	---	X	X	A,K,P
	3		UINT16	VARAN fieldbus error code	65535	0	1	1	---	X	X	K
	3		UINT32	PROFINET fieldbus error code	9568255	0	1	1	---	X	X	A
	4		UINT16	POWERLINK fieldbus error code	65535	0	1	1	---	X	X	A
	5		UINT16	EtherNet/IP fieldbus error code	65535	0	1	1	---	X	X	A
	6		UINT16	ModbusTCP fieldbus error code	65535	0	1	1	---	X	X	A

Index	SubIdx	CAN	Type	Name	Upper limit	Lower limit	Mult.	Div	Unit	nPD	RO	HW
2B64h	0	V	UINT8	node ID switch value	255	0	1	1	---	X	X	A
2B65h	0	V	UINT8	adjusted node ID value	255	0	1	1	---	X	---	A
2B66h	0	V	UINT8	effective node ID	255	0	1	1	---	X	X	A
2B66h	0	V	UINT8	effective node ID	255	0	1	1	---	X	---	P
2B67h	0	V	UINT32	FB MAC Address (Base)	4211081215	0	1	1	---	X	---	A
2B68h	0	V	UINT32	PNET MAC Address (Port1)	4211081215	0	1	1	---	X	---	A
2B69h	0	V	UINT32	PNET MAC Address (Port2)	4211081215	0	1	1	---	X	---	A
2B69h	0	V	UINT32	MAC Address (EoE Channel)	4211081215	0	1	1	---	X	---	P
2B6Ah	0	V	UINT32	MAC Address (EthChannel)	4211081215	0	1	1	---	X	X	P
2B6Ah	0	V	UINT32	MAC Address (EthChannel)	4211081215	0	1	1	---	X	---	A,K
2B6Bh	0	A	UINT8	PROFINET NameOfStation	240	240	1	1	---	X	X	A
	1 - 240				32	122						
2B6Ch	0	ST	UINT8	Ethernet over fieldbus IP configuration	3	3	1	1	---	X	X	A,K,P
	1		UINT32	IP address	4294967295	0	1	1	---	X	---	A,K,P
	2		UINT32	subnet mask	4294967295	0	1	1	---	X	---	A,K,P
	3		UINT32	gateway address	4294967295	0	1	1	---	X	---	A,K,P
2B6Dh	0	ST	UINT8	basic IP configuration	3	3	1	1	---	X	X	A,P
	1		UINT32	IP address	4294967295	0	1	1	---	X	---	A,P
	2		UINT32	subnet mask	4294967295	0	1	1	---	X	---	A,P
	3		UINT32	gateway address	4294967295	0	1	1	---	X	---	A,P
2B6Eh	0	A	UINT8	Scan names	2	2	1	1	---	X	---	P
	1 - 2				32	0						
2B6Fh	0	A	UINT8	POWERLINK RPDO offset	8	8	1	1	---	X	---	A
	1 - 8				255	0						
2B70h	0	A	UINT8	POWERLINK TPDO offset	8	8	1	1	---	X	---	A
	1 - 8				255	0						
2B71h	0	V	UINT8	EtherNet/IP Configuration	6	0	1	1	---	X	---	A
2B72h	0	V	UINT8	ModbusTCP Configuration	2	0	1	1	---	X	---	A
2B73h	0	V	UINT8	ModbusTCP Subindex	255	0	1	1	---	X	---	A

### 13.3 Solutions for KEB specific fieldbus error codes

The following table describes KEB specific fieldbus errors. The listed solutions should be carried out in consultation with a KEB service employee. If other error codes occur in [fb91](#), please check whether they are error codes of the respective fieldbus system. If this is not the case, please contact KEB Support.

Error code	Description	Solution
INIT_ERR_PD	PD mapping error, PD mapping is deactivated.	Check and reactivate the PD mapping, then perform a software reset.
INIT_ERR_INV_NODE_ID	Node address value is not supported by the selected fieldbus system.	Change node address value or active fieldbus system and perform a software reset.
INIT_ERR_FB_TYPE_INVALID	Selected fieldbus system is not supported in the current configuration.	Change fieldbus system or check connection to MRTE module( <a href="#">fb80</a> ). After changes software reset.
INIT_ERR_I2C	No access to I2C, detection of the MRTE module and reading of the node address is not possible.	Check the physical connection between the MRTE module and the control board. Perform software reset.
INIT_ERR_BASE_DRV_NOT_ONLINE	Driver of the MRTE module does not receive a response from the MRTE module.	Check the connection between the MRTE module and the control board. ( <a href="#">fb80</a> )
INIT_ERR_NETX_BASE	Error in the initialization of the MRTE module driver.	Check the connection between the MRTE module and the control board. ( <a href="#">fb80</a> )
INIT_ERR_INV_NODE_ID	Error during configuration due to incorrect node address	Set node address value by <a href="#">fb100</a> or <a href="#">fb101</a> to a valid value for POWERLINK CN
XXX_NO_STACK_INIT_ERR	MRTE module file for selected fieldbus system is not in the file system	Check files in the file system via KEB FTP. Check MRTE module version via <a href="#">fb80</a> .
XXX_MAC_ADDR_ERROR	Missing MAC addresses.	Add MAC addresses ( <a href="#">fb103</a> – <a href="#">fb105</a> ).
XXX_SLAVE_FLAGS_INIT_ERR	Error in communication between software and MRTE module.	Check software version, contents of the file system and NetX version. Follow the start-up manual (chapter 5)
ECAT_DMA_ERROR	Error in the communication with the EtherCAT Core	Hardware error. Please contact your KEB support.
ECAT_DCSYNCMISSEDERROR	No process data received between two SYNCs	Checking the EtherCAT master configuration
ECAT_INVALID_SYNC_CYCLE_TIME	Specified cycle time is not supported by the slave	Check <a href="#">is22</a> and <a href="#">fb19</a> (cycle time must be integer multiple of MidIRQ cycle)
CAN301_ERROR_GENERIC_ERROR	Special case: Is displayed when CAN is the active fieldbus system and an error has occurred in the drive.	Check <a href="#">ru01</a> and <a href="#">ru02</a> .



### 13.4 Revision history

Version	Chapter	Change	Date
00		First version created	26.10.2020
	5.3.4	Changed behaviour of the node address on the P cards described	26.10.2020
	7.5.1	Added instructions for correct configuration of EoE networks	02.02.2021
	5.5.1.3, 10.3.2	Note that the node address values 0, 240 and 255 for POWERLINK invalid are added.	25.02.2021
	6.8	Description of the "transmission type" revised	01.03.2021
	6.6	Added information about data length for CAN SDO telegrams	01.04.2021
	12	Chapter about Modbus/TCP added	18.05.2021
	5	Modbus/TCP added to the general fieldbus chapter	19.05.2021
	5.8	General chapter about the fieldbus watchdog added	19.05.2021
	13.2	List with fieldbus parameters updated	20.05.2021
01		Editorial changes	28.05.2021
02	5.3.1, 12	Modbus/TCP supplement only available as special version	05.08.2021



**Belgium** | KEB Automation KG  
Herenveld 2 9500 Geraardsbergen Belgium  
Tel: +32 544 37860 Fax: +32 544 37898  
E-Mail: vb.belgien@keb.de Internet: www.keb.de

**Brazil** | KEB SOUTH AMERICA - Regional Manager  
Rua Dr. Omar Pacheco Souza Riberio, 70  
CEP 13569-430 Portal do Sol, São Carlos Brazil Tel: +55 16  
31161294 E-Mail: roberto.arias@keb.de

**Germany | Headquarters**  
KEB Automation KG  
Südstraße 38 32683 Barntrup Germany  
Telefon +49 5263 401-0 Telefax +49 5263 401-116  
Internet: www.keb.de E-Mail: info@keb.de

**Germany | Geared Motors**  
KEB Antriebstechnik GmbH  
Wildbacher Straße 5 08289 Schneeberg Germany  
Telefon +49 3772 67-0 Telefax +49 3772 67-281  
Internet: www.keb-drive.de E-Mail: info@keb-drive.de

**France** | Société Française KEB SASU  
Z.I. de la Croix St. Nicolas 14, rue Gustave Eiffel  
94510 La Queue en Brie France  
Tel: +33 149620101 Fax: +33 145767495  
E-Mail: info@keb.fr Internet: www.keb.fr

**United Kingdom** | KEB (UK) Ltd.  
5 Morris Close Park Farm Industrial Estate  
Wellingborough, Northants, NN8 6 XF United Kingdom  
Tel: +44 1933 402220 Fax: +44 1933 400724  
E-Mail: info@keb.co.uk Internet: www.keb.co.uk

**Italia** | KEB Italia S.r.l. Unipersonale  
Via Newton, 2 20019 Settimo Milanese (Milano) Italia  
Tel: +39 02 3353531 Fax: +39 02 33500790  
E-Mail: info@keb.it Internet: www.keb.it

**Japan** | KEB Japan Ltd.  
15 - 16, 2 - Chome, Takanawa Minato-ku  
Tokyo 108 - 0074 Japan  
Tel: +81 33 445-8515 Fax: +81 33 445-8215  
E-Mail: info@keb.jp Internet: www.keb.jp

**Austria** | KEB Automation GmbH  
Ritzstraße 8 4614 Marchtrenk Austria  
Tel: +43 7243 53586-0 Fax: +43 7243 53586-21  
E-Mail: info@keb.at Internet: www.keb.at

**Russian Federation** | KEB RUS Ltd.  
Lesnaya str, house 30 Dzerzhinsky MO  
140091 Moscow region Russian Federation  
Tel: +7 495 6320217 Fax: +7 495 6320217  
E-Mail: info@keb.ru Internet: www.keb.ru

**Switzerland** | KEB Automation KG  
Witzbergstraße 24 8330 Pfäffikon/ZH Switzerland  
Tel: +41 43 2886060 Fax: +41 43 2886088  
E-Mail: info@keb.ch Internet: www.keb.ch

**Republic of Korea** | KEB Automation KG  
Room 1709, 415 Missy 2000 725 Su Seo Dong  
Gangnam Gu 135- 757 Seoul Republic of Korea  
Tel: +82 2 6253 6771 Fax: +82 2 6253 6770  
E-Mail: vb.korea@keb.de

**Spain** | KEB Automation KG  
c / Mitjer, Nave 8 - Pol. Ind. LA MASIA  
08798 Sant Cugat Sesgarrigues (Barcelona) Spain  
Tel: +34 93 8970268 Fax: +34 93 8992035  
E-Mail: vb.espana@keb.de

**United States** | KEB America, Inc  
5100 Valley Industrial Blvd. South Shakopee, MN 55379 United States  
Tel: +1 952 2241400 Fax: +1 952 2241499  
E-Mail: info@kebamerica.com Internet: www.kebamerica.com

**Volksrepublik China** | KEB Power Transmission Technology Co. Ltd.  
No. 435 QianPu Road Chedun Town Songjiang District  
201611 Shanghai P.R. China  
Tel: +86 21 37746688 Fax: +86 21 37746600  
E-Mail: info@keb.cn Internet: www.keb.cn



**WEITERE KEB PARTNER WELTWEIT:**

...www.keb.de/de/contact/contact-worldwide



**Automation mit Drive**

**[www.keb.de](http://www.keb.de)**

KEB Automation KG   Südstraße 38   32683 Barntrop   Tel. +49 5263 401-0   E-Mail: [info@keb.de](mailto:info@keb.de)