

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

**Artisan Technology Group**
(217) 352-9330 | sales@artisantg.com | artisantg.com

**Full-service, independent repair center**
with experienced engineers and technicians on staff.

**We buy your excess, underutilized, and idle equipment**
along with credit for buybacks and trade-ins.

**Custom engineering**
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

**Expert team | Trust guarantee | 100% satisfaction**

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

**Visit our website - *Click HERE***

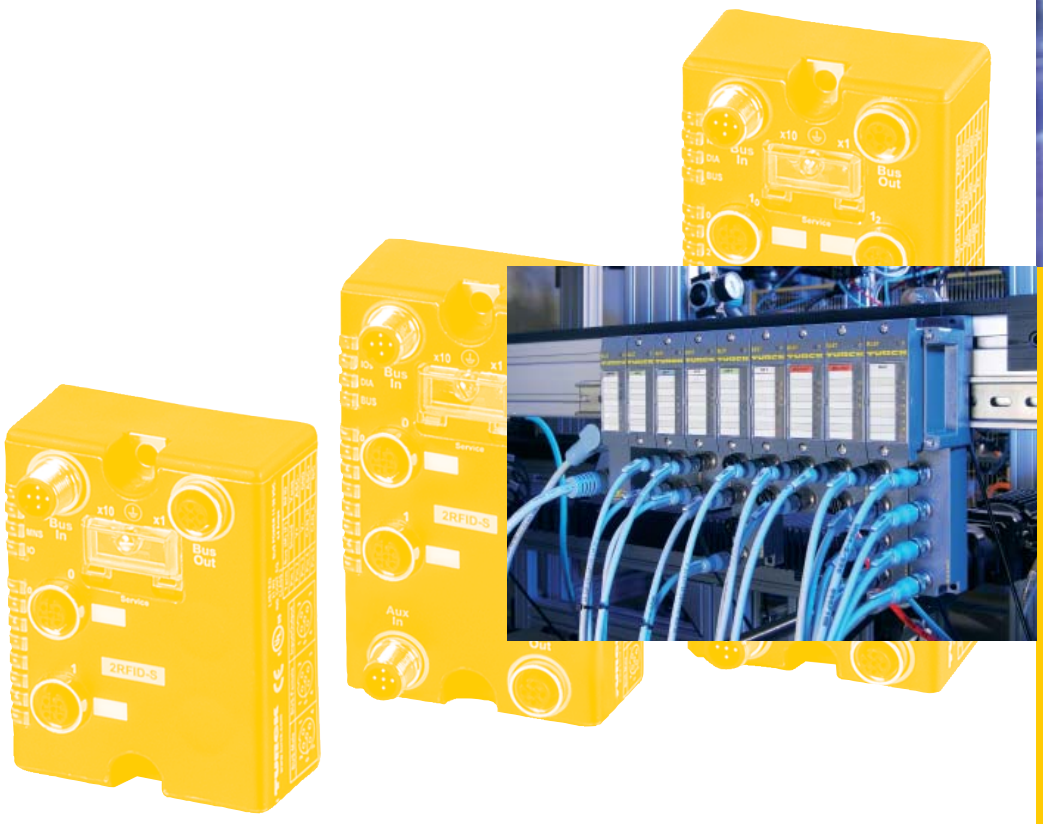


TURCK

Industrial Automation

# USER MANUAL BLxx – CANOPEN-OBJECT REGISTER





***Sense it! Connect it! Bus it! Solve it!***

All brand and product names are trademarks or registered trade marks of the owner concerned.

Edition 0511

© Hans Turck GmbH, Muelheim an der Ruhr

All rights reserved, including those of the translation.

No part of this manual may be reproduced in any form (printed, photocopy, microfilm or any other process) or processed, duplicated or distributed by means of electronic systems without written permission of Hans Turck GmbH & Co. KG, Muelheim an der Ruhr.

Subject to alterations without notice

## Warning!
## Before commencing the installation

- Disconnect the power supply of the device.
- Ensure that devices cannot be accidentally restarted.
- Verify isolation from the supply.
- Earth and short circuit.
- Cover or enclose neighboring units that are live.
- Follow the engineering instructions of the device concerned.
- Only suitably qualified personnel in accordance with EN 50 110-1/-2 (VDE 0 105 Part 100) may work on this device/system.
- Before installation and before touching the device ensure that you are free of electrostatic charge.
- The functional earth (FE) must be connected to the protective earth (PE) or to the potential equalization. The system installer is responsible for implementing this connection.
- Connecting cables and signal lines should be installed so that inductive or capacitive interference do not impair the automation functions.
- Install automation devices and related operating elements in such a way that they are well protected against unintentional operation.
- Suitable safety hardware and software measures should be implemented for the I/O interface so that a line or wire breakage on the signal side does not result in undefined states in the automation devices.
- Ensure a reliable electrical isolation of the low voltage for the 24 volt supply. Only voltage supplies which meet the requirements of IEC 60 364-4-41 or. HD 384.4.41 S2 (VDE 0100 part 410).
- Deviations of the mains voltage from the rated value must not exceed the tolerance limits given in the specifications, otherwise this may cause malfunction and dangerous operation.
- Emergency stop devices complying with IEC/EN 60 204-1 must be effective in all operating modes of the automation devices. Unlatching the emergency-stop devices must not cause restart.
- Devices that are designed for mounting in housings or control cabinets must only be operated and controlled after they have been installed with the housing closed.
- Measures should be taken to ensure the proper restart of programs interrupted after a voltage dip or failure. This should not cause dangerous operating states even for a short time. If necessary, emergency-stop devices should be implemented.
- Wherever faults in the automation system may cause damage to persons or property, external measures must be implemented to ensure a safe operating state in the event of a fault or malfunction (for example, by means of separate limit switches, mechanical interlocks etc.).
- The electrical installation must be carried out in accordance with the relevant regulations (e. g. with regard to cable cross sections, fuses, PE).
- All work relating to transport, installation, commissioning and maintenance must only be carried out by qualified personnel. (respect IEC 60 364 or HD 384 or DIN VDE 0100 and national work safety regulations).
- All shrouds and doors must be kept closed during operation.

# Table of Contents


**1 About this manual**

**1.1 Documentation concept ..... 1-2**

1.1.1 Additional documentation ..... 1-2

**1.2 General ..... 1-2**

1.2.1 Prescribed use ..... 1-2
1.2.2 Notes concerning planning/ installation of this product ..... 1-2

**1.3 Description of symbols used ..... 1-3**

**2 CANopen**

**2.1 CANopen ..... 2-2**

2.1.1 General ..... 2-2
2.1.2 Communication ..... 2-2

**2.2 EDS-file – Electronic data sheet ..... 2-5**

**2.3 BLxx and CANopen ..... 2-5**

2.3.1 Setting up communication ..... 2-5
2.3.2 Parameterization through Service Data Objects (SDO) ..... 2-10
2.3.3 Transmission of Process Data Objects (PDO) ..... 2-14
2.3.4 Inhibit Time ..... 2-16
2.3.5 Event Timer ..... 2-16
2.3.6 Mapping Objects in PDOs ..... 2-16
2.3.7 Commands for "Parameter Save" and "Restore Defaults" ..... 2-24

**3 Object dictionary - overview of all objects**

**3.1 Overview of all objects ..... 3-2**

**4 Device (gateway) objects**

**4.1 Objects of the Communication Profile (acc. to CiA DS-301) ..... 4-2**

4.1.1 Object $1000_{hex}$ – Device Type ..... 4-5
4.1.2 Object $1001_{hex}$ – Error Register ..... 4-5
4.1.3 Object $1005_{hex}$ – SYNC COB-ID ..... 4-6
4.1.4 Object $1008_{hex}$ – Device Name ..... 4-7
4.1.5 Object $1009_{hex}$ – Manufacturer Hardware Version ..... 4-8
4.1.6 Object $100A_{hex}$ – Manufacturer Software Version ..... 4-8
4.1.7 Object $100C_{hex}$ – Guard Time ..... 4-9
4.1.8 Object $100D_{hex}$ – Lifetime Factor ..... 4-9
4.1.9 Object $1010_{hex}$ – Store Parameters ..... 4-10
4.1.10 Object $1011_{hex}$ – Restore Default Parameters ..... 4-11
4.1.11 Object $1014_{hex}$ – Emcy COB-ID ..... 4-12
4.1.12 Object $1016_{hex}$ – Consumer Heartbeat Time ..... 4-13
4.1.13 Object $1017_{hex}$ – Producer Heartbeat Time ..... 4-15
4.1.14 Object $1018_{hex}$ – Identity Object ..... 4-16
4.1.15 Object $1020_{hex}$ – Verify Configuration ..... 4-17
4.1.16 Object $1027_{hex}$ – Module List ..... 4-19

**4.2 Objects for the Transfer of Service Data ..... 4-20**

4.2.1 Objects $1200_{hex}$ to $1203_{hex}$ – Server SDO Default Parameters ..... 4-20

**4.3 Objects for the Transfer of Process Output Data ..... 4-21**

4.3.1 Object $1400_{hex}$ to $141F_{hex}$ – Receive PDO Comm. Default Parameters ..... 4-21
4.3.2 Object $1600_{hex}$ to $161F_{hex}$ – Receive PDO-Mapping Parameter ..... 4-24

**4.4 Objects for the Transfer of Process Input Data ..... 4-27**

4.4.1 Object $1800_{hex}$ to $181F_{hex}$ – Transmit PDO-Parameters ..... 4-27
4.4.2 Objects $1A00_{hex}$ to $1A1F_{hex}$ - Transmit PDO Mapping Parameter ..... 4-30

**4.5 Objects for network management (not valid for BLCCO) ..... 4-33**

4.5.1 Object $1F80_{hex}$ – NMT Start-up ..... 4-33
4.5.2 Object $1F81_{hex}$ – Slave Assignment ..... 4-34
4.5.3 Object $1F82_{hex}$ – Request NMT ..... 4-36
4.5.4 Object $1F83_{hex}$ – Request Guarding ..... 4-38

**4.6 Overview about the objects of the Device Profile (acc. to CiA DS-401 and 406) ..... 4-39**

4.6.1 Object $67FF_{hex}$ – Device Type ..... 4-39
4.6.2 Object $6FFF_{hex}$ – Device Type ..... 4-40

**4.7 Manufacturer specific device objects ..... 4-40**

4.7.1 Object $2000_{hex}$ – Serial Number ..... 4-40
4.7.2 Object $2010_{hex}$ – Node ResetModifiers ..... 4-41
4.7.3 Object $2400_{hex}$ - System Voltages (only BL67) ..... 4-45
4.7.4 Object $2401_{hex}$ - System Currents (only BL67) ..... 4-45

## 5 Objects for digital input modules

**5.1 Digital input modules BL×× ..... 5-2**

**5.2 General object overview for digital input modules ..... 5-2**

5.2.1 Object $6000_{hex}$ – Read Input 8 Bit ..... 5-4
5.2.2 Object $6020_{hex}$ – Read Input Bit (1 to 128), Object $6021_{hex}$ – Read Input Bit (129 to 256), Object $6022_{hex}$ – Read Input Bit (257 to 288) ..... 5-5
5.2.3 Object $6100_{hex}$ – Read Input 16 Bit ..... 5-6
5.2.4 Object $6120_{hex}$ – Read Input 32 Bit ..... 5-6

## 6 Objects for digital output modules

**6.1 Digital output modules BL×× ..... 6-2**

**6.2 General object overview for digital output modules ..... 6-3**

6.2.1 Object $6200_{hex}$ – Write Output 8 Bit ..... 6-4
6.2.2 Object $6206_{hex}$ – Error Mode Output 8 Bit ..... 6-5
6.2.3 Object $6207_{hex}$ – Error State Output 8 Bit ..... 6-6
6.2.4 Object $6220_{hex}$ – Write Output Bit (1 to 128), Object $6221_{hex}$ – Write Output Bit (129 to 256), Object $6222_{hex}$ – Write Output Bit (257 to 288) ..... 6-7
6.2.5 Object $6250_{hex}$ – Error Mode Output Bit (1 to 128), Object $6251_{hex}$ – Error Mode Output Bit (129 to 256), Object $6252_{hex}$ – Error Mode Output Bit (257 to 288) ..... 6-8
6.2.6 Object $6260_{hex}$ – Error State Output Bit (1 to 128), Object $6261_{hex}$ – Error State Output Bit (129 to 256), Object $6262_{hex}$ – Error State Output Bit (257 to 288) ..... 6-9
6.2.7 Object $6300_{hex}$ – Write Output 16 Bit ..... 6-10
6.2.8 Object $6306_{hex}$ – Error Mode Output 16 Bit ..... 6-10
6.2.9 Object $6307_{hex}$ – Error State Output 16 Bit ..... 6-11
6.2.10 Object $6320_{hex}$ – Write Output 32 Bit ..... 6-12
6.2.11 Object $6326_{hex}$ – Error Mode Output 32 Bit ..... 6-13
6.2.12 Object $6327_{hex}$ – Error State Output 32 Bit ..... 6-14

**7 Objects for digital combi modules**

**7.1 Digital combi modules BL×× ..... 7-2**

**7.2 General object overview for digital combi modules ..... 7-2**

**8 Objects for analog input modules**

**8.1 Analog input modules BL×× ..... 8-2**

**8.2 General object overview for analog input modules ..... 8-2**

8.2.1 Object $5420_{hex}$ – Manu Spec Analog Input Range ..... 8-3
8.2.2 Object $6401_{hex}$ – Read Analog Input 16 Bit ..... 8-10
8.2.3 Object $6421_{hex}$ – Analog Input Interrupt Trigger Selection ..... 8-11
8.2.4 Object $6422_{hex}$ – Analog Input Interrupt Source ..... 8-13
8.2.5 Object $6423_{hex}$ – Analog Input Global Interrupt Enable ..... 8-14
8.2.6 Object $6424_{hex}$ – Analog Input Interrupt Upper Limit Integer ..... 8-14
8.2.7 Object $6425_{hex}$ – Analog Input Interrupt Lower Limit Integer ..... 8-15
8.2.8 Object $6426_{hex}$ – Analog Input Interrupt Delta Unsigned ..... 8-15
8.2.9 Object $6427_{hex}$ – Analog Input Interrupt Negative Delta Unsigned ..... 8-16
8.2.10 Object $6428_{hex}$ – Analog Input Interrupt Positive Delta Unsigned ..... 8-16

**9 Objects for analog output modules**

**9.1 Analog output modules BL×× ..... 9-2**

**9.2 General object overview for analog output modules ..... 9-2**

9.2.1 Object $5440_{hex}$ – Manu Spec Analog Output Range ..... 9-4
9.2.2 Object $6411_{hex}$ – Write Analog Output 16 Bit ..... 9-6
9.2.3 Object $6443_{hex}$ - Analog Output Error Mode ..... 9-7
9.2.4 Object $6444_{hex}$ – Analog Output Error State ..... 9-8

**10 Objects for RS232/RS4xx modules**

**10.1 RS××× modules BL×× ..... 10-2**

**10.2 Allgemeine Objektübersicht für RS232/RS4××-Module ..... 10-2**

10.2.1 Object $5600_{hex}$ – RS232/RS4xx Parameters ..... 10-2
10.2.2 Object $5601_{hex}$ – RS232/RS4xx RxD ..... 10-4
10.2.3 Object $5602_{hex}$ – RS232/RS4xx TxD ..... 10-7

**11 Objects for Encoder modules (SSI, CNT)**

**11.1 Encoder module BL×× ..... 11-4**

**11.2 General object overview for encoder modules ..... 11-4**

11.2.1 Object $5800_{hex}$ – Encoder Basic Mode ..... 11-6
11.2.2 Object $5801_{hex}$ – Encoder Config ..... 11-7
11.2.3 Object $5802_{hex}$ – Encoder Status ..... 11-11
11.2.4 Object $5803_{hex}$ – Encoder Flags ..... 11-13
11.2.5 Objekt $5804_{hex}$ – Encoder Diag ..... 11-15
11.2.6 Object $5805_{hex}$ – Encoder Native Status ..... 11-16
11.2.7 Object $5806_{hex}$ – Optional Encoder Status ..... 11-18
11.2.8 Object $5808_{hex}$ – Encoder Control ..... 11-19
11.2.9 Object $5810_{hex}$ – Encoder Load Prepare Value ..... 11-21
11.2.10 Object $5811_{hex}$ – Encoder Pulse Width ..... 11-22
11.2.11 Object $5820_{hex}$ – Measuring Integration Time ..... 11-22
11.2.12 Object $5821_{hex}$ – Measuring Low Limit ..... 11-23

11.2.13 Object $5822_{hex}$ – Measuring High Limit ... 11-25
11.2.14 Object $5823_{hex}$ – Measuring Units Per Revolution ... 11-25
11.2.15 Object $5824_{hex}$ – Measuring Divisor ... 11-26
11.2.16 Object $5825_{hex}$ – Measuring Factor ... 11-27
11.2.17 Object $5827_{hex}$ – Measuring Timeout ... 11-27
11.2.18 Object $5830_{hex}$ – Measuring Value ... 11-28
11.2.19 Object $5831_{hex}$ – Encoder Latch Value ... 11-29
11.2.20 Object $5840_{hex}$ – Diag Mapping ... 11-29
11.2.21 Object $5901_{hex}$ – PWM Config ... 11-31
11.2.22 Object $5902_{hex}$ – PWM Status ... 11-33
11.2.23 Object $5903_{hex}$ – PWM Flags ... 11-34
11.2.24 Object $5904_{hex}$ – PWM Diag ... 11-35
11.2.25 Object $5908_{hex}$ – PWM Control ... 11-36
11.2.26 Object $5910_{hex}$ – PWM Load Prepare Value ... 11-38
11.2.27 Object $5913_{hex}$ – PWM Duty Cycle ... 11-39
11.2.28 Object $5920_{hex}$ – PWM Period Duration ... 11-39
11.2.29 Object $5931_{hex}$ – PWM Latch Value ... 11-40
11.2.30 Object $6800_{hex}$ – Operating Parameters ... 11-40
11.2.31 Objekt $6810_{hex}$ – Preset Values for Multi-Sensor Devices ... 11-41
11.2.32 Object $6820_{hex}$ – Position Value ... 11-42
11.2.33 Object $6B00_{hex}$ – CAM State Register ... 11-42
11.2.34 Object $6B01_{hex}$ – CAM1 Enable Register ... 11-44
11.2.35 Object $6B02_{hex}$ – CAM Polarity Register ... 11-45
11.2.36 Object $6B10_{hex}$ – CAM1 Low Limit ... 11-46
11.2.37 Object $6B20_{hex}$ – CAM1 High Limit ... 11-47
11.2.38 Object $6B30_{hex}$ – CAM1 Hysteresis ... 11-48
11.2.39 Object $6C00_{hex}$ – Area State Register ... 11-49
11.2.40 Object $6C01_{hex}$ – Work Area Low Limit ... 11-50
11.2.41 Object $6C02_{hex}$ – Work Area High Limit ... 11-51
11.2.42 Object $6D00_{hex}$ – Operating Status, Object $6D01_{hex}$ – SingleTurn Resolution (rotary), Measuring step (linear), Object $6D02_{hex}$ – Number of Distinguishable Revolutions ... 11-52
11.2.43 Object $6FFF_{hex}$ – Device Type ... 11-52

**11.3 Process output/ control interface of the Encoder modules ... 11-52**

11.3.1 Meaning of the process output bits of BL××-1SSI ... **11-52**
11.3.2 Meaning of the process output bits of BL20-1CNT-24VDC ... 11-53
11.3.3 Meaning of the process output bits of BL20-E-2CNT-2PWM ... 11-56
11.3.4 Meaning of the process output bits of BL67-1CNT/ENC ... 11-58

**11.4 Meaning of the process input bits of the Encoder modules ... 11-59**

11.4.1 Meaning of the process input bits of BL××-1SSI ... **11-59**
11.4.2 Meaning of the process input bits of BL20-1CNT-24VDC ... 11-61
11.4.3 Meaning of the process input bits of BL20-E-2CNT-2PWM ... 11-63
11.4.4 Meaning of the process input bits of BL67-1CNT/ENC ... 11-66

**11.5 Parameter interface of the Encoder modules ... 11-67**

11.5.1 Meaning of the parameter bits of BL××-1SSI ... **11-67**
11.5.2 Meaning of the parameter bits of BL20-1CNT-24VDC ... 11-69
11.5.3 Meaning of the parameter bits of BL20-E-2CNT-2PWM ... 11-71
11.5.4 Meaning of the parameter bits of BL67-1CNT/ENC ... 11-74

**11.6 Diagnostic interface of the Encoder modules ... 11-76**

11.6.1 Meaning of the diagnostic bits of BL××-1SSI ... **11-76**
11.6.2 Meaning of the diagnostic bits of BL20-1CNT-24VDC ... 11-77
11.6.3 Meaning of the diagnostic bits of BL20-E-2CNT-2PWM ... 11-78
11.6.4 Meaning of the diagnostic bits of BL67-1CNT/ENC ... 11-79

**12 Objects for SWIRE modules**

**12.1 Motor starter modules BL20 ..... 12-2**

**12.2 General object overview for SWIRE modules ..... 12-2**

12.2.1 Representation of process input data ..... 12-3
12.2.2 Representation of process output data ..... 12-4
12.2.3 Representation of diagnostic data ..... 12-6
12.2.4 Representation of parameter data ..... 12-10

**13 Objects for RFID-modules**

**13.1 RFID-S-module BL20 ..... 13-2**

**13.2 General object overview for RFID-S-modules ..... 13-2**

**13.3 Object $5700_{hex}$ - 8 byte process input data ..... 13-2**

**13.4 Object $5701_{hex}$ - 12 byte process input data ..... 13-3**

**13.5 Object $5702_{hex}$ - 8 byte process output data ..... 13-3**

**13.6 Object $5703_{hex}$ - 12 byte process output data ..... 13-4**

**13.7 Object $5708_{hex}$ - 1 byte status messages ..... 13-4**

**13.8 Object $5722_{hex}$ - 1 byte parameter ..... 13-5**

**14 Manufacturer specific objects**

**14.1 General ..... 14-2**

14.1.1 Module related manufacturer specific objects ..... 14-2
14.1.2 Slot-related manufacturer specific objects ..... 14-4
14.1.3 Process input objects ..... 14-5
14.1.4 Process output objects ..... 14-6
14.1.5 Diagnosis objects ..... 14-8
14.1.6 Parameter objects ..... 14-9

**15 Diagnostics - Emergency Frames**

**15.1 General ..... 15-2**

**15.2 Structure of the emergency frames ..... 15-2**

15.2.1 Emergency Error-Codes ..... 15-2
15.2.2 Error register ..... 15-4

**15.3 Emergency codes for module diagnostics ..... 15-5**

15.3.1 General module error codes ..... 15-5
15.3.2 Digital output modules ..... 15-5
15.3.3 Analog Input Modules ..... 15-6
15.3.4 Technology modules ..... 15-8

**16 Index**

# 1 About this manual

**1.1 Documentation concept ..... 2**

1.1.1 Additional documentation ..... 2
– BL67 ..... 2
– BL20 ..... 2
– BLC ..... 2

**1.2 General ..... 2**

1.2.1 Prescribed use ..... 2
1.2.2 Notes concerning planning/ installation of this product ..... 2

**1.3 Description of symbols used ..... 3**

## 1.1 Documentation concept

This manual describes the CANopen object directory for TURCK BLxx-products.

If not marked especially, the object descriptions are valid for all products of the product lines BL20, BL67 and BLcompact.

### 1.1.1 Additional documentation

**BL67**

- D301006 - BL67 – User manual for CANopen
- Data sheets for the BL20-products www.turck.com

**BL20**

- D301108 BL20 – ECO gateway for CANopen
- D301109 - BL20 – User manual for CANopen
- Data sheets for the BL20-products www.turck.com

**BLC**

- Manuals and data sheets for the BLC-CANopen-products www.turck.com

## 1.2 General



**Attention**

Please read this section carefully. Safety aspects cannot be left to chance when dealing with electrical equipment.

### 1.2.1 Prescribed use

Appropriate transport, storage, deployment and mounting as well as careful operating and thorough maintenance guarantee the trouble-free and safe operation of these devices.



**Danger**

The devices described in this manual must be used only in applications prescribed in this manual or in the respective technical descriptions, and only with certified components and devices from third party manufacturers.

### 1.2.2 Notes concerning planning/ installation of this product



**Danger**

All respective safety measures and accident protection guidelines must be considered carefully and without exception.

## 1.3 Description of symbols used



**Danger**

This sign can be found next to all notes that indicate a source of hazards. This can refer to danger to personnel or damage to the system (hardware and software) and to the facility.

This sign means for the operator: work with extreme caution.



**Attention**

This sign can be found next to all notes that indicate a potential hazard.

This can refer to possible danger to personnel and damages to the system (hardware and software) and to the facility.



**Note**

This sign can be found next to all general notes that supply important information about one or more operating steps.

These specific notes are intended to make operation easier and avoid unnecessary work due to incorrect operation.

# 2 CANopen


**2.1 CANopen ..... 2**

2.1.1 General ..... 2
2.1.2 Communication ..... 2
– Network Management Messages ..... 2
– Service Data Objects (SDOs) ..... 3
– Process Data Objects (PDOs) ..... 3
– Special Function Objects ..... 4

**2.2 EDS-file – Electronic data sheet ..... 5**

**2.3 BL×× and CANopen ..... 5**

2.3.1 Setting up communication ..... 5
– Minimum Boot-up ..... 5
– Identifier for the Standard Objects ..... 8
– Set up Node Guarding Protocol ..... 9
– Boot-up Message ..... 10
2.3.2 Parameterization through Service Data Objects (SDO) ..... 10
– Write (Write to Object Dictionary) ..... 12
– Parameter Storing/Restoring ..... 13
2.3.3 Transmission of Process Data Objects (PDO) ..... 14
– Communication parameter COB-ID ..... 14
– Transmission Type ..... 15
2.3.4 Inhibit Time ..... 16
2.3.5 Event Timer ..... 16
– Available PDOs ..... 16
2.3.6 Mapping Objects in PDOs ..... 16
– Default-PDOs and PDO-mappings ..... 17
– BL××-spezifische Default-PDOs (gilt nicht für BL compact) ..... 18
– Mappable Objects ..... 22
– Procedure for Altering PDO-Mappings ..... 23
2.3.7 Commands for "Parameter Save" and "Restore Defaults" ..... 24

## 2.1 CANopen

**Note**

The following description of CANopen is an excerpt from the homepage of CiA (CAN in Automation), the international users' and manufacturers' organization for CAN.

### 2.1.1 General

CANopen is an open, non-proprietary network protocol. It consists of a profile family, based on a communication profile and several device profiles. The CANopen communication profile is standardized as CiA DS-301 (Application Layer and Communication Profile).

The CANopen device profile for I/O-modules has been published as CiA DS-401 (Device Profile for I/O-Modules).

CANopen is based on the following standards:

- ISO 11 898 (Physical and Data Link Layer)Layers 1 and 2 of the ISO/OSI communication model
- CiA DS-301 (Application Layer and Communication Profile) CANopen communication profile
- CiA DS-302 (Framework for Programmable CANopen Devices) CANopen Network Management NMT
- CiA DS-401 (Device Profile for I/O-modules)
- CiA DS-406 (Device Profile for Encoders) CANopen device profile for counter modules
- CiA DS-102 (CAN Physical Layer for Industrial Applications) General application in the field sector (connectors and bit rates) on the basis of ISO 11898

### 2.1.2 Communication

The lower layers of CANopen are defined according to the ISO-OSI model in the ISO 11898 standard.

Communication between the individual nodes is made by transmitting "Telegrams".

4 different types of telegram message are defined for CANopen:

- Network management messages
- Service data objects SDO
- Process data objects PDO
- Predefined messages

**Network Management Messages**

Network management messages are used in the network to control the nodes and their operating states. This type of message makes it possible, for instance, to configure the data transmission mechanism of a node.

The Network Management objects include Boot-up message, Heartbeat protocol and NMT message.

Boot-up message, Heartbeat and Node Guarding are implemented as single CAN frames with 1-byte data field.

**The NMT message** is mapped to a single CAN frame with a data length of 2 byte. The CAN-Identifier is 0. The first byte contains the command specifier and the second contains the Node-ID of the device that must perform the command (in the case of Node-ID 0 all nodes have to perform the command). The NMT message transmitted by the NMT master forces the nodes to transit to another NMT state. CANopen defines the following statsus: "Initialization", "Pre-Operational", "Operational" and "Stopped".

After a "power-on", each CANopen devices is in the status "Initialization" and automatically changes to the Pre-Operational status . In this state the transmission of SDOs is allowed. If the NMT master has set one or more nodes into the state Operational, they are allowed to transmit and to receive PDOs. In the state Stopped no communication is allowed except that of NMT objects.

The state Initialization is devided into 3 sub-states. in order to enable a complete or partial reset of a node. In the sub-state Reset Application the parameters of the manufacturer-specific profile area and the standardized device profile area are set to their power-on values. In the sub-state Reset Communication the parameters of the communication profile area are set to their power-on values. The third sub-state is initializing, which a node enters automatically after power-on. Power-on values are the last stored parameters.

The **Heartbeat** protocol and **Node Guarding** (see Set up Node Guarding Protocol (page 2-9)) are for error control purposes and signals the presence of a node and its state. The Heartbeat message is a periodic message of the node to one or several other nodes. It indicates that the sending node is still working properly.

A device sends the **Boot-up message** to indicate to the NMT master that it has changed from „Initialization“ tot he state Pre-operational. This occurs whenever the device initially boots-up but also after a power-out during operation. The Boot-up message has the same identifier as the Heartbeat object, however, its data content is zero.

### Service Data Objects (SDOs)

A Service Data Object (SDO) reads from entries or writes to entries of the Object Dictionary.

The SDO transport protocol allows transmitting objects of any size. The first byte of the first segment contains the necessary flow control information including a toggle bit to overcome the problem of doubly received CAN frames. The next three bytes of the first segment contain index and sub-index of the Object Dictionary entry to be read or written. The last four bytes of the first segment are available for user data. The second and the following segments (using the very same CAN identifier) contain the control byte and up to seven bytes of user data. The receiver confirms each segment or a block of segments, so that a peer-to-peer communication (client/server) takes place.

### Process Data Objects (PDOs)

Process Data Objects (PDOs) are mapped to a single CAN frame using up to 8 bytes of the data field to transmit application objects. Each PDO has a unique identifier and is transmitted by only one node, but it can be received by more than one (producer/consumer communication). PDO transmissions may be driven by an internal event, by an internal timer, by remote requests and by the Sync message received:

#### PDO transmissions

- „Event-“ or „timer-driven“:
  An event (specified in the device profile) triggers message transmission. An elapsed timer additionally triggers the periodically transmitting of PDO-messages, even if no event has ocurred.
- Remotely requested (RTR):
  Another device may initiate the transmission of an asynchronous PDO by sending a remote transmission request (remote frame).
- Sync Master (Synchronous transmission:)
  In order to initiate simultaneous sampling of input values of all nodes, a periodically transmitted Sync message is required. Synchronous transmission of PDOs takes place in cyclic and acyclic transmission mode. Cyclic transmission means that the node waits for the Sync message, after which it sends its measured values. Acyclically transmitted synchronous PDOs are triggered by a defined application-specific event. The device transmits it's input values. Further transmission is only done if a further Sync messages occurs.

### Special Function Objects

CANopen also defines three specific protocols for synchronization, emergency indication, and time-stamp transmission.

- **Synchronization object (Sync)**
  The Sync Object is broadcast periodically by the Sync Producer. This object is a central timer. The Sync Object is broadcast periodically by the Sync Producer. The time period between Sync messages is defined by the Communication Cycle Period, which may be reset by a configuration tool to the application devices during the boot-up process. There can be a time jitter in transmission by the Sync Producer due to some other objects with higher prior identifiers or by one frame being transmitted just before the Sync message. The Sync message is mapped to a single CAN frame with the identifier 128 by default.
- **Emergency object (Emcy)**
  The Emergency message is triggered by the occurrence of a device internal error situation and are transmitted from an Emergency producer on the concerned application device. This makes them suitable for interrupt type error alerts. An Emergency message is transmitted only once per 'error event'. As long as no new errors occurs on a device, no further Emergency message can be transmitted. Zero or more Emergency consumers may receive these. The reaction of the Emergency consumer is application-specific. CANopen defines several Emergency Error Codes to be transmitted in the Emergency message, which is a single CAN frame with 8 data byte.
- **Time stamp object (Time)**
  By means of Time-Stamp, a common time frame reference is provided to application devices. It contains a value of the type Time-of-Day. This object transmission follows the producer/consumer push model. The associated CAN frame has the pre-defined identifier 256 and a data field of 6-byte length.

## 2.2 EDS-file – Electronic data sheet

CANopen nodes are embedded in the CANopen structure by the help of a standardized EDS file (Electronic Data Sheet).

The EDS file lists all necessary Objects with their corresponding Sub-indices and the matching entries.

The latest version of a particular EDS file can be downloaded directly from the TURCK homepage www.turck.com.

## 2.3 BLxx and CANopen

BLxx supports the following CANopen-functions:

- SDO transfer, any length of information
- Emergency object
- Sync frame evaluation
- Event-driven PDOs
- Synchronous PDOs (clock-synchronous)
- Remote-requested PDO/polling

### 2.3.1 Setting up communication

**Minimum Boot-up**

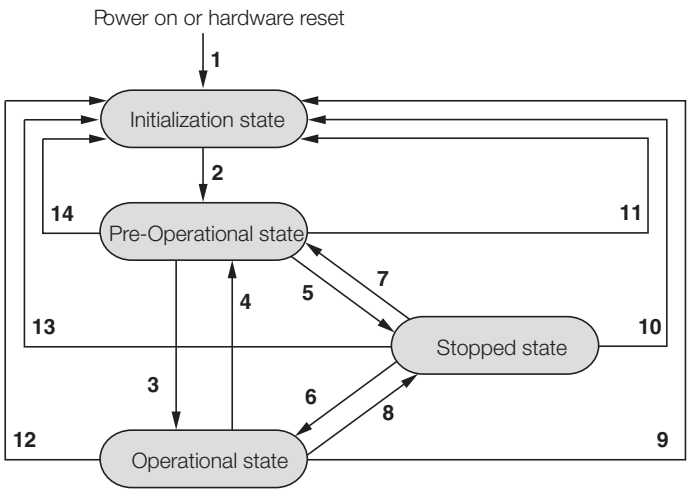
BLxx supports the Minimum Boot-up function described in CiA DS-301.

*Table 1: Meaning of the abbreviations*

| Abbreviation | Meaning | Description |
|---|---|---|
| cs | NMT command specifier | A designation label for the required service |
| Node-ID | Node Identifier | Identifier for the node; an identification byte that is set through the DIP switches for the CAN node. |

Booting with the Minimum Boot-up function is the typical application option for CANopen, and runs according to the following state diagram:

*Figure1: Boot procedure with Minimum Boot-up*



**1** Power-on (automatic change of state to the "Initialization" condition)

**2** Initialization Finished (automatic change of state to "Pre-Operational")

**3** Start Remote Node (start the CAN node)

**4** Enter Pre-Operational (change over to "Pre-Operational")

**5** Stop Remote Node (stop the CAN node)

**6** Start Remote Node (start the CAN node)

**7** Enter Pre-Operational (change over to "Pre-Operational")

**8** Stop Remote Node (stop the CAN node)

**9** Reset Node (reset the complete CAN node)

**10** Reset Node (reset the complete CAN node)

**11** Reset Node (reset the complete CAN node)

**12** Reset Communication (reset communication for the CAN node)

**13** Reset Communication (reset communication for the CAN node)

**14** Reset Communication (reset communication for the CAN node)

The following messages are exchanged in the states mentioned:

- Operational: PDO and SDO communication
- Pre-Operational: only SDO-communication

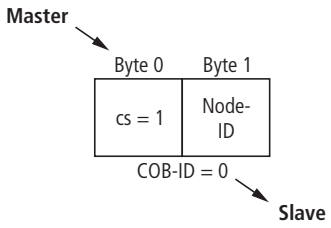
The services listed above (1 to 14) are required by CANopen or performed independently by the nodes in order to change from one state to another.

The "Stopped" state can be skipped when using Minimum Boot-up.

**1** Power-on (automatic change of state to the "Initialization" condition)

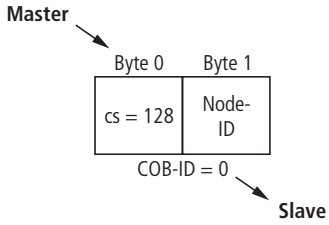
**2** Initialization finished (automatic change of state to "Pre-Operational")

**3, 6** Start Remote Node (start the CAN node)



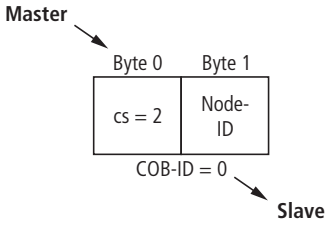

The internal change of state of the CANopen slave now requires a **pause of at least 20 ms**, before another request may be made by the master.

**15, 7** Enter Pre-Operational (change over to "Pre-Operational")



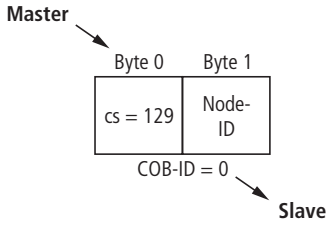

The internal change of state of the CANopen slave now requires a **pause of at least 20 ms**, before another request may be made by the master.

**15, 8** Stop Remote Node (stop the CAN node)



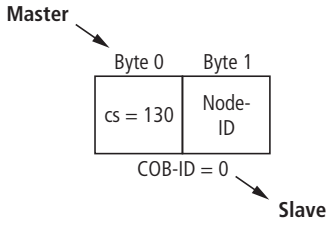

The internal change of state of the CANopen slave now requires a **pause of at least 20 ms**, before another request may be made by the master.

**15, 10, 11** Reset Node (reset the complete CAN node)




The execution of this command is confirmed by a boot-up message. This is in the form of a guard frame with the data contents $00_{hex}$.

**15, 13, 14** Reset Communication (reset communication for the CAN node)




The execution of this command is confirmed by a boot-up message. This is in the form of a guard frame with the data contents $00_{hex}$.

### Identifier for the Standard Objects

#### Node-ID

The identifier for each device in a CANopen network is the Node-ID. he CANopen slaves can be assigned the Node-IDs 1 to 127. The maximum number of Node-IDs to be set may vary depending on the BLxx-product.

#### COB-ID (Communication Object Identifier)

The identifier for each communication object in a CANopen network is the COB-ID. The COB-IDs for the standard objects (digital input, digital output, analog input, analog output) are assigned automatically. The ranges for the COB-IDs are defined by the "Predefined Master-Slave Connection Set".

Each range for the COB-IDs has 127 numerical values.

The COB-IDs are calculated according to the following rule:

**COB-ID = Base-ID + Node-ID**

Base-ID: 128; 384; 512; 640; 768; 896; 1024; 1152; 1280; 1408; 1536; 1792

Node-ID: max. 1 to 127

*Table 2: Identifiers for basic objects*

| **COB-ID** dec. | hex. | **Function** | **Application** |
|---|---|---|---|
| 0 | $000_{hex}$ | Network Management (NMT) | Broadcast object |
| 01 to 127 | $001_{hex}$ to $07F_{hex}$ | free | |
| 128 | $080_{hex}$ | Synchronization (Sync) | Broadcast object |
| 129 to 255 | $081_{hex}$ to $0FF_{hex}$ | Emergency Message | |
| 256 | $100_{hex}$ | Timestamp | Broadcast object |
| 257 to 384 | $101_{hex}$ to $180_{hex}$ | free | |
| 385 to 511 | $181_{hex}$ to $1FF_{hex}$ | Transmit PDO 1 | Digital input |
| 512 | $200_{hex}$ | free | |
| 513 to 639 | $201_{hex}$ to $27F_{hex}$ | Receive PDO 1 | Digital output |
| 640 | $280_{hex}$ | free | |
| 641 to 767 | $281_{hex}$ to $2FF_{hex}$ | Transmit PDO 2 | Analog input |
| 768 | $300_{hex}$ | free | |
| 769 to 895 | $301_{hex}$ to $37F_{hex}$ | Receive PDO 2 | Analog output |
| 896 | $380_{hex}$ | free | |
| 897 to 1023 | $381_{hex}$ to $3FF_{hex}$ | Transmit PDO 3 | Analog input |
| 1024 | $400_{hex}$ | free | |
| 1025 to 1151 | $401_{hex}$ to $47F_{hex}$ | Receive PDO 3 | Analog output |
| 1152 | $480_{hex}$ | free | |
| 1153 to 1279 | $481_{hex}$ to $4FF_{hex}$ | Transmit PDO 4 | Analog input |
| 1280 | $500_{hex}$ | free | |

*Table 2: Identifiers for basic objects*

| COB-ID dec. | COB-ID hex. | Function | Application |
|---|---|---|---|
| 1281 to 1407 | $501_{hex}$ to $57F_{hex}$ | Receive PDO 4 | Analog output |
| 1408 | $580_{hex}$ | free | |
| 1409 to 1535 | $581_{hex}$ to $5FF_{hex}$ | Transmit SDO | |
| 1536 | $600_{hex}$ | free | |
| 1537 to 1663 | $601_{hex}$ to $67F_{hex}$ | Receive SDO | |
| 1664 to 1772 | $680_{hex}$ to $6EC_{hex}$ | free | |
| 1793 to 1919 | $701_{hex}$ to $77F_{hex}$ | NMT Error (Node Guarding, Heartbeat, Boot-Up) | |
| 1920 to 2014 | $800_{hex}$ to $7DE_{hex}$ | free | |
| 2015 to 2031 | $7DF_{hex}$ to $7EF_{hex}$ | NMT, LMT, DBT | |

### Set up Node Guarding Protocol

**Note**

Further information on Node Guarding can be found in CiA DS-301.

Node Guarding is the name for the monitoring of network nodes by a network manager. You distinguish between active and passive Node Guarding.

In addition, the CANopen network nodes check that their network manager is operating correctly and that the network is functioning reliably.

In the default state, Node Guarding is inactive.

#### Active Node Guarding

To activate the Node Guarding protocol for a node, various parameters must be set for the Object Dictionary:

- [100C] = Guard time
  Given in milliseconds; the query interval (polling) that is to be expected by the network slave.
  Default = 0
- [100D] = Lifetime factor
  This factor, multiplied by the Guard time, is the time that should elapse after a Node Guarding protocol error before the network slave generates an error message via EMCY. In this way, a temporary communication problem, such as may be caused by heavy bus loading, can be bridged without a Guarding Error.
  Default = 0
- Guard-ID
  This is fixed and cannot be changed.

Guarding is initiated with the first Guard-Remote frame (Guarding-RTR) from the CANopen network manager.

The Guarding Frame of the network manager has the COBID "1793 - 1 + Node-ID" and does **not** have a data field.

Furthermore, the RTR bit in the message header must be set and the Data Length code = 1.

The node answers the telegram sent out by the network manager within the preset time (Guard time) in the "Operational" state, with the data contents **5**. The gateway answers the next polling query with the contents **133**. The following response from the gateway is with **5** again, and so on. This means that the gateway changes the state of the most significant bit after every query (i.e. the bit is toggled). If the node is in the "Pre-Operational" state, then the value of the data contents of the response telegram toggles between 127 and 255. If the node is in the "Stop" state, the value toggles between 4 and 132.

If there is no query from the network manager within the preset time, then the gateway changes to the state "Guard Fail". If output modules are fitted in the BL×× station, then their outputs will be put into defined states, depending on the objects "Error mode output" and "Error state output", or will retain the last state that was received. Any RxPDOs that are received will still be processed and output again. If the Guarding starts up again, the gateway leaves the "Guard Fail" state, but remains in the Pre-Operational state. A "Start Node" command must be generated by the network manager in order to restart the gateway (see CiA DS-301).

**Passive Node Guarding**

If a Guard-Time = 0 is set (see Object 100Chex – Guard Time (page 4-9)), than, passive guarding is activated. This means that the gateway answers the Guard-Remote-Frames without itself starting a Guard-Timer. The gateway does not change to the Pre-Operational state.

As an alternative to Node-/Life-Guarding, the Heartbeat mechanism (see Object 1016hex – Consumer Heartbeat Time (page 4-13) and Object 1017hex – Producer Heartbeat Time (page 4-15)) newly introduced with DS301 V4.0 is supported, which, unlike Guarding, does not require Remote frames.

**Boot-up Message**

After initialization (after Power-On, Reset-Node and Reset-Communication), a Boot-up message as per CiA DS-301 V4.0 is sent out. This is in the form of a guard frame with the contents $00_{hex}$.

Under certain circumstances, a network manager may fail to detect a short drop-out of an BL20 gateway (for example, as a result of voltage variations). This could occur under the following conditions:

- The drop-out and initialization of the gateway happen in the time between two Guarding-Frames
- The gateway was already in the Pre-Operational state beforehand
- The last state of the toggle bit was 1

If a Boot-up message is sent out after a reset or initialization, then the drop-out mentioned above will also not be missed by the network manager.

### 2.3.2 Parameterization through Service Data Objects (SDO)

SDO (= Service Data Object) is a confirmed CANopen service that is primarily used for parameterization and configuration of the CANopen slaves (BL××) and less frequently for transmitting process data. "Confirmed" means that an BL××-CANopen gateway (SDO server) that is addressed by this procedure must acknowledge it through a response. In this way, the SDO client obtains information about whether the BL×× gateway that it addressed was contacted, and whether the access was achieved without any errors (error code in the response from the SDO server). SDO access means that the contents of the Object Dictionary entries for an SDO server can be read or written, and that the settings for a BL×× station can be made in this way.

Four parallel SDO servers are supported. There are three "additional" SDOs, as well as the default SDO. As a default, these are inactive, but can be parameterized and enabled through the Object Dictionary entries $1201_{hex}$ to $1203_{hex}$.

The communication parameters for the default SDO follow the Predefined Connection Set, and cannot be modified (see CiA DS-301, V4.01).

In the following representations of the messages, the identifier of the CANopen message that is to be sent can be found below the frame, and the contents of the data byte to be transmitted are within the frame.

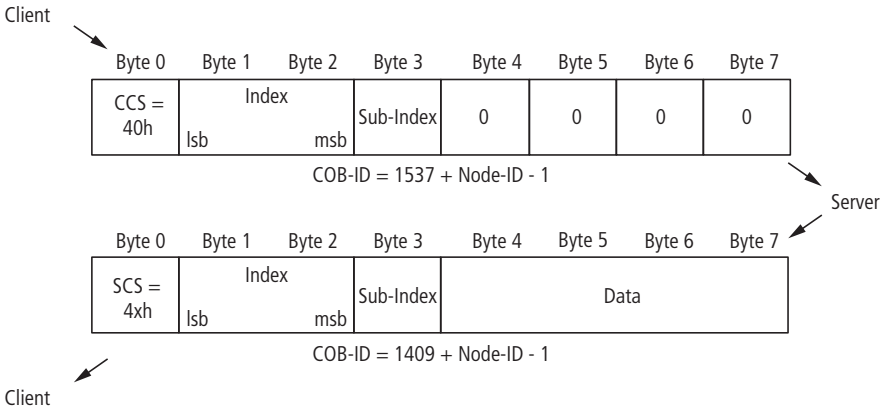
The following representations use the Expedited SDO Transfer, i.e. a maximum of 4 bytes of user data can be transferred within one telegram.

**Note**

CANopen also offers the possibility of segmented SDO-transfer of data with data length of more than 4 bytes.

### Read (Read from Object Dictionary)

*Figure2: Read (Read from Object Dictionary)*




x... depending on the length of data read

LSB= Least Significant byte ? lowest value byte

MSB= Most Significant byte ? highest value byte

SCS = Server Command Specifier

CCS = Client Command Specifier

(see CiA DS-301)

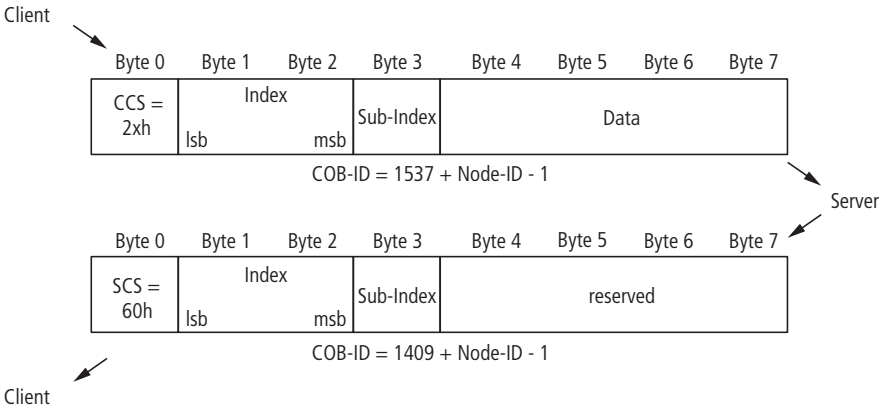
The stated COB-ID refers to the default SDO server.

**Note**

The information in byte 0 "SCS " can optionally contain the length information for the transmitted data bytes (see CiA DS-301, Page 9-21 ff). The information in byte 0 "SCS = $4x_{hex}$" means that no length information is present.

### Write (Write to Object Dictionary)

*Figure3: Write (Write to Object Dictionary)*




LSB= Least Significant byte ? lowest value byte

MSB= Most Significant byte ? highest value byte

SCS = Server Command Specifier

CCS = Client Command Specifier

(see CiA DS-301)

The stated COB-ID refers to the default SDO server.

**Note**

The information in byte 0 "SCS " can optionally contain the length information for the transmitted data bytes (see CiA DS-301). The information in byte 0 „SCS = 60h“ means that no length information is present.

**Attention**

If an incorrect data length is given, the error code "Abort SDO Transfer Service" will be generated (see CiA DS-301).

*Table 3: Abort codes for errors in SDO transfer*

| Abort code | Description |
|---|---|
| 0503 $0000_{hex}$ | Toggle bit not altered. |
| 0504 $0001_{hex}$ | Client server command specifier not valid or unknown. |
| 0601 $0000_{hex}$ | Unsupported access to an object. |
| 0601 $0001_{hex}$ | Attempt to write a read only object. |
| 0601 $0002_{hex}$ | Attempt to read a write only object. |
| 0602 $0000_{hex}$ | Object does not exist in the object dictionary. |
| $06040041_{hex}$ | Object cannot be mapped to the PDO. |
| $06040042_{hex}$ | The number and length of objects exceeds PDO length. |

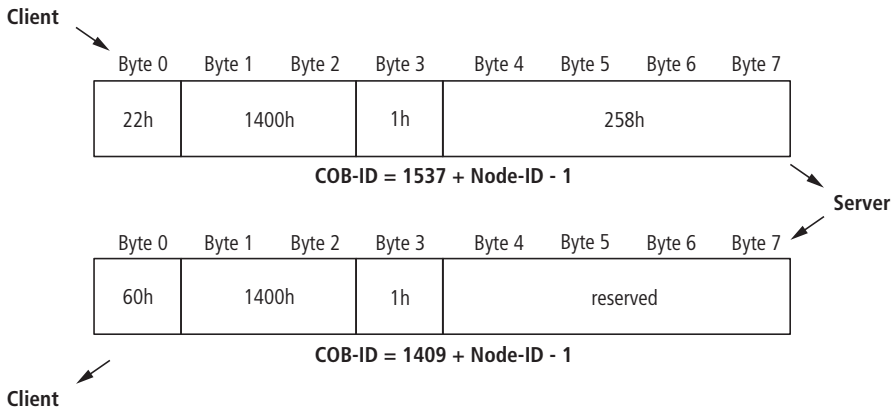
*Table 3: Abort codes for errors in SDO transfer*

| Abort code | Description |
|---|---|
| $06040043_{hex}$ | General parameter incompatibility reason. |
| $06040047_{hex}$ | General internal incompatibility in the device. |
| $06070010_{hex}$ | Data type does not match - wrong length. |
| $0607\ 0012_{hex}$ | Data type does not match- length too high. |
| $0607\ 0013_{hex}$ | Data type does not match- length too low. |
| $06090011_{hex}$ | Sub-index does not exist. |
| $06090030_{hex}$ | Value range of parameter exceeded. |
| $06090031_{hex}$ | Value range of parameter written too high. |
| $06090032_{hex}$ | Value range of parameter written too low |
| $06090036_{hex}$ | Maximum value is less than minimum value. |
| $08000000_{hex}$ | Other error |
| $08000020_{hex}$ | Data cannot be stored to the application. |
| $08000021_{hex}$ | Data cannot be stored to the app. because of local control. |
| $08000022_{hex}$ | Data cannot be stored to the app. because of device state. |

**Example:**

Write a new COB-ID for RxPDO 1 (ID = $258_{hex}$)

*Figure4: Example COB-ID*




### Parameter Storing/Restoring

Saving of communication and application parameters is executed by a command. This means that the parameters transferred through an SDO are held in volatile memory, until they are saved by using the command "Store parameters" (Object $1010_{hex}$, Sub-indices 0 to 3). All the communication and application parameters that are supported by the gateway will be saved.

The command "Restore Default parameters" (Object $1011_{hex}$, Sub-indices 0 to 3) is also supported. This command resets all the communication and/or application parameters to the default values.

### 2.3.3 Transmission of Process Data Objects (PDO)

PDOs are fast real-time process data that are handled as unconfirmed services without a protocol overhead.

We distinguish between:

- Transmit PDOS = TPDOs
- Receive PDOs = RPDOs

PDOs can contain a maximum of 8 bytes of data. They can be assembled and configured by the user to suit the specific requirements. In addition, there are a number of transmission/transfer settings (Transmission types) for process data.

The following attributes can be set for each PDO through the object "PDO communication parameter":

**Communication parameter COB-ID**

The COB-ID is the CAN identifier that is used for the transmission of a PDO (object $1800_{hex}$ ff and $1400_{hex}$ ff.).

COB-IDs are used to define the priority of the message telegrams. The lowest COB-ID has the highest priority.

For communication between 2 nodes, the COB-ID of the transmit PDO must be the same as the COB-ID of the receive PDO..



**Note**

As delivered, each BL×× gateway has a maximum number of 8 active PDOs (0 to 8 are possible), with COB-IDs that are taken from the Predefined Master-Slave Connection Set. The compact modules of the BLCCO product line provide up to four active PDOs.

All other PDOs are inactive. This state can be seen from the invalid bit (bit 31) of the COB-ID (see also The COB-ID entries in the tables on page 2-17 ff.).

### Transmission Type

The Transmission type determines under which circumstances a PDO can be transmitted or received.

The following PDO Transmission types are supported by BL××:

- Type 0 (sync, acyclic)
- Type 1 (sync, cyclic)
- Type 253 (Remote Request)
- Type 255 (Event-Driven)

*Table 4: Overview of PDO Transmission Types*

| Transmission Type | PDO transmission | | | | |
|---|---|---|---|---|---|
| | cyclic | acyclic | synchr. | asynchr. | only with RTR |
| 0 | | × | × | | |
| 1 | × | | × | | |
| 253 | | | | × | × |
| 255 | | | | × | |

**Type 0**
The PDO will always be transmitted (TPDO) or evaluated (RPDO) if this is permitted by a Sync-Frame transmitted by the SYNC producer and the mapped contents of the BL××-CANopen gateway have changed since the last transmission.

**Type 1**
Immediately after receiving each Sync-Frame, the BL××-CANopen gateway puts out the mapped contents as a PDO on the network, even if they have not changed since the last transmission.

**Type 253**
The PDO is only transmitted if a transmitted Remote-Frame requests this from the BL××-CANopen gateway.



**Attention**
This Transmission Type is only allowed for TPDOs.

**Type 255**

In this mode of operation, the BL××-CANopen gateway does not depend on any Sync or Remote-Request for PDO communication. Whenever this is envisaged for an internal event within the BL××CANopen gateway, the gateway will transmit a PDO to the CANopen network.

The Transmission types of the individual PDOs are independent, which means that a freely mixed operation with synchronous and asynchronous PDOs is possible.

### 2.3.4 Inhibit Time

The setting of an Inhibit time for the PDOs (Object $1800_{hex}$ ff, Sub-Index $03_{hex}$) is only supported for TPDOs. Unlike the other time values, which are given as multiples of 1 ms, the Inhibit time is defined as a multiple of 100 µs. However, since the time resolution of the system clock in the BL××-CANopen gateway is 1 ms, Inhibit time values below 10 x 100 µs are pointless.

### 2.3.5 Event Timer

The Event timer (Object $1800_{hex}$ ff, Sub-Index $05_{hex}$) defines the maximum interval after which a TPDO will be transmitted, even though no event has occurred. This means that the Event timer determines the maximum interval between two transmissions of a TPDO.

The expiry of the interval set for the Event timer is detected as an event. If any other event occurs, the Event timer is reset and restarted.

The value of the object is interpreted as a multiple of 1 ms.

**Available PDOs**

BL××:
64 PDOs are supported:

- 32 Transmit-PDOs: TPDO1 to TPDO32 (Index $1800_{hex}$ to $181F_{hex}$)
- 32 Receive-PDOs: RPDO1 to RPDO32 (Index $1400_{hex}$ to $141F_{hex}$)

BLCCO:
8 PDOs are supported:

- 4 Transmit-PDOs: TPDO1 to TPDO4 (Index $1800_{hex}$ to $1803_{hex}$)
- 4 Receive-PDOs: RPDO1 to RPDO4 (Index $1400_{hex}$ to $1403_{hex}$)

The corresponding Default Master-Slave Connection Set is supported for each of the PDOs 1 to 4, so that a COB-ID distribution is not necessary for these PDOs.

If one of the COB-IDs from xPDO1 to xPDO4 is reconfigured, then the use of a COB-ID from the Default Master-Slave Connection Set can be achieved by setting this COB-ID to 0.

### 2.3.6 Mapping Objects in PDOs

Mapping is the assignment of objects from an Object Dictionary in a PDO for transmission/reception through the CAN-bus. More than one object can be transmitted in a single PDO.

The Mapping parameters determine which items of information are transmitted in a PDO:

*Table 5: Object Dictionary for mapping parameters*

| **PDO** | | **Object Dictionary entries** | |
|---|---|---|---|
| **Type** | **Product family** | **Range** | **Range** |
| Transmit-PDOs | BL20/BL67 | TPDO1 to TPDO32 | $1A00_{hex}$ to $1A1F_{hex}$ |
| | BLCCO | TPDO1 to TPDO4 | $1A00_{hex}$ to $1A03_{hex}$ |
| Receive-PDOs | BL20/BL67 | RPDO1 to RPDO32 | $1600_{hex}$ to $161F_{hex}$ |
| | BLCCO | RPDO1 to RPDO4 | $1600_{hex}$ to $1603_{hex}$ |

### Default-PDOs and PDO-mappings

The 4 Transmit and 4 Receive-PDOs which are specified by the Communication Profile CiA DS-301 are supported by BL××. The mapping of these PDOs and their Transmission types are specified by the I/O-Device Profile CiA DS-401.

**Note**

BL67/BL20:
The Default-PDOs are only activated if the planned objects and sub-indices actually exist for the corresponding PDO. If, in a station, for example no analog I/Os are used, then the PDOs 2 to 4 are set to "Invalid" and no mapping entries will be present.

In addition to the default PDOs which are standardized by the CiA DS-301 and DS-401 profiles, other PDOs for a -CANopen gateway may be provided with mapping entries and communication parameters.

These additional PDOs (5 to 16) will be set to "Invalid" as a default (see Table 6: Overview Default-TPDOs as per CiA DS-301 und DS-401 to Table 9: Overview of the BL××-specific Default-RPDOs).

### Default-PDOs as per CiA DS-301 und DS-401

The TPDOs in the following table have the following characteristics:

- The COB-ID is part of sub-index $01_{hex}$
- The PDO is active!
  The first digit of the 8-digit hexadecimal COB-ID-number shows amongst others, if the PDO is valid. Active PDOs are marked by a hex-digit 7. Normally, the digit is 0 or 4. → Description of the COB-ID entry (sub-index 01hex ) (page 4-29).

**Overview of the Default-TPDOs as per CiA DS-301 and DS-401**

*Table 6: Overview Default-TPDOs as per CiA DS-301 und DS-401*

| Meaning | TPDO | Sub-Index $01_{hex}$ - "COB-ID" |
|---|---|---|
| 1 group, digital input channels, (bits 0 to 63) | PDO1 $1800_{hex}$ | 0000 $0180_{hex}$ + Node-ID |
| 1 group, analog input channels, (channels 0 to 3) | PDO2 $1801_{hex}$ | 0000 $0280_{hex}$ + Node-ID |
| 2 group, analog input channels, (channels 4 to 7) | PDO3 $1802_{hex}$ | 0000 $0380_{hex}$ + Node-ID |
| 3 group, analog input channels, (channels 8 to 11) | PDO4 $1803_{hex}$ | 0000 $0480_{hex}$ + Node-ID |

**Overview of the Default-RPDOs as per CiA DS-301 and DS-401**

*Table 7: Overview Default-RPDOs as per CiA DS-301 und DS-401*

| Meaning | RPDO | COB-ID |
|---|---|---|
| 1 group, digital output channels, (bits 0 to 63) | PDO1 $1400_{hex}$ | 0000 $0200_{hex}$ + Node-ID |
| 1 group, analog output channels, (channels 0 to 3) | PDO2 $1401_{hex}$ | 0000 $0300_{hex}$ + Node-ID |
| 2 group, analog output channels, (channels 4 to 7) | PDO3 $1402_{hex}$ | 0000 $0400_{hex}$ + Node-ID |
| 3 group, analog output channels, (channels 8 to 11) | PDO4 $1403_{hex}$ | 0000 $0500_{hex}$ + Node-ID |

**BLxx-spezifische Default-PDOs (gilt nicht für BL compact)**

These additional PDOs will be set to "Invalid" as a default.

Before enabling these PDOs, the corresponding parameters must be checked. This applies especially to the COB-IDs, since these are taken from the Default Master-Slave Connection Set, and are assigned to other Node-IDs. For this reason, other nodes with the corresponding Node-ID must not be present in the network, or such nodes must not use the corresponding COB-IDs.

The Transmission type of these PDOs is generally 255.

**Overview of the BLxx-specific Default-TPDOs (not valid for BL compact)**



**Note**

The COB-ID definition for the TPDOs depends on the gateway used in the application (see EDS file for the gateways).

*Table 8: Overview of the BL××-specific Default-TPDOs*

| | | BL20- -GW-CO | BL67-GW-CO/ BL20-GWBR-CANopen |
|---|---|---|---|
| **Meaning** | **TPDOs** | **COB-ID TPDO** | **COB-ID TPDO** |
| 2 group digital input channels (bits 64 to 127) | PDO5 $1804_{hex}$ | 8000 $01C0_{hex}$ + Node-ID | 8000 $01C0_{hex}$ + Node-ID |
| 3 group digital input channels (bits 128 to 191) | PDO6 $1805_{hex}$ | 8000 $02C0_{hex}$ + Node-ID | 8000 $02C0_{hex}$ + Node-ID |
| 4 group digital input channels (bits 192 to 255) | PDO7 $1806_{hex}$ | 8000 $03C0_{hex}$ + Node-ID | C000 $03C0_{hex}$ + Node-ID |
| 5 group digital input channels (bits 256 to 319) | PDO8 $1807_{hex}$ | 8000 $04C0_{hex}$ + Node-ID | C000 $04C0_{hex}$ + Node-ID |
| 1 group encoders (channels 0 +1) | PDO9 $1808_{hex}$ | 8000 $01E0_{hex}$ + Node-ID | C000 $01E0_{hex}$ + Node-ID |
| 2 group encoders (channels 2 3) | PDO10 $1809_{hex}$ | 8000 $02E0_{hex}$ + Node-ID | C000 $02E0_{hex}$ + Node-ID |
| 3 group encoders (channels 4 5) | PDO11 $180A_{hex}$ | 8000 $03E0_{hex}$ + Node-ID | C000 $03E0_{hex}$ + Node-ID |
| 4 group encoders (channels 6 7) | PDO12 $180B_{hex}$ | 8000 $04E0_{hex}$ + Node-ID | C000 $04E0_{hex}$ + Node-ID |
| 4 group analog input channels (channels 12 to 15) | PDO13 $180C_{hex}$ | 8000 $01A0_{hex}$ + Node-ID | C000 $01A0_{hex}$ + Node-ID |
| 5 group analog input channels (channels 16 to 19) | PDO14 $180D_{hex}$ | 8000 $02A0_{hex}$ + Node-ID | C000 $02A0_{hex}$ + Node-ID |
| 6 group analog input channels (channels 20 to 23) | PDO15 $180E_{hex}$ | 88000 000 $03A0_{hex}$ + Node-ID | C8000 000 $03A0_{hex}$ + Node-ID |
| 7 group analog input channels (channels 24 to 27) | PDO16 $180F_{hex}$ | 8000 $04A0_{hex}$ + Node-ID | C000 $04A0_{hex}$ + Node-ID |
| 1. group RS×× (channel 0) | PDO18 $1811_{hex}$ | 8000 $0000_{hex}$ | C000 $0000_{hex}$ |
| 1. group RS×× (channel 1) | PDO19 $1812_{hex}$ | 8000 $0000_{hex}$ | C000 $0000_{hex}$ |

**Overview of the BL××-specific Default-TPDOs (not valid for BL compact)**

*Table 9: Overview of the BL××-specific Default-RPDOs*

| Meaning | RPDOs | COB-ID RPDO |
|---|---|---|
| 2 group, digital input I/Os, (bits 64 to 127) | PDO5<br>$1404_{hex}$ | 8000 $0240_{hex}$ + Node-ID |
| 3 group, digital input I/Os, (bits 128 to 191) | PDO6<br>$1405_{hex}$ | 8000 $0340_{hex}$ + Node-ID |
| 4 group, digital input I/Os, (bits 192 to 255) | PDO7<br>$1406_{hex}$ | 8000 $0440_{hex}$ + Node-ID |
| 5 group, digital input I/Os, (bits 256 to 319) | PDO8<br>$1407_{hex}$ | 8000 $0540_{hex}$ + Node-ID |
| 1 group encoders (channel 0 + 1) | PDO9<br>$1408_{hex}$ | 8000 $0260_{hex}$ + Node-ID |
| 2 group encoders (channel 2 + 3) | PDO10<br>$1409_{hex}$ | 8000 $0360_{hex}$ + Node-ID |
| 3 group encoders (channel 4 + 5) | PDO11<br>$140A_{hex}$ | 8000 $0460_{hex}$ + Node-ID |
| 4 group encoders (channel 6 + 7) | PDO12<br>$140B_{hex}$ | 8000 $0560_{hex}$ + Node-ID |
| 4 group analog I/Os, (channels 12 to 15) | PDO13<br>$140C_{hex}$ | 8000 $0220_{hex}$ + Node-ID |
| 5 group analog I/Os, (channels 16 to 19) | PDO14<br>$140D_{hex}$ | 8000 $0320_{hex}$ + Node-ID |
| 6 group analog I/Os, (channels 20 to 23) | PDO15<br>$140E_{hex}$ | 8000 $0420_{hex}$ + Node-ID |
| 7 group analog I/Os, (channels 24 to 27) | PDO16<br>$140F_{hex}$ | 8000 $0520_{hex}$ + Node-ID |
| 1. group RS×× (channel 0) | PDO18<br>$1411_{hex}$ | 8000 $0000_{hex}$ |
| 1. group RS×× (channel 1) | PDO19<br>$1412_{hex}$ | 8000 $0000_{hex}$ |



**Attention**

The COB-ID for the RS×××-Module must be defined by the user!

**Example (not valid for BLCCO)**

The own Node-ID of an BL××-CANopen gateway is 1. There are more than 12 analog input channels. As a result, appropriate mapping entries are set up for TPDO13 (Object $1A0C_{hex}$), and the COB-ID (Object 180C, Sub-Index 1) is pre-loaded with the value 8000 $01A1_{hex}$. This PDO can only be enabled without alteration if a node with the Node-ID 33 (own Node-ID + 32) does not exist, or at least its TPDO1 is not used.+

The following table illustrates the systematic relationship:

*Table 10: Relationship between a Node-ID and BL20-specific PDOs*

| **PDO** | **Node-ID**<br>assigned to this COB-ID in the Default Master-Slave Connection Set | **Original PDO**<br>assigned to this COB-ID in the Default Master-Slave Connection Set |
|---|---|---|
| PDO5 | own Node-ID + 64 ($40_{hex}$) | PDO1 |
| PDO6 | own Node-ID + 64 ($40_{hex}$) | PDO2 |
| PDO7 | own Node-ID + 64 ($40_{hex}$) | PDO3 |
| PDO8 | own Node-ID + 64 ($40_{hex}$) | PDO4 |
| PDO9 | own Node-ID + 96 ($60_{hex}$) | PDO1 |
| PDO10 | own Node-ID + 96 ($60_{hex}$) | PDO2 |
| PDO11 | own Node-ID + 96 ($60_{hex}$) | PDO3 |
| PDO12 | own Node-ID + 96 ($60_{hex}$) | PDO4 |
| PDO13 | own Node-ID + 32 ($20_{hex}$) | PDO1 |
| PDO14 | own Node-ID + 32 ($20_{hex}$) | PDO2 |
| PDO15 | own Node-ID + 32 ($20_{hex}$) | PDO3 |
| PDO16 | own Node-ID + 32 ($20_{hex}$) | PDO4 |

### Mappable Objects

The maximum of 64 mapping entries per PDO that is specified by Communication Profile CiA DS-301 is supported.

The following objects from the Object Dictionary can be mapped:

*Table 11: Overview of mappable objects*

| Name | Index | Sub-index | Direction |
|---|---|---|---|
| Dummy Mapping Boolean | $0001_{hex}$ | - | Receive |
| Dummy Mapping Boolean Integer8 | $0002_{hex}$ | - | Receive |
| Dummy Mapping Boolean Integer16 | $0003_{hex}$ | - | Receive |
| Dummy Mapping Boolean Integer32 | $0004_{hex}$ | - | Receive |
| Dummy Mapping Boolean Unsigned8 | $0005_{hex}$ | - | Receive |
| Dummy Mapping Boolean Unsigned16 | $0006_{hex}$ | - | Receive |
| Dummy Mapping Boolean Unsigned32 | $0007_{hex}$ | - | Receive |
| Error register | $1001_{hex}$ | – | Transmit |
| Manu Spec Analog Input Range | $5420_{hex}$ | 1 to n **A** | Transmit |
| RS232/RS4xx RxD | $5601_{hex}$ | 1 to n | Receive |
| RS232/RS4xx TxD | $5602_{hex}$ | 1 to n | Transmit |
| RFID Input Data U64 | 5700 | 1 to n | |
| RFID Output Data U64 | 5702 | 1 to n | |
| RFID Status | 5708 | 1 to n | |
| Encoder Status | $5802_{hex}$ | 1 to n | Transmit |
| Encoder Flags | $5803_{hex}$ | 1 to n | Transmit |
| SSI Native Status | $5805_{hex}$ | 1 to n | Transmit |
| SSI Optional Encoder Status | $5806_{hex}$ | 1 to n | Transmit |
| Encoder Control | $5808_{hex}$ | 1 to n | Receive |
| PwmStatus | $5902_{hex}$ | 1 to n | |
| PwmFlags | $5903_{hex}$ | 1 to n | |
| PwmControl | $5908_{hex}$ | 1 to n | |
| PwmPeriodDuration | $5920_{hex}$ | 1 to n | |
| Read Input 8 Bit | $6000_{hex}$ | 1 to n | Transmit |
| Read Input Bit (1 to 128) **A** | $6020_{hex}$ | 1 to n | Transmit |
| Read Input Bit (129 to 256) | $6021_{hex}$ | 1 to n | Transmit |
| Read Input Bit (257 to 288) | $6022_{hex}$ | 1 to n | Transmit |

**A** *"n" depends on the used object as well as on the used product*

*Table 11: Overview of mappable objects*

| Name | Index | Sub-index | Direction |
|---|---|---|---|
| Read Input 16 Bit | $6100_{hex}$ | 1 to n | Transmit |
| Read Input 32 Bit | $6120_{hex}$ | 1 to n | Transmit |
| Write Output 8 Bit | $6200_{hex}$ | 1 to n | Receive |
| Write Output Bit (1 to 128) **A** | $6220_{hex}$ | 1 to n | Receive |
| Write Output Bit (129 to 256) A | $6221_{hex}$ | 1 to n | Receive |
| Write Output Bit (257 to 288) | $6222_{hex}$ | 1 to n | Receive |
| Write Output 16 Bit | $6300_{hex}$ | 1 to n | Receive |
| Write Output 32 Bit | $6320_{hex}$ | 1 to n | Receive |
| Read Analog Input 16 Bit | $6401_{hex}$ | 1 to n | Transmit |
| Write Analog Output 16 Bit | $6411_{hex}$ | 1 to n | Receive |
| Preset Value for Multi-Sensor Devices | $6810_{hex}$ | 1 to n | Transmit |
| Position Value for Multi-Sensor Devices | $6820_{hex}$ | 1 to n | Transmit |
| CAM1 State Register | $6B00_{hex}$ | 1 to n | Transmit |
| Area State Register | $6C00_{hex}$ | 1 to n | Transmit |

### Procedure for Altering PDO-Mappings

The Communication Profile CiA DS-301 Version 4 defines a detailed procedure for altering PDO-mappings.

For the gateway, this results in the following method for modifying PDO-mappings:

- The node state of the gateway must be "Pre-Operational".
- The number of mapping entries (Sub-index 0) for a PDO must be set to 0.
- The mapping entries (Sub-index 1 to 64) can be written.
- The number of mapping entries (Sub-index 0) must now be set to the valid number of mapped objects.
- Alternatively, the new mapping can be saved in non-volatile memory (Store Communication parameters).

The following abort codes (Abort-Domain-Protocol) may be sent back by the gateway in case of an error:

*Table 12: Abort code*

| Abort code | Description as per CiA DS-301 | Possible cause |
|---|---|---|
| $0604\ 0041_{hex}$ | Object cannot be mapped | Invalid object-index transferred while writing the mapping entries. |
| $0604\ 0042_{hex}$ | Number or length of the objects exceeds the PDO length | Attempted to map too many or excessively long objects to a PDO. This will be returned on writing to the sub-index 0. |

*Table 12: Abort code*

| Abort code | Description as per CiA DS-301 | Possible cause |
|---|---|---|
| 0609 0011$_{hex}$ | Sub-index does not exist. | A sub-index > 64 was addressed. |
| 0800 0022$_{hex}$ | Access not possible in this node state | Write access is only possible in the "Pre-operational" node state. Write access to Sub-indices 1 to 64 is only possible if sub-index 0 is written with value 0. |

### 2.3.7 Commands for "Parameter Save" and "Restore Defaults"

Parameter changes that are made through SDO access are only stored in volatile memory. All alterations that are made by the user will be replaced by default values at the next Reset Communication, Reset Node or Power-ON-Boot-Up.

With BL××0, it is possible to use a command to make a permanent save of the communication and/or application parameters. This is done through the "Store parameters" command (Object 1010$_{hex}$ sub-index 1 to 3).

The command is executed by using an SDO to write the data contents 6576 6173$_{hex}$ ("save") to one of the following entries:

- 1010$_{hex}$ sub-index 1 saves all parameters
- 1010$_{hex}$ sub-index 2 saves all parameters
- 1010$_{hex}$ sub-index 3 saves all device parameters (see CiA DS-301 V4.01)

*Table 13: Data contents 0x6576 6173 („save“)*

| | MSB | | | LSB |
|---|---|---|---|---|
| **ASCII** | e | v | a | s |
| hex | 65$_{hex}$ | 76$_{hex}$ | 61$_{hex}$ | 73$_{hex}$ |

Since in some circumstances it may not be possible to restore the original memory contents after a lot of alterations, BL×× supports the "Restore default Parameter" command (Object 1011$_{hex}$ sub-index 1 to 3) with the following data contents: 6461 6F6C$_{hex}$ ("load").

*Table 14: Data contents 0x6461 6F6C („load“)*

| | MSB | | | LSB |
|---|---|---|---|---|
| **ASCII** | d | a | o | l |
| hex | 64$_{hex}$ | 61$_{hex}$ | 6F$_{hex}$ | 6C$_{hex}$ |

The division of the Sub-entries corresponds to that for the "Store parameters" command.

After the command "Restore default parameters", a Reset Node must be carried out, followed by a "Store parameters" command. The default values are only saved again when this last command is executed.

# 3 Object dictionary - overview of all objects

3.1 Overview of all objects ..... 2

## 3.1 Overview of all objects

The following table provides an overview of all objects supported by the BLxx-CANopen gateway.

*Table 15: Overview of all objects*

| Index | Name | BLxx-GWBR-CANOPEN | B67-GW-CANOPEN | BLxx-E-GW-CO | BLCCO-xxx | |
|---|---|---|---|---|---|---|
| **Objects (according to CiA CS-301)** | | | | | | |
| $1000_{hex}$ | Device Type | X | X | X | X | see chapter 4 |
| $1001_{hex}$ | Error register | X | X | X | X | |
| $1005_{hex}$ | SYNC COB-ID | X | X | X | X | |
| $1008_{hex}$ | Device Name | X | X | X | X | |
| $1009_{hex}$ | Manufacturer Hardware Version | X | X | X | X | |
| $100A_{hex}$ | Manufacturer Software Version | X | X | X | X | |
| $100C_{hex}$ | Guard Time | X | X | X | X | |
| $100D_{hex}$ | Lifetime Factor | X | X | X | X | |
| $1010_{hex}$ | Store Parameters | X | X | X | X | |
| $1011_{hex}$ | Restore Default Parameters | X | X | X | X | |
| $1014_{hex}$ | Emcy COB-ID | X | X | X | X | |
| $1016_{hex}$ | Consumer Heartbeat Time | X | X | X | X | |
| $1017_{hex}$ | Producer Heartbeat Time | X | X | X | X | |
| $1018_{hex}$ | Identity Object | X | X | X | X | |
| $1020_{hex}$ | Verify Configuration | X | X | X | X | |
| $1027_{hex}$ | Module List | X | X | X | | |
| $1200_{hex}$ to $1203_{hex}$ | Server SDO Parameters | X | X | X | X | |
| $1400_{hex}$ to $1403_{hex}$ | Receive PDO Communication Parameters (1 to 4) | X | X | X | X | |
| $1404_{hex}$ to $141F_{hex}$ | Receive PDO Communication Parameters (5 to 32) | X | X | X | | |
| $1600_{hex}$ to $1603_{hex}$ | Receive PDO-Mapping Parameters (1 to 4) | X | X | X | X | |
| $1604_{hex}$ to $161F_{hex}$ | Receive PDO-Mapping Parameters (5 to 32) | X | X | X | | |

*Table 15: Overview of all objects*

| Index | Name | BLxx-GWBR-CANOPEN | B67-GW-CANOPEN | BLxx-E-GW-CO | BLCCO-xxx | |
|---|---|---|---|---|---|---|
| $1800_{hex}$ to $1803_{hex}$ | Transmit PDO-Parameters (1 to 4) | X | X | X | X | see chapter 4 |
| $1800_{hex}$ to $181F_{hex}$ | Transmit PDO-Parameters (5 to 32) | X | X | X | | |
| $1A00_{hex}$ to $1A03_{hex}$ | Transmit PDO-Mapping Parameters (1 to 4) | X | X | X | X | |
| $1A04_{hex}$ to $1A1F_{hex}$ | Transmit PDO-Mapping Parameters (5 to 32) | X | X | X | | |
| $1F80_{hex}$ | NMT Startup | X | X | X | | |
| $1F81_{hex}$ | Slave Assignment | X | X | X | | |
| $1F82_{hex}$, | Request NMT | X | X | X | | |
| $1F83_{hex}$ | Request Guarding | X | X | X | | |
| **Manufacturer specific objects** | | | | | | |
| $2000_{hex}$ | Serial Number | X | X | X | X | |
| $2010_{hex}$ | Node Reset Modifiers | X | X | X | X | |
| $2400_{hex}$ | System Voltages | | X | | | |
| $2401_{hex}$ | System Voltages | | X | | | |
| $3000_{hex}$ | XBIInputByte | X | X | X | X | see chapter 14 |
| $3002_{hex}$ | XBIInputWord | X | X | X | X | |
| $3004_{hex}$ to $300B_{hex}$ | XBIInputDWord0 to XBIInputDWord8 | X | X | X | X | |
| $3020_{hex}$ | XBIOutputByte | X | X | X | X | |
| $3022_{hex}$ | XBIOutputByte | X | X | X | X | |
| $3024_{hex}$ to $302B_{hex}$ | XBIOutputDWord0 to XBIOutputDWord8 | X | X | X | X | |
| $3040_{hex}$ | XBIDiagByte | X | X | X | X | |
| $3042_{hex}$ | XBIDiagWord | X | X | X | X | |
| $3044_{hex}$ to $304B_{hex}$ | XBIDiagDWord0 to XBIDiagDWord8 | X | X | X | X | |
| $3060_{hex}$ | XBIParamByte | X | X | X | X | |

*Table 15: Overview of all objects*

| Index | Name | BLxx-GWBR-CANOPEN | B67-GW-CANOPEN | BLxx-E-GW-CO | BLCCO-xxx | |
|---|---|---|---|---|---|---|
| $3062_{hex}$ | XBIParamWord | X | X | X | X | see chapter 14 |
| $3064_{hex}$ to $306B_{hex}$ | XBIParamDWord0 to XBIParamDWord8 | X | X | X | X | |
| $3081_{hex}$ | XBIReferenceModuleType | X | X | X | X | |
| $3084_{hex}$ | XBIReferenceInputSize | X | X | X | X | |
| $3085_{hex}$ | XBIReferenceOutputSize | X | X | X | X | |
| $3086_{hex}$ | XBIReferenceDiagSize | X | X | X | X | |
| $3087_{hex}$ | XBIReferenceParamSize | X | X | X | X | |
| $3090_{hex}$ | XBICurrentModuleId | X | X | X | X | |
| $3091_{hex}$ | XBICurrentModuleType | X | X | X | X | |
| $3094_{hex}$ | XBICurrentInputSize | X | X | X | X | |
| $3095_{hex}$ | XBICurrentOutputSize | X | X | X | X | |
| $3096_{hex}$ | XBICurrentDiagSize | X | X | X | X | |
| $3097_{hex}$ | XBICurrentParamSize | X | X | X | X | |
| $5420_{hex}$ | Analog Input Mode | X | X | X | X | see chapter 8 |
| $5440_{hex}$ | Analog Output Mode | X | X | X | X | see chapter 9 |
| $5600_{hex}$ | RS232/RS4xx parameters | X | X | X | X | see chapter 10 |
| $5601_{hex}$ | RS232/RS4xx RxD | X | X | X | X | |
| $5602_{hex}$ | RS232/RS4xx TxD | X | X | X | X | |
| $5801_{hex}$ | Encoder Config | X | X | X | X | see chapter 11 |
| $5802_{hex}$ | Encoder Status | X | X | X | X | |
| $5803_{hex}$ | Encoder Flags | X | X | X | X | |
| $5804_{hex}$ | Encoder Diag | X | X | X | X | |
| $5805_{hex}$ | Encoder Native Status | X | X | X | X | |
| $5806_{hex}$ | Encoder Optional Encoder | X | X | X | X | |
| $5808_{hex}$ | Encoder Control | X | X | X | X | |
| $5810_{hex}$ | Encoder Load Prepare Value | X | | X | | |

*Table 15: Overview of all objects*

| Index | Name | BLxx-GWBR-CANOPEN | B67-GW-CANOPEN | BLxx-E-GW-CO | BLCCO-xxx | |
|---|---|---|---|---|---|---|
| $5811_{hex}$ | Encoder Pulse Width | X | | X | | see chapter 11 |
| $5820_{hex}$ | Measuring Integration Time | X | | X | | |
| $5821_{hex}$ | Measuring Low Limit | | | | | |
| $5822_{hex}$ | Measuring High Limit | | | | | |
| $5823_{hex}$ | Measuring Units Per Revolution | | | | | |
| $5824_{hex}$ | Encoder Measuring Divisor | X | | X | | |
| $5825_{hex}$ | Encoder Measuring Factor | X | | X | | |
| $5827_{hex}$ | Encoder Measuring Time Out | X | | X | | |
| $5830_{hex}$ | Encoder Measuring Value | X | | X | | |
| $5831_{hex}$ | Encoder Latch Value | X | | X | | |
| $5840_{hex}$ | SSI Diag Mapping | X | X | X | | |
| $5901_{hex}$ | PWM Config | X | | X | | |
| $5902_{hex}$ | PWM Status | X | | X | | |
| $5903_{hex}$ | PWM Flags | X | | X | | |
| $5904_{hex}$ | PWM Diag | X | | X | | |
| $5908_{hex}$ | PWM Control | X | | X | | |
| $5910_{hex}$ | PWM Load Prepare Value | X | | X | | |
| $5913_{hex}$ | PWM Duty Cycle | X | | X | | |
| $5920_{hex}$ | PWM Period Duration | X | | X | | |
| $5931_{hex}$ | PWM Latch Value | X | | X | | |
| **I/O module objects (according to CiA CS-401)** | | | | | | |
| $6000_{hex}$ | Read Input 8 Bit | X | X | X | X | see chapter 5 and chapter 7 |
| $6020_{hex}$ | Read Input Bit 1 to 128 | X | X | X | X | |
| $6021_{hex}$ | Read Input Bit 129 to 256 | X | X | X | X | |
| $6022_{hex}$ | Read Input Bit 257 to 288 | X | X | X | X | |
| $6100_{hex}$ | Read Input 16 Bit | X | X | X | X | |
| $6120_{hex}$ | Read Input 32 Bit | X | X | X | X | |

*Table 15: Overview of all objects*

| Index | Name | BLxx-GWBR-CANOPEN | B67-GW-CANOPEN | BLxx-E-GW-CO | BLCCO-xxx | |
|---|---|---|---|---|---|---|
| $6200_{hex}$ | Write Output 8 Bit | X | X | X | X | see chapter 6 and chapter 7 |
| $6206_{hex}$ | Error Mode Output 8 Bit | X | X | X | X | |
| $6207_{hex}$ | Error Value Output 8 Bit | X | X | X | X | |
| $6220_{hex}$ to $6222_{hex}$ | Write Output Bit 1 to 128 to Write Output Bit 257 to 288 | X | X | X | X | |
| $6250_{hex}$ to $6252_{hex}$ | Error Mode Output Bit 1 to 128 to Error Mode Output Bit 257 to 288 | X | X | X | X | |
| $6260_{hex}$ to $6262_{hex}$ | Error Value Output Bit 1 to 128 to Error Value Output Bit 257 to 288 | X | X | X | X | see chapter 6 |
| $6300_{hex}$ | Write Output 16 Bit | X | X | X | X | |
| $6306_{hex}$ | Error Mode Output 16 Bit | X | X | X | X | |
| $6307_{hex}$ | Error Value Output 16 Bit | X | X | X | X | |
| $6320_{hex}$ | Write Output 32 Bit | X | X | X | X | |
| $6326_{hex}$ | Error Mode Output 32 Bit | X | X | X | X | |
| $6327_{hex}$ | Error Value Output 32 Bit | X | X | X | X | |
| $6401_{hex}$ | Read Analog Input 16 Bit | X | X | X | X | see chapter 8 |
| $6411_{hex}$ | Write Analog Output 16 Bit | X | X | X | X | see chapter 9 |
| $6421_{hex}$ | Analog Input Interrupt Trigger Auswahl | X | X | X | X | see chapter 8 |
| $6422_{hex}$ | Analog Input Interrupt Source | X | X | X | X | |
| $6423_{hex}$ | Analog Input global Interrupt Enable | X | X | X | X | |
| $6424_{hex}$ | Analog Input Interrupt Upper Limit Integer | X | X | X | X | |
| $6425_{hex}$ | Analog Input Interrupt lower Limit Integer | X | X | X | X | |
| $6426_{hex}$ | Analog Input Interrupt Delta Unsigned | X | X | X | X | |
| $6427_{hex}$ | Analog Input Interrupt Negative Delta Unsigned | X | X | X | X | |
| $6428_{hex}$ | Analog Input Interrupt Positive Delta Unsigned | X | X | X | X | |

*Table 15: Overview of all objects*

| Index | Name | BLxx-GWBR-CANOPEN | B67-GW-CANOPEN | BLxx-E-GW-CO | BLCCO-xxx | |
|---|---|---|---|---|---|---|
| $6443_{hex}$ | Analog Output Error Mode | X | X | X | X | see chapter 9 |
| $6444_{hex}$ | Analog Output Error Value Integer | X | X | X | X | |
| $67FF_{hex}$ | Device Type | X | X | X | X | see chapter 4 |
| **I/O module objects (according to CiA DS-406)** | | | | | | |
| $6800_{hex}$ | Operating parameters | X | X | X | X | see chapter 11 |
| $6810_{hex}$ | Preset Value for Multi-Sensor Devices | X | X | X | X | |
| $6820_{hex}$ | Position Value for Multi-Sensor Devices | X | X | X | X | |
| $6B00_{hex}$ | CAM State Register | X | X | X | X | |
| $6B00_{hex}$ | CAM Enable Register | X | X | X | X | |
| $6B00_{hex}$ | CAM Polarity Register | X | X | X | X | |
| $6B10_{hex}$ | CAM1 Low Limit | X | X | X | X | |
| $6B20_{hex}$ | CAM1 High Limit | X | X | X | X | |
| $67FF_{hex}$ | Device TypeDS401 | X | X | X | X | |
| $6FFF_{hex}$ | Device TypeDS406 | X | X | X | X | |

# 4 Device (gateway) objects


**4.1 Objects of the Communication Profile (acc. to CiA DS-301) ..... 2**

4.1.1 Object $1000_{hex}$ – Device Type ..... 5
4.1.2 Object $1001_{hex}$ – Error Register ..... 5
4.1.3 Object $1005_{hex}$ – SYNC COB-ID ..... 6
4.1.4 Object $1008_{hex}$ – Device Name ..... 7
4.1.5 Object $1009_{hex}$ – Manufacturer Hardware Version ..... 8
4.1.6 Object $100A_{hex}$ – Manufacturer Software Version ..... 8
4.1.7 Object $100C_{hex}$ – Guard Time ..... 9
4.1.8 Object $100D_{hex}$ – Lifetime Factor ..... 9
4.1.9 Object $1010_{hex}$ – Store Parameters ..... 10
4.1.10 Object $1011_{hex}$ – Restore Default Parameters ..... 11
4.1.11 Object $1014_{hex}$ – Emcy COB-ID ..... 12
4.1.12 Object $1016_{hex}$ – Consumer Heartbeat Time ..... 13
4.1.13 Object $1017_{hex}$ – Producer Heartbeat Time ..... 15
4.1.14 Object $1018_{hex}$ – Identity Object ..... 16
4.1.15 Object $1020_{hex}$ – Verify Configuration ..... 17
4.1.16 Object $1027_{hex}$ – Module List ..... 19

**4.2 Objects for the Transfer of Service Data ..... 20**

4.2.1 Objects $1200_{hex}$ to $1203_{hex}$ – Server SDO Default Parameters ..... 20

**4.3 Objects for the Transfer of Process Output Data ..... 21**

4.3.1 Object $1400_{hex}$ to $141F_{hex}$ – Receive PDO Comm. Default Parameters ..... 21
4.3.2 Object $1600_{hex}$ to $161F_{hex}$ – Receive PDO-Mapping Parameter ..... 24

**4.4 Objects for the Transfer of Process Input Data ..... 27**

4.4.1 Object $1800_{hex}$ to $181F_{hex}$ – Transmit PDO-Parameters ..... 27
– Inhibit time ..... 30
– Event Timer ..... 30
4.4.2 Objects $1A00_{hex}$ to $1A1F_{hex}$ - Transmit PDO Mapping Parameter ..... 30

**4.5 Objects for network management (not valid for BLCCO) ..... 33**

4.5.1 Object $1F80_{hex}$ – NMT Start-up ..... 33
4.5.2 Object $1F81_{hex}$ – Slave Assignment ..... 34
4.5.3 Object $1F82_{hex}$ – Request NMT ..... 36
4.5.4 Object $1F83_{hex}$ – Request Guarding ..... 38

**4.6 Overview about the objects of the Device Profile (acc. to CiA DS-401 and 406) ..... 39**

4.6.1 Object $67FF_{hex}$ – Device Type ..... 39
4.6.2 Object $6FFF_{hex}$ – Device Type ..... 40

**4.7 Manufacturer specific device objects ..... 40**

4.7.1 Object $2000_{hex}$ – Serial Number ..... 40
4.7.2 Object $2010_{hex}$ – Node ResetModifiers ..... 41
4.7.3 Object $2400_{hex}$ - System Voltages (only BL67) ..... 45
4.7.4 Object $2401_{hex}$ - System Currents (only BL67) ..... 45

## 4.1 Objects of the Communication Profile (acc. to CiA DS-301)

The following table provides an overview of the supported entries in the object directory which are defined through the Communication Profile CiA DS-301.

The column **Object** shows the object type.

The column **Name** shows the predefined symbolic name of the entry.

The column **Typ** shows the entry's data type predefined in CiA DS-301.

The column **Access** shows the access options for the entry. These are:

- rw (read/write)
- ro (read only)
- const (constant) = a read-only constant

The column **M/O/C** indicates if the entry is **M**andatory, **O**ptional or **C**onditional (depending on the Device Type, e. g. modular or compact device).

*Table 16: Object overview for the communication Profile CiA DS-301*

| Index (hex) | Object | Name | Type | Access | M/O/C |
|---|---|---|---|---|---|
| $1000_{hex}$ | VAR | Device Type (page 4-5) | Unsigned32 | const | M |
| $1001_{hex}$ | ARRAY | Error Register (page 4-5) | Unsigned8 | ro | M |
| $1005_{hex}$ | VAR | SYNC COB-ID (page 4-6) | Unsigned32 | rw | O |
| $1008_{hex}$ | VAR | Device Name (page 4-7) | Vis-String | const | O |
| $1009_{hex}$ | VAR | Manufacturer Hardware Version (page 4-8) | Vis-String | const | O |
| $100A_{hex}$ | VAR | Manufacturer Software Version (page 4-8) | Vis-String | const | O |
| $100C_{hex}$ | VAR | Guard Time (page 4-9) | Unsigned32 | rw | O |
| $100D_{hex}$ | VAR | Lifetime Factor (page 4-9) | Unsigned32 | rw | O |
| $1010_{hex}$ | ARRAY | Store Parameters (page 4-10) | Unsigned32 | rw | O |
| $1011_{hex}$ | ARRAY | Restore Default Parameters (page 4-11) | Unsigned32 | rw | O |
| $1014_{hex}$ | VAR | Emcy COB-ID (page 4-12) | Unsigned32 | rw | O |
| $1016_{hex}$ | ARRAY | Consumer Heartbeat Time (page 4-13) | Unsigned32 | rw | O |
| $1017_{hex}$ | VAR | Producer Heartbeat Time (page 4-15) | Unsigned16 | rw | O |
| $1018_{hex}$ | RECORD | Identity Object (page 4-16) | Identity | ro | O |
| $1020_{hex}$ | ARRAY | Verify Configuration (page 4-17) | Unsigned32 | rw | O |
| $1027_{hex}$ | ARRAY | Module List (page 4-19) | Unsigned16 | ro | C |

*Table 16: Object overview for the communication Profile CiA DS-301*

| Index (hex) | Object | Name | Type | Access | M/O/C |
|---|---|---|---|---|---|
| **Server SDO Parameter** | | | | | |
| $1200_{hex}$ | RECORD | 1st Server SDO Parameter (page 4-20) | SDO-Parameters | ro | O |
| $1201_{hex}$ | RECORD | 2nd Server SDO Parameter (page 4-20) | SDO-parameters | rw | O |
| $1202_{hex}$ | RECORD | 3rd Server SDO Parameter (page 4-20) | SDO-parameters | rw | O |
| $1203_{hex}$ | RECORD | 4th Server SDO Parameter (page 4-20) | SDO-parameters | rw | O |
| **Receive PDO Communication Parameters** | | | | | |
| $1400_{hex}$ | RECORD | 1st Receive PDO Parameter (page 4-21) | PDO CommPar | rw | O |
| $1401_{hex}$ | RECORD | 2nd Receive PDO Parameter (page 4-21) | PDO CommPar | rw | O |
| $1402_{hex}$ | RECORD | 3rd Receive PDO Parameter (page 4-21) | PDO CommPar | rw | O |
| ... | ... | ... | ... | ... | ... |
| $141F_{hex}$ | RECORD | 32nd Receive PDO Parameter (page 4-21) | PDO CommPar | rw | O |
| **Receive PDO-Mapping Parameters** | | | | | |
| $1600_{hex}$ | ARRAY | 1st Receive PDO-Mapping (page 4-24) | PDO mapping | rw | O |
| $1601_{hex}$ | ARRAY | 2nd Receive PDO-Mapping (page 4-24) | PDO mapping | rw | O |
| $1602_{hex}$ | ARRAY | 3rd Receive PDO-Mapping (page 4-24) | PDO mapping | rw | O |
| ... | ... | ... | ... | ... | ... |
| $161F_{hex}$ | ARRAY | 32nd Receive PDO-Mapping (page 4-24) | PDO mapping | rw | O |
| **Transmit PDO Communication Parameter** | | | | | |
| $1800_{hex}$ | RECORD | 1st Transmit PDO Parameter (page 4-27) | PDO CommPar | rw | O |
| $1801_{hex}$ | RECORD | 2nd Transmit PDO Parameter (page 4-27) | PDO CommPar | rw | O |

*Table 16: Object overview for the communication Profile CiA DS-301*

| Index (hex) | Object | Name | Type | Access | M/O/C |
|---|---|---|---|---|---|
| $1802_{hex}$ | RECORD | 3rd Transmit PDO Parameter (page 4-27) | PDO CommPar | rw | O |
| ... | ... | ... | ... | ... | ... |
| $181F_{hex}$ | RECORD | 32nd Transmit PDO Parameter (page 4-27) | PDO CommPar | rw | O |
| **Transmit PDO-Mapping Parameter** | | | | | |
| $1A00_{hex}$ | ARRAY | 1st Transmit PDO-Mapping (page 4-30) | PDO mapping | rw | O |
| $1A00_{hex}$ | ARRAY | 2nd Transmit PDO-Mapping ( page 4-30) | PDO mapping | rw | O |
| $1A02_{hex}$ | ARRAY | 3rd Transmit PDO-Mapping (page 4-30) | PDO mapping | rw | O |
| ... | ... | ... | ... | ... | ... |
| $1A1F_{hex}$ | ARRAY | 32nd Transmit PDO Mapping (page 4-30) | PDO mapping | rw | O |
| **NMT Master Objects** | | | | | |
| $1F80_{hex}$ | VAR | NMT Start-up (page 4-33) | Unsigned32 | rw | O |
| $1F81_{hex}$ | ARRAY | Slave Assignment (page 4-34) | Unsigned32 | rw | O |
| $1F82_{hex}$ | ARRAY | Request NMT (page 4-36) | Unsigned8 | rw | O |
| $1F83_{hex}$ | ARRAY | Request Guarding (page 4-38) | Unsigned8 | rw | O |

### 4.1.1 Object $1000_{hex}$ – Device Type

Object $1000_{hex}$ contains the Type and Function for the station.

The value FFFF $0191_{hex}$ indicates that all Device Profiles are supported.

*Table 17: Object $1000_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1000_{hex}$ |
| Name | Device Type |
| Object code | VAR |
| Data type | Unsigned32 |
| **Value range** | |
| Access | ro |
| PDO mapping | No |
| Value range | Unsigned32 |
| Default value, BL×× | FFFF $0191_{hex}$ |
| Default value, BL×× | depends on the respective module |

### 4.1.2 Object $1001_{hex}$ – Error Register

The object $1001_{hex}$ contains the Error register for the CANopen gateway. It thus contains, in one byte, the internal errors that occur.

*Table 18: Object $1001_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1001_{hex}$ |
| Name | Error register |
| Object code | VAR |
| Data type | Unsigned8 |
| **Value range** | |
| Access | ro |
| PDO mapping | Optional |
| Value range | Unsigned8 |
| Default value, BL×× | $00_{hex}$ |

| **Error register** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 0** | **Bit 1** | **Bit 2** | **Bit 3** | **Bit 4** | **Bit 5** | **Bit 6** | **Bit 7** |
| Generic | Current | Voltage | 0 | Comm. | 0 | 0 | Manu. |

**Abbreviations:**

| Abbr. | Meaning | Valid for Modules |
|---|---|---|
| generic | General error | all modules |
| Current | Output short-circuit/ current error | DO, AI,TC |
| Voltage | voltage error | PF, DO, AI, AO |
| Comm. | communication error | all modules |
| Manu. | Manufacturer-specific error | all modules |

### 4.1.3 Object $1005_{hex}$ – SYNC COB-ID

Object $1005_{hex}$ defines the COB-ID and thus the priority (→ „Identifier for the die standard objects“) of the synchronization object (SYNC). The BLxx-CANopen-gateway is not able to generate, but only to receive, SYNC messages.

Structure of the SYNC COB-ID entry (Unsigned32):

| | **MSB** | | | | **LSB** |
|---|---|---|---|---|---|
| **Bits** | **31** | **30** | **29** | **28 to 11** | **10 to 0** |
| 11-bit ID | × | 0 | 0 | 00 0000 0000 0000 0000 | 11-bit Identifier |
| 29-bit ID | × | 0 | 1 | 29-bit Identifier | |

*Table 19: Description of the SYNC COB-ID entry*

| **Bit no.** | **Value** | **Description** |
|---|---|---|
| 31 (MSB) | X | fixed |
| 30 | 0<br>1 | Module does not generate a SYNC message<br>Module generates a SYNC message |
| 29 | 0<br>1 | 11-bit ID (CAN 2.0A) for BLxx<br>29-bit ID (CAN 2.0B) |
| 28 to 11 | 0<br>X | if bit 29 = 0<br>if bit 29 = 1 bits 28 to 11 of the SYNC-COB-ID |
| 10 to 0 (LSB) | X | bits 10 to 0 of the SYNC-COB-ID |

**Note**

Bit 30 is static, i.e. cannot be changed.

*Table 20: Object $1005_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1005_{hex}$ |
| Name | COB-ID |
| Object code | VAR |
| Data type | Unsigned32 |
| **Value range** | |
| Access | rw |
| PDO mapping | No |
| Value range | Unsigned32 |
| Default value, BL×× | 0000 0080h |

### 4.1.4 Object $1008_{hex}$ – Device Name

Object $1008_{hex}$ contains the manufacturer-specific device name

*Table 21: Object $1008_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1008_{hex}$ |
| Name | manufacturer specific device name |
| Object code | VAR |
| Data type | Visible String |
| **Value range** | |
| Access | const. |
| PDO mapping | No |
| Value range, BL×× | – |
| Default value | depends on the gateway/ module |

### 4.1.5 Object $1009_{hex}$ – Manufacturer Hardware Version

Object $1009_{hex}$ contains the designation for the Hardware Version.

*Table 22: Object $1009_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1009_{hex}$ |
| Name | Hardware version |
| Object code | VAR |
| Data type | Visible String |
| **Value range** | |
| Access | const. |
| PDO mapping | No |
| Value range, BL×× | – |
| Default value, BL×× | depends on the gateway/ module |

### 4.1.6 Object $100A_{hex}$ – Manufacturer Software Version

Object $100A_{hex}$ contains the device's software version.

*Table 23: Object $100A_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $100A_{hex}$ |
| Name | Software version |
| Object code | VAR |
| Data type | Visible String |
| **Value range** | |
| Access | const. |
| PDO mapping | No |
| Value range, BL×× | – |

### 4.1.7 Object $100C_{hex}$ – Guard Time

Object $100C_{hex}$ contains the Guard time in milliseconds. The product of "Lifetime factor" (Object $100D_{hex}$) and Guard time is the "Lifetime" for the Node Guarding.

*Table 24: Object $100C_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $100C_{hex}$ |
| Name | Guard Time |
| Object code | VAR |
| Data type | Unsigned16 |
| **Value range** | |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned16 |
| Default value, BL×× | 0 |

### 4.1.8 Object $100D_{hex}$ – Lifetime Factor

If the Lifetime factor is multiplied by the Guard time, the result is the Lifetime for Node Guarding.

Example:

Guard Time .....100 ms

Lifetime Factor:.....3

With a Guard Time of 100 ms, the network nodes expect a guard telegram every 100 ms. The Lifetime factor enables a setting to be made in the BL××-gateway for how often a Guard Frame from the Master can be missed before an error condition is recognized. In this example, the relevant time would be 300 ms. The evaluation would only become active after the message had been missing for 300 ms.

*Table 25: Object $100D_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $100D_{hex}$ |
| Name | Lifetime Factor |
| Object code | VAR |
| Data type | Unsigned8 |
| **Value range** | |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned8 |
| Default value, BL×× | 0 |

### 4.1.9 Object $1010_{hex}$ – Store Parameters

The object $1010_{hex}$ can be used to save parameter changes in non-volatile memory. The command is executed by writing the data contents $6576\ 6173_{hex}$ ("save") to one of the Sub-indices.

*Table 26: Object $1010_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1010_{hex}$ |
| Name | Store Parameters |
| Object code | $3_{hex}$ |
| Data type | Unsigned32 |
| **Value description** | |
| Sub-index | $00_{hex}$ |
| Description | Largest Supported Index |
| Access | ro |
| PDO mapping | No |
| Default value, BL×× | $3_{hex}$ |
| Sub-index | $01_{hex}$ |
| Description | Save all Parameters |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | - |
| Sub-index | $02_{hex}$ |
| Description | Save Communication Parameters |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | - |
| Sub-index | $03_{hex}$ |
| Description | Save Application Parameters |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | - |

### 4.1.10 Object $1011_{hex}$ – Restore Default Parameters

The object $1011_{hex}$ can be used to restore the default parameters. The command is executed by writing the data contents 6461 6F6C$_{hex}$ ("load") to one of the Sub-indices.

*Table 27: Object $1011_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1011_{hex}$ |
| Name | Restore Default Parameters |
| Object code | ARRAY |
| Data type | $3_{hex}$ |
| **Value description** | |
| Sub-index | $00_{hex}$ |
| Description | Largest Supported Index |
| Access | ro |
| PDO mapping | No |
| | |
| Sub-index | $01_{hex}$ |
| Description | Restore All Default Parameters |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | - |

*Table 27: Object $1011_{hex}$*

| **Object description** | |
|---|---|
| **Value description** | |
| Sub-index | $02_{hex}$ |
| Description | Restore Communication Parameters |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | - |
| | |
| Sub-index | $03_{hex}$ |
| Description | Restore Application Parameters |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | - |

### 4.1.11 Object $1014_{hex}$ – Emcy COB-ID

Object $1014_{hex}$ contains the Identifier for the Emergency Messages

| | **MSB** | | | | **LSB** |
|---|---|---|---|---|---|
| **Bits** | **31** | **30** | **29** | **28 to 11** | **10 to 0** |
| 11-bit ID | 0/1 | 0 | 0 | 00 0000 0000 0000 0000 | 11-bit Identifier |
| 29-bit ID | 0/1 | 0 | 1 | 29-bit Identifier | |

*Table 28: Description of the Emcy COB-ID entry*

| **Bit no.** | **Value** | **Description** |
|---|---|---|
| 31 (MSB) | 0<br>1 | EMCY exists/ is valid<br>EMCY does not exist/ is not valid |
| 30 | 0 | reserved (always 0) |
| 29 | 0<br>1 | 11-bit ID (CAN 2.0A) ∅ for BL××<br>29-bit ID (CAN 2.0B) |
| 28 to 11 | 0<br>X | if bit 29 = 0<br>if bit 29 = 1 bits 28 to 11 of 29-bit-COB-ID |

*Table 28: Description of the Emcy COB-ID entry*

| Bit no. | Value | Description |
|---|---|---|
| 10 to 0 (LSB) | X | bits 10 to 0 of COB-ID |

**Note**
Bit 30 is static, i.e. cannot be changed.

*Table 29: Object $1014_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1014_{hex}$ |
| Name | Emcy COB-ID |
| Object code | VAR |
| Data type | Unsigned32 |
| **Value description** | |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | 0000 $0080_{hex}$ + Node-ID |

### 4.1.12 Object $1016_{hex}$ – Consumer Heartbeat Time

The Heartbeat Protocol is used to monitor the operational capability of other CANopen bus nodes. The Heartbeat Protocol must be seen as an alternative to Node-/Life-Guarding, which, unlike Guarding, does not use Remote-Frames.

A device generates the Heartbeat with a specific cycle time (see object $1017_{hex}$ "Producer heartbeat time"). Another device receives the Heartbeat and monitors the cycle time. Another device receives the Heartbeat and monitors the cycle time.

Object 1016 defines the cycle time (interval) at which the Heartbeat is expected. This cycle time should be longer than the corresponding cycle time for the transmitter (see object $1017_{hex}$). The monitoring of the Heartbeat starts when the first Heartbeat Frame is received. If the Consumer heartbeat time = 0, then the corresponding entry will not be used.

The time is set as a multiple of 1 ms.

Structure of the entry for Consumer heartbeat time (Unsigned32):

| | **MSB** | | **LSB** |
|---|---|---|---|
| **Bits** | 31 to 24 | 23 to 16 | 15 to 0 |
| Value | reserved (default: $00_{hex}$ | Node-ID | Heartbeat Time |
| Data type | – | Unsigned8 | Unsigned16 |

*Table 30: Object $1016_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1016_{hex}$ |
| Name | Consumer Heartbeat Time |
| Object code | ARRAY |
| Data type | Unsigned32 |
| **Value description** | |
| Sub-index | $00_{hex}$ |
| Description | Number of Entries |
| Category | Mandatory |
| Access | ro |
| PDO mapping | No |
| Value range, BL×× | 1 |
| Default value, BL×× | 1 |
| | |
| Sub-index | $01_{hex}$ |
| Description | Consumer Heartbeat Time |
| Category | Mandatory |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | 0 |

### 4.1.13 Object $1017_{hex}$ – Producer Heartbeat Time

The object $1017_{hex}$ defines the cycle time for the Heartbeat.

If the cycle time = 0, then Heartbeat will not be used. The content of the object is interpreted as a multiple of 1 ms.

*Table 31: Object $1017_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1017_{hex}$ |
| Name | Producer Heartbeat Time |
| Object code | VAR |
| Data type | Unsigned16 |
| **Value description** | |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned16 |
| Default value, BL×× | 0 |

### 4.1.14 Object $1018_{hex}$ – Identity Object

Object $1018_{hex}$ contains general information about the BL×× gateway.

he Vendor-ID (Sub-index $_{hex}$) is a unique ID which precisely identifies the manufacturer. The manufacturer-specific Product-Code (Sub-index $02_{hex}$) identifies a specific device version. The manufacturer-specific Revision-Number (Sub-index $03_{hex}$) consists of a major revision number and a minor revision number. The major revision number defines a special CANopen functionality. If the CANopen functionality is expanded, then the major revision number must be incremented. The minor revision number identifies various versions that have the same CANopen functionality.

| | **MSB** | **LSB** |
|---|---|---|
| **Bits** | 31 to 16 | 15 to 0 |
| Value | Major Revision Number | Minor Revision Number |

*Table 32: Object $1018_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1018_{hex}$ |
| Name | Device Specification |
| Object code | RECORD |
| Data type | Identity |
| **Value description** | |
| Sub-index | $00_{hex}$ |
| Description | Number of Entries |
| Category | Mandatory |
| Access | ro |
| PDO mapping | No |
| Default value, BL×× | 4 |
| **Value description** | |
| Sub-index | $01_{hex}$ |
| Description | Manufacturer-ID |
| Category | Mandatory |
| Access | ro |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | 0000 009C$_{hex}$ |

*Table 32: Object $1018_{hex}$*

| **Object description** | |
|---|---|
| Sub-index | $02_{hex}$ |
| Description | Product Code |
| Category | Option |
| Access | ro |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| | |
| Sub-index | $03_{hex}$ |
| Description | Revision number |
| Category | Option |
| Access | ro |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | No |
| Sub-index | $04_{hex}$ |
| Description | Serial number |
| Category | Option |
| Access | ro |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |

### 4.1.15 Object $1020_{hex}$ – Verify Configuration

Object $1020_{hex}$ is used to check the station configuration after a device reset.

The BL××-CANopen gateway supports the non-volatile storage of parameters. A network configuration tool or an CANopen Manager can use object $1020_{hex}$ to test the station configuration after a reset, and so check whether a reconfiguration is necessary. The configuration tool saves the time and date simultaneously in object $1020_{hex}$ and the corresponding DCF file. After a reset, the most recent configuration and the signature will be restored, either automatically on request. If the configuration values are altered by some other command, then the object will be set to 0.

The Configuration Manager compares the signature and the configuration with the values from the DCF file. If it discovers any deviations, a reconfiguration will be necessary.

*Table 33: Object $1020_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1020_{hex}$ |
| Name | Verify Configuration |
| Object code | ARRAY |
| Data type | Unsigned16 |
| **Value description** | |
| Sub-index | $00_{hex}$ |
| Description | Number of entries |
| Access | rw |
| PDO mapping | No |
| Default value, BL×× | $02_{hex}$ |
| Sub-index | $01_{hex}$ |
| Description | Configuration date |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | No |
| | |
| Sub-index | $02_{hex}$ |
| Description | Configuration time |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | No |



**Note**

The configuration date contains the number of days since January 01 1984. The configuration time contains the number of milliseconds since midnight.

### 4.1.16 Object $1027_{hex}$ – Module List

Object $1027_{hex}$ describes all the actually installed modules in a station.

It is only valid for modular devices and thus not for BLCCO.

*Table 34: Object $1027_{hex}$*

| **Object description** | | |
|---|---|---|
| INDEX | $1027_{hex}$ | |
| Name | Module List | |
| Object code | ARRAY | |
| Data type | Unsigned16 | |
| Access | ro | |
| **Value description** | | |
| Sub-index | $00_{hex}$ | |
| Description | Number of connected modules | |
| Access | ro | |
| PDO mapping | No | |
| Value range, BL×× | $00_{hex}$ to $4A_{hex}$ | |
| Default value, BL×× | No | |
| | | |
| Sub-index | $01_{hex}$ | |
| Description | Module 1 | |
| ... | ... | |
| | BL20 | BL67 |
| Sub-index | $4A_{hex}$ | $20_{hex}$ |
| Description | Module 74 | Module 32 |

The sequential Sub-indices $01_{hex}$ to $4A_{hex}$ (BL20) or resp. $20_{hex}$ (BL67) describe the corresponding BL×× modules in the sequence in which they are installed in the station. Each entry contains a number that identifies the particular module. Each entry contains a number which identifies the respective module.

**Note**

There is one identifier for each type of BL×× module.

In the EDS-file, the individual extensions to object $1027_{hex}$ (e.g. M1SubExt1027) for all the optional BL×× module types are listed in the section [Supported Modules]. The default values (e.g. default value=$8000_{hex}$) correspond to the identifiers for the particular module types (e.g.[M1ModuleInfo] ProductName=Generic BL××-BR/-PF).

## 4.2 Objects for the Transfer of Service Data

### 4.2.1 Objects $1200_{hex}$ to $1203_{hex}$ – Server SDO Default Parameters

Objects $1200_{hex}$ to $1203_{hex}$ define the priority for the transmission of SDO1 to SDO4.

The priority of the data is defined through the Identifier/ COB-ID

*Table 35: Object $1200_{hex}$ to $1203_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1200_{hex}$ to $1203_{hex}$ |
| Name | Server SDO Parameters |
| Object code | RECORD |
| Number of Elements | $3_{hex}$ |
| Data type | SDO-parameters |
| **Value description** | |
| Sub-index | $00_{hex}$ |
| Description | Number of entries |
| Access | ro |
| PDO mapping | No |
| Default value, BL×× | $02_{hex}$ |
| | |
| Sub-index | $01_{hex}$ |
| Description | COB-ID Client  Server (rx) |
| Access | Index $1200_{hex}$:ro<br>Index $1201_{hex}$ to $1203_{hex}$:rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | Index $1200_{hex}$:<br>0000 $0600_{hex}$ + Node-ID<br>Index $1201_{hex}$ to $1203_{hex}$:rwNo |
| Sub-index | $02_{hex}$ |
| Description | COB-ID Server  Client (rx) |
| Access | Index $1200_{hex}$:ro<br>Index $1201_{hex}$ to $1203_{hex}$:rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |

| *Table 35: Object $1200_{hex}$ to $1203_{hex}$* | **Object description** | |
|---|---|---|
| | Default value, BL×× | Index $1200_{hex}$:<br>$0000\ 0580_{hex}$ + Node-ID<br>Index $1201_{hex}$ to $1203_{hex}$:rwNo |

## 4.3 Objects for the Transfer of Process Output Data

The objects $1400_{hex}$ to $141F_{hex}$ define, together with objects $1600_{hex}$ to $161F_{hex}$, which output data have to be transferred via RPDO. In addition to that, the priority and the transmission type for the RPDO-transfer are defined.

Object $1400_{hex}$ defines the priority and the transmission type for the RPDO**1**. Object $1600_{hex}$ defines the object-index, the sub-index and the data length for the data which have to be transferred via RPDO**1**.

Objects $1401_{hex}$ and $1601_{hex}$ thus define the RPDO**2**, etc.

The priority of the data is defined through the Identifier/ COB-ID

The values are already entered by default for objects $1400_{hex}$ to $1403_{hex}$ and $1600_{hex}$ to $1603_{hex}$.

A station with up to 64 digital outputs and 12 analog outputs therefore transfers the process output data automatically via RPDOs.

### 4.3.1 Object $1400_{hex}$ to $141F_{hex}$ – Receive PDO Comm. Default Parameters

Objects $1400_{hex}$ to $141F_{hex}$ define the priority for the transmission of RPDO11 to RPDO32.

The priority is defined via the identifier/COB-ID (see „Identifier for the Standard Objects") in sub-index $01_{hex}$.

With the highest bit of sub-index $01_{hex}$, the further content can be defined as valid/invalid. The respective highest $_{hex}$adecimal number is then > 8.

The transmission type is defined with the sub-index $02_{hex}$.

Which data content is to be transferred with the RPDO1 to RPDO32 is defined with the objects $1600_{hex}$ to $161F_{hex}$.

| *Table 36: Object $1400_{hex}$ to $141F_{hex}$* | **Object description** | **BL××** | **BLC** |
|---|---|---|---|
| | INDEX | $1400_{hex}$ to $141F_{hex}$ | $1400_{hex}$ to $1403_{hex}$ |
| | Name | Receive PDO Parameters | |
| | Object code | RECORD | |
| | Data type | PDO CommPar | |
| | **Value description** | | |
| | Sub-index | $00_{hex}$ | |
| | Description | Maximum Number of Entries | |
| | Access | ro | |

*Table 36: Object $1400_{hex}$ to $141F_{hex}$*

| Object description | BLxx | BLC |
|---|---|---|
| PDO mapping | No | |
| Value range, BLxx | 2 | |
| Default value, BLxx | $02_{hex}$ | |
| | | |
| Sub-index | $01_{hex}$ | |
| Description | COB-ID for the PDOs (see Table 37:) | |
| Access | rw | |
| PDO mapping | No | |
| Value range, BLxx | Unsigned32 | |
| Default value, BLxx | Index $1400_{hex}$: 0000 $0200_{hex}$ + Node-ID<br>Index $1401_{hex}$: 0000 $0300_{hex}$ + Node-ID<br>Index $1402_{hex}$: 0000 $0400_{hex}$ + Node-ID<br>Index $1403_{hex}$: 0000 $0500_{hex}$ + Node-ID | |
| | Index $1404_{hex}$ to $141F_{hex}$: reserved | - |
| | | |
| Sub-index | $02_{hex}$ | |
| Description | Transmission type (see Table 38:) | |
| Access | rw | |
| PDO mapping | No | |
| Value range, BLxx | Unsigned8 | |
| Default value, BLxx | $FF_{hex}$ | |

Structure of the COB-ID entry

| | MSB | | | | LSB |
|---|---|---|---|---|---|
| **Bits** | **31** | **30** | **29** | **28 to 11** | **10 to 0** |
| 11-bit ID | 0/1 | 0/1 | 0 | 00 0000 0000 0000 0000 | 11-bit Identifier |
| 29-bit ID | 0/1 | 0/1 | 1 | 29-bit Identifier | |

*Table 37: Description of the COB-ID-entry*

| Bit no. | Value | Meaning |
|---|---|---|
| 31 (msb) | 0 | PDO exists / is valid |
| | 1 | PDO does not exist / is invalid |
| 30 | 0 | RTR is possible in this PDO |
| | 1 | RTR is not possible in this PDO |
| 29 | 0 | 11-Bit-ID (CAN 2.0A) (standard application) |
| | 1 | 29-bit ID (CAN 2.0B) |
| 28 to 11 | 0 | if bit 29=0 (Standard application) |
| | X | if bit 29 = 1 bits 28 to 11 of COB-ID |
| 10 to 0 (lsb) | X | bits 10 to 0 of COB-ID |

The Transmission type (Sub-index $02_{hex}$) can have the following values:

*Table 38: Description of the Transmission Type*

| Transmission Type | PDO transmission | | | | |
|---|---|---|---|---|---|
| | cyclic | acyclic | synchr. | asynchr. | only with RTR |
| 0 | | × | × | | |
| 1 | × | | × | | |
| 2 to 254 | | | reserved | | |
| 255 | | | | × | |

### 4.3.2 Object $1600_{hex}$ to $161F_{hex}$ – Receive PDO-Mapping Parameter

Objects $1600_{hex}$ to $161F_{hex}$ (BLCCO: $1600_{hex}$ to $1603_{hex}$) define, which data have to be transferred with RPDO1 to RPDO32 (BLCCO: RPDO1 to RPDO 4).

The data content (here: process output data) is represented by product specific mappable objects.

→ Mappable Objects (page 2-22)

For example, the process output data for the digital channels are entered in objects $6200_{hex}$, $6220_{hex}$ etc.

The description of these objects can be found in the following chapter.

Sub-indices $01_{hex}$ ff. of the objects $1600_{hex}$ to $161F_{hex}$ (BLCCO: $1600_{hex}$ to $1603_{hex}$) contain the object number, the sub-index and the length of the data that have to be transferred via the respective RPDO.

An RPDO can transfer a maximum number of 8 bytes (64 bit).

The number of sub-indices depends on the data length. For larger projects, it has to be calculated and entered by user.

At a data length of 8 bit, 8 sub-indices are needed to represent a total number of 64 bit. At a data length of 1 bit, 64 sub-indices are needed to represent a total number of 64 bit.

Objects $1600_{hex}$ to $1603_{hex}$ (RPDO1 to RPDO4) references by default the values for the first 64 digital output channels and for the first 12 analog output channels, provided that the values are represented by the objects $6200_{hex}$ (digital values) and $6411_{hex}$ (analog values).

*Table 39: Object $1600_{hex}$ to $161F_{hex}$*

| **Object description** | **BLxx** | **BLCCO** |
|---|---|---|
| INDEX | $1600_{hex}$ to $161F_{hex}$ | $1600_{hex}$ to $1603_{hex}$ |
| Name | Receive PDO-Mapping Parameters | |
| Object code | RECORD | |
| Data type | PDO mapping | |
| **Value description** | | |
| Sub-index | $00_{hex}$ | |
| Description | NumberOfMappedApplication Objects | Receive PDO-Mapping Parameters |
| Access | rw | rw |
| PDO mapping | No | No |
| Value range, BLxx | 0 to 64 | - |
| Default value, BLxx | see Table 16: | - |
| | | |
| Sub-index | $01_{hex}$ | |
| Description | 1st Mapping Object | |
| Access | rw | |
| PDO mapping | No | |

*Table 39: Object $1600_{hex}$ to $161F_{hex}$*

| **Object description** | **BL××** | **BLCCO** |
|---|---|---|
| Value range, BL×× | | Unsigned32 |
| Default value, BL×× | | see Table 16: |
| ... | | ... |
| Sub-index | $40_{hex}$ | $03_{hex}$ |
| Description | 64th Mapping Object | 4th Mapping Object |
| Access | rw | rw |
| PDO mapping | No | No |
| Value range, BL×× | Unsigned32 | Unsigned32 |
| Default value, BL×× | No | No |

Default values for objects $1600_{hex}$ to $1603_{hex}$:

**Note**

The number of mapping objects, which are automatically generated by the gateway during start-up, depends on the actual physical structure of the BL××-station.

*Table 40: Default values for objects $1600_{hex}$ to $1603_{hex}$*

| **Object** | **Product** | **Sub-index** | **Default value** | **Description** | **valid for** |
|---|---|---|---|---|---|
| $1600_{hex}$ | BL×× | $01_{hex}$ | $6200\ 0108_{hex}$ | 1st Mapping Object (digital output) | RPDO1 |
| | | ... | ... | ... | |
| | | $08_{hex}$ | $6200\ 0808_{hex}$ | 8th Mapping Object (digital output) | |
| | BLCCO | $01_{hex}$ | $6200\ 0108_{hex}$ | 1st Mapping Object (digital output) | RPDO1 |
| $1601_{hex}$ | BL×× | $01_{hex}$ | $6411\ 0110_{hex}$ | 1st Mapping Object (analog output) | RPDO2 |
| | | ... | ... | ... | |
| | | $04_{hex}$ | $6411\ 0410_{hex}$ | 4th Mapping Object (analog output) | |
| | BLCCO | $01_{hex}$ | $6411\ 0110_{hex}$ | 1st Mapping Object (analog output) | RPDO2 |

*Table 40: Default values for objects $1600_{hex}$ to $1603_{hex}$*

| Object | Product | Sub-index | Default value | Description | valid for |
|---|---|---|---|---|---|
| $1602_{hex}$ | BL×× | $01_{hex}$ | 6411 $0510_{hex}$ | 1st Mapping Object (analog output) | RPDO3 |
| | | ... | ... | ... | |
| | | $04_{hex}$ | 6411 $0810_{hex}$ | 4th Mapping Object (analog output) | |
| | BLCCO | | | | |
| $1603_{hex}$ | BL×× | $01_{hex}$ | 6411 $0910_{hex}$ | 1st Mapping Object (analog output) | RPDO4 |
| | | ... | ... | ... | |
| | | $04_{hex}$ | 6411 $0C10_{hex}$ | 4th Mapping Object (analog output) | |
| | BLCCO | $01_{hex}$ | 6411 $0510_{hex}$ | 1st Mapping Object (analog output) | RPDO4 |

The following structure applies to the parameters for sub-indices:

Structure of the PDO Mapping entries:

| MSB | | LSB |
|---|---|---|
| Index (16 bit) | Sub-Index (8 bit) | Object length (8 bit) |

**Note**

To change the number of mapping entries, follow the instructions in section .
Procedure for Altering PDO-Mappings (page 2-23).

## 4.4 Objects for the Transfer of Process Input Data

Objects $1800_{hex}$ to $181F_{hex}$ (BLCCO: $1800_{hex}$ to $1803_{hex}$) define, together with objects $1A00_{hex}$ to $1A1F_{hex}$ (BLCCO: $1A00_{hex}$ to $1A03_{hex}$), which input data are transferred. Additionally they define the priority and the mode for the data transfer via TPDO.

Object $1800_{hex}$ defines the priority, the minimum inhibit time, the event timer and the transmission type for TPDO1. Object $1A00_{hex}$ defines the object-index, the sub-index and the data length for the data which have to be transferred via TPDO1.

Objects $1801_{hex}$ and $1A01_{hex}$ thus define the TPDO2, etc.

The priority of the data is defined through the Identifier/ COB-ID

The values are already entered by default for objects $1800_{hex}$ to $1803_{hex}$ and $1A00_{hex}$ to $1A03_{hex}$.

A station with up to 64 digital outputs and 12 analog outputs therefore transfers the process input data automatically via TPDOs.

### 4.4.1 Object $1800_{hex}$ to $181F_{hex}$ – Transmit PDO-Parameters

Objects $1800_{hex}$ to $181F_{hex}$ (BLCCO: $1800_{hex}$ to $1803_{hex}$) define the priority, the transmission type, the inhibit time and the event timer for TPDO1 to TPDO32.

The priority is defined via the identifier/COB-ID (see „Identifier for the Standard Objects") in sub-index $01_{hex}$.

With the highest bit of sub-index $01_{hex}$, the further content can be defined as valid/invalid. The respective highest $_{hex}$adecimal number is then > 8.

The transmission type is defined with the sub-index $02_{hex}$.

The inhibit time is defined in sub-index $03_{hex}$.

The maximum time between 2 transmissions, the event timer, is defined in sub-index $05_{hex}$.

Which data content is to be transferred with the RPDO1 to RPDO32 is defined with the objects $1A00_{hex}$ to $1A1F_{hex}$. (BLCCO: $1A00_{hex}$ to $1A03_{hex}$).

*Table 41: Object $1800_{hex}$ to $181F_{hex}$*

| Object description | BLxx | BLCCO |
|---|---|---|
| INDEX | $1800_{hex}$ to $181F_{hex}$ | $1800_{hex}$ to $1803_{hex}$ |
| Name | Transmit PDO-parameters | |
| Object code | RECORD | |

*Table 41: Object $1800_{hex}$ to $181F_{hex}$*

| **Object description** | **BL××** | **BLCCO** |
|---|---|---|
| Data type | | PDO CommPar |
| **Value description** | | |
| Sub-index | | $00_{hex}$ |
| Description | | Highest Sub-Index Used |
| Access | | ro |
| PDO mapping | | No |
| Value range | | 5 |
| Default value | | No |
| | | |
| Sub-index | | $01_{hex}$ |
| Description | | COB-ID of the PDO |
| Access | | rw |
| PDO mapping | | No |
| Value range, BL×× | | Unsigned32 |
| Default value, BL×× | | Index $1800_{hex}$: 0000 $0180_{hex}$ + Node-ID<br>Index $1801_{hex}$: 0000 $0280_{hex}$ + Node-ID<br>Index $1802_{hex}$: 0000 $0380_{hex}$ + Node-ID<br>Index $1803_{hex}$: 0000 $0480_{hex}$ + Node-ID |
| | Index $1804_{hex}$ to $181F_{hex}$: | - |
| **Value description** | | |
| Sub-index | | $02_{hex}$ |
| Description | | Transmission Type |
| Access | | rw |
| PDO mapping | | No |
| Value range, BL×× | | Unsigned8 |
| Default value, BL×× | | FFh |
| | | |
| Sub-index | | $03_{hex}$ |
| Description | | Inhibit Time |
| Access | | rw |
| PDO mapping | | No |
| Value range, BL×× | | Unsigned16 |

*Table 41: Object $1800_{hex}$ to $181F_{hex}$*

| **Object description** | **BL××** | **BLCCO** |
|---|---|---|
| Default value, BL×× | | 0 |
| | | |
| Sub-index | | $04_{hex}$ |
| Description | | reserved |
| | | |
| Sub-index | | $05_{hex}$ |
| Description | | Event Timer |
| Access | | rw |
| PDO mapping | | No |
| Value range, BL×× | | Unsigned16 (0 is not used) |
| Default value, BL×× | | 0 |

The COB-ID (sub-Index $01_{hex}$) shows the following structure:

| | **MSB** | **LSB** | | | |
|---|---|---|---|---|---|
| **Bits** | **31** | **30** | **29** | **28 to 11** | **10 to 0** |
| 11-bit ID | 0/1 | 0/1 | 0 | 00 0000 0000 0000 0000 | 11-bit Identifier |
| 29-bit ID | 0/1 | 0/1 | 1 | 29-bit Identifier | |

*Table 42: Description of the COB-ID entry (sub-index $01_{hex}$)*

| **Bit no.** | **Value** | **Description** |
|---|---|---|
| 31 (MSB) | 0 | PDO exists / is valid |
| | 1 | PDO does not exist / is invalid |
| 30 | 0 | RTR is possible in this PDO |
| | 1 | RTR is not possible in this PDO |
| 29 | 0 | 11-Bit-ID (CAN 2.0A) (standard application) |
| | 1 | 29-bit ID (CAN 2.0B) |
| 28 to 11 | 0 | if bit 29=0 (Standard application) |
| | X | if bit 29 = 1 bits 28 to 11 of COB-ID |
| 10 to 0 (LSB) | X | bits 10 to 0 of COB-ID |

The Transmission type (Sub-index $02_{hex}$) can have the following values:

*Table 43: Transmission Type*

| **Transmission Type** | **PDO transmission** | | | | |
|---|---|---|---|---|---|
| | cyclic | acyclic | synchr. | asynchr. | only with RTR |
| 0 | | × | × | | |
| 1 | × | | × | | |
| 2 to 252 | | | reserved | | |
| 253 | | | | × | × |
| 254 | | | reserved | | |
| 255 | | | | × | |

**Inhibit time**

In order to avoid a permanent occupation of the field bus by high-prior messages, the Inhibit Time is defined.

With sub-Index $03_{hex}$, the Inhibit time is defined as a multiple of 100 µs. However, since the time resolution of the system clock in the BL××-CANopen gateway is 1 ms, Inhibit time values below 10 x 100 µs are pointless.

**Event Timer**

Sub-index $03_{hex}$ defines the maximum interval after which a TPDO will be transmitted, even though no event has occurred. The expiry of the interval set for the Event timer is detected as an event.

If any other event occurs, the Event timer is reset and restarted.

The value of the object is interpreted as a multiple of 1 ms.

### 4.4.2 Objects $1A00_{hex}$ to $1A1F_{hex}$ - Transmit PDO Mapping Parameter

Die Objects $1A00_{hex}$ to $1A1F_{hex}$ (BLCCO: $1A00_{hex}$ to $1A03_{hex}$) define, which data have to be transferred with TPDO1 to TPDO32.

The data content (here: process input data) is represented by product specific mappable objects.

see Mappable Objects (page 2-22)

For example, the process input data for the digital channels are entered in objects $6000_{hex}$, $6020_{hex}$ etc.

The description of these objects can be found in the following chapter.

Sub-indices $01_{hex}$ ff. of the objects $1A00_{hex}$ to $1A1F_{hex}$ (BLCCO: $1A00_{hex}$ to $1A03_{hex}$) contain the object number, the sub-index and the length of the data that have to be transferred via the respective TPDO.

A TPDO can transfer a maximum number of 8 bytes (64 bit).

The number of sub-indices depends on the data length. For larger projects, it has to be calculated and entered by user.

At a data length of 8 bit, 8 sub-indices are needed to represent a total number of 64 bit. At a data length of 1 bit, 64 sub-indices are needed to represent a total number of 64 bit.

Objects $1600_{hex}$ to $1603_{hex}$ (RPDO1 to RPDO4) references by default the values for the first 64 digital output channels and for the first 12 analog output channels, provided that the values are represented by the objects $6000_{hex}$ (digital values) and $6401_{hex}$ (analog values).

*Table 44: Object $1A00_{hex}$ to $1A1F_{hex}$*

| **Object description** | **BL××** | **BLCCO** |
|---|---|---|
| INDEX | $1A00_{hex}$ to 1A1Fh | $1A00_{hex}$ to 1A03h |
| Name | Transmit PDO-Mapping Parameters | |
| Object code | RECORD | |
| Data type | PDO mapping | |
| **Value description** | | |
| Sub-index | $00_{hex}$ | |
| Description | Number of Mapped Application Objects in the PDO | Transmit PDO-Mapping Parameters |
| Access | rw | |
| PDO mapping | No | |
| Value range, BL×× | 0 deactivated<br>1 to 64: activated | |
| Default value, BL×× | see Table 38: | |
| | | |
| Sub-index | $01_{hex}$ | |
| Description | 1st Mapping Object | |
| Access | rw | |
| PDO mapping | No | |
| Value range, BL×× | Unsigned32 | |
| Default value, BL×× | see page 4-32 | |
| ... | | |
| Sub-index | $40_{hex}$ | |
| Description | 64th Mapping Object | |
| Access | rw | |
| PDO mapping | No | |
| Value range, BL×× | Unsigned32 | |
| Default value, BL×× | No | |

Default values for objects $1A00_{hex}$ to $1A03_{hex}$:

**Note**

The number of mapping objects, which are automatically generated by the gateway during start-up, depends on the actual physical structure of the BL××-station.

*Table 45: Default values for objects $1A00_{hex}$ to $1A03_{hex}$*

| Object | Product | Sub-Index | Default value | Description | valid for |
|---|---|---|---|---|---|
| $1A00_{hex}$ | BL×× | $01_{hex}$ | $6000\ 0108_{hex}$ | 1st Mapping Object (digital input) | TPDO1 |
| | | ... | ... | ... | |
| | | $08_{hex}$ | $6000\ 0808_{hex}$ | 8th Mapping Object (digital input) | |
| | BLCCO | $01_{hex}$ | $6000\ 0108_{hex}$ | 1st Mapping Object (digital input) | TPDO1 |
| $1A00_{hex}$ | BL×× | $01_{hex}$ | $6401\ 0110_{hex}$ | 1st Mapping Object (analog input) | TPDO2 |
| | | ... | ... | ... | |
| | | $04_{hex}$ | $6401\ 0410_{hex}$ | 4th Mapping Object (analog input) | |
| | BLCCO | $01_{hex}$ | $6401\ 0110_{hex}$ | 1st Mapping Object (analog input) | |
| $1A02_{hex}$ | BL×× | $01_{hex}$ | $6401\ 0510_{hex}$ | 1st Mapping Object (analog input) | TPDO3 |
| | | ... | ... | ... | |
| | | $04_{hex}$ | $6401\ 0810_{hex}$ | 4th Mapping Object (analog input) | |
| | BLCCO | $01_{hex}$ | $6401\ 0510_{hex}$ | 1st Mapping Object (analog input) | TPDO3 |
| $1A03_{hex}$ | BL×× | $01_{hex}$ | $6401\ 0910_{hex}$ | 1st Mapping Object (analog input) | TPDO4 |
| | | ... | ... | ... | |
| | | $04_{hex}$ | $6401\ 0C10_{hex}$ | 4th Mapping Object (analog input) | |
| | BLCCO | $01_{hex}$ | $6401\ 0910_{hex}$ | 1st Mapping Object (analog input) | TPDO4 |

The following structure applies to the parameters for sub-indices $01_{hex}$ ff.:

| MSB | | LSB |
|---|---|---|
| Index (16 bit) | Sub-Index (8 bit) | Object length (8 bit) |

**Note**

To change the number of mapping entries, follow the instructions in section Procedure for Altering PDO-Mappings (page 2-23).

## 4.5 Objects for network management (not valid for BLCCO)

Objects $1F80_{hex}$ to $1F83_{hex}$ are only relevant if the BL××-station is to work as a Network Management Master. The activation is done via bit 0 from object $1F80_{hex}$.

### 4.5.1 Object $1F80_{hex}$ – NMT Start-up

Object $1F80_{hex}$ describes the start-up behavior of BL×× in NMT (Network-Management).

*Table 46: Object $1F80_{hex}$ to $1A1F_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1F80_{hex}$ |
| Name | NMT Start-up |
| Object code | VAR |
| Data type | Unsigned32 |
| Access | rw |

*Table 47: Structure of the NMT-Start-up*

| Bit | Value | Meaning |
|---|---|---|
| 0 | 0 | BL×× station is not the NMT-Master All further bits will be ignored. The objects in the network list will be ignored. |
| | 1 | BL×× is the NMT-Master |
| 1 | 0 | Only the explicitly selected slaves will be started. |
| | 1 | After boot-up, the service "NMT Start Remote Node All Nodes" will be performed. |
| 2 | 0 | BL×× moves automatically to the "Operational" state. |
| | 1 | BL×× does not automatically move to the "Operational" state. The status change is defined through the respective application. |
| 3 | 0 | Starting of the slave is permitted. |
| | 1 | Starting of the slave is not permitted. |

*Table 47: Structure of the NMT-Start-up*

| Bit | Value | Meaning |
|---|---|---|
| 4 | 0 | An error event in an obligatory slave deals with the slave individually. |
| | 1 | An error event in an obligatory slave triggers an NMT Reset All Nodes (see object $1F81_{hex}$, bit 3). |
| 5 to 31 | | Reserved; set to 0 |

### 4.5.2 Object $1F81_{hex}$ – Slave Assignment

Object $1F82_{hex}$ describes, as per CiA DSP-302, all the slaves that can be connected to Network Management (NMT). It contains information on error control and on actions which will be enganged through error events. All other parameters for a slave are only valid if this slave is described in object $1F81_{hex}$ .

**Note**

Object $1F81_{hex}$ is only valid if BL×× is defined as NMT Master (see „Object $1F80_{hex}$ “, Bit 0).

*Table 48: Object $1F81_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1F81_{hex}$ |
| Name | Slave Assignment |
| Object code | ARRAY |
| Data type | Unsigned32 |
| Access | rw |
| **Value description** | |
| Sub-index | $00_{hex}$ |
| Description | Maximum Number of Entries |
| Access | rw |
| PDO mapping | No |
| Value range | 1 to 127 |
| Default value | 127 |
| Sub-index | $01_{hex}$ |
| Description | Slave with Node-ID 1 |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| ... | |

*Table 48: Object 1F81$_{hex}$*

| **Object description** | |
|---|---|
| Sub-index | 3Fh |
| Description | Slave with Node-ID 63 |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | No |

**Note**

Each sub-index corresponds to the slave with the respective Node-ID. The sub-index with the Node-ID of the NMT-Master will be ignored.

*Table 49: Structure of the object 1F81$_{hex}$ Slave Assignment*

| **Byte** | **Bit** | **Value** | **Meaning** |
|---|---|---|---|
| 0 | 0 | 0 | The node with this ID is not a slave. |
| | | 1 | The node with this ID is not a slave. After configuration, the node will be put into the "Operational" state. |
| | 1 | 0 | An error event or other event detection by a slave during boot-up leads to information from the application. |
| | | 1 | An error event or other event detection by a slave during boot-up leads to information from the application and to automatic start of Error Control Services. |
| | 2 | 0 | An error event or other event detection by a slave during boot-up does not lead to information from the application or automatic start of Error Control Services. |
| | | 1 | An error event or other event detection by a slave during boot-up leads to the start of "Start Boot Slaves". |
| | 3 | 0 | Optional slave: the network can also be started if this node is not connected. |
| | | 1 | Mandatory slave: the network will not be started if this node is not connected during the slave boot-up. |
| | 4 | 0 | The slave can be reset by the "NMT Reset Communication" command, depending on its state. |
| | | 1 | The NMT-Master does not have to send an "NMT Reset Communication" command for this slave, if the slave is in the "Operational" state. |

*Table 49: Structure of the object 1F81$_{hex}$ Slave Assignment*

| Byte | Bit | Value | Meaning |
|---|---|---|---|
| 0 | 5 | 0 | Verification of the application software version is not required for this node. |
| | | 1 | Verification of the application software version is required for this node. |
| | 6 | 0 | Automatic update of the application software (download) is not permitted. |
| | | 1 | Automatic update of the application software (download) is permitted. |
| | 7 | | Reserved; set to 0 |
| 1 | | | 8 bit value for the Retry Factor |
| 2 to 3 | | 0 | 16 bit value for the Guard time |

### 4.5.3 Object 1F82$_{hex}$ – Request NMT

Object 1F82$_{hex}$ describes, as per CiA DSP-302, all the slaves that can present queries to the Network Management (NMT).

*Table 50: Object 1F82$_{hex}$*

| **Object description** | |
|---|---|
| INDEX | 1F82$_{hex}$ |
| Name | Query NMT |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | ro/rw |
| **Value description** | |
| Sub-index | 00$_{hex}$ |
| Description | Supported Number of Slaves |
| Access | ro |
| PDO mapping | No |
| Value | 128 |
| | |
| Sub-index | 01$_{hex}$ |
| Description | Request NMT-Service for slave with Node-ID 1 |
| Access | rw |

*Table 50: Object 1F82$_{hex}$*

| **Object description** | |
|---|---|
| PDO mapping | No |
| Value range, BL×× | Unsigned8 |
| Default value, BL×× | No |
| | |
| Sub-index | 3Fh |
| Description | Request NMT-Service for slave with Node-ID 63 |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned8 |
| Default value, BL×× | No |
| | |
| Sub-index | $80_{hex}$ |
| Description | Request NMT-Service for all slaves |
| Access | wo |
| PDO mapping | No |
| Value range, BL×× | Unsigned8 |
| Default value, BL×× | No |

For a write access to this object, the value corresponds to the state for the node to which the query is directed. For read access, the object contains the present state of the node.

*Table 51: Value range*

| **Status** | **Value for Write access** | **Value for Read access** |
|---|---|---|
| Stopped | 4 | 4 |
| Operational | 5 | 5 |
| Reset Node | 6 | – |
| Reset Communication | 7 | – |
| Pre-Operational | 127 | 127 |
| unknown | – | 0 |
| Node missing | – | 1 |

### 4.5.4 Object $1F83_{hex}$ – Request Guarding

Object $1F83_{hex}$ describes, as per CiA DSP-302, all the slaves that can be monitored through the Network Management (NMT).

*Table 52: Object $1F83_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $1F83_{hex}$ |
| Name | Request Guarding |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | ro/rw |
| **Value description** | |
| Sub-index | $00_{hex}$ |
| Description | Supported Number of Slaves |
| Access | ro |
| PDO mapping | No |
| Value | 128 |
| | |
| Sub-index | $01_{hex}$ |
| Description | Request NMT-Service for slave with Node-ID 1 |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned8 |
| Value | 0 = Slave being monitored at present<br>1 = Slave not being monitored at present |
| ... | |
| Sub-index | 7Fh |
| Description | Request Guarding for Slave with Node-ID 63 |
| Access | rw |
| PDO mapping | No |
| Value range | Unsigned8 |
| Value | 0 = Slave being monitored at present<br>1 = Slave not being monitored at present |

*Table 52: Object $1F83_{hex}$*

| **Object description** | |
|---|---|
| Sub-index | $80_{hex}$ |
| Description | Request Start/Stop Guarding for all slaves |
| Access | wo |
| PDO mapping | No |
| Value range, BL×× | Unsigned8 |
| Default value, BL×× | No |

## 4.6 Overview about the objects of the Device Profile (acc. to CiA DS-401 and 406)

### 4.6.1 Object $67FF_{hex}$ – Device Type

The object $67FF_{hex}$ specifies the type of the first device profile supported.

The object contains the value $000x0191_{hex}$.

Das Low-Word ($0191_{hex}$) specifies the device profile (acc. to CiA DS-401: I/O modules).

the high word ($000x_{hex}$) the I/O-types, see CiA DS-401.

*Table 53: Object $67FF_{hex}$*

| **Feature** | **Description/ Value** |
|---|---|
| Name | Device Type |
| Object code | VAR |
| Data type | Unsigned32 |
| Access | ro |
| Default value | No |
| PDO mapping | No |

### 4.6.2 Object $6FFF_{hex}$ – Device Type

Object $6FFF_{hex}$ (corresponds to object $67FF_{hex}$ in accordance with CiA DS-406) specifies the type of the second device profile supported.

The object contains the value $000A0191_{hex}$.

The low word ($0196_{hex}$ = 406dez) specifies the device profile.

The high word ($000A_{hex}$) describes the encoder type in accordance with CiA DS-406 (10dec = Multi Sensor Encoder Interface).

*Table 54: Object $6FFF_{hex}$*

| Feature | Description |
|---|---|
| Name | Device Type |
| Object code | VAR |
| PDO Mapping | No |
| Data type | Unsigned32 |
| Access | ro |

## 4.7 Manufacturer specific device objects

### 4.7.1 Object $2000_{hex}$ – Serial Number

Object $2000_{hex}$ corresponds to sub-index $04_{hex}$ of object $1018_{hex}$ and contains the ident-number of the used BLxx-gateway.

**Note**

We recommend using object $1018_{hex}$, sub-index $04_{hex}$ for the serial number.

### 4.7.2 Object $2010_{hex}$ – Node ResetModifiers

Object $2010_{hex}$ is used for a temporary (volatile) modification to the module behavior.

*Table 55: Object $2010_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $2010_{hex}$ |
| Name | Node ResetModifiers |
| Object code | ARRAY |
| Data type | Unsigned32 |
| **Value description** | |
| Sub-index | $00_{hex}$ |
| Description | Number of Entries |
| Access | ro |
| PDO mapping | No |
| Value range, BL×× | Unsigned8 |
| Default value, BL×× | No |
| | |
| Sub-index | $01_{hex}$ |
| Description | Hard Reset Node Identifier |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | No |
| | |
| Sub-index | $02_{hex}$ |
| Description | Save Reference Reset Modifier |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | No |

*Table 55: Object 2010$_{hex}$*

| **Object description** | |
|---|---|
| Sub-index | $03_{hex}$ |
| Description | Save Current Reset Modifier |
| Access | rw |
| PDO mapping | No |
| Value range, BL×× | Unsigned32 |
| Default value, BL×× | No |

The Hard-Reset-Modifier determines whether, in the event of a Reset Node command, a normal fast reset should be performed, or a hard processor reset, which can take several seconds to be carried out.

For writing, the value that is transferred in Unsigned 32 format will be interpreted as a string:

*Table 56: Hard Reset (processor reset) selection*

| **MSB** | | | **LSB** |
|---|---|---|---|
| t | s | r | h |
| $74_{hex}$ | $73_{hex}$ | $72_{hex}$ | $68_{hex}$ |

*Table 57: Normal Reset selection*

| **MSB** | | | **LSB** |
|---|---|---|---|
| t | s | r | s |
| $74_{hex}$ | $73_{hex}$ | $72_{hex}$ | $73_{hex}$ |

After the next Reset-Node or the next "Reset Communication" command, the operating mode will in all cases be reset to "normal Reset Node".

**$2010_{hex}$ at BL××**

The Index "Save reference module list" determines that, in the event of a Reset Node command, the BL×× reference module list (Objects $3080_{hex}$ and $3081_{hex}$) will be saved in non-volatile memory, and then followed by a hard processor reset. This hardware reset is necessary, because changes to the BL××0 reference module list cannot be dynamically accepted in the CANopen I/O mirror. If the module list is altered, all the CANopen parameters will be reset to the default values.

For writing, the value that is transferred in Unsigned 32 format will be interpreted as a string:

*Table 58: Selection: "Save and Hardware-Reset"*

| **MSB** | | | **LSB** |
|---|---|---|---|
| v | a | s | r |
| $76_{hex}$ | $61_{hex}$ | $73_{hex}$ | $72_{hex}$ |

*Table 59: Normal Reset selection*

| **MSB** | | | **LSB** |
|---|---|---|---|
| t | s | r | s |
| $74_{hex}$ | $73_{hex}$ | $72_{hex}$ | $73_{hex}$ |

After the next Reset-Node or the next "Reset Communication" command, the operating mode will in all cases be reset to "normal Reset Node".

The Index "Save Current Reset Modifier" determines that, in the event of a Reset Node command, the actual BLxx module list (Objects $3090_{hex}$ and $3091_{hex}$) will be saved in non-volatile memory, and then followed by a hard processor reset. This hardware reset is necessary, because changes to the reference module list cannot be dynamically accepted in the CANopen I/O mirror. If the module list is altered, all the CANopen parameters will be reset to the default values.

For writing, the value that is transferred in Unsigned 32 format will be interpreted as a string:

*Table 60: Save and Hard-ware-Reset selection*

| **MSB** | | | **LSB** |
|---|---|---|---|
| v | a | s | c |
| $76_{hex}$ | $61_{hex}$ | $73_{hex}$ | $63_{hex}$ |

*Table 61: Normal Reset selection*

| **MSB** | | | **LSB** |
|---|---|---|---|
| t | s | r | s |
| $74_{hex}$ | $73_{hex}$ | $72_{hex}$ | $73_{hex}$ |

After the next Reset-Node or the next "Reset Communication" command, the operating mode will in all cases be reset to "normal Reset Node".

### 4.7.3 Object $2400_{hex}$ - System Voltages (only BL67)

This object enables the reading of up to 4 system voltages.

At the moment, the BL67-GW-CO supports the reading of $U_{sys}$ from sub-index 1. The sub-indices 2...4 give out 0.

*Table 62: Object $2400_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $2400_{hex}$ |
| Name | System Voltages |
| Object code | ARRAY |
| Data type | Unsigned16 |
| Access | ro |
| PDO mapping | No |
| Default value | No |

### 4.7.4 Object $2401_{hex}$ - System Currents (only BL67)

This object enables the reading of up to 4 system currents.

*Table 63: Object $2401_{hex}$*

| **Object description** | |
|---|---|
| INDEX | $2401_{hex}$ |
| Name | System Currents |
| Object code | ARRAY |
| Data type | Unsigned16 |
| Access | ro |
| PDO mapping | No |
| Default value | No |

# 5 Objects for digital input modules

**5.1 Digital input modules BLxx ..... 2**

**5.2 General object overview for digital input modules ..... 2**

5.2.1 Object $6000_{hex}$ – Read Input 8 Bit ..... 4
5.2.2 Object $6020_{hex}$ – Read Input Bit (1 to 128), Object $6021_{hex}$ – Read Input Bit (129 to 256), Object $6022_{hex}$ – Read Input Bit (257 to 288) ..... 5
5.2.3 Object $6100_{hex}$ – Read Input 16 Bit ..... 6
5.2.4 Object $6120_{hex}$ – Read Input 32 Bit ..... 6

## 5.1 Digital input modules BL××

The objects are used in the following modules:

*Table 64: BL×× - digital input modules*

| Product family | Module |
|---|---|
| BL20 | BL20-×DI-24VDC-P |
| | BL20-×DI-24VDC-N |
| | BL20-2DI-120/230VAC |
| | BL20-4DI-NAMUR |
| | BL20-E-×DI-24VDC-P |
| BL67 | BL67-×DI-P |
| | BL67-×DI-N |
| | BL67-×DI-PD |
| BLC | miscellaneous |

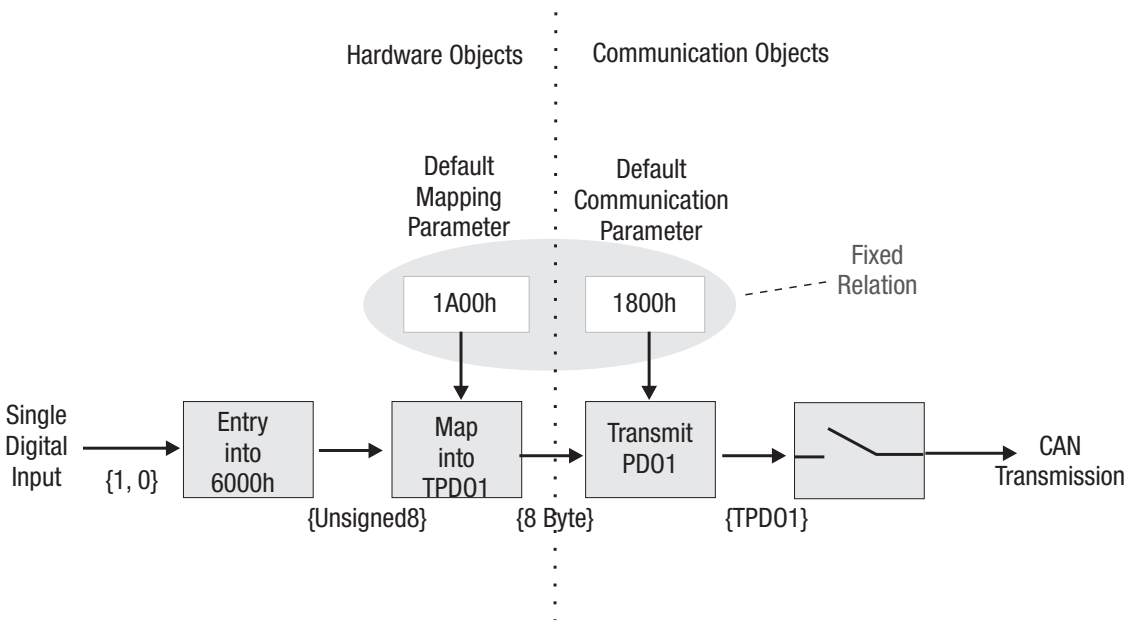
## 5.2 General object overview for digital input modules

*Table 65: General object overview for digital input modules*

| Object | Name | Page |
|---|---|---|
| $3064_{hex}$ | XBIParam Dword | page 14-10 |
| $6000_{hex}$ | Read Input 8 Bit | page 5-4 |
| $6020_{hex}$ | Read Input Bit 1 to 128 | page 5-5 |
| $6021_{hex}$ | Read Input Bit 129 to 256 | page 5-5 |
| $6022_{hex}$ | Read Input Bit 257 to 288 | page 5-5 |
| $6100_{hex}$ | Read Input 16 Bit | page 5-6 |
| $6120_{hex}$ | Read Input 32 Bit | page 5-6 |

The following figure shows the relationship between the digital input objects for an 8-bit access:

*Figure5: Relationship between the digital digital input modules (according to CiA Standard DS401)*



### 5.2.1 Object $6000_{hex}$ – Read Input 8 Bit

The object presents the values for the digital input modules in 8-bit groups.

A total of 36 groups (each 8 bit) can be displayed (288 digital input channels).

PDO mapping of this object is always executed by default and automatically for the first 8 sub-indexes. This corresponds to 64 input channels.

If more than 64 input channels are present, then the PDO-mapping must be carried out by the user.

*Table 66: Object $6000_{hex}$*

| **Feature** | **Description/ Value** |
|---|---|
| Name | Read Input 8 Bit |
| Object code | ARRAY |
| PDO mapping | Yes |
| | |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $24_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |

### 5.2.2 Object $6020_{hex}$ – Read Input Bit (1 to 128), Object $6021_{hex}$ – Read Input Bit (129 to 256), Object $6022_{hex}$ – Read Input Bit (257 to 288)

The objects defined the values bit-wise. Each sub-index for these objects is a Boolean value.

A total of 128 bits can be represented (128 digital input channels).

If more than 128 input channels are present, then object $6021_{hex}$ is used.

If more than 256 input channels are present, then object $6022_{hex}$ is used.

Since the number of digital input channels in a station is limited to 288, it is not possible to make use of the complete range of the array in object $6022_{hex}$.

*Table 67: Objects $6020_{hex}$, $6021_{hex}$ and $6022_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Read Input Bit |
| Object code | ARRAY |
| PDO mapping | Yes |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $80_{hex}$ |
| Data type | Boolean |
| Access | ro |
| Default value, BL×× | No |

### 5.2.3 Object $6100_{hex}$ – Read Input 16 Bit

The object presents the values for the digital input modules in 16-bit groups.

A total of 18 groups (each 16 bit) can be displayed (288 digital input channels).

*Table 68: Object $6100_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Read Input 16 Bit |
| Object code | ARRAY |
| PDO mapping | Yes |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Sub-index | $01_{hex}$ to $12_{hex}$ |
| Data type | Unsigned16 |
| Access | ro |

### 5.2.4 Object $6120_{hex}$ – Read Input 32 Bit

The object presents the values for the digital input modules in 32-bit groups.

A total of 9 groups (each 32 bit) can be displayed (288 digital input channels).

*Table 69: Object $6120_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Read Input 32 Bit |
| Object code | ARRAY |
| PDO mapping | Yes |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Sub-index | $01_{hex}$ to $09_{hex}$ |
| Data type | Unsigned32 |
| Access | ro |

# 6 Objects for digital output modules


**6.1 Digital output modules BL×× ..... 2**

**6.2 General object overview for digital output modules ..... 3**

6.2.1 Object $6200_{hex}$ – Write Output 8 Bit ..... 4
6.2.2 Object $6206_{hex}$ – Error Mode Output 8 Bit ..... 5
6.2.3 Object $6207_{hex}$ – Error State Output 8 Bit ..... 6
6.2.4 Object $6220_{hex}$ – Write Output Bit (1 to 128), Object $6221_{hex}$ – Write Output Bit (129 to 256), Object $6222_{hex}$ – Write Output Bit (257 to 288) ..... 7
6.2.5 Object $6250_{hex}$ – Error Mode Output Bit (1 to 128), Object $6251_{hex}$ – Error Mode Output Bit (129 to 256), Object $6252_{hex}$ – Error Mode Output Bit (257 to 288) ..... 8
6.2.6 Object $6260_{hex}$ – Error State Output Bit (1 to 128), Object $6261_{hex}$ – Error State Output Bit (129 to 256), Object $6262_{hex}$ – Error State Output Bit (257 to 288) ..... 9
6.2.7 Object $6300_{hex}$ – Write Output 16 Bit ..... 10
6.2.8 Object $6306_{hex}$ – Error Mode Output 16 Bit ..... 10
6.2.9 Object $6307_{hex}$ – Error State Output 16 Bit ..... 11
6.2.10 Object $6320_{hex}$ – Write Output 32 Bit ..... 12
6.2.11 Object $6326_{hex}$ – Error Mode Output 32 Bit ..... 13
6.2.12 Object $6327_{hex}$ – Error State Output 32 Bit ..... 14

## 6.1 Digital output modules BL××

The objects are used in the following modules:

*Table 70: BL×× - digital output modules*

| Product family | Module |
|---|---|
| BL20 | BL20-×DO-24VDC-0.5A-P |
| | BL20-×DO-24VDC-0.5A-N |
| | BL20-2DO-24VDC-2A-P |
| | BL20-×DO-24VDC-0.5A-P |
| | BL20-2DO-120/230VAC-0.5A |
| | BL20-2DO-R-NC |
| | BL20-2DO-R-NO |
| | BL20-2DO-R-CO |
| BL67 | BL67-×DO-0.5A-P |
| | BL67-4DO-2A-P |
| | BL67-4DO-4A-P |
| | BL67-16DO-0.1A-P |
| | BL67-4DO-2A-N |
| | BL67-8DO-0.5A-N |
| | BL67-8DO-R-NO |
| | BL67-4DI4DO-PD |
| | BL67-8XSG-PD |
| BLC | miscellaneous |

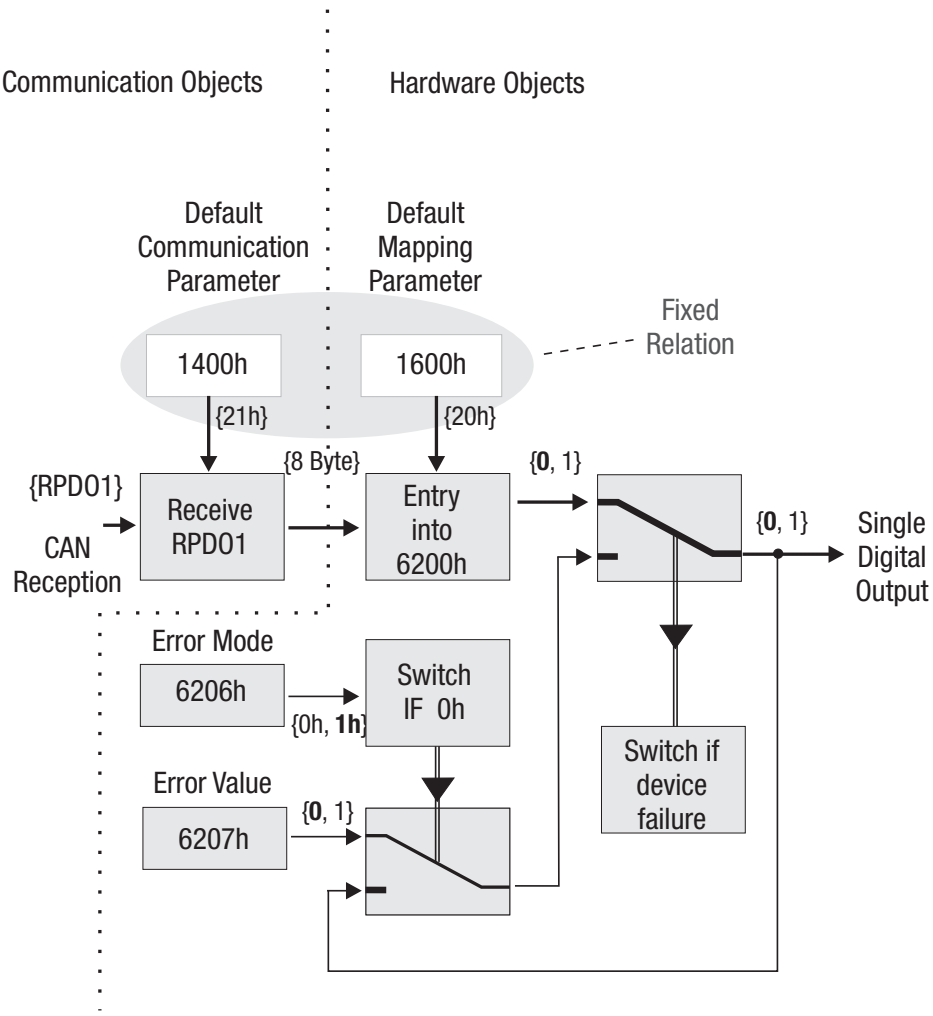
## 6.2 General object overview for digital output modules

*Table 71: General object overview for digital output modules*

| Object | Name | page |
|---|---|---|
| $6200_{hex}$ | Write Output 8 Bit | page 6-4 |
| $6206_{hex}$ | Error Mode Output 8 Bit | page 6-5 |
| $6207_{hex}$ | Error State Output 8 Bit | page 6-6 |
| $6220_{hex}$ | Write Output 8 Bit (1 to 128) | page 6-7 |
| $6221_{hex}$ | Write Output 8 Bit (129 to 256) | page 6-7 |
| $6222_{hex}$ | Write Output 8 Bit (257 to 288) | page 6-7 |
| $6250_{hex}$ | Error Mode Output Bit (1 to 128) | page 6-8 |
| $6251_{hex}$ | Error Mode Output Bit (129 to 256) | page 6-8 |
| $6252_{hex}$ | Error Mode Output Bit (257 to 288) | page 6-8 |
| $6260_{hex}$ | Error State Output Bit (1 to 128) | page 6-9 |
| $6261_{hex}$ | Error State Output Bit (129 to 256) | page 6-9 |
| $6300_{hex}$ | Write Output 16 Bit | page 6-10 |
| $6306_{hex}$ | Error Mode Output 16 Bit | page 6-10 |
| $6307_{hex}$ | Error State Output 16 Bit | page 6-11 |
| $6320_{hex}$ | Write Output 32 Bit | page 6-12 |
| $6326_{hex}$ | Error Mode Output 32 Bit | page 6-13 |
| $6327_{hex}$ | Error State Output 32 Bit | page 6-14 |

The following figure shows the relationship between the digital output objects for an 8-bit access:

*Figure6: Relationship between the digital output objects (according to CiA Draft Standard DS401)*




### 6.2.1 Object $6200_{hex}$ – Write Output 8 Bit

The object presents the values for the digital output modules in 8-bit groups.

A total of 36 groups (each 8 bit) can be displayed (288 digital output channels).

PDO mapping of this object is always executed by default and automatically for the first 8 sub-indexes. This corresponds to 64 digital output channels. If more than 64 output channels are present, then the PDO-mapping must be carried out by the user.

*Table 72: Object $6200_{hex}$*

| **Feature** | **Description/ Value** |
|---|---|
| Name | Write Output 8 Bit |
| Object code | ARRAY |
| PDO mapping | Yes |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $24_{hex}$ |
| Data type | Unsigned8 |
| Access | rw |
| Default value, BL×× | 0 |

### 6.2.2 Object $6206_{hex}$ – Error Mode Output 8 Bit

The object defines values in 8-bit groups.

A total of 36 groups (each 8 bit) can be displayed (288 digital output channels).

It defines (for each digital output channel) whether or not the output should take on a substitute value in the event of an error.

The following applies:

0 The output maintains its value if an error occurs.

1 The output is set to a substitute value if an error occurs.

The substitute values for the digital output channels are defined by the Error state output object (e.g. $6207_{hex}$).

*Table 73: Object $6206_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Error Mode Output 8 Bit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $24_{hex}$ |
| Data type | Unsigned8 |
| Access | rw |
| Default value, BL×× | $FF_{hex}$ |

### 6.2.3 Object $6207_{hex}$ – Error State Output 8 Bit

The object defines values in 8-bit groups. A total of 36 groups (each 8 bit) can be displayed (288 digital output channels).

A substitute value is assigned to each analog channel. The substitute values will only be sent in case of an error, object Error Mode Output ($6206_{hex}$) is set to "1" for the respective channel.

Substitute values:

0 The output will be switched off if an error occurs.

1 The output will be switched on if an error occurs.

*Table 74: Object $6207_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Error State Output 8 Bit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $24_{hex}$ |
| Data type | Unsigned8 |
| Access | rw |
| Default value, BL×× | $00_{hex}$ |

### 6.2.4 Object $6220_{hex}$ – Write Output Bit (1 to 128), Object $6221_{hex}$ – Write Output Bit (129 to 256), Object $6222_{hex}$ – Write Output Bit (257 to 288)

The objects defined the values bit-wise.

Each sub-index for these objects is a Boolean value.

A total of 128 bits can be represented (128 digital output channels).

If more than 128 output channels are present, then object $6221_{hex}$ is used.

If more than 256 output channels are present, then object $6222_{hex}$ is used.

Since the number of digital output channels in a station is limited to 288, it is not possible to make use of the complete range of the array in object $6222_{hex}$.

*Table 75: Objected $6220_{hex}$, $6221_{hex}$, $6222_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Write Output Bit |
| Object code | ARRAY |
| PDO mapping | Yes |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |

*Table 75: Objected $6220_{hex}$, $6221_{hex}$, $6222_{hex}$*

| Feature | Description/ Value |
|---|---|
| Sub-index | $01_{hex}$ to $80_{hex}$ |
| Data type | Boolean |
| Access | rw |
| Default value, BL×× | 0 |

### 6.2.5 Object $6250_{hex}$ – Error Mode Output Bit (1 to 128), Object $6251_{hex}$ – Error Mode Output Bit (129 to 256), Object $6252_{hex}$ – Error Mode Output Bit (257 to 288)

The objects define the values bit-wise. Each sub-index for these objects is a Boolean value.

A total of 128 bits can be represented (128 digital output channels).

If more than 128 output channels are present, then object $6251_{hex}$ is used. If more than 256 output channels are present, then object $6252_{hex}$ is used.

Since the number of digital output channels in a station is limited to 288, it is not possible to make use of the complete range of the array in object $6522_{hex}$.

It defines (for each digital output channel) whether or not the output should take on a substitute value in the event of an error. The following applies:

0 The output maintains its value if an error occurs.

1 The output is set to a substitute value if an error occurs.

The substitute values for the digital output channels are defined by the Error state output objects (e.g. $6260_{hex}$, $6261_{hex}$ and $6262_{hex}$).

*Table 76: Objected $6250_{hex}$, $6251_{hex}$, $6252_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Error Mode Output Bit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $80_{hex}$ |
| Data type | Boolean |
| Access | rw |
| Default value, BL×× | 1 |

### 6.2.6 Object $6260_{hex}$ – Error State Output Bit (1 to 128), Object $6261_{hex}$ – Error State Output Bit (129 to 256), Object $6262_{hex}$ – Error State Output Bit (257 to 288)

The objects define the values bit-wise. Each sub-index for these objects is a Boolean value.

A total of 128 bits can be represented (128 digital output channels).

If more than 128 output channels are present, then object $6261_{hex}$ is used.If more than 256 output channels are present, then object $6262_{hex}$ is used.

Since the number of digital output channels in a station is limited to 288, it is not possible to make use of the complete range of the array in object $6262_{hex}$.

A substitute value is assigned to each analog channel. The substitute values will only be sent in case of an error, object Error Mode Output object ($62506250_{hex}$, $6251_{hex}$ and $6252_{hex}$) is set to "1" for the respective channel.

Substitute values:

0 The output will be switched off if an error occurs.

1 The output will be switched on if an error occurs.

*Table 77: Objected $6260_{hex}$, $6261_{hex}$, $6262_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Error State Output Bit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $80_{hex}$ |
| Data type | Boolean |
| Access | rw |
| Default value, BL×× | 0 |

### 6.2.7 Object $6300_{hex}$ – Write Output 16 Bit

The object presents the values for the digital output modules in 16-bit groups.

A total of 18 groups (each 16 bit) can be displayed (288 digital output channels).

*Table 78: Object $6300_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Write Output 16 Bit |
| Object code | ARRAY |
| PDO mapping | Yes |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $12_{hex}$ |
| Data type | Unsigned16 |
| Access | rw |
| Default value, BL×× | $0000_{hex}$ |

### 6.2.8 Object $6306_{hex}$ – Error Mode Output 16 Bit

The object defines values in 16-bit groups. A total of 18 groups (each 16 bit) can be displayed (288 digital output channels).

It defines (for each digital output channel) whether or not the output should take on a substitute value in the event of an error. The following applies:

0 The output maintains its value if an error occurs.

1 The output is set to a substitute value if an error occurs.

The substitute values for the digital output channels are defined by the Error state output object (e.g. $6307_{hex}$).

*Table 79: Object $6306_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Error Mode Output 16 Bit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |

| *Table 79: Object $6306_{hex}$* | **Feature** | **Description/ Value** |
|---|---|---|
| | Sub-index | $01_{hex}$ to $12_{hex}$ |
| | Data type | Unsigned16 |
| | Access | rw |
| | Default value, BL×× | $FFFF_{hex}$ |

### 6.2.9 Object $6307_{hex}$ – Error State Output 16 Bit

The object defines values in 16-bit groups. A total of 18 groups (each 16 bit) can be displayed (288 digital output channels).

A substitute value is assigned to each analog channel. The substitute values will only be sent in case of an error, object Error Mode Output ($6306_{hex}$) is set to "1" for the respective channel.

Substitute values:

0 The output will be switched off if an error occurs.

1 The output will be switched on if an error occurs.

| *Table 80: Object $6307_{hex}$* | **Feature** | **Description/ Value** |
|---|---|---|
| | Name | Error State Output 16 Bit |
| | Object code | ARRAY |
| | PDO mapping | No |
| | Sub-index | $00_{hex}$ |
| | Data type | Unsigned8 |
| | Access | ro |
| | Default value, BL×× | No |
| | Sub-index | $01_{hex}$ to $12_{hex}$ |
| | Data type | Unsigned16 |
| | Access | rw |
| | Default value, BL×× | $0000_{hex}$ |

### 6.2.10 Object $6320_{hex}$ – Write Output 32 Bit

The object presents the values for the digital output modules in 32-bit groups.

A total of 9 groups (each 32 bit) can be displayed (288 digital output channels).

*Table 81: Object $6320_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Write Output Bit |
| Object code | ARRAY |
| PDO mapping | Yes |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $09_{hex}$ |
| Data type | Unsigned32 |
| Access | rw |
| Default value, BL×× | $0000\ 0000_{hex}$ |

### 6.2.11 Object $6326_{hex}$ – Error Mode Output 32 Bit

The object defines values in 32-bit groups. A total of 9 groups (each 32 bit) can be displayed (288 digital output channels).

It defines (for each digital output channel) whether or not the output should take on a substitute value in the event of an error. The following applies:

0 The output maintains its value if an error occurs.

1 The output is set to a substitute value if an error occurs.

The substitute values for the digital output channels are defined by the Error state output object (e.g. $6327_{hex}$).

*Table 82: Object $6326_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Error Mode Output 32 Bit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $09_{hex}$ |
| Data type | Unsigned32 |
| Access | rw |
| Default value, BL×× | FFFF $FFFF_{hex}$ |

### 6.2.12 Object $6327_{hex}$ – Error State Output 32 Bit

The object defines values in 32-bit groups. A total of 9 groups (each 32 bit) can be displayed (288 digital output channels).

A substitute value is assigned to each analog channel. The substitute values will only be sent in case of an error, object Error Mode Output ($6326_{hex}$) is set to "1" for the respective channel.

Substitute values:

0 The output will be switched off if an error occurs.

1 The output will be switched on if an error occurs.

*Table 83: Object $6327_{hex}$*

| **Feature** | **Description/ Value** |
|---|---|
| Name | Error State Output 32 Bit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $09_{hex}$ |
| Data type | Unsigned32 |
| Access | rw |
| Default value, BL×× | $0000\ 0000_{hex}$ |

# 7 Objects for digital combi modules


**7.1 Digital combi modules BL×× ..... 2**

**7.2 General object overview for digital combi modules ..... 2**

## 7.1 Digital combi modules BL××

The objects are used in the following modules:

*Table 84: BL×× - digital combi modules*

| Product family | Module |
|---|---|
| BL67 | BL67-4DI-4DO-PD |
| | BL67-8XSG-PD |
| BLC | miscellaneous |

## 7.2 General object overview for digital combi modules

**Note**

The objects for digital combi modules correspond to those of the digital in- and output modules. Please read chapter 5, Objects for digital input modules and chapter 6, Objects for digital output modules.

The parameterization of the combi modules is realized through the manufacturer specific objects described in the following.

*Table 85: General object overview for digital combi modules*

| Object | Name | page |
|---|---|---|
| **Input objects** | | |
| $6000_{hex}$ | Read Input 8 Bit | page 5-4 |
| $6020_{hex}$, $6021_{hex}$, $6022_{hex}$ | Read Input 8 Bit | page 5-5 |
| $6100_{hex}$ | Read Input 16 Bit | page 5-6 |
| $6120_{hex}$ | Read Input 32 Bit | page 5-6 |
| **Output objects** | | |
| $6200_{hex}$ | Write Output 8 Bit | page 6-4 |
| $6206_{hex}$ | Error Mode Output 8 Bit | page 6-5 |
| $6207_{hex}$ | Error State Output 8 Bit | page 6-6 |
| $6220_{hex}$, $6221_{hex}$, $6222_{hex}$ | Write Output 8 Bit | page 6-7 |
| $6250_{hex}$, $6251_{hex}$, $6252_{hex}$ | Error Mode Output Bit | page 6-8 |
| $6260_{hex}$, $6261_{hex}$ | Error State Output Bit (1 to 128), Bit (129 to 256) | page 6-9 |
| $6300_{hex}$ | Write Output 16 Bit | page 6-10 |
| $6306_{hex}$ | Error Mode Output 16 Bit | page 6-10 |
| $6307_{hex}$ | Error State Output 16 Bit | page 6-11 |
| $6320_{hex}$ | Write Output 32 Bit | page 6-12 |
| $6326_{hex}$ | Error Mode Output 32 Bit | page 6-13 |
| $6327_{hex}$ | Error State Output 32 Bit | page 6-14 |

| | Parameter object | | |
|---|---|---|---|
| | $3064_{hex}$ | XBIParam Dword | page 14-10 |

# 8 Objects for analog input modules


**8.1 Analog input modules BL××..... 2**

**8.2 General object overview for analog input modules..... 2**

8.2.1 Object $5420_{hex}$ – Manu Spec Analog Input Range ..... 3
– Analog input modules, current ..... 4
– Analog input modules, voltage ..... 4
– analog input modules, current/ voltage ..... 5
– Analog input modules, PT/Ni ..... 5
– Analog input modules, 2AI-THERMO/TC ..... 6
– Analog input modules, 4AI-TC ..... 7
– Analog input modules, -8AI-U/I-4PT/NI ..... 8
8.2.2 Object $6401_{hex}$ – Read Analog Input 16 Bit ..... 10
8.2.3 Object $6421_{hex}$ – Analog Input Interrupt Trigger Selection ..... 11
8.2.4 Object $6422_{hex}$ – Analog Input Interrupt Source ..... 13
8.2.5 Object $6423_{hex}$ – Analog Input Global Interrupt Enable ..... 14
8.2.6 Object $6424_{hex}$ – Analog Input Interrupt Upper Limit Integer ..... 14
8.2.7 Object $6425_{hex}$ – Analog Input Interrupt Lower Limit Integer ..... 15
8.2.8 Object $6426_{hex}$ – Analog Input Interrupt Delta Unsigned ..... 15
8.2.9 Object $6427_{hex}$ – Analog Input Interrupt Negative Delta Unsigned ..... 16
8.2.10 Object $6428_{hex}$ – Analog Input Interrupt Positive Delta Unsigned ..... 16

## 8.1 Analog input modules BL××

The objects are used in the following modules:

*Table 86: BL×× - analog input modules*

| Product family | Module |
|---|---|
| BL20 | BL20-×AI-I(0/4...20MA) |
| | BL20-×AI-U(-10/0...+10VDC) |
| | BL20-2AI-PT/NI-2/3 |
| | BL20-2AI-THERMO-PI |
| | BL20-4AI-U/I |
| | BL20-E-8AI-U/I-4PT/NI |
| | BL20-2AIH-I |
| BL67 | BL67-2AI-I |
| | BL67-2AI-V |
| | BL67-2AI-PT |
| | BL67-2AI-TC |
| | BL67-4AI-V/I |
| BLC | miscellaneous |

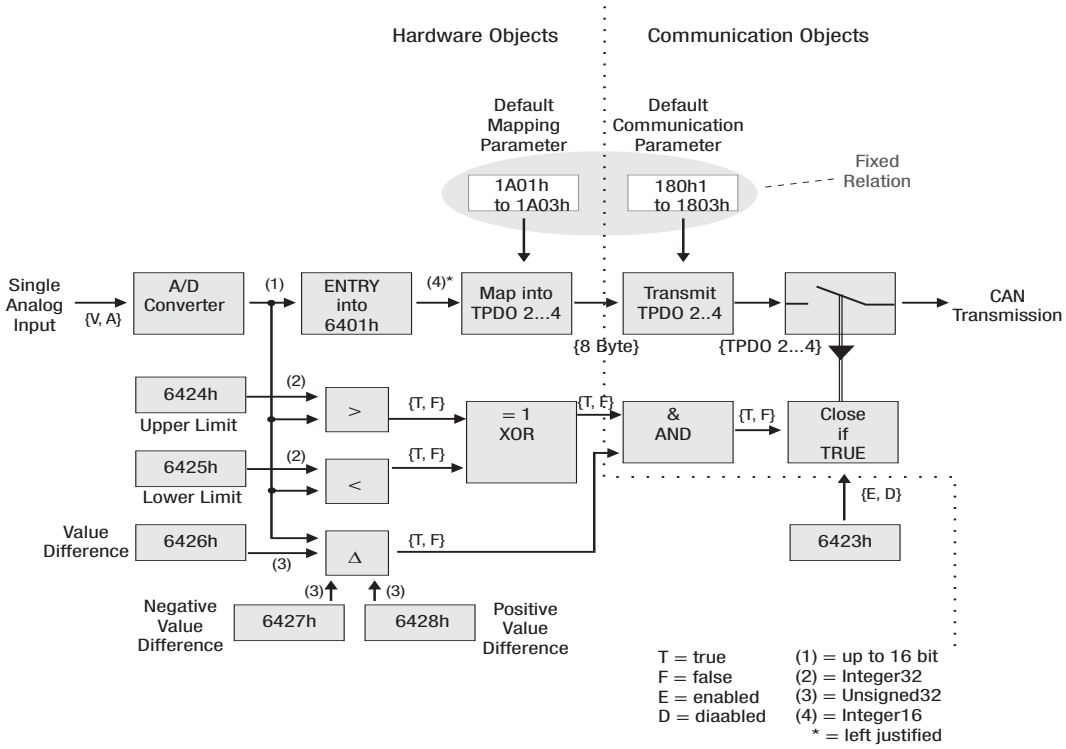
## 8.2 General object overview for analog input modules

*Table 87: General object overview for analog input modules*

| Object | Name | page |
|---|---|---|
| $5420_{hex}$ | Manu Spec Analog Input Range | page 8-3 |
| $6401_{hex}$ | Read analog Input 16 Bit | page 8-10 |
| $6421_{hex}$ | Analog Input Interrupt Trigger Selection | page 8-11 |
| $6422_{hex}$ | Analog Input Interrupt Source | page 8-13 |
| $6423_{hex}$ | Analog Input Global Interrupt Enable | page 8-14 |
| $6424_{hex}$ | Analog Input Interrupt Upper Limit Integer | Seite 8-14 |
| $6425_{hex}$ | Analog Input Interrupt Lower Limit Integer | Seite 8-15 |
| $6426_{hex}$ | Analog Input Interrupt Delta Unsigned | Seite 8-15 |
| $6427_{hex}$ | Analog Input Interrupt Negative Delta Unsigned | Seite 8-16 |
| $6428_{hex}$ | Analog Input Interrupt Positive Delta Unsigned | Seite 8-16 |

The following figure shows the relationship between the analog input objects for an 16-bit access:

*Figure7: Relationship between the analog output objects (according to CiA Draft Standard DS401)*




### 8.2.1 Object $5420_{hex}$ – Manu Spec Analog Input Range

The object defines the parameters of the analog input channels.

Write accesses initiate a parameter update via the internal BLxx-module bus.

The parameter is stored as a non-volatile parameter in the gateway and is restored with every node reset.

The Sub-indices $01_{hex}$ – $8E_{hex}$ define the parameters for the analog input channel 1 to 142.

*Table 88: Object $5420_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Manu Spec Analog Input Range |
| Object code | ARRAY |
| PDO mapping | No |
| | |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BLxx | No |
| | |
| Sub-index | $01_{hex}$ bis $8E_{hex}$ |

*Table 88: Object 5420$_{hex}$*

| Feature | Description/ Value |
|---|---|
| Data type | Unsigned16 |
| Access | rw |
| Default value, BL×× | No |

The structure of the 2 bytes of parameter data depends on the module concerned.

A sub-index is assigned for **each** channel. The following explains the structure for each module type.

**Analog input modules, current**

*Table 89: Parameters, analog input modules, current*

**A***Default-setting*

| Byte | Bit | Parameters | Value/ meaning |
|---|---|---|---|
| n | 0 | current mode | 0 = 0...20 mA **A**<br>1 = 4...20 mA |
| | 1 | Value representation | 0 = Integer (15 bit + sign) **A**<br>1 = reserved |
| | 2 | Diagnostic | 0 = release **A**<br>1 = block |
| | 3 | Channel x | 0 = activate **A**<br>1 = deactivate |
| | 4 to 7 | reserved | |

**Analog input modules, voltage**

*Table 90: Parameter, analog input modules, voltage*

**A***Default-setting*

| Byte | Bit | Parameters | Value/ meaning |
|---|---|---|---|
| n | 0 | voltage mode | 0 = 0...10 V **A**<br>1 = -10...10 V |
| | 1 | Value representation | 0 = Integer (15 bit + sign) **A**<br>1 = reserved |
| | 2 | Diagnostic | 0 = release **A**<br>1 = block |
| | 3 | Channel x | 0 = activate **A**<br>1 = deactivate |
| | 4 to 7 | reserved | |

### analog input modules, current/ voltage

*Table 91: Parameter, analog input modules, current/voltage*

**A***Default-setting*

| Byte | Bit | Parameters | Value/ meaning |
|---|---|---|---|
| n | 0 | Range | 0 = 0...10 V/0...20 mA **A**<br>1 = -10...10 V/4...20 mA |
| | 1 | Value representation | 0 = Integer (15 bit + sign) **A**<br>1 = reserved |
| | 2 | Diagnostic | 0 = release **A**<br>1 = block |
| | 3 | Channel x | 0 = activate **A**<br>1 = deactivate |
| | 4 | Operation mode | 0 = voltage **A**<br>1 = current |
| | 5 to 7 | reserved | |

### Analog input modules, PT/Ni

*Table 92: Parameter, analog input modules, PT/Ni*

**A***Default-setting*

| Byte | Bit | Parameters | Value/ meaning |
|---|---|---|---|
| n | 0 | Mains suppression | 0 = 50 Hz **A**<br>1 = 60 Hz |
| | 1 | Value representation | 0 = Integer (15 bit + sign) **A**<br>1 = reserved |
| | 2 | Diagnostic | 0 = release **A**<br>1 = block |
| | 3 | Channel x | 0 = activate **A**<br>1 = deactivate |
| | 4 to 7 | Element | 0000 = Pt100, -200...850 °C **A**<br>0001 = Pt100, -200...150 °C<br>0010 = Ni100, -60...250 °C<br>0011 = Ni100, -60...150 °C<br>0100 = Pt200, -200...850 °C<br>0101 = Pt200, -200...150 °C<br>0110 = Pt500, -200...850 °C<br>0111 = Pt500, -200...150 °C<br>1000 = Pt1000, -200...850 °C<br>1001 = Pt1000, -200...150 °C<br>1010 = Ni1000, -60...250 °C<br>1011 = Ni1000, -60...150 °C<br>1100 = resistance, 0...100 Ω<br>1101 = resistance, 0...200 Ω<br>1110 = resistance, 0...400 Ω<br>1111 = resistance, 0...1000 Ω |
| n +1 | 0 | Measurement mode | 0 = 2-wire **A**<br>1 = 3-wire |

### Analog input modules, 2AI-THERMO/TC

*Table 93: Parameter, analog input modules, 4AI-TC*

| Byte | Bit | Parameters | Value/ meaning |
|---|---|---|---|
| n | 0 | Cold junction sensor | 0 = Pt1000 **A**<br>1 = Pt100 |
| | 1 | Value representation | 0 = Integer (15 bit + sign) **A**<br>1 = reserved |
| | 2 | Diagnostic | 0 = release **A**<br>1 = block |
| | 3 | Channel x | 0 = activate **A**<br>1 = deactivate |
| | 6 to 4 | Element | 0000 = Type K, -270...1370 °C<br>0001 = Type B, +100....1820 °C<br>0010 = Type E, -270...1000 °C<br>0011 = Type J, -210...1200 °C<br>0100 = Type N, -270...1300 °C<br>0101 = Type R, -50...1760 °C<br>0110 = Type S, -50...1540 °C<br>0111 = Type T, -270...400 °C<br>1000 = ±50 mV<br>1001 = ±100 mV<br>1010 = ±500 mV<br>1011 = ±1000 mV<br>1100 = Type J, -454... 2498 °F<br>1101 = Type J, -346... 2192 °F<br>1110 = Type C, 0...2320 °C<br>1111 = Type G, 0...2320 °C |
| | 7 | reserved | |

**A** *Default-setting*

### Analog input modules, 4AI-TC

*Table 94: Parameter, analog input modules, THERMO*

| Byte | Bit | Parameters | Value/ meaning |
|---|---|---|---|
| n | 0 | Mains suppression | 0 = 50 Hz **A**<br>1 = 60 Hz |
| | 1 | Value representation | 0 = Integer (15 bit + sign) **A**<br>1 = reserved |
| | 2 | Diagnostic | 0 = release **A**<br>1 = block |
| | 3 | Channel x | 0 = activate **A**<br>1 = deactivate |
| | 6 to 4 | Element | 0000 = Type K, -270...1370 °C<br>0001 = Type B, +100....1820 °C<br>0010 = Type E, -270...1000 °C<br>0011 = Type J, -210...1200 °C<br>0100 = Type N, -270...1300 °C<br>0101 = Type R, -50...1760 °C<br>0110 = Type S, -50...1540 °C<br>0111 = Type T, -270...400 °C<br>1000 = ±50 mV<br>1001 = ±100 mV<br>1010 = ±500 mV<br>1011 = ±1000 mV |
| | 7 | reserved | |

**A***Default-setting*

### Analog input modules, -8AI-U/I-4PT/NI

*Table 95: Module parameters BL20-E-8AI-U/I-4PT/NI*

| Assignment | | Parameters | Meaning |
|---|---|---|---|
| Byte | Bit | | |
| 0 - 7 | 0 -5 | Operation mode Kx | 000000 = Voltage 0...10 V DC Standard **A** |
| | | | 000001 = Voltage -10...10 V DC Standard |
| | | | 000010 = Voltage 0...10 V DC PA (NE 43) |
| | | | 000011 = Voltage -10...10 V DC PA (NE 43) |
| | | | 000100 = Voltage -10...10 V DC Ext. range |
| | | | 000101 = Voltage 0...10 V DC Ext. range |
| | | | 001000 = Current 0 ... 20 mA Standard |
| | | | 001001 = Current 4 ... 20 mA Standard |
| | | | 001010 = Current 0 ... 20 mA PA (NE 43) |
| | | | 001011= Current 4 ... 20 mA PA (NE 43) |
| | | | 001100 = Current 0 ... 20 mA Ext. range |
| | | | 001001 = Current 4 ... 20 mA Ext. range |
| | | | 010000 = Pt100-200 °C ... 850 °C, 2-wire **B** |
| | | | 010001 = Pt 100-200 °C ... 150 ? 2-wire |
| | | | 010010 = Pt200-200 °C ... 850 °C 2-wire |
| | | | 010011 = Pt200-200 °C ... 150 °C 2-wire |
| | | | 010100 = Pt500-200 °C ... 850 °C 2-wire |
| | | | 010101 = Pt500-200 °C ... 150 °C 2-wire |
| | | | 010110 = Pt1000-200 °C ... 850 °C 2-wire |
| | | | 010111 = Pt1000-200 °C ... 150 °C 2-wire |
| | | | 011000 = Pt100-200 °C ... 850 °C 3-wire |
| | | | 011001 = Pt100-200 °C ... 150 °C 3-wire |
| | | | 011010 = Pt200-200 °C ... 850 °C 3-wire |
| | | | 011011 = Pt200-200 °C ... 150 °C 3-wire |
| | | | 011100 = Pt500-200 °C ... 850 °C 3-wire |
| | | | 011101 = Pt500-200 °C ... 150 °C 3-wire |
| | | | 011110 = Pt1000-200 °C ... 850 °C 3-wire |
| | | | 011111 = Pt1000-200 °C ... 150 °C 3-wire |
| | | | 100000 = Ni100, -60°C..150°C, 2-wire |
| | | | 100001 = Ni100, -60°C..150°C, 2-wire |

**A** *Default-settings*

**B** *In Pt, Ni and R-, measurement, only the first of the used channels is parameterized (channel 1,3,5,7).. The parameterization of the second channel will be ignored.*

*Table 95: Module parameters BL20-E-8AI-U/I-4PT/NI*

| Assignment | | Parameters | Meaning |
|---|---|---|---|
| Byte | Bit | | |
| 0 - 7 | 0 -5 | Operation mode Kx | 100010 = Ni1000, -60°C..250°C, 2-wire |
| | | | 100011 = Ni1000, -60°C..250°C, 2-wire |
| | | | 100100 = NI1000TK5000, -60 °C .. 150°C, 2-wire |
| | | | 100101 = Ni100, -60°C..150°C, 3-wire |
| | | | 100110 = Ni100, -60°C..150°C, 3-wire |
| | | | 100111 = Ni1000, -60°C..250°C, 3-wire |
| | | | 101010 = Ni1000, -60°C..250°C, 3-wire |
| | | | 101001 = NI1000TK5000, -60 °C .. 150°C, 3-wire |
| | | | 110000 = resistance, 0 ... 250 W |
| | | | 110001 = resistance, 0 ... 400 W |
| | | | 110010 = resistance, 0 ... 800 W |
| | | | 110011 = resistance, 0 ... 1000 W |
| | | | 110100 = resistance, 0 ... 2000 W |
| | | | 110101 = channel not active |
| 0 - 7 | 6 | Value representation Kx | Integer (15bit + sign) A **A** |
| | | | 12 bit (left-justified) |
| | 7 | Diagnostic Kx | release **A** |
| | | | block |

### 8.2.2 Object $6401_{hex}$ – Read Analog Input 16 Bit

The object represents the measured values for the analog input modules with 16 bits for each channel.



**Attention**

The process data traffic for the analog input values is not started until the object $6423_{hex}$ is switched from the default setting FALSE to TRUE!



**Attention**

he possibility of 12-bit value representation (left-justified) is not useful for CANopen since all reference values (upper limit, lower limit) must be defined with 16 bits.

*Table 96: Object 6401$_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Read Analog Input 16 Bit |
| Object code | ARRAY |
| PDO mapping | Yes |
| | |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ bis $8E_{hex}$ |
| Data type | Integer16 |
| Access | ro |
| Default value, BL×× | No |

### 8.2.3 Object 6421$_{hex}$ – Analog Input Interrupt Trigger Selection

**Note**

Objects 6421 – 6428$_{hex}$ can be used to control the event-triggered transmission of the process input data. As well as these event-triggered control objects, the transmission frequency of the process input data is also controlled by means of objects 1800$_{hex}$ to 181F$_{hex}$.

**Attention**

Object 6423$_{hex}$ Analog Input Global Interrupt Enable, page 8-14 must be used in order to enable the possibility of transmitting the process input data using an interrupt signal!

The object defines which event is to trigger the transmitting of the analog input data (TPDOs) by means of an interrupt signal.

The triggering event is defined for each input channel using a corresponding sub-index of the object.

*Table 97: Object 6421$_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Analog Input Interrupt Trigger Selection |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ bis $8E_{hex}$ |
| Data type | Unsigned8 |
| Access | rw |
| Default value | |
| – BL20 | Firmware:<br>≤ version 4.02/ 2.02 = 0<br>≥ version 4.02/ 2.02 = 7 |
| – BL67/BLCCO | 7 |

*Table 98: Structure of the Sub-Index 01$_{hex}$ to 8E$_{hex}$*

| Bit | Trigger event |
|---|---|
| 0 | 1 „upper limit“A exceeded **A** -<br>the value at the input has exceeded the upper limit. |
| 1 | 1 Input below „lower limit“ **A** -<br>the value at the input is below the lower limit. |
| 2 | 1 Input changed by more than „Delta“ **A** -<br>the value at the input has changed by a defined "Delta" value. |
| 3 | 1 Input reduced by more than „negative delta“**A** -<br>the value at the input has reduced by a defined "Delta" value. |
| 4 | 1 Input increased by more than „positive delta“ **A** -<br>the value at the input has increased by a defined "Delta" value. |
| 5 – 7 | reserved |

**A** *The upper/lower limit values and the delta values are defined using the objects 6424$_{hex}$, 6425$_{hex}$, 6426$_{hex}$, 6427$_{hex}$ and 6428$_{hex}$.*

**Note**

Several bits can be set simultaneously so that the transmitting of the input process data can be triggered by several events.

**Note**

The transmitting of the analog input data (TPDOs) by means of an interrupt signal is triggered repeatedly with every change of the analog input value if the value stays above the upper limit or below the lower limit.

If another triggering event occurs at the same time (e.g. increase by "Delta value"), the repeated transmitting is aborted.

### 8.2.4 Object $6422_{hex}$ – Analog Input Interrupt Source

The object indicates if an analog input channel has fulfilled a condition for triggering an interrupt signal.

The conditions were defined with object $6421_{hex}$. If a condition for triggering an interrupt signal on a channel is fulfilled, the corresponding bit is set to 1. The corresponding bits for channels 0 to 31 are set in sub-index $01_{hex}$ and the bits for channels 32 to 63 in sub-index $02_{hex}$ etc.

The bits can be read using a SDO. The reading causes a bit reset to "0".

*Table 99: Object $6422_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Analog Input Interrupt Source |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $08_{hex}$ |
| Data type | Unsigned32 |
| Access | ro |
| Default value, BL×× | $00_{hex}$ |

### 8.2.5 Object $6423_{hex}$ – Analog Input Global Interrupt Enable

With this object the possibility of generating an interrupt signal will be enabled. If the value is set as defined per default from FALSE to TRUE, than sending the analog input values (TPDOs) can be triggered using an interrupt signal.

*Table 100: Object $6423_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Analog Input Global Interrupt Enable |
| Object code | VAR |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Boolean |
| Access | rw |
| Default value, BL×× | FALSE |

### 8.2.6 Object $6424_{hex}$ – Analog Input Interrupt Upper Limit Integer

The object $6424_{hex}$ defines the value for an upper limit.

Values above this "upper limit" can be defined as the condition for generating an interrupt signal.

→ „$6421_{hex}$ Analog input interrupt trigger selection“, page 8-11.

*Table 101: Object $6424_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Analog Input Interrupt Upper Limit Integer |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ bis $8E_{hex}$ |
| Data type | Integer32 |
| Access | rw |
| Default value, BL×× | $00000000_{hex}$ |

### 8.2.7 Object $6425_{hex}$ – Analog Input Interrupt Lower Limit Integer

The object $6425_{hex}$ defines the value for an lower limit.

Values below this "lower limit" can be defined as the condition for generating an interrupt signal. → „$6421_{hex}$ Analog input interrupt trigger selection“, page 8-11.

*Table 102: Object $6425_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Analog Input Interrupt Lower Limit Integer |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ bis $8E_{hex}$ |
| Data type | Integer32 |
| Access | rw |
| Default value, BL×× | $00000000_{hex}$ |

### 8.2.8 Object $6426_{hex}$ – Analog Input Interrupt Delta Unsigned

The object $6426_{hex}$ defines a Delta value.

Values that deviate from the input value by this "Delta value" can be defined as the condition for generating an interrupt signal. → „$6421_{hex}$ Analog input interrupt trigger selection“, page 8-11.

*Table 103: Object $6426_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Analog Input Interrupt Delta Unsigned |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ bis $8E_{hex}$ |
| Data type | Unsigned32 |
| Access | rw |
| Default value, BL×× | $00000000_{hex}$ |

### 8.2.9 Object $6427_{hex}$ – Analog Input Interrupt Negative Delta Unsigned

The object $6427_{hex}$ defines a Delta value.

Values with a **negative** deviation of the input value by more than this "Delta value" can be defined as the condition for generating an interrupt signal.

→ „$6421_{hex}$ Analog input interrupt trigger selection“, page 8-11.

*Table 104: Object $6427_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Analog Input Interrupt Negative Delta Unsigned |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ bis $8E_{hex}$ |
| Data type | Unsigned32 |
| Access | rw |
| Default value, BL×× | $0000\ 0000_{hex}$ |

### 8.2.10 Object $6428_{hex}$ – Analog Input Interrupt Positive Delta Unsigned

The object $6428_{hex}$ defines a Delta value.

Values with a **positive** deviation of the input value by more than this "Delta value" can be defined as the condition for generating an interrupt signal.

→ „$6421_{hex}$ Analog input interrupt trigger selection“, page 8-11.

*Table 105: Object $6428_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Analog Input Interrupt Positive Delta Unsigned |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ bis $8E_{hex}$ |

*Table 105: Object 6428$_{hex}$*

| Feature | Description/ Value |
|---|---|
| Data type | Unsigned32 |
| Access | rw |
| Default value, BL×× | 00000000$_{hex}$ |

# 9 Objects for analog output modules


**9.1 Analog output modules BL×× ..... 2**

**9.2 General object overview for analog output modules ..... 2**

9.2.1 Object $5440_{hex}$ – Manu Spec Analog Output Range ..... 4
– Analog output modules, current ..... 4
– Analog output modules, voltage ..... 5
– analog output modules, current/ voltage ..... 5
9.2.2 Object $6411_{hex}$ – Write Analog Output 16 Bit ..... 6
9.2.3 Object $6443_{hex}$ - Analog Output Error Mode ..... 7
9.2.4 Object $6444_{hex}$ – Analog Output Error State ..... 8

## 9.1 Analog output modules BL××

The objects are used in the following modules:

*Table 106: BL×× - analog output modules*

| Product family | Module |
|---|---|
| BL20 | BL20-×AO-I(0/4...20MA) |
| | BL20-2AO-U(-10/0...+10VDC) |
| | BL20-E-4AO-U/I |
| | BL20-2AOH-I |
| BL67 | BL67-2AO-I |
| | BL67-2AO-V |
| | BL67-4AO-V |
| BLC | miscellaneous |

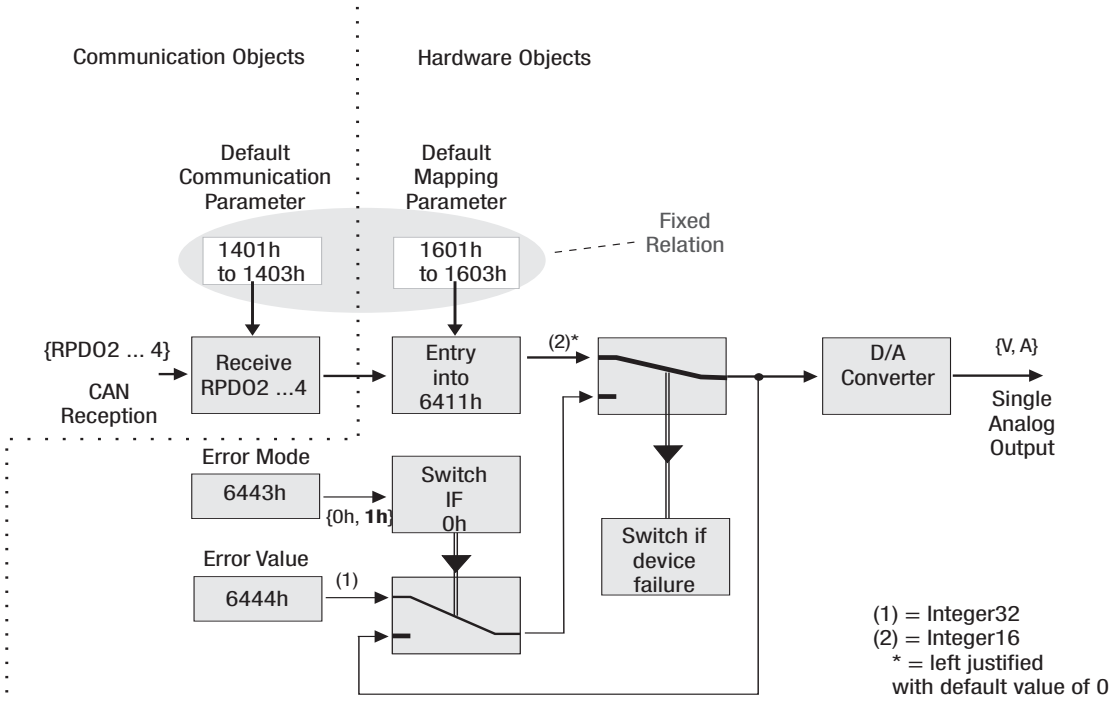
## 9.2 General object overview for analog output modules

*Table 107: General object overview for analog output modules*

| Object | Name | page |
|---|---|---|
| $5440_{i}$ | Manu Spec Analog Output Range | page 9-4 |
| $6411_{hex}$ | Write Analog Output 16 Bit | page 9-6 |
| $6443_{hex}$ | Analog Output Error Mode | page 9-6 |
| $6444_{hex}$ | Analog Output Error State | page 9-8 |

The following figure shows the relationship between the analog output objects for an 16-bit access:

*Figure8: Relationship between the objects for analog output modules (according to CiA Draft Standard 401)*



### 9.2.1 Object $5440_{hex}$ – Manu Spec Analog Output Range

The object defines the parameters of the analog output channels. Write accesses initiate a parameter update via the internal BL××-module bus.

The parameter is stored as a non-volatile parameter in the gateway and is restored with every node reset.

The Sub-indices $01_{hex}$ – $8E_{hex}$ define the parameters for the analog output channels 1 to 142.

*Table 108: Object $5440_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Manu Spec Analog Output Range |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ – $8E_{hex}$ |
| Data type | Unsigned16 |
| Access | rw |
| Default value, BL×× | No |

The structure of the 2 bytes of parameter data depends on the module concerned.

A sub-index is assigned for each channel. The following explains the structure for each module type.

**Analog output modules, current**

*Table 109: Parameter, analog Analog output modules, current*

**A***Default-setting*

| Byte | Bit | Parameters | Value/ meaning |
|---|---|---|---|
| n | 0 | Current mode | 0 = 0...20 mA **A**<br>1 = 4...20 mA |
| | 1 | Value representation | 0 = Integer (15 bit + sign) **A**<br>1 = reserved |
| | 3 | Channel x | 0 = activate **A**<br>1 = deactivate |
| n +1 and n + 2 | | substitute value A1 | The substitute value will be transmitted if the respective parameters of the gateway have been set to "output substitute value". |

### Analog output modules, voltage

*Table 110: Parameters, analog output modules, voltage*

**A***Default-setting*

| Byte | Bit | Parameters | Value/ meaning |
|---|---|---|---|
| n | 0 | Current mode | 0 = 0...10 V **A**<br>1 = -10...10 V |
| | 1 | Value representation | 0 = Integer<br>(15 bit + sign) **A**<br>1 = reserved |
| | 3 | Channel x | 0 = activate **A**<br>1 = deactivate |
| n +1<br>and<br>n + 2 | | substitute value A1 | The substitute value will be transmitted if the respective parameters of the gateway have been set to "output substitute value". |

### analog output modules, current/ voltage

*Table 111: Parameters, analog output modules, current/ voltage*

**A***Default-setting*

| Byte | Bit | Parameters | Value/ meaning |
|---|---|---|---|
| n | 0 - 3 | voltage | 0000 = -10...10 V DC Std **A**<br>0001 = 0...10 V DC Std<br>0010 = -10...10 V DC PA (NE 43)<br>0011 = 0...10 V DC PA (NE 43)<br>0100 = -10...10 V DC Ext. range<br>0101 = 0...10 V DC Ext. range |
| | | current | 1000 = 0 ... 20 mA Std **A**<br>1001 = 4 ... 20 mA Std<br>1010 = 0 ... 20 mA PA (NE 43)<br>1011 = 4 ... 20 mA PA (NE 43)<br>1100 = 0 ... 20 mA Ext. range<br>1101 = 4 ... 20 mA Ext. range |
| | | deactivate | 1111 |
| | 4 | Value representation | 0 = Integer (15 bit + sign) **A**<br>1 = 12 bit (left-justified) |
| | 5 | Diagnostics | 0 = release **A**<br>1 = block |
| | 6+7 | Behavior on module bus error Ax | 00 = output substitute value<br>01 = hold current value |
| n + 1 | | Substitute value A x / LOW Byte | |
| n + 2 | | Substitute value A x / HIGH Byte | |

### 9.2.2 Object $6411_{hex}$ – Write Analog Output 16 Bit

The object represents the values for the analog output modules with 16 bits for each channel.

*Table 112: Object $6411_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | Write Analog Output 16 Bit |
| Object code | ARRAY |
| PDO mapping | Yes |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $8E_{hex}$ |
| Data type | Integer16 |
| Access | rw |
| Default value, BL×× | $00_{hex}$ |

### 9.2.3 Object $6443_{hex}$ - Analog Output Error Mode

It defines (for each digital output channel) whether or not the output should take on a substitute value in the event of an error. The Sub-indices $01_{hex}$ – $8E_{hex}$ define the parameters for the analog output channels 1 to 142.

The following applies:

$00_{hex}$ The output maintains its value if an error occurs.

$_{hex}$The output is set to the substitute value if an error occurs.

The substitute values for the analog output channels are defined by the Analog Output Error State object ($6444_{hex}$).

*Table 113: Object $6443_{hex}$*

| **Feature** | **Description/ Value** |
|---|---|
| Name | Analog Output Error Mode |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $8E_{hex}$ |
| Data type | Unsigned16 |
| Access | rw |
| Default value, BL×× | $00_{hex}$ |

### 9.2.4 Object $6444_{hex}$ – Analog Output Error State

A substitute value is assigned to each analog channel. The substitute values will only be sent in case of an error, object Analog Output Error Mode ($6443_{hex}$) is set to "$01_{hex}$".

The Sub-indices $01_{hex}$ to $8E_{hex}$ define the parameters for the analog output channels 1 to 142.

*Table 114: Object $6444_{hex}$*

| **Feature** | **Description/ Value** |
|---|---|
| Name | Analog Output Error State |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ to $8E_{hex}$ |
| Data type | Unsigned16 |
| Access | rw |
| Default value, BL×× | 0000 $0000_{hex}$ |

# 10 Objects for RS232/RS4xx modules

**10.1 RS××× modules BL×× ..... 2**

**10.2 Allgemeine Objektübersicht für RS232/RS4××-Module ..... 2**

10.2.1 Object $5600_{hex}$ – RS232/RS4xx Parameters ..... 2
10.2.2 Object $5601_{hex}$ – RS232/RS4xx RxD ..... 4
10.2.3 Object $5602_{hex}$ – RS232/RS4xx TxD ..... 7

## 10.1 RS××× modules BL××

The objects are used in the following modules:

*Table 115: BL×× - RS××× modules*

| Product family | Module |
|---|---|
| BL20 | BL20-1RS232 |
| | BL20-1RS485/422 |
| BL67 | BL67-1RS232 |
| | BL67-1RS485/422 |
| BLC | miscellaneous |

## 10.2 Allgemeine Objektübersicht für RS232/RS4××-Module

*Table 116: General object overview for RS232/RS4×× modules*

| Object | Name | page |
|---|---|---|
| $5600_{hex}$ | RS232/RS4xx parameters | page 10-2 |
| $5601_{hex}$ | RS232/RS4xx RxD | page 10-4 |
| $5602_{hex}$ | RS232/RS4xx TxD | page 10-7 |

### 10.2.1 Object $5600_{hex}$ – RS232/RS4xx Parameters

The parameter setting of the BL20-1RS××× module enables communication with different data terminal devices. The handshake procedure (software/hardware) can be selected. The number of data bits embedded in the telegram, the type of parity, the number of stop bits, the bit transmission rate and the XON/XOFF character used must be configured in the module with the appropriate parameters in order to adapt it to the data format of the data terminal device.

4 bytes are used for the module parameters.

*Table 117: Object $5600_{hex}$*

| Feature | Description |
|---|---|
| Name | RS232/RS4xx parameters |
| Object code | ARRAY |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |
| PDO mapping | No |

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0** | Disable Diagnostics | Disable Reduced Control | X | Select RS485 | Bit Rate | | | |
| **Byte 1** | X | X | Flow Control | | Data | Parity | | Stop |
| **Byte 2** | XONChar | | | | | | | |
| **Byte 3** | XOFFChar | | | | | | | |

*Table 118: Parameters BL××-1RS×××*

**A***Default-setting*

| Parameters | Value/ meaning |
|---|---|
| DisableDiagnostics (Diagnostics) | 0 = Diagnostics activated |
| | 1 = Diagnostics deactivated<br>This controls the separate field bus-specific diagnostics signal - not the diagnostics signal embedded in the process input data (object $5601_{hex}$). |
| DisableReducedCtrl | The diagnostics messages are not part of the process input data (object $5601_{hex}$). Bytes1 to 7 can therefore be used for the user data. |
| | 1 = The diagnostics messages are contained in byte 1 of the object $5601_{hex}$ (irrespective of "DisableDiagnostics"). Byte 0 contains the status and control byte. Bytes 2 to 7 are available for the user data.<br>The structure of the object $5602_{hex}$ is also changed by this setting. At the location of the first data byte, byte 1 of the object $5602_{hex}$ contains a byte with two control bits. These can initiate a clearing of the receive and transmit buffer. |
| BitRate<br>Data rate | 0000 = reserved<br>0001 = 300 bps<br>0010 = 600 bps<br>0011 = 1200 bps<br>0100 = 2400 bps<br>0101 = 4800 bps<br>0110 = 9600 bps **A**<br>0111 = 14400 bps<br>1000 = 19200 bps<br>1001 = 28800 bps<br>1010 = 38400 bps<br>1011 = 57600 bps<br>1100 = 115200 bps<br>1101 = reserved<br>1110 = reserved<br>1111 = reserved |

| | |
|---|---|
| FlowControl | 00 = none **A**<br>01 = XON/XOFF<br>10 = RTS/CTS<br>11 = reserved |
| Data bits | 0 = 7 **A**<br>1 = 8 |
| Parity | 00 = none<br>01 = odd (The parity bit is set so that the number of bits set to 1 (data and parity bit together) is odd.) **A**<br>10 = even (The parity bit is set so that the number of bits set to 1 (data and parity bit together) is even.) |
| Stop bits | 0 = 1<br>1 = 2 **A** |
| XONChar | 0 to 255<br>XON character (17 **A**)<br>This character is used to start the data transfer of the data terminal device with software handshake activated. |
| XOFFChar | 0 to 255<br>XOFF character (19 **A**)<br>This character is used to stop the data transfer of the data terminal device with software handshake activated. |

### 10.2.2 Object $5601_{hex}$ – RS232/RS4xx RxD

Process input data is data that is transmitted from the connected field device via the BL20-1RS××× module to the communication partner (e.g. PLC). The data received from the device by the BL20-1RS××× module is entered in a 128 byte receive buffer and then transferred in segments to the communication partner via the module bus and the gateway.

The transmission is realized in a 8-byte format which is structured as follows:

- 6 bytes or 7 bytes are used to contain the user data.
- 1 byte contains the diagnostics data depending on the parameter setting.
- 1 status byte is required to ensure trouble-free transmission of the data.

*Table 119: Object $5601_{hex}$*

| **Feature** | **Description** |
|---|---|
| Name | RS232/RS4xx RxD |
| Object code | ARRAY |
| Data type | Unsigned64 |
| Access | ro |
| Default value | No |
| PDO mapping | Yes |

Structure of the data bytes with DisableReducedControl = 1 (in object $5600_{hex}$):

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0 (Status)** | STAT | TX_CNT_ACK | | RX_CNT | | RX_BYTE_CNT | | |
| **Byte 1 (Diag.)** | BufOfl | Frame Err | HndShErr | Hw_ Failure | PrmErr | X | X | X |
| **Byte 2** | data byte 0 | | | | | | | |
| **...** | ... | | | | | | | |
| **Byte 7** | data byte 5 | | | | | | | |

Structure of the data bytes with DisableReducedControl = 0 (in object $5600_{hex}$):

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0 (Status)** | STAT | TX_CNT_ACK | | RX_CNT | | RX_BYTE_CNT | | |
| **Byte 1** | data byte 0 | | | | | | | |
| **...** | ... | | | | | | | |
| **Byte 7** | data byte 6 | | | | | | | |

*Table 120: Meaning of the data bits*

| Designation | Value | Description |
|---|---|---|
| STAT | 0-1 | 1 The communication with the data terminal equipment (DTE) is error free<br>0 The communication with the data terminal equipment (DTE) is disturbed. A diagnosis message is generated if the parameter „Diagnostics“ is set to "0" = release. The diagnostic data show the cause of the communication disturbance. The user has to set back this bit in the process output data by using STATRES (Object $5602_{hex}$). |
| TX_CNT_ACK | 0-3 | The value TX_CNT_ACK is a copy of the value TX_CNT.<br>The value TX_CNT was transferred together with the last data segment of the process output data. The value TX_CNT_ACK is a confirmation of successful acceptance of the data segment using TX_CNT. |
| RX_CNT | 0-3 | This value is transferred together with every data segment. The RX_CNT values are sequential:<br>00->01->10->11->00... (decimal: 0->1->2->3->0...) Errors in this sequence show the loss of data segments. |
| RX_BYTE_CNT | 0-7 | Number of the valid bytes in this data segment. |

*Table 120: Meaning of the data bits*

| Designation | Value | Description |
|---|---|---|
| BufOvfl | Bit 7 | 0 = ok |
| | | 1 = buffer overflow<br>Overflow of the receive-buffer (RX-buffer). |
| FrameErr | Bit 6 | 0 = ok |
| | | 1 = frame error<br>The module has to be parameterized for adaptation to the data structure of the data terminal equipment (DTE).<br>A frame error occurs in case of inconsequent parameterization (number of data bits, stop bits, method of parity,...). |
| HndShErr | Bit 5 | 0 = ok |
| | | 11 = Data flow control error<br>The DTE connected t the module does not react to XOFF or RTS handshake.<br>The internal receive-buffer may overflow (buffer-overflow = 1). |
| HwFailure | Bit 4 | 0 = ok |
| | | 1 = Hardware failure<br>The module has to be replaced (e.g. error in EEPROM or UART). |
| PrmErr | Bit 3 | 0 = ok |
| | | 1 = Parameterization error<br>The parameter settings can not be supported. |

### 10.2.3 Object $5602_{hex}$ – RS232/RS4xx TxD

Process output data are data which are sent from the PLC via the gateway and the RS×××-module to a connected field device.

The data received from the PLC are loaded into the 64-bit transmit-buffer in the RS×××-module.

The transmission is realized in a 8-byte format which is structured as follows:

- 6 bytes or 7 bytes are used to contain the user data.
- If parameterized, 1 byte contains, signals to start the flushing of transmit- and receive buffer.
- 1 control byte is required to ensure trouble-free transmission of the data.

*Table 121: Object $5602_{hex}$*

| Feature | Description |
|---|---|
| Name | RS232/RS4xx TxD |
| Object code | ARRAY |
| Data type | Unsigned64 |
| Access | rw |
| Default value | No |
| PDO mapping | Yes |

Structure of the data bytes with DisableReducedControl = 1 (in object $5600_{hex}$):

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0 (Status)** | STAT RES | RX_CNT_ACK | | TX_CNT | | TX_BYTE_CNT | | |
| **Byte 1 (Diag.)** | reserved | | | | | | RXBUF FLUSH | TXBUF FLUSH |
| **Byte 2** | data byte 0 | | | | | | | |
| **...** | ... | | | | | | | |
| **Byte 7** | data byte 5 | | | | | | | |

Structure of the data bytes with DisableReducedControl = 0 (in object $5600_{hex}$):

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0 (Status)** | STAT RES | RX_CNT_ACK | | TX_CNT | | TX_BYTE_CNT | | |
| **Byte 1** | data byte 0 | | | | | | | |
| **...** | ... | | | | | | | |
| **Byte 7** | data byte 6 | | | | | | | |

*Table 122: Meaning of the data bits*

| Designation | Value | Description |
|---|---|---|
| STATRES | 0-1 | The STATRES bit is used for resetting the STAT bit of the process input data (object $5601_{hex}$). The STAT bit is reset (from 0 to 1) with the transition from 1 to 0 (falling edge). If this bit is 0, all changes in the data fields TX_BYTE_CNT, TX_CNT and RX_CNT_ACK are ignored.<br>The clearing of the receive and transmit buffer by RXBUF FLUSH/TXBUF FLUSH is possible. The value 1 or the transition from 0 to 1 disables the clearing of the receive and transmit buffer by the RXBUF FLUSH/TXBUF FLUSH. |
| RX_CNT_ACK | 0-3 | The value RX_CNT_ACK is a copy of the value RX_CNT. The value RX_CNT was transferred together with the last data segment of the process input data (Object $5601_{hex}$). The value RX_CNT_ACK is a confirmation of successful acceptance of the data segment using RX_CNT. |
| TX_CNT | 0-3 | The value TX_CNT is transferred together with every data segment of the process output data.<br>The TX_CNT values are sequential: 00->01->10->11->00... (decimal: 0->1->2->3->0...) Errors in this sequence show the loss of data segments. |
| TX_BYTE_CNT | 0-7 | Number of the valid bytes in this data segment. |
| RXBUF FLUSH | 0-1 | The RXBUF FLUSH bit is used for clearing the receive buffer.<br>If STATRES = 1: A request with RXBUF FLUSH = 1 will be ignored.<br>If STATRES = 0: RXBUF FLUSH = 1 will clear the receive buffer. |
| TXBUF FLUSH | 0-1 | The TXBUF FLUSH bit is used for clearing the transmit buffer.<br>If STATRES = 1: A request with TXBUF FLUSH = 1 will be ignored.<br>If STATRES = 0: TXBUF FLUSH = 1 will clear the receive buffer. |

# 11 Objects for Encoder modules (SSI, CNT)


**11.1 Encoder module BL×× ..... 4**

**11.2 General object overview for encoder modules ..... 4**

11.2.1 Object $5800_{hex}$ – Encoder Basic Mode ..... 6
– BL20-1CNT-24VDC ..... 7
– BL67-1CNT/ENC ..... 7
11.2.2 Object $5801_{hex}$ – Encoder Config ..... 7
– BL××-1SSI ..... 8
– BL20-1CNT-24VDC ..... 9
– BL20-E-2CNT-2PWM ..... 10
– BL67-1CNT/ENC ..... 11
11.2.3 Object $5802_{hex}$ – Encoder Status ..... 11
– BL××-1SSI ..... 11
– BL20-1CNT-24VDC ..... 12
– BL20-E-2CNT-2PWM ..... 12
– BL67-1CNT/ENC ..... 12
11.2.4 Object $5803_{hex}$ – Encoder Flags ..... 13
– BL××-1SSI ..... 13
– BL20-1CNT-24VDC ..... 13
– BL20-E-2CNT-2PWM ..... 14
– BL67-1CNT/ENC ..... 14
11.2.5 Objekt $5804_{hex}$ – Encoder Diag ..... 14
– BL××-1SSI ..... 15
– BL20-1CNT-24VDC ..... 15
– BL20-E-2CNT-2PWM ..... 15
– BL67-1CNT/ENC ..... 16
11.2.6 Object $5805_{hex}$ – Encoder Native Status ..... 16
– BL××-1SSI ..... 16
– BL20-E-2CNT-2PWM ..... 17
– BL67-1CNT/ENC ..... 17
11.2.7 Object $5806_{hex}$ – Optional Encoder Status ..... 18
– BL××-1SSI ..... 18
– BL20-E-2CNT-2PWM ..... 18
– BL67-1CNT/ENC ..... 19
11.2.8 Object $5808_{hex}$ – Encoder Control ..... 19
– BL××-1SSI ..... 19
– BL20-1CNT-24VDC ..... 20
– BL20-E-2CNT-2PWM ..... 20
– BL67-1CNT/ENC ..... 20
11.2.9 Object $5810_{hex}$ – Encoder Load Prepare Value ..... 21
– BL20-1CNT-24VDC ..... 21
– BL20-E-2CNT-2PWM ..... 21
– BL67-1CNT/ENC ..... 21
11.2.10 Object $5811_{hex}$ – Encoder Pulse Width ..... 22
– BL20-1CNT-24VDC ..... 22
– BL20-E-2CNT-2PWM ..... 22
11.2.11 Object $5820_{hex}$ – Measuring Integration Time ..... 23
– BL20-1CNT-24VDC ..... 23
– BL20-E-2CNT-2PWM ..... 23
11.2.12 Object $5821_{hex}$ – Measuring Low Limit ..... 24
– BL20-1CNT-24VDC ..... 24
11.2.13 Object $5822_{hex}$ – Measuring High Limit ..... 24
– BL20-1CNT-24VDC ..... 24

11.2.14 Object $5823_{hex}$ – Measuring Units Per Revolution ..... 25
– BL20-1CNT-24VDC ..... 25
11.2.15 Object $5824_{hex}$ – Measuring Divisor ..... 26
– BL20-E-2CNT-2PWM ..... 26
– BL67-1CNT/ENC ..... 26
11.2.16 Object $5825_{hex}$ – Measuring Factor ..... 26
– BL20-E-2CNT-2PWM ..... 27
– BL67-1CNT/ENC ..... 27
11.2.17 Object $5827_{hex}$ – Measuring Timeout ..... 27
– BL20-E-2CNT-2PWM ..... 27
– BL67-1CNT/ENC ..... 27
11.2.18 Object $5830_{hex}$ – Measuring Value ..... 28
– BL20-E-2CNT-2PWM ..... 28
– BL67-1CNT/ENC ..... 28
11.2.19 Object $5831_{hex}$ – Encoder Latch Value ..... 28
– BL20-E-2CNT-2PWM ..... 28
11.2.20 Object $5840_{hex}$ – Diag Mapping ..... 29
– BL××-1SSI ..... 29
11.2.21 Object $5901_{hex}$ – PWM Config ..... 30
– BL20-E-2CNT-2PWM ..... 31
11.2.22 Object $5902_{hex}$ – PWM Status ..... 33
– BL20-E-2CNT-2PWM ..... 33
11.2.23 Object $5903_{hex}$ – PWM Flags ..... 34
11.2.24 Object $5904_{hex}$ – PWM Diag ..... 35
11.2.25 Object $5908_{hex}$ – PWM Control ..... 36
11.2.26 Object $5910_{hex}$ – PWM Load Prepare Value ..... 38
11.2.27 Object $5913_{hex}$ – PWM Duty Cycle ..... 38
11.2.28 Object $5920_{hex}$ – PWM Period Duration ..... 39
11.2.29 Object $5931_{hex}$ – PWM Latch Value ..... 39
11.2.30 Object $6800_{hex}$ – Operating Parameters ..... 40
11.2.31 Objekt $6810_{hex}$ – Preset Values for Multi-Sensor Devices ..... 40
– BL××-1SSI ..... 40
– BL20-1CNT-24VDC ..... 40
– BL20-E-2CNT-2PWM ..... 41
– BL67-1CNT/ENC ..... 41
11.2.32 Object $6820_{hex}$ – Position Value ..... 41
– BL××-1SSI ..... 41
– BL20-1CNT-24VDC ..... 41
– BL20-E-2CNT-2PWM ..... 41
– BL67-1CNT/ENC ..... 41
11.2.33 Object $6B00_{hex}$ – CAM State Register ..... 42
– BL××-1SSI ..... 42
– BL20-1CNT-24VDC ..... 42
– BL20-E-2CNT-2PWM ..... 43
11.2.34 Object $6B01_{hex}$ – CAM1 Enable Register ..... 44
11.2.35 Object $6B02_{hex}$ – CAM Polarity Register ..... 45
11.2.36 Object $6B10_{hex}$ – CAM1 Low Limit ..... 46
– BL××-1SSI ..... 46
– BL20-1CNT-24VDC ..... 46
– BL20-E-2CNT-2PWM ..... 46
11.2.37 Object $6B20_{hex}$ – CAM1 High Limit ..... 47
– BL××-1SSI ..... 47
– BL20-1CNT-24VDC ..... 47
– BL20-E-2CNT-2PWM ..... 47
11.2.38 Object $6B30_{hex}$ – CAM1 Hysteresis ..... 48

– BL20-1CNT-24VDC ..... 48
– BL20-E-2CNT-2PWM ..... 48
11.2.39 Object $6C00_{hex}$ – Area State Register ..... 49
– BL20-1CNT-24VDC ..... 49
11.2.40 Object $6C01_{hex}$ – Work Area Low Limit ..... 50
– BL20-1CNT-24VDC ..... 51
– BL20-E-2CNT-2PWM ..... 51
11.2.41 Object $6C02_{hex}$ – Work Area High Limit ..... 51
– BL20-1CNT-24VDC ..... 51
– BL20-E-2CNT-2PWM ..... 51
11.2.42 Object $6D00_{hex}$ – Operating Status, Object $6D01_{hex}$ – SingleTurn Resolution (rotary), Measuring step (linear), Object $6D02_{hex}$ – Number of Distinguishable Revolutions ..... 52
11.2.43 Object $6FFF_{hex}$ – Device Type ..... 52

**11.3 Process output/ control interface of the Encoder modules ..... 53**

11.3.1 Meaning of the process output bits of BL××-1SSI ..... **53**
11.3.2 Meaning of the process output bits of BL20-1CNT-24VDC ..... 54
11.3.3 Meaning of the process output bits of BL20-E-2CNT-2PWM ..... 56
11.3.4 Meaning of the process output bits of BL67-1CNT/ENC ..... 58

**11.4 Meaning of the process input bits of the Encoder modules ..... 59**

11.4.1 Meaning of the process input bits of BL××-1SSI ..... **59**
11.4.2 Meaning of the process input bits of BL20-1CNT-24VDC ..... 61
11.4.3 Meaning of the process input bits of BL20-E-2CNT-2PWM ..... 63
11.4.4 Meaning of the process input bits of BL67-1CNT/ENC ..... 66

**11.5 Parameter interface of the Encoder modules ..... 68**

11.5.1 Meaning of the parameter bits of BL××-1SSI ..... **68**
11.5.2 Meaning of the parameter bits of BL20-1CNT-24VDC ..... 69
11.5.3 Meaning of the parameter bits of BL20-E-2CNT-2PWM ..... 72
11.5.4 Meaning of the parameter bits of BL67-1CNT/ENC ..... 74

**11.6 Diagnostic interface of the Encoder modules ..... 77**

11.6.1 Meaning of the diagnostic bits of BL××-1SSI ..... **77**
11.6.2 Meaning of the diagnostic bits of BL20-1CNT-24VDC ..... 77
11.6.3 Meaning of the diagnostic bits of BL20-E-2CNT-2PWM ..... 79
11.6.4 Meaning of the diagnostic bits of BL67-1CNT/ENC ..... 79

## 11.1 Encoder module BL××

The objects are used in the following modules:

*Table 123: BL×× -Encoder modules*

| Product family | Module |
|---|---|
| BL20 | BL20-1CNT-24VDC |
| | BL20-1SSI |
| | BL20-E-2CNT-2PWM |
| BL67 | BL67-1SSI |
| | BL67-1CNT/ENC |
| BLC | miscellaneous |

## 11.2 General object overview for encoder modules

*Table 124: General object overview for encoder modules*

| Object | Name | Page |
|---|---|---|
| $5800_{hex}$ | Encoder Basic Mode | Seite 11-6 |
| $5801_{hex}$ | Encoder Config | Seite 11-7 |
| $5802_{hex}$ | Encoder Status | Seite 11-11 |
| $5803_{hex}$ | Encoder Flags | Seite 11-13 |
| $5804_{hex}$ | Encoder Diag | Seite 11-14 |
| $5805_{hex}$ | SSI Native Status | Seite 11-16 |
| $5806_{hex}$ | SSI Optional Encoder status | Seite 11-18 |
| $5808_{hex}$ | Encoder Control | Seite 11-19 |
| $5810_{hex}$ | Encoder Load Prepare | page 11-19 |
| $5811_{hex}$ | Encoder Pulse Width | page 11-22 |
| $5820_{hex}$ | Measuring Integration Time | page 11-23 |
| $5821_{hex}$ | Measuring Low Limit | page 11-24 |
| $5822_{hex}$ | Measuring High Limit | page 11-24 |
| $5823_{hex}$ | Measuring Units Per Revolution | page 11-25 |
| $5824_{hex}$ | Measuring Divisor | page 11-26 |
| $5825_{hex}$ | Measuring Factor | page 11-26 |
| $5827_{hex}$ | Measuring Timeout | page 11-27 |
| $5830_{hex}$ | Measuring Value | page 11-28 |

| | | |
|---|---|---|
| $5831_{hex}$ | Encoder Latch Value | page 11-28 |
| $5840_{hex}$ | Diag Mapping | page 11-29 |
| $5901_{hex}$ | PWM Config | page 11-30 |
| $5902_{hex}$ | PWM Status | page 11-33 |
| $5903_{hex}$ | PWM Flags | page 11-34 |
| $5904_{hex}$ | PWM Diag | page 11-35 |
| $5908_{hex}$ | PWM Control | page 11-36 |
| $5910_{hex}$ | PWM Load Prepare | page 11-38 |
| $5913_{hex}$ | PWM Duty Cycle | page 11-38 |
| $5920_{hex}$ | PWM Period Duration | page 11-39 |
| $5931_{hex}$ | PWM Latch Value | page 11-39 |
| $6800_{hex}$ | Operating Parameters | Seite 11-40 |
| $6810_{hex}$ | Preset Value for Multi-Sensor Devices | Seite 11-40 |
| $6820_{hex}$ | Position Value | Seite 11-41 |
| $6B00_{hex}$ | CAN State Register | Seite 11-42 |
| $6B00_{hex}$ | CAM Enable Register | Seite 11-44 |
| $6B00_{hex}$ | CAM Polarity Register | Seite 11-45 |
| $6B10_{hex}$ | CAM Low Limit | Seite 11-46 |
| $6B20_{hex}$ | CAM High Limit | Seite 11-47 |
| $6B30_{hex}$ | CAM1 Hysteresis | Seite 11-48 |
| $6C00_{hex}$ | Area State Register | Seite 11-49 |
| $6C01_{hex}$ | Work Area Low Limit | Seite 11-50 |
| $6C02_{hex}$ | Work Area High Limit | Seite 11-51 |
| $6D00_{hex}$ | Operating Status | Seite 11-52 |
| $6D01_{hex}$ | SingleTurn Resolution (rotary), Measuring step (linear) | Seite 11-52 |
| $6D02_{hex}$ | Number of Distinguishable Revolutions | Seite 11-52 |
| $6FFF_{hex}$ | Device Type | Seite 11-52 |

### 11.2.1 Object $5800_{hex}$ – Encoder Basic Mode

The object $5800_{hex}$ affects parts of the module parameters:

Write accesses initiate a parameter update via the internal module bus. The parameter is stored as a non-volatile parameter in the gateway and is restored with every node reset.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC
- BL67-1CNT/ENC

*Table 125: Object $5800_{hex}$*

| Feature | Description |
|---|---|
| Name | Encoder Basic Mode |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value, BL×× | No |
| Sub-index | $01_{hex}$ - $47_{hex}$ |
| Data type | Unsigned32 |
| Access | rw |
| Default value, BL×× | No |

**BL20-1CNT-24VDC**

The object $5800_{hex}$ affects the operation mode parameters of the BL20 counter module. It serves amongst others for setting the count operation or the measurement operation mode.

Structure of the data bytes for the count operation:

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0** | reserved | | Counter mode | | | | | |
| **Byte 1** | reserved | Main count direction | | Synchroni-zation | Function DI | | Digital input DI | Gate func-tion |
| **Byte 2** | reserved | | | | | | | |
| **Byte 3** | reserved | | | | | | | |

X = reserved

**Note**

Please find a detailed description of the parameters in the table Meaning of the parameter bits of BL20-1CNT-24VDC (page 11-69).

Structure of the data bits in the measurement mode:

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0** | reserved | | Measurement mode | | | | | |
| **Byte 1** | reserved | | | | Function DI | | Digital input DI | reserved |
| **Byte 2** | reserved | | | | | | | |
| **Byte 3** | reserved | | | | | | | |

**Note**

Please find a detailed description of the parameters in the table Meaning of the parameter bits of BL20-1CNT-24VDC (page 11-69).

**BL67-1CNT/ENC**

For this module, this object describes **parameter byte 14**.

Byte 14 of the modules' parameter data serves for defining the number of the register from the register interface (REG_PARA; REG_COUNTER_VALUE, REG_LOWER_LIMIT etc. see D300529) whose content has to be mapped into the process input data of the module.

The default value for the module BL67-1CNT/ENC is 32 (register-no. 32 „REG_COUNTER_VALUE“).

### 11.2.2 Object $5801_{hex}$ – Encoder Config

The object takes effect on parts of the respective module parameters and serves for configuration setting. Write accesses initiate a parameter update via the internal module bus. The parameter is stored as a non-volatile parameter in the gateway and is restored with every node reset.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BLxx-1SSI
- BL20-1CNT-24VDC

- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 126: Object 5801$_{hex}$*

| Feature | Description |
|---|---|
| Name | Encoder Config |
| Object code | ARRAY |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |
| PDO mapping | No |

**BLxx-1SSI**

The Object Encoder Config takes effect on parameter bytes 0 ... 3 of the BL20-SSI module.

Structure of the data bytes 0 to 3 for the SSI-module:

| | **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
|---|---|---|---|---|---|---|---|---|
| **Byte 0 (Status)** | reserved | | DIS_ERR_ SSI | reserved | | | | |
| **Byte 1** | | INVALID_BITS_MSB | | | INVALID_BITS_LSB | | | |
| **Byte 2** | reserved | | | | SSI_BIT_RATE | | | |
| **Byte 3** | SSI_CODE_ G/D | reserved | SSI_FRAME_LENGTH | | | | | |

**Note**

Please find a detailed description of the parameters in the table Meaning of the parameter bits of BLxx-1SSI (page 11-68).

### BL20-1CNT-24VDC

The Object Encoder Config takes effect on parameter bytes 12 ... 15 of the counter module.

**Count operation:**

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0** | Function Do2 | | | Function DO1 | | | diagnostic DO1 | substitute value DO1 |
| **Byte 1** | direction input (B) | sensor (A) | | sensor/ input filter (DI) | sensor/ input filter (BI) | sensor/ input filter (A) | signal evaluation (A B) | |
| **Byte 2** | reserved | | behavior CPU/ master stop | | reserved | | | Common diagnostics |
| **Byte 3** | reserved | | | | | | | |

**Note**

Please find a detailed description of the parameters in the table Meaning of the parameter bits of BL20-1CNT-24VDC (page 11-69).

**Structure of the data bytes for the measurement operation:**

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0** | | | | Function DO1 | | | diagnostic DO1 | substitute value DO1 |
| **Byte 1** | direction input (B) | sensor (A) | | sensor/ input filter (DI) | sensor/ input filter (BI) | sensor/ input filter (A) | signal evaluation (A B) | |
| **Byte 2** | reserved | | behavior CPU/ master stop | | reserved | | | Common diagnostics |
| **Byte 3** | reserved | | | | | | | |

**Note**

Please find a detailed description of the parameters in the table Meaning of the parameter bits of BL20-1CNT-24VDC (page 11-69).

### BL20-E-2CNT-2PWM

Object Encoder Config takes effect on parameter bytes 0...2 and 10 (channel 1), or respectively 3...5 and 12 (channel 2) of the BL20-E-2CNT-2PWM.

| Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0 | input A1 | input B1 | input Z1 | res. | diag-nostic CNT1 | measure-ment mode CNT1 | main count direction CNT1 | |
| 1 | filter Z1 | | filter A1, B1 | | res. | pull up Z1 | res. | threshold input A,B,Z CNT1 |
| 2 | mode Z1 | | | | mode CNT1 | | | |
| 3 | input A2 | input B2 | input Z2 | res. | diagnostic | measure-ment mode CNT2 | main count direction CNT2 | |
| 4 | filter Z2 | | filter A2, B2 | | res. | pull up Z2 | res. | threshold input A,B,Z CNT2 |
| 5 | mode Z2 | | | | mode CNT2 | | | |
| ... | Object 5901 | | | | | | | |
| 10 | res. | ADR AUX REG1 RD DATA | | | | | | |
| ... | Object 5901 | | | | | | | |
| 12 | res. | ADR AUX REG3 RD DATA | | | | | | |
| ... | Object 5901 | | | | | | | |

**Note**

Please find a detailed description of the parameters in the table Meaning of the parameter bits of BL20-E-2CNT-2PWM (page 11-72).

**BL67-1CNT/ENC**

The Object Encoder Config has effect on parameter bytes 0 ... 0.3 of the BL67-1CNT/ENC module.

| Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0 | input A | input B | reserved | count direction | Signal evaluation | | input filter (A B) | |
| 1 | input Z | reserved | Sync. with Z | reserved | Encoder signal | Function DO3 | Function DI3 | PullUp Z |
| 2 | threshold input A, B and Z | | | | | | | |
| 3 | reserved | | Measure-ment mode | Counter mode | Gate | Gate function | | |
| ... | | | | | | | | |

input = input

Sync. = synchronization

**Note**

Please find a detailed description of the parameters in table Meaning of the parameter bits of BL67-1CNT/ENC (page 11-74).

### 11.2.3 Object $5802_{hex}$ – Encoder Status

The object Encoder Status displays the status of the modules' count direction.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BLxx-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 127: Object $5802_{hex}$*

| Feature | Description |
|---|---|
| Name | Encoder Status |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| PDO mapping | Yes |

**BLxx-1SSI**

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| STS_DNS | STS_UP | reserved | | | | | |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL××-1SSI (page 11-59).

**BL20-1CNT-24VDC**

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| STS_DN | STS_UP | reserved | STS_DO2 | STS_DO1 | reserved | STS_DI | STS_ GATE |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL20-1CNT-24VDC (page 11-61).

**BL20-E-2CNT-2PWM**

The object $5802_{hex}$ supplies the following status messages:

- count direction
- status of in- and outputs
- operation status of the counter

**Structure of the data bytes**

| Sub-index 0×00 ≤ **n** ≥ 0×47 | Bit 7 (msb) | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 (lsb) |
|---|---|---|---|---|---|---|---|---|
| **n** = CNT1 | STS_CNT1_ DN | STS_CNT1_ UP | STS_CNT1 _SFKT_EN | STS_DBP1 | D1 | STS_CNT1 _GENERAL _EN | Z1 | STS_CNT1 _RUN |
| **n + 1** = CNT2 | STS_CNT2_ DN | STS_CNT2_ UP | STS_CNT2 _SFKT_EN | STS_DBP2 | D2 | STS_CNT2 _GENERAL _EN | Z2 | STS_CNT2 _RUN |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL20-E-2CNT-2PWM (page 11-63).

**BL67-1CNT/ENC**

Bits 6 and 7 of the object Encoder Status describe the up/down direction of the current values.

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| STS_DNS | STS_UP | reserved | | | | | |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL67-1CNT/ENC (page 11-66).

### 11.2.4 Object $5803_{hex}$ – Encoder Flags

The object contains volatile status messages as well as module flags.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL××-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 128: Object $5803_{hex}$*

| Feature | Description |
|---|---|
| Name | Encoder Flags |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | rw |
| Default value | No |
| PDO mapping | Yes |

**BL××-1SSI**

The bits STS_OFLW and STS_UFLW are volatile status bits for the SSI module.

Writing the object with any value will reset the markers FLAG_CMP1 and FLAG_CMP2.

**Exception:**

if the relevant condition for setting a marker is still fulfilled, this marker will continue to remain set.

**Structure of the data byte**

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| reserved | STS_ UFLW | STS_ OFLW | FLAG_ CMP2 | FLAG_ CMP1 | reserved | | |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL××-1SSI (page 11-59).

**BL20-1CNT-24VDC**

The object reads byte 6 of the check-back interface of the counter module.

This byte sends the stored status of some module flags.

The flags are reset by writing this object with any value.

**Exception:**

if the relevant condition for setting a marker is still fulfilled, this marker will continue to remain set.

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| STS_ND | STS_ UFLW | STS_ OFLW | STS_ CMP2 | STS_ CMP1 | reserved | | STS_SYN |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL20-1CNT-24VDC (page 11-61).

**BL20-E-2CNT-2PWM**

| Sub-index<br>0×00 ≤ **n** ≥ 0×47 | Bit 7<br>(msb) | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0<br>(lsb) |
|---|---|---|---|---|---|---|---|---|
| **n** = CNT1 | MSG_ CNT1_ND | MSG_ CNT1_ UFLW | MSG_ CNT1_ OFLW | MSG_ CNT1_ CMP1 | MSG_ CNT1_ CMP0 | MSG_ CNT1_SW_ LR | MSG_ CNT1_ SFKT | MSG_ CNT1_ FQE |
| **n + 1** = CNT2 | MSG_ CNT2_ND | MSG_ CNT2_ UFLW | MSG_ CNT2_ OFLW | MSG_ CNT2_ CMP1 | MSG_ CNT2_ CMP0 | MSG_ CNT2_SW_ LR | MSG_ CNT2_ SFKT | MSG_ CNT2_ FQE |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL20-E-2CNT-2PWM (page 11-63).

**BL67-1CNT/ENC**

The flags are reset by writing this object with any value.

**Exception:**

if the relevant condition for setting a marker is still fulfilled, this marker will continue to remain set.

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| STS_ZC | STS_ UFLW | STS_ OFLW | reserved | | | | |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL67-1CNT/ENC (page 11-66).

### 11.2.5 Objekt $5804_{hex}$ – Encoder Diag

This object reads the diagnostic byte of the encoder module.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL××-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

- BL67-1CNT/ENC

*Table 129: Object 5804$_{hex}$*

| Feature | Description |
|---|---|
| Name | Encoder Diag |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| PDO mapping | No |

**BLxx-1SSI**

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| reserved | | | ERR_ PARA | STS_ UFLW | STS_ OFLW | ERR_ SSI | SSI_ DIAG |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the diagnostic bits of BLxx-1SSI (page 11-77).

**BL20-1CNT-24VDC**

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| MEAS._ MODE | OPER._ MODE | ERR_PARA | | | | ERR_ 24Vdc | ERR_DO |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the diagnostic bits of BL20-1CNT-24VDC (page 11-77).

**BL20-E-2CNT-2PWM**

| Sub-index 0×00 ≤ **n** ≥ 0×47 | Bit 7 (msb) | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 (lsb) |
|---|---|---|---|---|---|---|---|---|
| **n** = CNT1 | HW_ERR | CNT1_ PAR_ERR | CNT1_ERR _UFLW | CNT1_ERR _OFLW | reserved | | | |
| **n + 1** = CNT2 | HW_ERR | CNT2_ PAR_ERR | CNT2_ERR _UFLW | CNT2_ERR _OFLW | reserved | | | |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the diagnostic bits of BL20-E-2CNT-2PWM (page 11-79).

**BL67-1CNT/ENC**

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| ERR_ PARA | reserved | DIA_DO3 | DIA_DO2 | DIA_DO1 | DIA_DO0 | STS_ OFLW | STS_ UFLW |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the diagnostic bits of BL67-1CNT/ENC (page 11-79).

### 11.2.6 Object $5805_{hex}$ – Encoder Native Status

The object Encoder Native Status reads bytes 0 to 1 the the module's process input data.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL××-1SSI
- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 130: Object $5805_{hex}$*

| Feature | Description |
|---|---|
| Name | Encoder Native Status |
| Object code | ARRAY |
| Data type | Unsigned16 |
| Access | rw |
| Default value | No |
| PDO mapping | Yes |

**BL××-1SSI**

Writing the object with any value will reset the non-volatile flags FLAG_CMP1 and FLAG_CMP2.

Exception:

if the relevant condition for setting a marker is still fulfilled, this marker will continue to remain set.

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0** | STS_ STOP | reserved | | ERR_ PARA | STS_ UFLW | STS_ OFLW | ERR_ SSI | SSI_ DIAG |
| **Byte 1** | STS_ UP | STS_ DN | REL_ CMP2 | FLAG_ CMP2 | STS_ CMP2 | REL_ CMP1 | FLAG_ CMP1 | STS_ CMP1 |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL××-1SSI (page 11-59).

### BL20-E-2CNT-2PWM

The object reads the process input bytes 0 to 1 (channel 1) and 2 to 3 (channel 2) of the module.

| Sub-index<br>0×00 ≤ **n** ≥ 0×47 | Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| **n** = CNT1 | 0 | A1 | B1 | Z1 | STS_CNT1_DIR | STS_CNT1_LOGMSG | STS_CNT1_SFKT_EN | STS_CNT1_RUN | STS_CNT1_GENERAL_EN |
| | 1 | MSG_CNT1_SW_LR | MSG_CNT1_SFKT | MSG_CNT1_FQE | MSG_CNT1_ND | MSG_CNT1_OFLW | MSG_CNT1_UFLW | MSG_CNT1_CMP1 | MSG_CNT1_CMP0 |
| **n + 1** = CNT2 | 2 | A2 | B2 | Z2 | STS_CNT2_DIR | STS_CNT2_LOGMSG | STS_CNT2_SFKT_EN | STS_CNT2_RUN | STS_CNT2_GENERAL_EN |
| | 3 | MSG_CNT2_SW_LR | MSG_CNT2_SFKT | MSG_CNT2_FQE | MSG_CNT2_ND | MSG_CNT2_OFLW | MSG_CNT2_UFLW | MSG_CNT2_CMP1 | MSG_CNT2_CMP0 |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL20-E-2CNT-2PWM (page 11-63).

### BL67-1CNT/ENC

The object reads the module's process input bytes 1 and 2.

Writing the object with any value will reset the non-volatile flags STS_ZC, STS_OFLW and STS_UFLW.

| Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 1 | ERR_PARA | SYNC_AKN | reserved | | | | | count direction |
| 2 | REG_WR_ACEPT | REG_WR_AKN | reserved | | | STS_ZC | STS_OFLW | STS_UFLW |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL67-1CNT/ENC (page 11-66).

### 11.2.7 Object $5806_{hex}$ – Optional Encoder Status

The object Optional Encoder Status reads different bytes of the module's process input data.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL××-1SSI
- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 131: Object $5806_{hex}$*

| Feature | Description |
|---|---|
| Name | Optional Encoder Status |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| PDO mapping | Yes |

#### BL××-1SSI

The object Optional Encoder Status reads byte 2 of the process input data of the BL××-1SS module. Bits 6 and 7 are masked out.

Structure of the data byte

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| masked (0) | masked (0) | reserved | | SSI_ STS3 | SSI_ STS2 | SSI_ STS1 | SSI_ STS0 |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL××-1SSI (page 11-59).

#### BL20-E-2CNT-2PWM

The object reads process input byte 6 of the 2CNT-2PWM module. Bits 5 to 7 are masked out.

| Sub-index 0×00 ≤ n ≥ 0×47 | Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 (lsb) |
|---|---|---|---|---|---|---|---|---|---|
| **n** = CNT1 | 0 | reserved | | | | | | STS_DBP1 | D1 |
| **n + 1** = CNT2 | 0 | reserved | | | | | | STS_DBP2 | D2 |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL20-E-2CNT-2PWM (page 11-63).

**BL67-1CNT/ENC**

The object reads process input byte 0 of the BL67-1CNT/ENC module.

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| reserved | A | B | Z | DI3 | DI2 | DI1 | DI0 |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL67-1CNT/ENC (page 11-66).

## 11.2.8 Object $5808_{hex}$ – Encoder Control

Object $5808_{hex}$ writes and reads the different bytes of the modules' control interface and therefore serves for influencing the module during operation.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BLxx-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 132: Object $5808_{hex}$*

| Feature | Description |
|---|---|
| Name | Encoder Control |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | rw |
| Default value | No |
| PDO mapping | Yes |

**BLxx-1SSI**

The object writes or reads byte 0 of the control interface of the SSI-module.

At the moment, only bit 7 (STOP) is used.

**Note**

Please find a detailed description of the control bits in the process output in table Meaning of the process output bits of BLxx-1SSI (page 11-53).

### BL20-1CNT-24VDC

The object Encoder Control writes or reads byte 4 of the module's control interface.

**Count operation**

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| reserved | CTRL_ DO2 | SET_DO2 | CTRL_ DO1 | SET_DO1 | reserved | CTRL_SY N | SW_GAT E |

**Measurement mode**

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| reserved | | | CTRL_ DO1 | SET_DO1 | reserved | | SW_GAT E |

**Note**

Please find a detailed description of the control bits in the process output in table Meaning of the process output bits of BL20-1CNT-24VDC (page 11-54).

### BL20-E-2CNT-2PWM

The object Encoder Control writes or reads byte 0 (channel 1), or respectively byte 1 (channel 2) of the module's process output data.

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **n** = CNT1 | reserved | CNT1_ SINGLE | CNT1_ SW_LR | CNT1_ SFKT_ DISABLE | reserved | CNT1_ LOGMSG | CNT1_ ENABLE | CNT1_ GENERAL _DISABLE |
| **n + 1** = CNT2 | reserved | CNT2_ SINGLE | CNT2_ SW_LR | CNT2_ SFKT_ DISABLE | reserved | CNT2_ LOGMSG | CNT2_ ENABLE | CNT2_ GENERAL _DISABLE |

**Note**

Please find a detailed description of the control bits in the process output in table Meaning of the process output bits of BL20-E-2CNT-2PWM (page 11-56).

### BL67-1CNT/ENC

The object reads process input byte 0 of the BL67-1CNT/ENC module.

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| DO3 | DO2 | DO1 | DO0 | reserved | RES_ STS | SYNC_ REQ | GATE |

**Note**

Please find a detailed description of the control bits in the process output in table Meaning of the process output bits of BL67-1CNT/ENC (page 11-58).

### 11.2.9 Object $5810_{hex}$ – Encoder Load Prepare Value

The object Setting the counter's count value to this value is event-driven. This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 133: Object $5810_{hex}$*

| Feature | Description |
|---|---|
| Name | Encoder Control |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | rw |
| Default value | No |
| PDO mapping | Yes |

**BL20-1CNT-24VDC**

The object takes effect on the function „Load in preparation LOAD_PREPARE" in the module's control interface.

**BL20-E-2CNT-2PWM**

The object writes or reads register $23_{hex}$ (channel 1), or respectively register $43_{hex}$ (channel 2) of the module's register interface.

**BL67-1CNT/ENC**

The object writes or reads register $2c_{hex}$ (44) of the module's register interface.

### 11.2.10 Object $5811_{hex}$ – Encoder Pulse Width

The object serves for setting the pulse duration.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 134: Object $5811_{hex}$*

| Feature | Description |
|---|---|
| Name | Encoder Pulse Width |
| Object code | ARRAY |
| Data type | Unsigned16 |
| Access | rw |
| Default value | No |
| PDO mapping | No |

**BL20-1CNT-24VDC**

The object takes effect on the module'S parameter byte 11 and serves for setting the pulse duration at DO1 and DO2 in the count operation.

**BL20-E-2CNT-2PWM**

The object writes or reads register $30_{hex}$ (channel 1), or respectively register $50_{hex}$ (channel 2) of the module's register interface.

The time is set in 1ms/ bit.

Value range: 0 ms to 65535 ms (1min 5s).

### 11.2.11 Object $5820_{hex}$ – Measuring Integration Time

The object serves for setting the integration time.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 135: Object $5820_{hex}$*

| Feature | Description |
|---|---|
| Name | Measuring Integration Time |
| Object code | ARRAY |
| Data type | Unsigned 32 |
| Access | rw |
| Default value | - |
| PDO mapping | No |

**BL20-1CNT-24VDC**

The object takes effect on parameter byte 8 and 9 and serves for setting the integration time in measurement mode.

Valid values are 1 to 1000.

**Note**

The object is only valid, if the counter module works in measurement mode.

**BL20-E-2CNT-2PWM**

The object writes or reads register $29_{hex}$ (channel 1), or respectively register $49_{hex}$ (channel 2) of the module's register interface.

For the frequency measurement and the rotation speed measurement, the integration time is entered in 10 ms/ bit.

### 11.2.12 Object $5821_{hex}$ – Measuring Low Limit

This object is used to set the module's lower limit in measurement mode.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC

*Table 136: Object $5821_{hex}$*

| Feature | Description |
|---|---|
| Name | Measuring Low Limit |
| Object code | ARRAY |
| Data type | Unsigned 32 |
| Access | rw |
| Default value | - |
| PDO mapping | No |

**BL20-1CNT-24VDC**

The object takes effect on parameter byte 2 and 4 and serves for setting the lower limit in measurement mode.

Permissible value range:
0 to 16777214



**Note**
The object is only valid, if the counter module works in measurement mode.

### 11.2.13 Object $5822_{hex}$ – Measuring High Limit

This object is used to set the module's upper limit in measurement mode.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC

*Table 137: Object $5822_{hex}$*

| Feature | Description |
|---|---|
| Name | Measuring High Limit |
| Object code | ARRAY |
| Data type | Unsigned 32 |
| Access | rw |
| Default value | - |
| PDO mapping | No |

**BL20-1CNT-24VDC**

The object takes effect on parameter byte 5 and 7 and serves for setting the upper limit in measurement mode.

Permissible value range: 1 to 16777215

**Note**

The object is only valid, if the counter module works in measurement mode.

### 11.2.14 Object $5823_{hex}$ – Measuring Units Per Revolution

Object 5823hex is used to set the sensor pulses per revolution.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC

*Table 138: Object $5823_{hex}$*

| Feature | Description |
|---|---|
| Name | Measuring Units Per Revolution |
| Object code | ARRAY |
| Data type | Unsigned 16 |
| Access | rw |
| Default value | - |
| PDO mapping | No |

**BL20-1CNT-24VDC**

The object is used to set the sensor pulses per revolution for the counter module.

Value range: 1 to 65535.

### 11.2.15 Object $5824_{hex}$ – Measuring Divisor

The object serves for scaling the measured value.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 139: Object $5824_{hex}$*

| Feature | Description |
|---|---|
| Name | Measuring Divisor |
| Object code | ARRAY |
| Data type | Unsigned32 |
| Access | rw |
| Default value | - |
| PDO mapping | No |

**BL20-E-2CNT-2PWM**

The object writes or reads register $2B_{hex}$ (channel 1), or respectively Register $4B_{hex}$ (channel 2) of the module's register interface.

**BL67-1CNT/ENC**

The object writes or reads register $38_{hex}$ (56) of the module's register interface.

### 11.2.16 Object $5825_{hex}$ – Measuring Factor

The object serves for scaling the measured value.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 140: Object $5825_{hex}$*

| Feature | Description |
|---|---|
| Name | Measuring Factor |
| Object code | ARRAY |
| Data type | Unsigned 32 |
| Access | rw |
| Default value | - |
| PDO mapping | No |

**BL20-E-2CNT-2PWM**

The object writes or reads register $2A_{hex}$ (channel 1), or respectively Register $4A_{hex}$ (channel 2) of the module's register interface.

**BL67-1CNT/ENC**

The object writes or reads register $34_{hex}$ (52) of the module's register interface.

### 11.2.17 Object $5827_{hex}$ – Measuring Timeout

The object defines the time out after which, in period duration measurement, a message is generated. The current period duration measurement is monitored.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 141: Object $5827_{hex}$*

| Feature | Description |
|---|---|
| Name | Measuring Timeout |
| Object code | ARRAY |
| Data type | Unsigned 32 |
| Access | rw |
| Default value | - |
| PDO mapping | No |

**BL20-E-2CNT-2PWM**

The object writes or reads register $2D_{hex}$ (channel 1), or respectively Register $4D_{hex}$ (channel 2) of the module's register interface.

**BL67-1CNT/ENC**

The object writes or reads register $3c_{hex}$ (60) of the module's register interface.

### 11.2.18 Object $5830_{hex}$ – Measuring Value

The object reads the measured value.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 142: Object $5830_{hex}$*

| Feature | Description |
|---|---|
| Name | Measuring Value |
| Object code | ARRAY |
| Data type | Unsigned 32 |
| Access | rw |
| Default value | - |
| PDO mapping | No |

**BL20-E-2CNT-2PWM**

The object writes or reads register $21_{hex}$ (channel 1), or respectively register $41_{hex}$ (channel 2) of the module's register interface.

**BL67-1CNT/ENC**

The object writes or reads register $30_{hex}$ (48) of the module's register interface.

### 11.2.19 Object $5831_{hex}$ – Encoder Latch Value

The object reads the value of the counters' latch-register.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-E-2CNT-2PWM

*Table 143: Object $5831_{hex}$*

| Feature | Description |
|---|---|
| Name | Encoder Latch Value |
| Object code | ARRAY |
| Data type | Unsigned 32 |
| Access | rw |
| Default value | - |
| PDO mapping | No |

**BL20-E-2CNT-2PWM**

The object writes or reads register $28_{hex}$ (channel 1), or respectively register $48_{hex}$ (channel 2) of the module's register interface.

### 11.2.20 Object $5840_{hex}$ – Diag Mapping

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BLxx-1SSI

**BLxx-1SSI**

The object writes or reads register 51 (REG_SSI_MASK) of the module.

REG_SSI_MASK contains the SSI encoder diagnostics transferred. Some SSI encoders not only transfer the position value in the data frame that they transfer to the module but also supply additional status messages.

It is advisable to include these status messages in the application in order to analyze the measured value.

Writing the REG_SSI_MASK register allows up to four individual bits to be taken from the SSI encoder data frame and transferred to the SSI_STSx bits of the process input data.

*Table 144: Object $5840_{hex}$*

| **Feature** | **Description** |
|---|---|
| Name | SSI Diag Mapping |
| Object code | ARRAY |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |
| PDO mapping | No |

Masking through REG_SSI_MASK:

| **process input data** | | **REG_SSI_MASK** | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Byte | Bit 7 | Bit 6 | B 5 | B 4 | B 3 | B 2 | B 1 | B 0 |
| SSI_STS0 | 0 | EN_ D0_RMS0 | EN_D0_DS | X | SSI_FRAME_BIT_SEL0 | | | | |
| SSI_STS1 | 1 | EN_ D1_RMS1 | EN_D1_DS | X | SSI_FRAME_BIT_SEL1 | | | | |
| SSI_STS2 | 2 | EN_ D2_RMS2 | EN_D2_DS | X | SSI_FRAME_BIT_SEL2 | | | | |
| SSI_STS3 | 3 | EN_ D3_RMS3 | EN_D3_DS | X | SSI_FRAME_BIT_SEL3 | | | | |

*Table 145: Bit-meaning*

**A***Default-setting*

| **Designation** | **Value** | **Description** |
|---|---|---|
| EN_Dx_RMSx | 0 **A** | The mapping of the SSI status messages into the process input data is not activated. |
| | 1 | The mapping of the SSI status messages into the process input data is not activated. |

*Table 145: Bit-meaning*

| Designation | Value | Description |
|---|---|---|
| EN_ Dx_DS | 0 **A** | The evaluation of the SSI status messages for bit 0 of the diagnostics is not activated. |
| | 1 | The evaluation of the SSI status messages for bit 0 of the diagnostics is activated. |
| SSI_FRAME_BIT_SEL | 0 **A** | Definition of the selected bits in the frame of the SSI encoder<br>Default: 0 |

The following applies to bit 0 (SSI group diagnostics) of the diagnostics interface and SSI_DIAG of the process input data:

(SSI_STS0 & EN_D0_DS) || (SSI_STS1 & EN_D1_DS) ||
(SSI_STS2 & EN_D2_DS) || (SSI_STS3 & EN_D3_DS)

### 11.2.21 Object $5901_{hex}$ – PWM Config

This object is used in the following modules.

- BL20-E-2CNT-2PWM

*Table 146: Object 5901$_{hex}$*

| Feature | Sub-index | Description |
|---|---|---|
| Name | | PWM Config |
| Object code | | ARRAY |
| Data type | 0×00 | Unsigned 8 |
| | 0×01 to 0×47 | Unsigned 32 |
| Access | 0×00 | ro |
| | 0×01 to 0×47 | rw |
| Default value | 0×00 | - |
| PDO mapping | | No |

**BL20-E-2CNT-2PWM**

The object affects the configuration parameters of the PWMx:

| **Sub-index** 0×00 ≤ **n** ≥ 0×47 | **Byte** | **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0 (lsb)** |
|---|---|---|---|---|---|---|---|---|---|
| **n** = PWM1 | 0 | diagnostic PWM1 | X | mode D1 (0×3F) | | | | | |
| | 1 | **Bit 15** | **Bit 14** | **Bit 13** | **Bit 12** | **Bit 11** | **Bit 10** | **Bit 9** | **Bit 8** |
| | | DBP1 STS MODE 0×00 | | substitute value P1 0×00 | substitute value D1 0×00 | mode PWM1 0×00 | | | |
| | 2 | **Bit 23** | **Bit 22** | **Bit 21** | **Bit 20** | **Bit 19** | **Bit 18** | **Bit 17** | **Bit 16** |
| | | X | | | | | | | |
| | 3 | **Bit 31 (msb)** | **Bit 30** | **Bit 29** | **Bit 28** | **Bit 27** | **Bit 26** | **Bit 25** | **Bit 24** |
| | | X | ADR AUX REG1 WR DATA (0×60) | | | | | | |

| **Sub-index**<br>0×00 ≤ **n** ≥ 0×47 | **Byte** | **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0 (lsb)** |
|---|---|---|---|---|---|---|---|---|---|
| **n + 1** = PWM2 | 0 | diagnostic PWM2 | X | mode D2 (0×3F) | | | | | |
| | 1 | **Bit 15** | **Bit 14** | **Bit 13** | **Bit 12** | **Bit 11** | **Bit 10** | **Bit 9** | **Bit 8** |
| | | DBP2 STS MODE 0×00 | | substitute value P2 (0×00) | substitute value D2 0×00 | mode PWM2 0×00 | | | |
| | 2 | **Bit 23** | **Bit 22** | **Bit 21** | **Bit 20** | **Bit 19** | **Bit 18** | **Bit 17** | **Bit 16** |
| | | X | | | | | | | |
| | 3 | **Bit 31 (msb)** | **Bit 30** | **Bit 29** | **Bit 28** | **Bit 27** | **Bit 26** | **Bit 25** | **Bit 24** |
| | | X | ADR AUX REG3 WR DATA (0×70) | | | | | | |

( ) = default parameterization

**Note**

Please find a detailed description of the parameters in the table Meaning of the parameter bits of BL20-E-2CNT-2PWM (page 11-72).

### 11.2.22 Object $5902_{hex}$ – PWM Status

This object is used in the following modules.

- BL20-E-2CNT-2PWM

**BL20-E-2CNT-2PWM**

The object supplies the following status messages:

- output status
- operation status of the PWM

*Table 147: Object $5902_{hex}$*

| Feature | Sub-index | Description |
|---|---|---|
| Name | | PWM Status |
| Object code | | ARRAY |
| PDO mapping | | Yes |
| Data type | 0×00 | Unsigned 8 |
| | 0×01 to 0×47 | Unsigned 8 |
| Access | 0×00 | ro |
| | 0×01 to 0×47 | ro |
| Default value | 0×00 | - |
| | 0×01 to 0×47 | - |

**Structure of the data byte**

| Sub-index 0×00 ≤ **n** ≥ 0×47 | Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| **n** = PWM1 | 0 | X | X | STS_ DBP1 | D1 | STS_ PWM1_ LOGMSG | STS_ PWM1_ SFKT_EN | STS_ PWM1_ RUN | STS_ PWM1_ GENERAL _EN |
| **n + 1** = PWM2 | 0 | X | X | STS_ DBP2 | D2 | STS_ PWM2_ LOGMSG | STS_ PWM2_ SFKT_EN | STS_ PWM2_ RUN | STS_ PWM2_ GENERAL _EN |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL20-E-2CNT-2PWM (page 11-63).

### 11.2.23 Object $5903_{hex}$ – PWM Flags

This object is used in the following modules.

- BL20-E-2CNT-2PWM

**BL20-E-2CNT-2PWM**

The object supplies the following status messages:

- reaching of limit values
- execution of a SW latch retrigger
- an event defined as special function (SFKT) has occurred
- a time out in the count pulse measurement occurred

*Table 148: Object $5903_{hex}$*

| Feature | Sub-index | Description |
|---|---|---|
| Name | | PWM Flags |
| Object code | | ARRAY |
| PDO mapping | | Yes |
| Data type | 0×00 | Unsigned 8 |
| | 0×01 to 0×47 | Unsigned 8 |
| Access SDO | 0×00 | ro |
| | 0×01 to 0×47 | rw |
| Access PDO | 0×01 to 0×47 | r |
| Default value | 0×00 | - |
| | 0×01 to 0×47 | - |

**Note**

By means of this access, the messages are reset automatically after reading.

**Structure of the data byte**

| Sub-index<br>0×00 ≤ **n** ≥ 0×47 | Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| **n** = PWM1 | 0 | X | X | X | X | MSG_ PWM1_ DO_ERR | MSG_ PWM1_ SFKT | MSG_ PWM1_ NDDC | MSG_ PWM1_ SW_LR |
| **n + 1** = PWM2 | 0 | X | X | X | X | MSG_ PWM2_ DO_ERR | MSG_ PWM2_ SFKT | MSG_ PWM2_ NDDC | MSG_ PWM2_ SW_LR |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL20-E-2CNT-2PWM (page 11-63).

### 11.2.24 Object $5904_{hex}$ – PWM Diag

This object is used in the following modules.

- BL20-E-2CNT-2PWM

**BL20-E-2CNT-2PWM**

The object reads the module's diagnostic byte.

*Table 149: Object $5904_{hex}$*

| Feature | Sub-index | Description |
|---|---|---|
| Name | | PWM Diag |
| Object code | | ARRAY |
| PDO mapping | | – |
| Data type | 0×00 | Unsigned 8 |
| | 0×01 to 0×47 | Unsigned 8 |
| Access | 0×00 | ro |
| | 0×01 to 0×47 | ro |
| Default value | 0×00 | - |
| | 0×01 to 0×47 | - |

**Note**

By means of this access, the messages are reset automatically after reading.

**Structure of the data byte**

| Sub-index<br>0×00 ≤ **n** ≥ 0×47 | Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|
| **n** = PWM1 | 0 | HW_ERR | PWM1_ PAR _ERR | X | X | X | X | P1_DIAG | D1_DIAG |
| **n + 1** = PWM2 | 0 | HW_ERR | PWM2_ PAR _ERR | X | X | X | X | P2_DIAG | D2_DIAG |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the diagnostic bits of BL20-E-2CNT-2PWM (page 11-79).

### 11.2.25 Object $5908_{hex}$ – PWM Control

This object is used in the following modules.

- BL20-E-2CNT-2PWM

**BL20-E-2CNT-2PWM**

The object provides the following control functions:

- general enable of the function
- start/ stop of the signal output
- Freezing the module's error messages to enable reading them without data loss
- enabling or disabling of the special function
- executing a software latch retrigger
- Setting the single/ continuous signal output

*Table 150: Object $5908_{hex}$*

| Feature | Sub-index | Description |
|---|---|---|
| Name | | PWM Control |
| Object code | | ARRAY |
| PDO mapping | | Yes |
| Data type | 0×00 | Unsigned 8 |
| | 0×01 to 0×47 | Unsigned 16 |
| Access SDO | 0×00 | ro |
| | 0×01 to 0×47 | rw |
| Access PDO | 0×01 to 0×47 | r |
| Default value | 0×00 | - |
| | 0×01 to 0×47 | - |

**Note**

By means of this access, the messages are reset automatically after reading.

### Structure of the data byte

| **Sub-index**<br>0×00 ≤ **n** ≥ 0×47 | **Byte** | **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0 (lsb)** |
|---|---|---|---|---|---|---|---|---|---|
| **n** = PWM1 | 0 | X | PWM1 _SINGLE | PWM1_ SW_LR | PWM1_ SFKT_ DISABLE | X | PWM1_ LOGMSG | PWM1_ ENABLE | PWM1_ GENERAL_ DISABLE |
| | 1 | **Bit 15 (msb)** | **Bit 14** | **Bit 13** | **Bit 12** | **Bit 11** | **Bit 10** | **Bit 9** | **Bit 8** |
| | | X | X | X | X | X | X | SET_P1 | SET_ D1 |
| **n + 1** = PWM2 | 0 | X | PWM2_SI NGLE | PWM2_ SW_LR | PWM2_ SFKT_ DISABLE | X | PWM2_ LOGMSG | PWM2_ ENABLE | PWM2_ GENERAL_ DISABLE |
| | 1 | **Bit 15 (msb)** | **Bit 14** | **Bit 13** | **Bit 12** | **Bit 11** | **Bit 10** | **Bit 9** | **Bit 8** |
| | | X | X | X | X | X | X | SET_P2 | SET_D2 |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process output bits of BL20-E-2CNT-2PWM (page 11-56).

### 11.2.26 Object $5910_{hex}$ – PWM Load Prepare Value

This object is used in the following modules.

- BL20-E-2CNT-2PWM

**BL20-E-2CNT-2PWM**

The object contains the load value (load value register, PWM1 no. 0×64, PWM2 no. 0×74) for the "prepared loading" of the counters. Setting the count value of the signals to be given out to this value is event-driven.

*Table 151: Object $5910_{hex}$*

| Feature | Sub-index | Description |
|---|---|---|
| Name | | PWM Load Prepare Value |
| Object code | | ARRAY |
| PDO mapping | | - |
| Data type | 0×00 | Unsigned 8 |
| | 0×01 to 0×47 | Integer 32 |
| Access | 0×00 | ro |
| | 0×01 to 0×47 | rw |
| Default value | 0×00 | - |
| | 0×01 to 0×47 | - |

### 11.2.27 Object $5913_{hex}$ – PWM Duty Cycle

This object is used in the following modules.

- BL20-E-2CNT-2PWM

**BL20-E-2CNT-2PWM**

The object serves for setting the Duty Cycle of the PWM pulse (Register for Duty Cycle): PWM1, no. 0×64/ PWM2, no. 0×71. )

*Table 152: Object $5913_{hex}$*

| Feature | Sub-index | Description |
|---|---|---|
| Name | | PWM Duty Cycle |
| Object code | | ARRAY |
| PDO mapping | | - |
| Data type | 0×00 | Unsigned 8 |
| | 0×01 to 0×47 | Integer 32 |
| Access | 0×00 | ro |
| | 0×01 to 0×47 | rw |
| Default value | 0×00 | - |
| | 0×01 to 0×47 | - |

### 11.2.28 Object $5920_{hex}$ – PWM Period Duration

The object contains the value for the period duration of the PWM (register for period duration: PWM1, no. 0×60/ PWM2, no. 0×70. )

*Table 153: Object $5920_{hex}$*

| Feature | Sub-index | Description |
|---|---|---|
| Name | | PWM Period Duration |
| Object code | | ARRAY |
| PDO mapping | | Yes |
| Data type | 0×00 | Unsigned 8 |
| | 0×01 to 0×47 | Integer 32 |
| Access SDO | 0×00 | ro |
| | 0×01 to 0×47 | rw |
| Access PDO | 0×01 to 0×47 | rw |
| Default value | 0×00 | - |
| | 0×01 to 0×47 | - |

### 11.2.29 Object $5931_{hex}$ – PWM Latch Value

The object reads the value of the PWMs' latch-register (PWM1, no. 0×66/ PWM2 no. 0×76).

*Table 154: Object $5931_{hex}$*

| Feature | Sub-index | Description |
|---|---|---|
| Name | | PWM Latch Value |
| Object code | | ARRAY |
| PDO mapping | | - |
| Data type | 0×00 | Unsigned 8 |
| | 0×01 to 0×47 | Unsigned 32 |
| Access | 0×00 | ro |
| | 0×01 to 0×47 | ro |
| Default value | 0×00 | - |
| | 0×01 to 0×47 | - |

### 11.2.30 Object $6800_{hex}$ – Operating Parameters

**Note**

Object $6800_{hex}$ (corresponds to object $6000_{hex}$ in accordance with CiA DS406) has no meaning with BL20×× and only exists because it is a "mandatory" object in accordance with DS406.

*Table 155: Object $6800_{hex}$*

| Feature | Description |
|---|---|
| Name | Operating Parameters |
| Object code | VAR |
| Data type | Unsigned16 |
| Access | rw |
| Default value | 00h |
| PDO mapping | No |

### 11.2.31 Objekt $6810_{hex}$ – Preset Values for Multi-Sensor Devices

The corresponds to object $6010_{hex}$ in accordance with CiA DS406 and is used for zero point adaption of the system.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL××-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM
- BL67-1CNT/ENC

*Table 156: Object $6810_{hex}$*

| Feature | Description |
|---|---|
| Name | Position Value for Multi-Sensor Devices |
| Object code | ARRAY |
| Data type | Integer32 |
| Access | rw |
| Default value | No |
| PDO mapping | Yes |

**BL××-1SSI**

The content of this object is added to the SSI encoder value. The calculated value is stored to object $6820_{hex}$.

**BL20-1CNT-24VDC**

The object writes the load value for the counter value.

**BL20-E-2CNT-2PWM**

The object writes the load value for the counter value.

**BL67-1CNT/ENC**

The object writes the load value for the counter value.

### 11.2.32 Object $6820_{hex}$ – Position Value

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL××-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWMI
- BL67-1CNT/ENC

*Table 157: Object $6820_{hex}$*

| Feature | Description |
|---|---|
| Name | Position Value for Multi-Sensor Devices |
| Object code | ARRAY |
| Data type | Integer32 |
| Access | ro |
| Default value | No |
| PDO mapping | Yes |

**BL××-1SSI**

Object $6820_{hex}$ (corresponds to object $6020_{hex}$ as per CiA DS406) contains the SSI encoder value of the BL××-1SSI module. The content of the object contains a value that is added to correct the measured value for a zero point adjustment.

**BL20-1CNT-24VDC**

Object $6820_{hex}$ (corresponds to object $6020_{hex}$ as per CiA DS406) contains the counter value of the module.

**BL20-E-2CNT-2PWM**

Object $6820_{hex}$ (corresponds to object $6020_{hex}$ as per CiA DS406) contains the counter value of the module.

**BL67-1CNT/ENC**

Object $6820_{\mathbf{hex}}$ (corresponds to object $6020_{\mathbf{hex}}$ as per CiA DS406) contains the counter value of the module.

### 11.2.33 Object $6B00_{hex}$ – CAM State Register

The object CAM State register indicates in accordance with DS406 whether the actual counter status is within the range defined by CAM1 Low limit and CAM1 High limit (object $6B10_{hex}$ and $6B20_{hex}$).

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BLxx-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 158: Object $6B00_{hex}$*

| Feature | Description |
|---|---|
| Name | CAM State Register |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| PDO mapping | Yes |

Structure of the data byte:

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| channel 1 | X | X | X | X | X | X | X | STAT_ CAM1 |

X = reserved

*Table 159: Meaning of the data bits*

| Designation | | Description |
|---|---|---|
| STAT_CAM1 **A** | 0 | The counter value lies within the following range:<br>compare value2 ≤ counter value compare value1<br>or<br>the conditions for activating this status message has not been fulfilled. |
| | 1 | The counter value lies out of the following range:<br>compare value2 ≤ counter value compare value1 |

**A** *Object $6B02_{hex}$ can be used to invert the values.*

**BLxx-1SSI**

For this module, the object behavior is as defined per DS406. Special configuration settings are not necessary.

**BL20-1CNT-24VDC**

The behavior of the BL20 counter module is only according to DS406, if the configuration and the operation mode is the following:

- Operation mode is set to „count".
- The operation mode for DO1 is set to "On when cnt. value ref. value1"

- The operation mode for DO2 is set to "On when cnt. value  ref. value2"
- The enable bits for DO1 and DO2 are set.

Please observe the following:

- Object $6B10_{hex}$ "CAM1 Low Limit" corresponds to reference value2 of the BL20-counter module.
- Object $6B20_{hex}$ "CAM1 High Limit" corresponds to reference value1 of the BL20-counter module.
- The both outputs DO1 and DO2 are assigned to the 2 reference values.
- Object $6B00_{hex}$ „CAM 1 State Register“ provides the result of the AND relation of the two outputs DO1 and DO2.

**BL20-E-2CNT-2PWM**

For this module, the object behavior is as defined per DS406. Special configuration settings are not necessary.

The following applies:

STS_DBPx = 1

at (REG_CNTx_CMP0) ≤ (REG_CNTx_CNT) < (REG_CNTx_CMP1)

**Structure of the data byte**

| **Sub-index**<br>0×00 ≤ **n** ≥ 0×47 | **Byte** | **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0 (lsb)** |
|---|---|---|---|---|---|---|---|---|---|
| **n** = CNT1 | 0 | X | X | X | X | X | X | X | STAT_ CAM1(STS _DBP1) |
| **n + 1** = CNT2 | 0 | X | X | X | X | X | X | X | STAT_ CAM2(STS _DBP2) |

**Note**

The function is only given, if DBPx STS MODE = 00 is parameterized, see Abschnitt „Meaning of the parameter bits of BL20-1CNT-24VDC“.

### 11.2.34 Object $6B01_{hex}$ – CAM1 Enable Register

The object activates or deactivates the status message concerning the comparison result (Object $6B00_{hex}$).

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BLxx-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 160: Object $6B01_{hex}$*

| Feature | Description |
|---|---|
| Name | CAM Enable register |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | rw |
| Default value | No |
| PDO mapping | No |

Structure of the data byte:

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| channel 1 | X | X | X | X | X | X | X | EN_CAM1 |
| channel 2 | X | X | X | X | X | X | X | EN_CAM2 |

X = reserved

*Table 161: Meaning of the data bits*

| Designation | | Description |
|---|---|---|
| EN_CAM1 | 0 | Object $6B00_{hex}$ is blocked. |
| | 1 | Object $6B00_{hex}$ is released. |

### 11.2.35 Object $6B02_{hex}$ – CAM Polarity Register

The object $6B02_{hex}$ can invert the status message for the comparison result (Object $6B00_{hex}$).

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL××-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 162: Object $6B02_{hex}$*

| Feature | Description |
|---|---|
| Name | CAM Polarity Register |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| Sub-index | $01_{hex}$ - $47_{hex}$ |
| Data type | Unsigned8 |
| Access | rw |
| Default value | $0_{hex}$ |

Structure of the data byte:

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| channel 1 | X | X | X | X | X | X | X | POL_ CAM1 |
| channel 2 | X | X | X | X | X | X | X | POL_ CAM2 |

X = reserved

*Table 163: Meaning of the data bits*

| Designation | | Description |
|---|---|---|
| POL_CAM1 | 0 | The status message of object $6B00_{hex}$ is not inverted. |
| | 1 | The status message of object $6B00_{hex}$ is inverted. |

### 11.2.36 Object $6B10_{hex}$ – CAM1 Low Limit

The object is the same as object $6310_{hex}$ in accordance with CiA DS406 which defines a lower switch limit for the count range.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL××-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 164: Object $6B10_{hex}$*

| Feature | Description |
|---|---|
| Name | CAM1 Low Limit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| Sub-index | $01_{hex}$ - $47_{hex}$ |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |

**BL××-1SSI**

Object CAM1 Low Limit corresponds to the reference value2 (CmpVal2) of the SSI module.

**BL20-1CNT-24VDC**

The object corresponds to the reference value2 of the counter module

The logical output DO2, whose behavior can be configured in manifold ways, is assigned to reference value 2 and thus to object CAM1 Low Limit.

If reference value2 serves as lower limit, depends on the configuration of DO2. In order to achieve a CAM-function as per DS406 a special configuration is necessary, see description of Object $6B00_{hex}$ – CAM State Register (page 11-42).

**BL20-E-2CNT-2PWM**

The object corresponds to the reference value CMP0 of the counter module

The object writes or reads register $26_{hex}$ (channel 1), or respectively register $46_{hex}$ (channel 2) of the module's register interface.

### 11.2.37 Object $6B20_{hex}$ – CAM1 High Limit

The object is the same as object $6320_{hex}$ in accordance with CiA DS406 which defines an upper switch limit for the count range.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL××-1SSI
- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 165: Object $6B20_{hex}$*

| Feature | Description |
|---|---|
| Name | CAM1 High Limit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| Sub-index | $01_{hex}$ - $47_{hex}$ |
| Data type | Integer32 |
| Access | rw |
| Default value | $0000_{hex}$ |

**BL××-1SSI**

Object CAM1 High Limit corresponds to the reference value1 (CmpVal1) of the SSI module.

**BL20-1CNT-24VDC**

The object corresponds to the reference value1 of the counter module

The logical output DO1, whose behavior can be configured in manifold ways, is assigned to reference value1 and thus to object CAM1 High Limit.

If reference value1 serves as upper limit, depends on the configuration of DO1.

In order to achieve a CAM-function as per DS406 a special configuration is necessary, see description of Object $6B00_{hex}$ – CAM State Register (page 11-42).

**BL20-E-2CNT-2PWM**

The object corresponds to the reference value CMP1 of the counter module

The object writes or reads register $27_{hex}$ (channel 1), or respectively register $47_{hex}$ (channel 2) of the module's register interface.

### 11.2.38 Object $6B30_{hex}$ – CAM1 Hysteresis

Per DS406, the object defines a hysteresis value which serves as offset for the switch points CAM1 Low Limit and CAM1 High Limit in case of switching-on and -off.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 166: Object $6B31_{hex}$*

| Feature | Description |
|---|---|
| Name | CAM1 Hysteresis |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| Sub-index | $01_{hex}$ - $47_{hex}$ |
| Data type | Unsigned16 |
| Access | rw |
| Default value | No |

#### BL20-1CNT-24VDC

The object defines the hysteresis value for reference value2 (CAM1 Low Limit - Object $6B10_{hex}$) and reference value1 (CAM1 High Limit Object $6B20_{hex}$).

**Note**

The object $6B30_{hex}$ is only valid, if the counter module works in count mode.

#### BL20-E-2CNT-2PWM

The object defines the hysteresis value for compare values CMP0 and CMP1 which affects the digital output Dx and the STS_DBPx assigned to the counter.

The object writes or reads register $2F_{hex}$ (channel 1), or respectively Register $4F_{hex}$ (channel 2) of the module's register interface.

### 11.2.39 Object $6C00_{hex}$ – Area State Register

Object $6C00_{hex}$ (corresponds to Object $6400_{hex}$ acc. to CiA DS406) contains two status bits per channel which report an underflow at the lower count limit (Object $6C01_{hex}$ Work Area Low Limit) and an overflow at the upper count limit (Object $6C02_{hex}$ Work Area High Limit).

The status bits are non-volatile. All status messages are reset by writing Object $5803_{hex}$ with any value.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 167: Object $6C00_{hex}$*

| Feature | Description |
|---|---|
| Name | Area State Register |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| Sub-index | $01_{hex}$ - $47_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |

#### BL20-1CNT-24VDC

Structure of the data byte

| Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|
| X | X | X | X | X | STS_ UFLW | STS_ OFLW | X |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL20-1CNT-24VDC (page 11-61).

**BL20-E-2CNT-2PWM**

Structure of the data byte

| **Sub-index**<br>0×00 ≤ **n** ≥ 0×47 | **Byte** | **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0 (lsb)** |
|---|---|---|---|---|---|---|---|---|---|
| **n** = CNT1 | 0 | X | X | X | X | X | MSG_ CNT1_ UFLW | MSG_ CNT1_ OFLW | X |
| **n + 1** = CNT2 | 0 | X | X | X | X | X | MSG_ CNT2_ UFLW | MSG_ CNT2_ OFLW | X |

**Note**

Please find a detailed description of the status bits in the process input in table Meaning of the process input bits of BL20-E-2CNT-2PWM (page 11-63).

### 11.2.40 Object $6C01_{hex}$ – Work Area Low Limit

Object $6C01_{hex}$ (corresponds to $6401_{hex}$ acc. to CiA DS406) defines the value for the lower count limit.

In case of an underflow, bit 2 in object $6C00_{hex}$ and bit 6 object $5803_{hex}$in are set.

This object is used in the following modules. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 168: Object $6C01_{hex}$*

| **Feature** | **Description** |
|---|---|
| Name | Work Area Low Limit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| Sub-index | $01_{hex}$ - $47_{hex}$ |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |

**BL20-1CNT-24VDC**

**Note**
The object $6B30_{hex}$ is only valid, if the counter module works in count mode.

**BL20-E-2CNT-2PWM**

The object writes or reads register $24_{hex}$ (channel 1), or respectively register $44_{hex}$ (channel 2) of the module's register interface.

## 11.2.41 Object $6C02_{hex}$ – Work Area High Limit

Object $6C02_{hex}$ (corresponds to $6402_{hex}$ acc. to CiA DS406) defines the value for the upper count limit. In case of an overflow, bit 1 in object $6C00_{hex}$ and bit 56 object $5803_{hex}$in are set. In all other encoder modules, this objects can only be found for compatibility reasons.

- BL20-1CNT-24VDC
- BL20-E-2CNT-2PWM

*Table 169: Object $6C02_{hex}$*

| Feature | Description |
|---|---|
| Name | Work Area High Limit |
| Object code | ARRAY |
| PDO mapping | No |
| Sub-index | $00_{hex}$ |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| Sub-index | $01_{hex}$ - $47_{hex}$ |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |

**BL20-1CNT-24VDC**

**Note**
The object $6B30_{hex}$ is only valid, if the counter module works in count mode.

**BL20-E-2CNT-2PWM**

The object writes or reads register $25_{hex}$ (channel 1), or respectively register $45_{hex}$ (channel 2) of the module's register interface.

### 11.2.42 Object $6D00_{hex}$ – Operating Status, Object $6D01_{hex}$ – SingleTurn Resolution (rotary), Measuring step (linear), Object $6D02_{hex}$ – Number of Distinguishable Revolutions

**Note**

The objects $6D00_{hex}$ to $6D02_{hex}$ (correspond to objects $6500_{hex}$ to $6502_{hex}$ acc. to CiA DS406) only exist because they are mandatory objects acc. to DS406 and do not have any importance for BL××. The objects are always set to 0 in BL××.

### 11.2.43 Object $6FFF_{hex}$ – Device Type

Object $6FFF_{hex}$ (corresponds to object $67FF_{hex}$ in accordance with CiA DS406) specifies the type of the second device profile supported.

The object contains the value $000A0191_{hex}$.

The low word ($0196_{hex} = 406_{dez}$) specifies the device profile.

The high word ($000A_{hex}$) describes the encoder type in accordance with CiA DS-406: ($10_{dec}$ = Multi Sensor Encoder Interface).

*Table 170: Object $6FFF_{hex}$*

| Feature | Description |
|---|---|
| Name | Device Type |
| Object code | VAR |
| PDO Mapping | No |
| Data type | Unsigned32 |
| Access | ro |

## 11.3 Process output/ control interface of the Encoder modules

### 11.3.1 Meaning of the process output bits of BL××-1SSI

*Table 171: Meaning of the process output bits of BL××-1SSI, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| CLR_CMPx | 0 | Default status, i.e. reset of FLAG_CMPx not active. |
| | 1 | Reset of FLAG_CMPx active. |
| EN_CMPx | 0 | Default status, i.e. the data bits REL_CMPx, STS_CMPx and FLAG_CMPx always have the value 0, irrespective of the actual SSI encoder value. |
| | 1 | Default status, i.e. the data bits REL_CMPx, STS_CMPx and FLAG_CMPx always have a value based on the result of the comparison with the SSI encoder value. |
| REG_RD_ADR | 0...63 | Address of the register to be read. If the read operation is successful (REG_RD_ABORT = 0), the user data is located in REG_RD_DATA of the process input data (bytes 0 to 3). |
| REG_WR | 0 | Default status, i.e. there is no request to overwrite the content of the register with the address stated at REG_WR_ADR with REG_WR_DATA. Bit REG_WR_AKN (seeAbschnitt „Meaning of the process input bits of BL××-1SSI“) is reset (0) if necessary. |
| | 1 | Request to overwrite the content of the register with address REG_WR_ADR with REG_WR_DATA. |
| REG_WR_ADR | 0...63 | Address of the register, which has to be written with REG_WR_DATA. |
| REG_WR_DATA | 0... 232-1 | Value which has to be written to the register with the address REG_WR_ADR. |
| STOP | 0 | Request to read the SSI encoder cyclically |
| | 1 | Request to interrupt communication with the encoder |

### 11.3.2 Meaning of the process output bits of BL20-1CNT-24VDC

*Table 172: Meaning of the process output bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| CTRL_DOx | 0 | The output DOx is blocked. |
| | 1 | The output DOx is released. |
| CTRL_SYN | 0 → 1 | Release synchronization:<br>A rising edge at the physical DI input enables the counter value to be set (synchronized) once/periodically to the load value. |
| EXTF_ACK | 1 | Error acknowledgement<br>The error bits must be acknowledged with the control bit EXTF_ACK after the cause of the fault has been rectified. This control bit must then be reset again. Any new error messages are not set while the EXTF_ACK control bit is set! |
| Hysteresis value | 0 to 255 | The reference value 1/2 can be assigned a hysteresis value in order to generate a response at DO1/DO2 with hysteresis. This will prevent the excessive on and off switching of DO1/DO2 if the count value fluctuates too quickly around the reference value. |
| Pulse duration | 0 to 255 | unit: ms<br>If the DO1/DO2 outputs are set to indicate counter status =reference value1/2, a longer pulse is some-times required to indicate equal values. |
| LOAD_CMP_VALx | 1 | Parameterization: "reference value x"<br>The value in bytes 0 to 3 is accepted as a reference value x. |
| LOAD_DO_ PARAM | 0 → 1 | Parameter definition of the DO1 physical output and the virtual output DO2.<br>DO1 and DO2 can indicate the status of data bit SET_DO1 and SET_DO2 or comparison results. The latest telegram (MODE_DO1 and MODE_DO2) indicates the function required for DO1 and DO2. |
| LOAD_INTTIME | 0 → 1 | Parameterization: "Integration time"<br>Bytes 0 to 1 of this process output represent a factor for defining the Integration time for frequency measurement and for determining the rotational speed. The integration time can be adjusted between 10 ms and 10 s in 10 ms increments and is produced by multiplying the factor x 10 ms.<br>With period duration measurement, this factor determines the number of periods measured in order to calculate a mean value.<br>A factor 1 to 1000 ($1_{hex}$ to $3E8_{hex}$) is permissible. |

*Table 172: Meaning of the process output bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| LOAD_LOLIMIT | 0 → 1 | Parameter setting of the lower measuring limit<br>The value in bytes 0 to 3 is accepted as lower measuring limit.<br>LOAD_LOLIMIT:<br>0 to 199 999 999 x$10^{-3}$ Hz<br>0 to 24 999 999 x $10^{-3}$ U/min<br>0 to 99 999 999 ms |
| LOAD_PREPARE | 0 → 1 | Parameterization: "Load counter in preparation"<br>The value in bytes 0 to 3 is accepted as a load value. |
| LOAD_UPLIMIT | 0 → 1 | Parameter setting of the upper measuring limit<br>The value in bytes 0 to 3 is accepted as upper measuring limit.<br>LOAD_UPLIMT:<br>1 to 200 000 000 x $10^{-3}$ Hz<br>1 to 25 000 000 x $10^{-3}$ U/min<br>1 to 100 000 000 ms |
| LOAD_VAL | 0 → 1 | Parameterization: "Load counter directly"<br>The value in bytes 0 to 3 is accepted as new counter value. |
| MODE_DOx | | MODE_DOx is only valid, if LOAD_DO_PARAM: 0 → 1<br>The output DOx can show the status of the data bit SET_DOx or comparison results, if CTRL_DOx = 1.<br>MODE_DOx defines which function DO2 is to accept:<br>– 00 The output DOx shows the status of the control bit SET_DOx. This must be released with CTRL_DOx = 1.<br>– 01 Output DOx indicates: Counter value ≥ reference value x<br>– 10 Output DOx indicates: Counter value ≤ reference value x<br>– 11 Output DOx indicates: Counter value = reference value x<br>A pulse is generated for indicating equal values. The pulse duration is defined by byte 2 of this process output. |
| RES_STS | 0 → 1 | 0 → 1 Initiate resetting of status bits. Status bits STS_ND, STS_UFLW, STS_OFLW, STS_CMP2, STS_CMP1, STS_SYN (process input) are reset. Bit RES_STS_A = 1 (process input) acknowledges that the reset command has been received. RES_STS can now be reset to 0. |
| SET_DOx | | f CTRL_DOx = 1 and the physical output DOx is set to indicate the value SET_DOx, DOx can be set and reset directly with SET_DOx.<br>DO1 can be set for this function via the process output (MODE_DOx = 00 and LOAD_DO_PARAM 0 → 1).<br>The output DOx can also be set before commissioning via the separate parameter data. The default setting for DOx is to display the value of SET_DOx. |
| SW_GATE | 0 → 1 | Counting is started (release). |
| | 1 → 0 | Counting is stopped. |

### 11.3.3 Meaning of the process output bits of BL20-E-2CNT-2PWM

*Table 173: Meaning of the process output bits of BL20-E-2CNT-2PWM, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| AUX_REGx_WR_DATA, Byte 0 ... AUX_REG1_WR_DATA, byte 3 | 0 ... $2^{32}$-1 | Value which is written to the register with the address defined in ADR AUX REGx WR DATA in the parameterization. |
| AUX_REG1_ WR_EN ... AUX_REG3_ WR_EN | 0 | Disabling the writing of register data with the register contents in AUX_REGx_WR_DATA. his option avoids an unintentional writing to the registers in the register interface during a module power-up. |
| | 1 | Writing of the Register interface with the register contents in AUX_REGx_WR_DATA is enabled. |
| CNTx_ENABLE | 0 | Not activated |
| | 1 | Enable counter CNTx (SW gate) |
| CNTx_GENERAL_ DISABLE | 0 | Count function unit CNTx is generally enabled |
| | 1 | Counter function unit is generally disabled |
| CNTx_LOGMSG | 0 | The messages in the MSG-bits in the process input/ check-back interface are active. |
| | 1 | With a change from 0 → 1 the MSG data are held and actual incoming messages are stored to the register REG_CNTx_LOGMSG.<br>Before switching to REG_CNTx_LOGMSG, this register is set to "0".<br>With a change from 1 1 → 0 , all data from REG_CNTx_LOGMSG are copied to the MSG-bits in the check-back interface. |
| CNT1_SFKT_DISABLE | 0 | The special function of input Zx is enabled depending on the parameterization in "mode Zx". |
| | 1 | The special function of input Zx is disabled. |
| CNTx _SINGLE | 0 | Single enabling of CNTx |
| | 1 | Continuous enabling of CNTx |
| CNTx_SW_LR | 0 | Not activated |
| | 1 | A (SW) latch retrigger has to be executed at counter CNTx with a change from 0 ®→ 1. |
| PWMx_ ENABLE | 0 | Not activated |
| | 1 | Enable output PWMx<br>(The enable is done either per SW- **or** per HW gate.) |

*Table 173: Meaning of the process output bits of BL20-E-2CNT-2PWM, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| PWMx_GENERAL_DISABLE | 0 | Not activated |
| | 1 | Output PWMx generally enabled |
| PWMx_LOGMSG | 0 | The messages in the MSG-bits in the process input/ check-back interface are active. |
| | 1 | With a change from 0 → 1 the MSG data are held and actual incoming messages are stored to the register REG_CNTx_LOGMSG.<br>Before switching to REG_CNTx_LOGMSG, this register is set to "0".<br>With a change from 1 → 0, all data from REG_CNTx_LOGMSG are copied to the MSG-bits in the check-back interface. |
| PWMx_SINGLE | 0 | Single enabling of PWMx |
| | 1 | Continuous enabling of PWM |
| PWMx_SFKT_DISABLE | 0 | Enable the special function of input Zx depending on the parameterization. |
| | 1 | Disable the special function of input Zx depending on the parameterization. |
| PWMx_SW_LR | 0 | Not activated |
| | 1 | A latch retrigger has to be executed at counter PWMx with a change from 0 → 1 . |
| REG_RD_ADR | 0...127 | Address of the register which has to be read. If RD_ABORT = 0, then the user data can be found in REG_RD_DATA in the process input/ check-back interface. |
| REG_WR | 0 | Initial state |
| | 1 | Triggering a write command. The register of which the address has been defined with REG_WR_ADR, will be written with data from REG_WR_DATA. |
| REG_WR_ADR | 0...127 | Address of the register, which has to be written with REG_WR_DATA (see below). |
| REG_WR_DATA, byte 0<br>...<br>REG_WR_DATA, byte 3 | 0 ... $2^{32}$-1 | Value which, during a write operation, has to be written to the register selected with REG_WR_ADR (see above). |
| SET_ Dx | 0 | Clearing of bit Dx |
| | 1 | Setting the bit Dx |

*Table 173: Meaning of the process output bits of BL20-E-2CNT-2PWM, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| SET_ Px | 0 | Clearing of bit Px |
| | 1 | Setting the bit Px |

### 11.3.4 Meaning of the process output bits of BL67-1CNT/ENC

*Table 174: Meaning of the process output bits of BL67-1CNT/ENC, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| DOx | 0 | Digital output DOx = 0 |
| | 1 | Digital output DOx = 1 |
| GATE | 0 | Counter inactive |
| | 1 | Counter active, depending on parameter Gate function |
| REG_RD_ADR | 0 to 127 | Address of the register which has to be read. If RD_ABORT = 0, the user data can be found in REG_RD_DATA in the status interface (bytes 4-7). |
| RES_STS | | During the change from 0 to 1 the counter status bits (STS_UFLW and STS_OFLW) are reset. |
| REG_WR | 0 | Initial state |
| | 1 | Command: write register<br>Request to overwrite the content of the register with address REG_WR_ADR with REG_WR_DATA. |
| REG_WR_ADR | 0 to 127 | Address of the register, which has to be written with REG_WR_DATA. |
| REG_WR_DATA | 0 to $2^{32}$-1 | Value which, during a write operation, has to be written to the register selected with REG_WR_ADR. |
| SYNC_REQ | 1 | Synchronization request<br>The bit SYNC_AKN of the status interface is reset. |

## 11.4 Meaning of the process input bits of the Encoder modules

### 11.4.1 Meaning of the process input bits of BL××-1SSI

*Table 175: Meaning of the process input bits of BL××-1SSI, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| ERR_PARA | 0 | The parameter set of the module has been accepted. |
| | 1 | Operation of the module is not possible with the present parameter set. |
| ERR_SSI | 0 | SSI encoder signal present. |
| | 1 | SSI encoder signal faulty. (e.g. due to a cable break). |
| FLAG_CMPx | 0 | Default status, i.e. the register contents have not yet matched (REG_SSI_POS) = (REG_CMPx) since the last reset. |
| | 1 | The contents of the registers match (REG_SSI_POS) = (REG_CMPx). This marker must be reset with CLR_CMPx = 1 in the process output data. |
| REG_RD_ABORT | 0 | The reading of the register defined in REG_RD_ADR has been accepted and executed. The content of the register can be found in the user data (REG_RD_DATA, byte 0-3). |
| | 1 | Reading of the register defined in REG_RD_ADR has not been accepted. The user data range (REG_RD_DATA Bytes 0-3) is zero. |
| REG_RD_DATA | 0... $2^{32}$-1 | Content of the register to be read if REG_RD_ABORT=0. If REG_RD_ABORT =1, then REG_RD_DATA=0. |
| REG_RD_ADR | 0...63 | Address of the register to be read. If the read operation is successful (REG_RD_ABORT = 0), the user data is located in REG_RD_DATA of the process input data (bytes 0 to 3). |
| REG_WR_ACEPT | 0 | Writing the user data from the process output to the register addressed with REG_WR_ADR in the process output could not be done. |
| | 1 | Writing the user data from the process output to the register addressed with REG_WR_ADR in the process output was successful. |

*Table 175: Meaning of the process input bits of BL××-1SSI, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| REG_WR_AKN | 0 | No modification of the data in the register bank by process output, i.e. REG_WR = 0.<br>A write job would be accepted with the next telegram of process output data.<br>(handshake for data transmission to the register.) |
| | 1 | A modification of the register contents by a process output was initiated, which means REG_WR = 1 → Abschnitt „Meaning of the process output bits of BL××-1SSI“.<br>A write job would not be accepted with the next telegram of process output data. |
| REL_CMPx | 0 | A comparison of the register contents has produced the following result:<br>(REG_SSI_POS) < (REG_CMP1) |
| | 1 | A comparison of the register contents has produced the following result:<br>(REG_ SSI_POS) ≥ (REG_CMP1) |
| REL_CMPx | 0 | A comparison of the register contents has produced the following result:<br>(REG_SSI_POS) < (REG_CMPx) |
| | 1 | A comparison of the register contents has produced the following result:<br>(REG_SSI_POS) ≥ (REG_CMP2) |
| SSI_DIAG | 0 | No enabled status signal is active (SSI_STSx = 0). |
| | 1 | At least one enabled status signal is active (SSI_STSx = 1) |
| SSI_STSx | 0 | These four bits transfer the status bits of the SSI encoder with the status messages of the SSI module. With some SSI encoders, the status bits are transferred together with the position value. |
| | 1 | |
| STS_CMP1 | 0 | A comparison of the register contents has produced the following result:<br>(REG_SSI_POS) ≠ (REG_CMP1) |
| | 1 | A comparison of the register contents has produced the following result:<br>(REG_ SSI_POS) = (REG_CMP1) |
| STS_CMP2 | 0 | A comparison of the register contents has produced the following result:<br>(REG_SSI_POS) ≠ (REG_CMP1) |
| | 1 | A comparison of the register contents has produced the following result:<br>(REG_ SSI_POS) = (REG_CMP2) |

*Table 175: Meaning of the process input bits of BL××-1SSI, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| STS_DN (LED DN) | 0 | The SSI encoder values are incremented or the values are constant. |
| | 1 | The SSI encoder values are decremented. |
| STS_OFLW | 0 | A comparison of the register contents has produced the following result: (REG_SSI_POS) ≤ (REG_UPPER_LIMIT) |
| | 1 | A comparison of the register contents has produced the following result: (REG_SSI_POS) > (REG_UPPER_LIMIT) |
| STS_UFLW | 0 | A comparison of the register contents has produced the following result: (REG_SSI_POS) ≥ (REG_LOWER_LIMIT) |
| | 1 | A comparison of the register contents has produced the following result: (REG_SSI_POS) < (REG_LOWER_LIMIT) |
| STS_UP (LED UP) | 0 | The SSI encoder values are decremented or the values are constant. |
| | 1 | The SSI encoder values are incremented. |
| STS_STOP | 0 | The SSI encoder is read cyclically. |
| | 1 | Communication with the SSI encoder is stopped as STOP = 1 (process output) or ERR_PARA = 1. |

### 11.4.2 Meaning of the process input bits of BL20-1CNT-24VDC

*Table 176: Meaning of the process input bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| ERR_24Vdc | 1 | Short-circuit sensor supply:<br>This diagnostics information must be acknowledged with the EXTF_ACK (process output) control bit. |
| ERR_DO | 1 | Short-/open circuit/excess temperature at the output DO1:<br>This diagnostics information must be acknowledged with the EXTF_ACK (process output) control bit. |

*Table 176: Meaning of the process input bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| ERR_LOAD | 1 | Error with load function<br>Control bits LOAD_DO_PARAM, LOAD_CMP_VAL2, LOAD_CMP_VAL1, LOAD_PREPARE and LOAD_VAL must not be set at the same time during the transfer. An incorrect value was transferred with the control bits.<br>Example:<br>The values for "load load value directly" or "load load value in preparation" have been chosen above the upper or below the lower count limit. |
| ERR_PARA | 0 | The parameter definition is correct as per specification. |
| | 1 | There is a parameter error. ERR_PARA is a group diagnostics bit. With the separate diagnostics message bits 3 to 6 describe the parameter errors in more detail. |
| RES_STS_A | 0 | The last process output telegram contained: RES_STS = 0. |
| | 1 | Resetting of status bits running. The last process output telegram contained: RES_STS = 1. |
| STS_CMP1 | 1 | Status comparator 1<br>This status bit indicates a comparison result for comparator 1 if: the output DO1 is released with CTRL_DO1 = 1 and a comparison is run via MODE_DO1 = 01, 10 or 11.<br>Otherwise STS_CMP1 simply indicates that the output is or was set.<br>STS_CMP1 is also set if DO1 SET_DO1 = 1 when the output is not released.<br>This bit must be reset by the RES_STS control bit. |
| STS_CMP2 | 1 | Status comparator 2<br>This status bit indicates a comparison result for comparator 2 if: the output DO2 is released with CTRL_DO2 = 1 and a comparison is run via MODE_DO2 = 01, 10 or 11.<br>Otherwise STS_CMP2 simply indicates that the output is or was set.<br>STS_CMP2 is also set if DO1 SET_DO2 = 1 when the output is not released.<br>This bit must be reset by the RES_STS control bit. |
| STS_DN | 1 | Status direction down. |
| STS_DI | 1 | The DI status bit indicates the status of digital input DI. |
| STS_DO1 | 1 | The DO1 status bit indicates the status of digital output DO1. |
| STS_DO2 | 1 | The DO2 status bit indicates the status of digital output DO2. |
| STS_GATE | 1 | Counting operation running. |

*Table 176: Meaning of the process input bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| STS_LOAD | 1 | Status of load function is set, if the Load function is running. |
| STS_ND | 1 | Status zero crossing<br>Set on crossing zero in counter range when counting without main direction.<br>This bit must be reset by the RES_STS control bit. |
| STS_OFLW | 1 | Status upper count limit<br>Set if the counter goes above the upper count limit.<br>This bit must be reset by the RES_STS control bit. |
| STS_UFLW | 1 | Status lower count limit<br>Set if the count value goes below the lower count limit.<br>This bit must be reset by the RES_STS control bit. |
| STS_UP | 1 | Status direction up. |
| STS_SYN | 1 | Status synchronization<br>After synchronization is successfully completed the STS_SYN status bit is set. This bit must be reset by the RES_STS control bit. |

### 11.4.3 Meaning of the process input bits of BL20-E-2CNT-2PWM

*Table 177: Meaning of the process input bits of BL20-E-2CNT-2PWM, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| **CNTx** | | |
| STS_CNTx_GENERAL_EN | 1 | Function enabled |
| STS_CNTx_RUN | 1 | CNTx Counter ready to count |
| STS_CNTx_SFKT_EN | 1 | Special function of Z disabled for CNTx |
| STS_CNTx_LOGMSG | 0 | Status of MSG bits actual |
| | 1 | Status of MSG bits are held |
| STS_CNTx_DIR | 0 | CNTx Counter counts down |
| | 1 | CNTx Counter counts up |

*Table 177: Meaning of the process input bits of BL20-E-2CNT-2PWM, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| Ax, Bx, Zx | 0 | Digital input is ‚LOW' |
| | 1 | Digital input is ‚HIGH' |
| MSG_CNTx_CMP0 | 1 | The counter CNTx reports that the compare value CMP0 was reached. |
| MSG_CNTx_CMP1 | 1 | The counter CNTx reports that the compare value CMP1 was reached. |
| MSG_CNTx_UFLW | 1 | The counter CNTx reports the lower count limit was reached. |
| MSG_CNTx_OFLW | 1 | The counter CNTx reports the upper count limit was reached. |
| MSG_CNTx_ND | 1 | The counter CNTx reports a zero crossing. |
| MSG_CNTx_FQE | 1 | The counter CNTx reports an error in frequency/ period duration measurement.<br>Error cause: Max. length of the no-pulse period reached.<br>The value cannot be displayed correctly in the register for the "pulses per integration time" REG_CNTx_IPI due to a multiplicator which has been set too high in register REG_CNTx_MUL. |
| MSG_CNTx_SFKT | 1 | The event according to there parameterized special function CNTx_SFKT_DISABLE did occur . |
| MSG_CNTx_SW_LR | 1 | The function Latch-Retrigger has been activated via bit CNTx_SW_LR = 1. |
| **PWMx** | | |
| MSG_PWM1x_SW_LR | 1 | The function Latch-Retrigger has been activated via bit PWMx_SW_LR = 1. |
| MSG_PWMx_NDDC | 1 | The counter PWMx reports a zero crossing. |
| MSG_PWMx_SFKT | 1 | The event according to there parameterized special function PWMx_SFKT_DISABLE did occur . |
| MSG_PWMx_DO_ERR | 1 | One of the outputs Px (Px_DIAG) or Dx (Dx_DIAG) of the corresponding PWMx-channel sent an error. |
| STS_PWMx_GENERAL_EN | 1 | Function enabled, with a change from 0 Æ→ 1 the channel is set to the initial state |
| STS_PWMx_RUN | 1 | PWMx-signal output active |
| STS_PWMx_SFKT_EN | 1 | Special function of Z enabled for PWMx |
| STS_PWMx_LOGMSG | 0 | Status of MSG bits actual |
| | 1 | Status of MSG bits are held |

*Table 177: Meaning of the process input bits of BL20-E-2CNT-2PWM, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| **Communication** | | |
| Dx | 0 | Digital output is ‚LOW' |
| | 1 | Digital output is 'HIGH' |
| STS_DBPx | 1 | Status of the information defined through DBPx STS MODE. |
| STS_CONFIG _ERR | 1 | In REG_CONFIG_ERR an error is reported |
| REG_RD_ABORT | 0 | The reading of the register defined in REG_RD_ADR has been accepted and executed. The content of the register can be found in the user data (REG_RD_DATA). |
| | 1 | Reading of the register defined in REG_RD_ADR has not been accepted. The register content (REG_RD_DATA) is zero. |
| REG_WR_AKN | 0 | A change of register contents had been assigned through a process output. |
| | 1 | No change of register contents through a process output. (Write access REG_WR to the register interface is only possible, if this bit was zero before; handshake for data transfer to the registers). |
| REG_WR_ACEPT | 0 | Writing the user data from the control interface to the register addressed with REG_WR_ADR in the control interface could not be done. |
| | 1 | Writing the user data from the control interface to the register addressed with REG_WR_ADR in the control interface was successful. |
| REG_RD_ADR | 0 to 127 | Address of the input register of which the content is shown in the user data (REG_RD_DATA) in the check-back interface if REG_RD_ABORT = 0. |
| **User data** | | |
| REG_RD_DATA | 0 ... $2^{32}$-1 | Content of the register of which the address is transferred in the process input data (REG_RD_ADR) if REG_RD_ABORT = 0. If not, If not,<br>If not, REG_RD_DATA = 0. |
| AUX_REGx_RD_DATA | 0 ... $2^{32}$-1 | Value, which is read from the register with the address defined in the parameterization in ADR_AUX_REGx_RD_DATA. |

### 11.4.4 Meaning of the process input bits of BL67-1CNT/ENC

*Table 178: Meaning of the process input bits of BL67-1CNT/ENC, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| A, B, Z | 0 | Digital input A, B or Z = 0 |
| | 1 | Digital input A, B or Z = 1 |
| AUX_RD_DATA | 0 to $2^{32}$-1 | Content of the register which has been defined via parameter byte 14 (see Abschnitt „Meaning of the parameter bits of BL67-1CNT/ENC“). |
| DI0 to DI3 | 0 | Digital input DIx = 0 |
| | 1 | Digital input DIx = 1 |
| ERR_PARA | 0 | The last change in parameters is valid. |
| | 1 | Faulty/ inconsistent parameter data. |
| REG_ACT_RD_ ADR | 0 to 127 | Address of the actually read input register. |
| REG_RD_ABORT | 0 | REG_RD_ADR valid<br>→ The reading of the register defined in REG_RD_ADR has been accepted and executed. The content of the register can be found in the user data (REG_RD_DATA, byte 0-3). |
| | 1 | REG_RD_ADR error<br>→ Reading the register defined in REG_RD_ADR not accepted. The value in the user data area (REG_RD_DATA, byte 0-3) is zero. |
| REG_RD_DATA | 0 to $2^{32}$-1 | Content of the register selected by REG_RD_ADR. If RD_ABORT = 0, if not REG_RD_DATA = 0. |
| REG_WR_ACCEPT | 0 | REG_WR_ADR error<br>→ During REG_WR = 1 the register addressed in REG_WR_ADR in the control interface could **not** successfully be written with user data. |
| | 1 | REG_WR_ADR valid<br>→ During REG_WR = 1 the register addressed in REG_WR_ADR in the control interface could not successfully be written with user data. |
| REG_WR_AKN | 0 | No change in the in the registers<br>→ No change of register contents through a process output. A write access (REG_WR) to the register bank is only possible when this bit was previously zero; handshake for data transfer to the registers. |
| | 1 | Register contents updated<br>→ A change of register contents had been assigned through a process output. |

*Table 178: Meaning of the process input bits of BL67-1CNT/ENC, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| STS_OFLW | 1 | Counter value exceeded upper limit of counter range. |
| STS_UFLW | 1 | Counter value below lower limit of counter range |
| STS_ZC | 1 | Counter value crossed zero value |
| SYNC_AKN | 1 | Encoder synchronized with zero-position. |
| count direction | 0 | Up |
| | 1 | Down |

## 11.5 Parameter interface of the Encoder modules

### 11.5.1 Meaning of the parameter bits of BL××-1SSI

*Table 179: Meaning of the parameter bits of BL××-1SSI, alphabetically sorted*

**A***Default-setting*

| Designation | Value | Description |
|---|---|---|
| DIS_ERR_SSI | Sensor idle data cable test | |
| | 0 **A** | Activate: ZERO test of data cable. |
| | 1 | Deactivate:<br>After the last valid bit, a ZERO test of the data cable is not carried out. |
| INVALID_BITS_ LSB | Invalid bits (LSB) | |
| | 0 to 15 | Number of invalid bits on the LSB side of the position value supplied by the SSI encoder.<br>The meaningful word width of the position value transferred to the module bus master is as follows:<br>SSI_FRAME_LEN - INVALID_BITS_MSB - INVALID_ BITS_LSB.<br>The invalid bits on the LSB side are removed by shifting the position value to the right, starting with the LSB.<br>INVALID_BITS_MSB +INVALID_BITS_LSB must always be less than SSI_FRAME_LEN.<br>Default 0 bit = $0_{hex}$ **A** |
| INVALID_BITS_ MSB | Number of invalid bits (MSB) | |
| | 0 to 7 | Number of invalid bits on the MSB side of the position value supplied by the SSI encoder.<br>The meaningful word width of the position value transferred to the module bus master is as follows:<br>SSI_FRAME_LEN - INVALID_BITS_MSB - INVALID_BITS_LSB.<br>The invalid bits on the MSB side are zeroed by masking the position value. INVALID_BITS_MSB +INVALID_BITS_LSB must always be less than SSI_FRAME_LEN. |

*Table 179: Meaning of the parameter bits of BL××-1SSI, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| SSI_BIT_RATE | Data rate | |
| | 0 | 1000000 bps |
| | 1 | 500000 bps |
| | 2 | 250000 bps |
| | 3 | 125000 bps |
| | 4 | 100000 bps |
| | 5 | 83000 bps |
| | 6 | 71000 bps |
| | 7 | 62500 bps |
| | 8 to 15 | reserved |
| SSI_CODE_G/D | 0 **A** | SSI encoder sends data in binary code |
| | 1 | SSI encoder sends data in Gray code |
| SSI_FRAME_ LENGTH | Data frame bits | |
| | 1 to 32 | Number of bits of the SSI data frame. SSI_FRAME_LEN must always be greater than INVALID_BITS. Default: 25 = $19_{hex}$ **A** |

## 11.5.2 Meaning of the parameter bits of BL20-1CNT-24VDC

*Table 180: Meaning of the parameter bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| Diagnostic DOx | 0 **A** | Diagnostic activated |
| | 1 | Diagnostic deactivated |
| Digital input DI | 0 **A** | normal |
| | 1 | inverted |
| Substitute value DO | 0 **A** | 0 |
| | 1 | 1 |

**A** *Default-setting*

*Table 180: Meaning of the parameter bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| Function DI | 00 **A** | input |
| | 01 | HW gate |
| | 10 | Latch-Retrigger when edge pos. (only count operation) |
| | 11 | Synchronization when edge pos. (only count operation) |
| Function DOx | 00 **A** | output |
| | 01 | – on when cnt value >= ref. value (count operation)<br>– outside of limit (measurement operation) |
| | 10 | – on when cnt value <= ref. value (count operation)<br>– below lower limit (measurement operation) |
| | 11 | – on when cnt value = ref. value (count operation)<br>– above upper limit (measurement operation) |
| Sensor pulse per revolution | 1**A** to 65535 | |
| Main count direction | 00 **A** | None |
| | 01 | Up |
| | 10 | Down |
| Hysteresis | 0 **A** to 255 | Unsigned8 |
| Pulse duration DOx | 0 **A** to 255 | [n*2ms], Unsigned8 |
| integration time | 1 to 1000 | [n*10ms]**,** 10 **A** |
| Measurement mode | 100000 | Frequency measurement **A** |
| | 100001 | Revolutions speed measurement |
| | 100010 | Period duration measurement |
| Upper limit | 1 to 16777215 x $10^{-3}$ | |
| Upper limit (HWORD) | 0 **A** to 255 | Unsigned8 |
| Upper limit (LWORD) | 0 **A** to 65535 | |
| Upper count limit | 0 to + 2147483647 ($2^{31}$-1) | |
| Upper count limit (HWORD) | 0 to 32767 **A** | Unsigned16 |
| Upper count limit (LWORD) | 0 to 65535 **A** | Unsigned16 |

*Table 180: Meaning of the parameter bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| direction input (B) | 0 **A** | normal |
| | 1 | inverted |
| group diagnostics | 0 **A** | release |
| | 1 | block |
| sensor (A) | 0 **A** | normal |
| | 1 | inverted |
| Sensor/ input filter (x) | 0 **A** | 2,5 µs (200 kHz) |
| | 1 | 25 µs (20 kHz) |
| Signal evaluation (A, B) | 00 **A** | pulse and direction |
| | 01 | rotary sensor: single |
| | 10 | rotary sensor: double |
| | 11 | rotary sensor: fourfold |
| Synchronization | 0 **A** | single-action |
| | 1 | periodical |
| Gate function | 0 **A** | abort count procedure |
| | 1 | interrupt count procedure |
| Lower limit | 0 to 16 777 214 x $10^{-3}$ | |
| Lower limit (HWORD) | 0 **A** to 255 | Unsigned8 |
| Lower limit (LWORD) | 0 **A** to 65535 | |
| Lower count limit | -2147483648 ($-2^{31}$) bis 0 | |
| Lower count limit (HWORD) | -32768**A** to 0 | Signed16 |
| Lower count limit (LWORD) | -3 2 768 to 32 767 | 0 **A**, Signed16 |
| Behaviour CPU/ master STOP | 00 **A** | turn off DO1 |
| | 01 | proceed with operating mode |
| | 10 | DO1 switch to substitute value |
| | 11 | DO1 hold last value |

*Table 180: Meaning of the parameter bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| Counter mode | 100000 **A** | continuous count |
| | 100001 | single counting |
| | 100010 | periodical counting |

### 11.5.3 Meaning of the parameter bits of BL20-E-2CNT-2PWM

*Table 181: Meaning of the parameter bits of BL20-E-2CNT-2PWM, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| ADR AUX REGx WR DATA | | Address of the basic write registers<br>(Default:<br>ADR AUX REG1 WR DATA = 0x60, ADR AUX REG2 WR DATA = 0x61, ADR AUX REG3 WR DATA = 0x70) |
| ADR AUX REGx RD DATA | | Address of the basic read registers<br>(Default:<br>ADR AUX REG1 RD DATA = 0x20, ADR AUX REG2 RD DATA = 0x21, ADR AUX REG3 RD DATA = 0x40) |
| DBPx STS MODE | 00 **A** | STS_DBPx = 1<br>at<br>(REG_CNTx_CMP0) <= (REG_CNTx_CNT) < (REG_CNTx_CMP1) |
| | 01 | reserved |
| | 10 | |
| | 11 | STS_DBPx = Px |
| Diagnostic CNTx,<br>Diagnostic PWMx | 0 | Diagnostic messages of the function unit activated in diagnostic interface **A** |
| | 1 | Diagnostic messages of the function unit deactivated in diagnostic interface A |
| Input Ax,<br>Input Bx,<br>Input Zx, | 0 **A** | Signal logic remains (LOW = 0 / HIGH = 1) |
| | 1 | Invert signal before processing |

*Table 181: Meaning of the parameter bits of BL20-E-2CNT-2PWM, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| Substitute value Px, Dx | 0 **A** | The output of the substitute value depends on the parameterization of the used gateway (see documentation for the BL20-gateways). |
| Filter Ax, Bx | 00 | 2 µs **A** |
| | 01 | 16 µs |
| | 10 | reserved |
| | 11 | |
| Filter Zx | 00 | 2 µs **A** |
| | 01 | 16 µs |
| | 10 | reserved |
| | 11 | |
| Main count direction CNTx | 00 | Basic function **A** |
| | 01 | None |
| | 10 | Up |
| | 11 | Down |
| Measurement mode CNTx | 0 | Frequency measurement **A** |
| | 1 | Period duration measurement |
| Mode CNTx | 0000 **A** | Pulse direction, single sample |
| | 0001 | Pulse direction, double sample |
| | 0010 | AB mode, single sample |
| | 0011 | AB mode, double sample |
| | 0100 | AB mode, four samples |
| | 0101 to 1110 | reserved |
| | 1111 | AB only input |
| Mode Dx | | Definition of the function for Dx<br>(Default = 11 1111 → simple output, can be controlled via process data)<br>Please find additional assignment in the general module documentation (Turck-documentation number: D301224) |

*Table 181: Meaning of the parameter bits of BL20-E-2CNT-2PWM, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| Mode PWMx | 0000 **A** | PD DC Definition: |
| | 0001 | HT LT Definition |
| | 0010 to 0111 | reserved |
| | 1111 | P just output |
| Mode Zx | 0000 | Alarm input CNT |
| | 0001 **A** | HW gate CNT |
| | 0010 | Single Latch-Retrigger CNT |
| | 0011 | Continuous latch retrigger CNT |
| | 0100 | Single L.-R. and HW gate CNT |
| | 0101 | Continuous L.-R. and HW gate CNT |
| | 0110 | reserved |
| Pull Up Zx | 0 **A** | Pull Up resistance 20 kΩ off |
| | 1 | Pull Up resistance 20 kΩ on |
| Threshold input A,B,Z CNTx | 0 **A** | Threshold 7.5V (only valid for Ax, Bx, Zx) |
| | 1 | Threshold 2.5V (only valid for Ax, Bx, Zx) |

### 11.5.4 Meaning of the parameter bits of BL67-1CNT/ENC

*Table 182: Meaning of the parameter bits of BL67-1CNT/ENC, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| Input x | 0 **A** | normal |
| | 1 | inverted |
| Input filter (A, B) | 00 **A** | 500 kHz |
| | 01 | 50 kHz |
| | 10 | 5 kHz |
| | 11 | reserved |

*Table 182: Meaning of the parameter bits of BL67-1CNT/ ENC, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| Function DI3 | 0 **A** | input |
| | 1 | encoder-GND |
| Function DO3 | 0 **A** | output |
| | 1 | encoder power supply |
| Encoder signal | 0 **A** | Push-pull input (single ended):<br>Signal evaluation between A, B, Z and GND. The inputs /A, /B and /Z are internally connected to GND. |
| | 1 | RS422-input: Signal evaluation between A, B, Z and /A, /B, /Z |
| Measurement mode | 0 **A** | Frequency measurement |
| | 1 | Period duration measurement |
| PullUp Z | 0 **A** | The PullUp-resistance for input Z is switched off. |
| | 1 | The PullUp-resistance for input Z is switched on. |
| REG_AUX_ADR | 0 to 127<br>Default:<br>48 **A** | Definition of the register number from the register interface (REG_PARA; REG_COUNTER_VALUE, REG_LOWER_LIMIT etc.), whose content should be mapped into byte 8 - 11 of the process input data. |
| Signal evaluation (A, B) | 00 | 1 x: rising edge at A |
| | 01 | 1 x: falling edge at A |
| | 10 **A** | 2 x: both edges at A |
| | 11 | 4 x: both edges at A and B (only incremental encoders) |
| Threshold input A, B, Z | 0000<br>0001<br>0010<br>0011<br>0100<br>0101<br>0110<br>0111<br>1000<br>1001<br>1010<br>1011 **A**<br>1100<br>1101<br>1110<br>1111 | 1V<br>1,5 V<br>2 V<br>2,5 V<br>3 V<br>4 V<br>5 V<br>6 V<br>7 V<br>8 V<br>9 V<br>10 V<br>12 V<br>14 V<br>16 V<br>18 V |

*Table 182: Meaning of the parameter bits of BL67-1CNT/ ENC, alphabetically sorted*

| Bit | Value | Meaning |
|---|---|---|
| Synchronization with Z | 0 **A** | single-action<br>If a signal is pending at Z and if, at the same time, the bit SYNC_REQ = 1 (see Abschnitt „Meaning of the process output bits of BL67-1CNT/ENC“ control interface: byte 1, bit 6), then the counter is synchronized once with the load value. |
| | 1 | periodical<br>If a signal is pending at Z and if, at the same time, the bit SYNC_REQ = 1 (see Abschnitt „Meaning of the process output bits of BL67-1CNT/ENC“ control interface: byte 1, bit 6), then the counter is synchronized once with the load value. |
| Gate | 0 **A** | normal |
| | 1 | inverted |
| Gate function | 000 | Counter permanently inactive |
| | 001 | DI0 is HW-gate |
| | 010 | DI1 is HW-gate |
| | 011 | DI2 is HW-gate |
| | 100 | DI3 is HW-gate |
| | 101 | Z is gate |
| | 110 **A** | only SW-gate |
| | 111 | reserved |
| Count mode | 0 **A** | encoder |
| | 1 | pulse and direction |
| Count direction | 0 **A** | Up |
| | 1 | Down |

## 11.6 Diagnostic interface of the Encoder modules

### 11.6.1 Meaning of the diagnostic bits of BL××-1SSI

*Table 183: Meaning of the diagnostic bits of BL××-1SSI, alphabetically sorted*

| Designation | Value | Description |
|---|---|---|
| ERR_PARA | 0 | The parameter set of the module has been accepted. |
| | 1 | Operation of the module is not possible with the present parameter set. |
| ERR_SSI | 0 | SSI encoder signal present. |
| | 1 | SSI encoder signal faulty. (e.g. due to a cable break). |
| SSI_DIAG | 0 | No enabled status signal is active (SSI_STSx = 0). |
| | 1 | At least one enabled status signal is active (SSI_STSx = 1). |
| SSI_OFLW | 0 | SSI encoder value below / equal to upper limit. |
| | 1 | SSI encoder value above upper limit. Overflow occurred. |
| SSI_UFLW | 0 | SSI encoder value above / equal to lower limit. |
| | 1 | SSI encoder value below lower limit. Underflow occurred. |

### 11.6.2 Meaning of the diagnostic bits of BL20-1CNT-24VDC

*Table 184: Meaning of the diagnostic bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Description |
|---|---|
| **Count operation** | |
| ERR_24Vdc | Short-circuit sensor supply:<br>This diagnostics information must be acknowledged with the EXTF_ACK (see Meaning of the process output bits of BL20-1CNT-24VDC (page 11-54)) control bit. |
| ERR_DO | Short-/open circuit/excess temperature at the output DO1:<br>This diagnostics information must be acknowledged with the EXTF_ACK (see Meaning of the process output bits of BL20-1CNT-24VDC (page 11-54)) control bit. |

*Table 184: Meaning of the diagnostic bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Description |
|---|---|
| ERR_PARA | **Bit 2:** End of counter range wrong<br>The following parameter errors are indicated:<br>Upper count limit = lower count limit<br>Upper count limit ≤ lower count limit<br>Upper count limit  0 |
| | **Bit 3:** Start of counter range wrong<br>Lower count limit = upper count limit<br>Lower count limit ≥  upper count limit<br>lower count limit  0 |
| | **Bit 4:** Invert-DI+latch-retr. not perm.<br>It is not permitted to invert the level of the digital input when using the latch-retrigger-function |
| | **Bit 5:** Main count direction wrong<br>Wrong parameter value for defining the main count direction (Object $5800_{hex}$ – Encoder Basic Mode (page 11-6), byte 1, bit 5 and 6).<br>Permissible values:<br>00 = none<br>01 = up<br>10 = down |
| OPER._MODE | Wrong parameter value for defining the operation mode (Object $5800_{hex}$ – Encoder Basic Mode (page 11-6), byte 0, bit 0-5). |
| **Measurement mode** | |
| ERR_24Vdc | see above |
| ERR_DO | see above |
| ERR_PARA | **Bit 2:** Sensor pulse wrong |
| | |
| | **Bit 3:** Integration time wrong<br>The value for the integration time is incorrect Object $5820_{hex}$ – Measuring Integration Time (page 11-23).<br>Permissible value range: 1 to 1000 |
| | **Bit 4:** Upper limit wrong<br>The value for the upper limit time is incorrect . Permissible value range: 1 to 16777215 |
| ERR_PARA | **Bit 5:** Lower limit wrong<br>The value for the lower limit time is incorrect . Permissible value range: 0 to 16777214 |
| OPER._MODE | Wrong parameter value for defining the operation mode (Object $5800_{hex}$ – Encoder Basic Mode (page 11-6), byte 0, bit 0-5). |

*Table 184: Meaning of the diagnostic bits of BL20-1CNT-24VDC, alphabetically sorted*

| Designation | Description |
|---|---|
| MEAS._MODE | This message is always shown in conjunction with other diagnostics messages and indicates that messages refer to an active measurement mode. |

### 11.6.3 Meaning of the diagnostic bits of BL20-E-2CNT-2PWM

*Table 185: Diagnostics of the BL20-E-2CNT-2PWM*

| Diagnostic message | Values | Meaning |
|---|---|---|
| CNT1_PAR_ERR, CNT2_PAR_ERR, PWM1_PAR_ERR, PWM2_PAR_ERR | 0 | Parameter set of function unit correct |
| | 1 | Faulty / inconsistent parameters, wrong parameterization |
| P1_DIAG, P2_DIAG, D1_DIAG, D2_DIAG | 0 | No diagnostic message |
| | 1 | Diagnosis pending at channel (short circuit) |
| HW_ERR | 0 | No diagnostic message |
| | 1 | "Hardware error"<br>Display of common errors of the module's hardware (e.g. CRC-error, adjustment error….).<br>Change of device necessary. |

### 11.6.4 Meaning of the diagnostic bits of BL67-1CNT/ENC

*Table 186: Meaning of the diagnostic bits of BL67-1CNT/ENC, alphabetically sorted*

| Diagnostic message | Values | Meaning |
|---|---|---|
| STS_OFLW | 0 | no overflow |
| | 1 | Counter value exceeded upper limit of counter range. |
| STS_UFLW | 0 | no underflow |
| | 1 | Counter value below lower limit of counter range. |
| DIA_DOx | 0 | output OK |
| | 1 | Short circuit or overload at output x |

*Table 186: Meaning of the diagnostic bits of BL67-1CNT/ ENC, alphabetically sorted*

| Diagnostic message | Values | Meaning |
|---|---|---|
| ERR_PARA | 0 | The last change in parameters is valid. |
| | 0 | Faulty/ inconsistent parameter data. |

# 12 Objects for SWIRE modules


**12.1 Motor starter modules BL20 ..... 2**

**12.2 General object overview for SWIRE modules ..... 2**

12.2.1 Representation of process input data ..... 3
– Process input ..... 3
12.2.2 Representation of process output data ..... 4
– Process output ..... 5
12.2.3 Representation of diagnostic data ..... 6
12.2.4 Representation of parameter data ..... 10
– Object $3064_{hex}$ – XBI Param Dword ..... 13
– Object $3065_{hex}$ – XBI Param Dword ..... 14
– Object $3066_{hex}$ – XBIParam Dword3 ..... 14
– Object $3067_{hex}$ – XBIParam Dword4 ..... 14
– Object $3068_{hex}$ – XBIParam Dword5 ..... 15
– Object $3069_{hex}$ – XBIParam Dword6 ..... 15

## 12.1 Motor starter modules BL20

The objects are used in the following modules:

*Table 187: BL20 - motor starter modules*

| Product family | Module |
|---|---|
| BL20 | BL20-E-1SWIRE |

## 12.2 General object overview for SWIRE modules

*Table 188: General object overview for SWIRE modules*

| Object | Name | page |
|---|---|---|
| **Input data** | | |
| $6000_{hex}$ | Read Input 8 bit | page 5-4 |
| ... | | |
| $6120_{hex}$ | Read Input 32 bit | page 5-6 |
| Output data | | |
| $6200_{hex}$ | Write Output 8 bit | page 6-4 |
| ... | | |
| $6320_{hex}$ | Write Output 32 bit | page 6-12 |
| **Diagnostic data** | | |
| $3040_{hex}$ | XBIDiag Byte | page 14-8ff. |
| $3042_{hex}$ | XBIDiag Word | |
| $3044_{hex}$ | XBIDiag Dword | |
| $3045_{hex}$ | XBIDiag Dword2 | |
| $3046_{hex}$ | XBIDiag Dword3 | |
| $3047_{hex}$ | XBIDiag Dword4 | |
| **Parameter data** | | |
| $3060_{hex}$ | XBIParam Byte | page 14-9ff. |
| $3062_{hex}$ | XBIParam Word | |
| $3064_{hex}$ | XBIParam DWord | |
| $3065_{hex}$ | XBIParam DWord2 | |
| ... | | |
| $306B_{hex}$ | XBIParam Dword8 | |

### 12.2.1 Representation of process input data

The process input data of the SWIRE modules is represented by objects $6000_{hex}$, $6020_{hex}$, $6021_{hex}$, and $6022_{hex}$, $6100_{hex}$ or $6120_{hex}$ for **digital input channels** of a station.

The user can choose between 4 different formats for representing the values:

- Only one bit represented by each SUB-index (objects $6020_{hex}$, $6021_{hex}$ and $6022_{hex}$), see page . page 5-5
- 8 bits represented by each SUB-index (object $6000_{hex}$), page 5-5
- 16 bits represented by each SUB-index (object $6100_{hex}$), page 5-5
- 32 bits represented by each SUB-index (object $6120_{hex}$), page 5-5

PDO mapping of object $6000_{hex}$ is always executed by default and automatically for the first 8 sub-indexes. This corresponds to the 64 bits of the SWIRE feedback interface. If more than 64 process input data items are present, the PDO mapping must be carried out by the user. A total of 36 groups of 8 bits each can be displayed (288 bits).

If the representation of the data bits is **not** required with the object $6000_{hex}$ or if another arrangement of the mapped objects is required, PDO mapping must be carried out by the user.

**Process input**

The field input data is transferred from the connected SWIRE-BUS to the SWIRE module. The process input data is the data that is transferred by the SWIRE module via a gateway to the PLC. The transfer is carried out in 8-byte format. 4 bits are reserved for each SWIRE slave. The following information can be transferred:

- Contactor coil on/off
- Motor-protective circuit-breaker off or tripped/ on
- Status of the slave o.k./diagnostics message present

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0** | SWIRE-Slave 2 | | | | SWIRE-Slave 1 | | | |
| **Byte 1** | SWIRE-Slave 4 | | | | SWIRE-Slave 3 | | | |
| **Byte 2** | SWIRE-Slave 6 | | | | SWIRE-Slave 6 | | | |
| **Byte 3** | SWIRE-Slave 8 | | | | SWIRE-Slave 7 | | | |
| **Byte 4** | SWIRE-Slave 10 | | | | SWIRE-Slave 9 | | | |
| **Byte 5** | SWIRE-Slave 12 | | | | SWIRE-Slave 11 | | | |
| **Byte 6** | SWIRE-Slave 14 | | | | SWIRE-Slave 13 | | | |
| **Byte 7** | SWIRE-Slave 16 | | | | SWIRE-Slave 15 | | | |

The data of SWIRE slave 1 is the data of the first physical slave on the SWIRE bus. The remaining slaves are assigned in consecutive order accordingly. The meaning of the data of an SWIRE slave depends on the product concerned.

Meaning of the 4-bit process input data on an SWIRE-DIL device:

| Bit 7 | Bit 6 | Bit 5 | Bit 4 |
|---|---|---|---|
| SDx / free | free | PKZSTx | SIx |

The following table shows the meaning of the data bits:

*Table 189: Meaning of the data bits*

| Designation | Status | Comment | |
|---|---|---|---|
| **SIx** | **Switch status, relay x** | | |
| | SIx supplies the switch status of the contactor coil of the SWIRE bus slave as a feedback signal.<br>SIx makes it possible to check whether the set switch status was executed by a mechanical connection. This must take into account the time delay between the setting of an output, a mechanical execution and the subsequent feedback signal. | | |
| | 0 | OFF | Contactor coil is switched off |
| | 1 | ON | Contactor coil is switched on |
| **PKZSTx** | **Switch status, PKZ x** | | |
| | 0 | OFF | The motor-protective circuit breaker is off or has tripped |
| | 1 | ON | The motor-protective circuit breaker is switched on |
| **SDx** | **Communication error, slave x** | | |
| | Setting the NDDIAG parameter copies the slave diagnostics message (input byte 1/bit 3) to the feed-back interface. The information is provided as status information in the PLC for the user. | | |
| | 0 | ON LINE | Status of slave x: |
| | 1 | OFF LINE | Status of slave x: diagnostics available |

### 12.2.2 Representation of process output data

The process input data of the SWIRE modules is represented by objects $6200_{hex}$, $6220_{hex}$, $6221_{hex}$, and $6222_{hex}$, $6300_{hex}$ or $6320_{hex}$ for **digital input channels** of a station.

The user can choose between 4 different formats for representing the values:

- Only one bit represented by each SUB-index (objects $6220_{hex}$, $6221_{hex}$ and $6222_{hex}$), page 6-7 ff.
- 8 bits represented by each SUB-index (object $6200_{hex}$), page 6-4.
- 16 bits represented by each SUB-index (object $6300_{hex}$), page 6-10.
- 32 bits represented by each SUB-index (object $6320_{hex}$), page 6-12.

PDO mapping of object $6200_{hex}$ is always executed by default and automatically for the first 8 sub-indexes. This corresponds to the 64 bits of the SWIRE control interface. If more than 64 process output data items are present, the PDO mapping must be carried out by the user. A total of 36 groups of 8 bits each can be displayed (288 bits).

If the representation of the data bits is **not** required with the object $6200_{hex}$ or if another arrangement of the mapped objects is required, PDO mapping must be carried out by the user.

### Process output

Field output data is output from an SWIRE module to a field device. The process output data is the data that is transferred by the PLC via a gateway to the SWIRE module. The transfer is carried out in 8-byte format. 4 bits are reserved for each SWIRE slave. The following information is transferred:

Switch status of contactor coil on/off

| | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Byte 0** | SWIRE-Slave 2 | | | | SWIRE-Slave 1 | | | |
| **Byte 1** | SWIRE-Slave 4 | | | | SWIRE-Slave 3 | | | |
| **Byte 2** | SWIRE-Slave 6 | | | | SWIRE-Slave 6 | | | |
| **Byte 3** | SWIRE-Slave 8 | | | | SWIRE-Slave 7 | | | |
| **Byte 4** | SWIRE-Slave 10 | | | | SWIRE-Slave 9 | | | |
| **Byte 5** | SWIRE-Slave 12 | | | | SWIRE-Slave 11 | | | |
| **Byte 6** | SWIRE-Slave 14 | | | | SWIRE-Slave 13 | | | |
| **Byte 7** | SWIRE-Slave 16 | | | | SWIRE-Slave 15 | | | |

The data of SWIRE slave 1 is the data of the first physical slave on the SWIRE bus. The remaining slaves are assigned in consecutive order accordingly. The meaning of the data of an SWIRE slave depends on the product concerned.

Meaning of the 4-bit process output data on an SWIRE-DIL device:

| Bit 7 | Bit 6 | Bit 5 | Bit 4 |
|---|---|---|---|
| free | free | free | SOx |

The following table shows the meaning of the data bits:

*Table 190: Meaning of the data bits*

| Designation | Status | Comment |
|---|---|---|
| **SOx** | **relay x**<br>**relay x** | |
| | SOx is transferred as the switch status of the contactor coil from the SWIRE bus master to the appropriate SWIRE bus slave. | |
| | 0 | OFF Contactor not switched on |
| | 1 | ON Contactor switched on |

### 12.2.3 Representation of diagnostic data

The 8 bytes of diagnostics data for the SWIRE module are represented by the manufacturer specific objects $3044_{hex}$, „XBIDiag Dword“ and $3045_{hex}$, „XBIDiag Dword2“ of the CANopen gateway.

A sub-index of these objects can represent a maximum number of 4 bytes. Larger diagnostics data volumes are distributed on the following objects. The sub-index also belongs to the module number.

The following table shows the assignment of the parameter data bytes of the SWIRE-module to the manufacturer specific objects:

| **Byte-No.** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8 Diagnostic bytes of the SWIRE-module | | | | | | | | free | | | | | | | |
| **Object-no. in (hex)** | 3044 (see page 14-8) | | | | 3045 (see page 14-8) | | | | 3046 | | | | 3047 | | | |

Use the objects $3044_{hex}$ and $3045_{hex}$ to access all the diagnostics data. The **sub-index** of the objects has to correspond to the slot-number of the module in the station.

| | **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
|---|---|---|---|---|---|---|---|---|
| Byte 1 | GENERAL $_{ERR}$ | $U_{SWERR}$ | free | $COM_{ERR}$ | free | $RDY_{ERR}$ | free | $SW_{ERR}$ |
| Byte 2 | free | $U_{AUXERR}$ | $TYP_{ERR}$ | free | $PKZ_{ERR}$ | free | $SD_{ERR}$ | free |
| **$TYP_{ERR}$ field** | | | | | | | | |
| Byte 3 | $TYP_{ERR}$ | $TYP_{ERR}S7$ | $TYP_{ERR}S6$ | $TYP_{ERR}S5$ | $TYP_{ERR}S4$ | $TYP_{ERR}S3$ | $TYP_{ERR}S2$ | $TYP_{ERR}S1$ |
| Byte 4 | $TYP_{ERR}$'S16 | $TYP_{ERR}S15$ | $TYP_{ERR}S14$ | $TYP_{ERR}S13$ | $TYP_{ERR}S12$ | $TYP_{ERR}S11$ | $TYP_{ERR}S10$ | $TYP_{ERR}S9$ |
| **Slave diagnostics bit field** | | | | | | | | |
| Byte 5 | $SD_{ERR}S8$ | $SD_{ERR}S7$ | $SD_{ERR}S6$ | $SD_{ERR}S5$ | $SD_{ERR}S4$ | $SD_{ERR}S3$ | $SD_{ERR}S2$ | $SD_{ERR}S1$ |
| Byte 6 | $SD_{ERR}S16$ | $SD_{ERR}S15$ | $SD_{ERR}S14$ | $SD_{ERR}S13$ | $SD_{ERR}S12$ | $SD_{ERR}S11$ | $SD_{ERR}S10$ | $SD_{ERR}S9$ |
| **PKZ field** | | | | | | | | |
| Byte 7 | $PKZ_{ERR}$ S8 | $PKZ_{ERR}$ S7 | $PKZ_{ERR}$ S6 | $PKZ_{ERR}$ S5 | $PKZ_{ERR}$ S4 | $PKZ_{ERR}$ S3 | $PKZ_{ERR}$ S2 | $PKZ_{ERR}$ S1 |
| Byte 8 | $PKZ_{ERR}$ S16 | $PKZ_{ERR}$ S15 | $PKZ_{ERR}$ S14 | $PKZ_{ERR}$ S13 | $PKZ_{ERR}$ S12 | $PKZ_{ERR}$ S11 | $PKZ_{ERR}$ S10 | $PKZ_{ERR}$ S9 |

The following table shows the meaning of the diagnostic bits:

| Byte | Design. | Status | | Comment |
|---|---|---|---|---|
| Byte 1 | $SW_{ERR}$ | **SWIRE MASTER** | | |
| | | The configuration was accepted according to the parameter setting and the SWIRE bus is in data exchange mode. | | |
| | | 0 | Data exchange | The bus is in data exchange mode |
| | | 1 | Offline | The configuration was not accepted, the bus does not switch to data exchange mode. (SW LED flashing) |
| | $RDY_{ERR}$ | **SPS SLAVE** | | |
| | | Parameter setting is faulty. The ACTUAL configuration was accepted according to the SET configuration and the data exchange with the higher-level is o.k. | | |
| | | 0 | Data exchange | The bus is in data exchange mode |
| | | 1 | Offline | The configuration was not accepted, the bus does not switch to data exchange mode. (Rdy LED flashing) |
| | $COM_{ERR}$ | **Communication SWIRE** | | |
| | | A communication error is present, such as a slave is no longer reached, its internal timeout has elapsed or communication is faulty. The master cannot carry out data exchange with at least one slave. | | |
| | | 0 | OK | No error present. |
| | | 1 | faulty | An error is present |
| | $U_{SWERR}$ | **Voltage $U_{SW}$** | | |
| | | Voltage fault in $U_{SW}$, voltage (17 V DC) for supplying the SWIRE slaves | | |
| | | 0 | OK | No error present. |
| | | 1 | under voltage | An error is present |
| | GENE-RAL$_{ERR}$ | **Error message** | | |
| | | The creation of a function block shows that systems/function blocks for the general checking of a slave for any diagnostics messages present only check the first byte. | | |
| | | 0 | None | No diagnostics message present |
| | | 1 | present | One/several diagnostics messages present |
| Byte 2 | $SD_{ERR}$ | **Communication SWIRE slave** | | |
| | | If the parameter $SD_{ERR}$A is set for group diagnostics, this bit indicates an error as soon as only one slave on the bus sets its SD error bit. | | |
| | | 0 | OK | No error is present or diagnostics function has been deactivated via the parameter setting. |
| | | 1 | faulty | An error is present |

| Byte | Design. | Status | | Comment |
|---|---|---|---|---|
| | $PKZ_{ERR}$ | **Overcurrent protective circuit-breaker** | | |
| | | If the parameter $PKZ_{ERR}A$ is set for group diagnostics, this bit indicates an error as soon as only one PKZ of a slave has tripped. | | |
| | | 0 | OK | No PKZ has tripped or diagnostics function has been deactivated via the parameter setting. |
| | | 1 | tripping | At least one PKZ has tripped. |
| | $TYP_{ERR}$ | **Configuration** | | |
| | | If the $TYP_{ERR}$ parameter is set for group diagnostics, this bit indicates an error as soon as the ACTUAL configuration of a slave does not match the SET configuration for this position. | | |
| | | 0 | OK | The ACTUAL configuration fully matches the SET configuration or diagnostics function has been deactivated via the parameter. |
| | | 1 | faulty | The actual configuration does not fully match set configuration. |
| | $U_{AUXERR}$ | $_{\text{Voltage U}}$**AUX** | | |
| | | If the $U_{AUXERR}A$ parameter is activated, $_{AUXERR}$ will generate an error message as soon as the power supply goes below the level at which the function of the relays is not guaranteed. | | |
| | | 0 | OK | Contactor supply voltage is o.k. (> 20 VDC) or diagnostics function has been deactivated via this parameter. |
| | | 1 | under voltage | Contactor supply voltage is not o.k. (< 18 VDC). |
| Byte 3.4 | $TYP_{ERR}Sx$ | **Device configuration, slave x** | | |
| | | Info field for the individual indication of a configuration error as error message. If the $TYP_{INFO}A$ parameter has been set for single diagnostics, this bit field indicates the error, as soon as the ACTUAL configuration of the slave was not accepted and is therefore not enabled for data exchange. The diagnostics LED of the slave flashes. | | |
| | | 0 | OK | No configuration error is present and the slave is in data exchange mode or diagnostics function has been deactivated via the parameter setting. |
| | | 1 | incorrect | No configuration error present and the slave is NOT in data exchange mode. |
| Byte 5.6 | $SD_{ERR}Sx$ | **Communication, slave x** | | |
| | | Info field for the individual indication of the release of the slave diagnostics as error message. If the $SD_{INFO}A$ is set for single diagnostics, this bit field indicates the error as soon as the slave diagnostic message of the slave Sx is triggered. | | |
| | | 0 | OK | No error is present or diagnostics function has been deactivated via the parameter setting. |
| | | 1 | Offline | A diagnostics message is present. |

| Byte | Design. | Status | | Comment |
|---|---|---|---|---|
| Byte 7.8 | $PKZ_{ERR}Sx$ | **Overcurrent protective circuit-breaker, slave x** | | |
| | | Info field for the individual indication of the tripping a motor-protective circuit-breaker (PKZ) as error message. If the $PKZ_{INFO}A$ is set for single diagnostics, this bit field indicates the error as soon as the PKZ of the slave Sx has tripped. | | |
| | | 0 | OK | The PKZ of the slave has not tripped or diagnostics function has been deactivated via the parameter setting. |
| | | 1 | tripped | The PKZ of the slave has tripped. |

### 12.2.4 Representation of parameter data

The 24 bytes of parameter data for the SWIRE module are represented by the manufacturer specific objects $3064_{hex}$, „XBIParam Dword“ to $3069_{hex}$, „XBIParam Dword6“ of the CANopen gateway.

A sub-index of these objects can represent a maximum number of 4 bytes. Larger parameter data volumes are distributed on the following objects. The sub-index also belongs to the module number.

The following table shows the assignment of the parameter data bytes of the SWIRE module to the manufacturer specific objects:

| **Byte no.** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
|---|---|---|---|---|---|---|---|---|
| Object no. | Parameter bytes 0 to 7 of the SWIRE-module | | | | | | | |
| | $3064_{hex}$ (see page 14-10) | | | | $3065_{hex}$ (see page 12-13) | | | |
| **Byte no.** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
| Object no. | Parameter bytes 8 to 15 of the SWIRE-module | | | | | | | |
| | $3066_{hex}$ (see page 12-13) | | | | $3067_{hex}$ (see page 12-13) | | | |
| **Byte no.** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** |
| Object no. | Parameter bytes 16 to 23 of the SWIRE-module | | | | | | | |
| | $3068_{hex}$ (see page 12-13) | | | | $3069_{hex}$ (see page 12-13) | | | |

Use the objects $3064_{hex}$ and $3069_{hex}$ to access all the parameter data. The sub-index of the objects has to correspond to the slot-number of the module in the station.

The structure of the parameter data will be described in the following:

| | **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
|---|---|---|---|---|---|---|---|---|
| **Byte 1** | reserved | free | free | free | MNA | Configuration | Disable Cfg | free |
| **Byte 2** | free | $U_{AUXERR}$ | $TYP_{ERR}$ | $TYP_{INFO}$ | $PKZ_{ERR}$ | $PKZ_{INFO}$ | $SD_{ERR}$ | $SD_{INFO}$ |
| **Byte 3** | reserved | | | | | | | |
| **Byte 4** | Life guarding time | | | | | | | |
| **Byte 5** | $SD_{DIAG}S8$ | $SD_{DIAG}S7$ | $SD_{DIAG}S6$ | $SD_{DIAG}S5$ | $SD_{DIAG}S4$ | $SD_{DIAG}S3$ | $SD_{DIAG}S2$ | $SD_{DIAG}S1$ |
| **Byte 6** | $SD_{DIAG}S16$ | $SD_{DIAG}S15$ | $SD_{DIAG}S14$ | $SD_{DIAG}S13$ | $SD_{DIAG}S12$ | $SD_{DIAG}S11$ | $SD_{DIAG}S10$ | $SD_{DIAG}S9$ |
| **Byte 7** | reserved | | | | | | | |
| **Byte 8** | reserved | | | | | | | |
| **Byte 9 - 24** | Type designation slave 1 - 16 | | | | | | | |

The following table shows the meaning of the parameter bits:

*Table 191: Module parameters*

| Parameter name | Value | |
|---|---|---|
| **Byte 1** | | |
| Disable Cfg | Disabling of the acceptance of the physically present configuration as ACTUAL configuration on manual pushbutton actuation. | |
| | 0 = inactive **A** | The physically present configuration of the SWIRE bus is only accepted as the ACTUAL configuration by pressing the CFG button. The comparison with the SET configuration is then carried out. |
| | 1 = active | The physically present configuration is automatically accepted as the ACTUAL configuration and then compared with the SET configuration. |
| Configuration | PLC configuration check<br>The configuration check parameter enables a comparison of the set and actual configuration based on the device ID. | |
| | 0 = active **A** | Configuration check based on device ID. Only SWIRE slaves with a device ID completely matching the set configuration are accepted on the bus. |
| | 1 = inactive | All slaves are mapped in 4Bit INPUT/ 4Bit OUTPUT without checking the device ID. |
| MNA active/ passive | Configuration check<br>If the ACTUAL configuration of the SWIRE bus does not match the SET configuration, the master only exchanges data with the correctly configured and functional slaves. | |
| | 0 = Bus based **A** | No data exchange with a slave with an incomplete / incorrect configuration. |
| | 1 = Slave based | The bus also goes into operation with the correctly configured slaves even if the configuration is incomplete. All slaves detected by the daisy chain configuration with a position that matches the set configuration are started up. Slaves that do not match the set configuration are inactive. |
| $SD_{INFO}$ | Slave error field<br>Activate slave diagnostics info field $SD_{ERR}Sx$. As soon as a slave on the bus sets its error bit, this is indicated individually as an error depending on the parameter setting. | |
| | active | Single diagnostics is activated |
| | inactive | Single diagnostics is not activate |

**A***Default-setting*

*Table 191: Module parameters*

**A***Default-setting*

| Parameter name | Value | |
|---|---|---|
| **Byte 2** | | |
| $SD_{ERR}$ | Group error - slave error<br>Activate slave diagnostics $SD_{ERR}$Sx. As soon as only one slave on the bus sets its error bit, this is indicated as a group error depending on the parameter setting. | |
| | 0 = active **A** | Group diagnostics is activated |
| | 1 = inactive | Group diagnostics is not activated |
| $PKZ_{INFO}$ | PKZ error field<br>Activate slave diagnostics info field $PKZ_{ERR}$Sx. As soon as a slave on the bus clears its PKZ bit, this is indicated as an individual error depending on the parameter setting. | |
| | 0 = active **A** | Single diagnostics is activated |
| | 1 = inactive | Single diagnostics is not activate |
| $PKZ_{ERR}$ | Group PKZ error field<br>Activate slave diagnostics $PKZ_{ERR}$Sx. As soon as a slave on the bus clears its PKZ bit, this is indicated as an individual error depending on the parameter setting. | |
| | 0 = active **A** | Single diagnostics is activated |
| | 1 = inactive | Single diagnostics is not activate |
| $TYP_{INFO}$ | Configuration error field<br>As soon as a slave on the bus does not match the set configuration and therefore cannot be started, this is indicated as an individual error depending on the parameter set. | |
| | 0 = active **A** | Single diagnostics is activated |
| | 1 = inactive | Single diagnostics is not activate |
| $TYP_{ERR}$ | Group configuration error field<br>Activate slave diagnostics $TYP_{ERR}$Sx. As soon as only one slave on the bus is incorrectly configured, this is indicated as an error depending on the parameter setting. | |
| | 0 = active **A** | Group diagnostics is activated |
| | 1 = inactive | Group diagnostics is not activated |
| $U_{AUXERR}$ | Error message Voltage $U_{AUX}$<br>Activate system diagnostics $U_{AUXERR}$. $U_{AUXERR}$ will generate an error message as soon as the power supply goes below a level at which the function of the relays is not guaranteed. | |
| | 0 = active **A** | Error message $U_{AUXERR}$ activated |
| | 1 = inactive | Error message $U_{AUXERR}$ not activated |
| **Byte 3** | reserved | |

*Table 191: Module parameters*

| Parameter name | Value | |
|---|---|---|
| **Byte 4** | | |
| Lifeguarding | 0x02-0xFF<br>0x64 **A** | Lifeguarding time of the SWIRE slaves<br>Setting of lifeguarding time of SWIRE slaves , timeout time up to automatic reset of the slaves in the event of communication failure. (n × 10ms) (Default 1s)<br>0xFF: Lifeguarding off |
| **Byte 5, 6** | | |
| $SD_{DIAG}Sx$ | Input bit communication error, slave x<br>Slave diagnostics message from Byte 1 / Bit 7 is accepted in the feedback interface as Bit 4 | |
| | 0 = active **A** | $SD_{DIAG}Sx$ is accepted |
| | 1 = inactive | $SD_{DIAG}Sx$ is not accepted |
| **Byte 7, 8** | reserved | |
| **Byte 9 to 24** | | |
| Device ID, slave x | TYPE setting for the LIN slave at position x on the SWIRE bus | |
| | 0x20 | SWIRE-DIL-MTB (: 0xFF) |
| | 0xFF | Basic setting (no slave) |

### Object $3064_{hex}$ – XBI Param Dword

The object XBIParam Dword reads the first Param-Dword (byte 0...3) of a module on the module bus. The sub-index 1 to 74 corresponds to the module number of the module in the station.

*Table 192: Object $3064_{hex}$*

| Feature | Description |
|---|---|
| Name | XBIParam Dword |
| Object code | ARRAY |
| PDO mapping | No |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |

### Object $3065_{hex}$ – XBI Param Dword

The object XBIParam Dword reads the first Param-Dword (byte 4..0.7) of a module on the module bus. The sub-index 1 to 74 corresponds to the module number of the module in the station.

*Table 193: Object $3065_{hex}$*

| Feature | Description |
|---|---|
| Name | XBIParam Dword2 |
| Object code | ARRAY |
| PDO mapping | No |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |

### Object $3066_{hex}$ – XBIParam Dword3

The object XBIParam Dword3 reads the third Param-Dword (byte 8...11) of a module on the module bus. The sub-index 1 to 74 corresponds to the module number of the module in the station.

*Table 194: Object $3066_{hex}$*

| Feature | Description |
|---|---|
| Name | XBIParam Dword3 |
| Object code | ARRAY |
| PDO mapping | No |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |

### Object $3067_{hex}$ – XBIParam Dword4

The object XBIParam Dword4 reads the fourth Param-Dword (byte 12...15) of a module on the module bus. The sub-index 1 to 74 corresponds to the module number of the module in the station.

*Table 195: Object $3067_{hex}$*

| Feature | Description |
|---|---|
| Name | XBIParam Dword4 |
| Object code | ARRAY |
| PDO mapping | No |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |

### Object $3068_{hex}$ – XBIParam Dword5

The object XBIParam Dword5 reads the fifth Param-Dword (byte 16...19) of a module on the module bus. The sub-index 1 to 74 corresponds to the module number of the module in the station.

*Table 196: Object $3068_{hex}$*

| Feature | Description |
|---|---|
| Name | XBIParam Dword5 |
| Object code | ARRAY |
| PDO mapping | No |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |

### Object $3069_{hex}$ – XBIParam Dword6

The object XBIParam Dword6 reads the first Param-Dword (byte 20...24) of a module on the module bus. The sub-index 1 to 74 corresponds to the module number of the module in the station.

*Table 197: Object $3069_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | XBIParam Dword6 |
| Object code | ARRAY |
| PDO mapping | No |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |

# 13 Objects for RFID-modules


13.1 RFID-S-module BL20 ..... 2

13.2 General object overview for RFID-S-modules ..... 2

13.3 Object $5700_{hex}$ - 8 byte process input data ..... 2

13.4 Object $5701_{hex}$ - 12 byte process input data ..... 3

13.5 Object $5702_{hex}$ - 8 byte process output data ..... 3

13.6 Object $5703_{hex}$ - 12 byte process output data ..... 4

13.7 Object $5708_{hex}$ - 1 byte status messages ..... 4

13.8 Object $5722_{hex}$ - 1 byte parameter ..... 5

## 13.1 RFID-S-module BL20

The objects are used in the following modules:

*Table 198: BL20-RFID-S*

| Product family | Module |
|---|---|
| BL20 | BL20-2RFID-S |

## 13.2 General object overview for RFID-S-modules

*Table 199: General object overview for RFID-S-modules*

| Object | Name | page |
|---|---|---|
| $5700_{hex}$ | 8 byte process input data | Seite 13-2 |
| $5701_{hex}$ | 12 byte process input data | Seite 13-3 |
| $5702_{hex}$ | 8 byte process output data | Seite 13-3 |
| $5703_{hex}$ | 12 byte process output data | Seite 13-4 |
| $5708_{hex}$ | 1 byte status messages | Seite 13-4 |
| $5722_{hex}$ | 1 byte parameters | Seite 13-5 |

## 13.3 Object $5700_{hex}$ - 8 byte process input data

The first 8 bytes of the process input data of a BL ident®-channel are displayed in Object $5700_{hex}$.

Thus the range of the read-data is limited to 4byte. The 8 byte transfer format consists of the following:

- 1 byte status messages
- 2 byte error code
- 1 byte reserved
- 4 byte read data

*Table 200: Object $5700_{hex}$*

| Feature | Description |
|---|---|
| Name | RS232/RS4xx parameters |
| Object code | ARRAY |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |
| PDO mapping | Yes |

## 13.4 Object 5701$_{hex}$ - 12 byte process input data

The 12 byte of the "Process Input Data" of BL ident ®-channel are displayed in object $_{hex}$.

The 12 byte transfer format consists of the following:

- 1 byte status messages
- 2 byte error code
- 1 byte reserved
- 8 byte read data

*Table 201: Object 5701$_{hex}$*

| Feature | Description |
|---|---|
| Name | RFID Input Data Segmented |
| Object code | ARRAY |
| Data type | Domain |
| Access | ro |
| Default value | No |
| PDO mapping | No |

## 13.5 Object 5702$_{hex}$ - 8 byte process output data

The 8 byte of the "Process output Data" of BL ident ®-channel are displayed in object 5702$_{hex}$.

The 8 byte transfer format consists of the following:

- 4 byte command and control bits
- 4 byte write data

*Table 202: Object 5702$_{hex}$*

| Feature | Description |
|---|---|
| Name | RFID Output Data U64 |
| Object code | ARRAY |
| Data type | Unsigned64 |
| Access | rw |
| Default value | No |
| PDO mapping | Yes |

## 13.6 Object $5703_{hex}$ - 12 byte process output data

The 12 byte of the "Process output Data" of BL ident ®-channel are displayed in object $5703_{hex}$.

The 12 byte transfer format consists of the following:

- 4 byte command and control bits
- 8 byte write data

*Table 203: Object $5703_{hex}$*

| Feature | Description |
|---|---|
| Name | RFID Output Data Segmented |
| Object code | ARRAY |
| Data type | Domain |
| Access | rw |
| Default value | No |
| PDO mapping | No |

## 13.7 Object $5708_{hex}$ - 1 byte status messages

1 byte of the "Process Input Data" of BL ident ®-channel is displayed in $5708_{hex}$. This byte contains all status messages (DONE, BUSY, ERROR...):

The 12 byte transfer format consists of the following:

*Table 204: Object $5708_{hex}$*

| Feature | Description |
|---|---|
| Name | RFID Status |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |
| PDO mapping | Yes |

## 13.8 Object $5722_{hex}$ - 1 byte parameter

The Object $5722_{hex}$ displays the parameter "Bypass time" to a BL ident ®-channel.

This parameter setting is only needed when the error message "Dwell period of the tag in the detection range was not sufficient for successful command processing." appears at start-up.

*Table 205: Object $5722_{hex}$*

| Feature | Description |
|---|---|
| Name | RFID Bypass Time Parameter |
| Object code | ARRAY |
| Data type | Unsigned8 |
| Access | rw |
| Default value | No |
| PDO mapping | No |

# 14 Manufacturer specific objects


**14.1 General ..... 2**

14.1.1 Module related manufacturer specific objects ..... 2
– Address assignment ..... 3
14.1.2 Slot-related manufacturer specific objects ..... 4
– Address assignment ..... 5
14.1.3 Process input objects ..... 5
– Object $3000_{hex}$ - XBIInputByte ..... 5
– Object $3002_{hex}$ - XBIInputWord ..... 6
– Object $3004_{hex}$ - XBIInputDWord0 to Object $300B_{hex}$- XBIInputDWord8 ..... 6
14.1.4 Process output objects ..... 6
– Object $3020_{hex}$- XBIOutputByte ..... 7
– Object $3022_{hex}$ - XBIOutputWord ..... 7
– Object $3024_{hex}$ - XBIOutputDWord0 to Object $302B_{hex}$- XBIOutputDWord8 ..... 8
14.1.5 Diagnosis objects ..... 9
– object $3040_{hex}$ – XBIDiag Byte ..... 9
– object $3042_{hex}$ – XBIDiagWord ..... 9
– Object $3044_{hex}$ – XBIDiagDword to Object $3047_{hex}$ – XBIDiagDword4 ..... 9
14.1.6 Parameter objects ..... 10
– Objekt $3060_{hex}$ - XBIParamByte ..... 10
– Object $3062_{hex}$ - XBIParamWord ..... 10
– object $3064_{hex}$ – XBIParamDword to object $3069_{hex}$ – XBIParamDword8 ..... 11

## 14.1 General

The manufacturer specific objects used in BL×× can be divided into 2 groups:

- module-related manufacturer specific objects ((2000$_{hex}$ to 2FFF$_{hex}$ and 4000$_{hex}$ to 5FFF$_{hex}$)
- slot-related objects (3000$_{hex}$ to 3FFF$_{hex}$)

### 14.1.1 Module related manufacturer specific objects

The module-related objects (2000$_{hex}$ to 2FFF$_{hex}$ and 4000$_{hex}$ to 5FFF$_{hex}$) refer to special BL×× device types. They describe special functions for these types of devices which cannot be represented using the device profiles defined by the CiA.

*Table 206: Module-related manufacturer specific objects*

| Object no. | | Description |
|---|---|---|
| **Gateway** | | |
| 2000$_{hex}$ | Serial Number | see chapter 4: Device (gateway) objects |
| 2010$_{hex}$ | Node Reset Modifiers | |
| 2400$_{hex}$ | System Voltages | |
| 2401$_{hex}$ | System Voltages | |
| **Analog input modules** | | |
| 5420$_{hex}$ | Analog Input Mode | see chapter 8: Objects for analog input modules |
| **Analog outputs** | | |
| 5440$_{hex}$ | Analog Output Mode | see chapter 9: Objects for analog output modules |
| **RS232/485/422** | | |
| 5600$_{hex}$ | RS232/RS4xx parameters | see chapter 10: Objects for RS232/RS4xx modules |
| 5601$_{hex}$ | RS232/RS4xx RxD | |
| 5602$_{hex}$ | RS232/RS4xx TxD | |
| **Encoder (SSI, counter)** | | |
| 5801$_{hex}$ | Encoder Config | see chapter 11: Objects for Encoder modules (SSI, CNT) |
| 5802$_{hex}$ | Encoder Status | |
| 5803$_{hex}$ | Encoder Flags | |
| 5804$_{hex}$ | Encoder Diag | |
| 5805$_{hex}$ | Encoder Native Status | |

*Table 206: Module-related manufacturer specific objects*

| Object no. | | Description |
|---|---|---|
| $5806_{hex}$ | Encoder Optional Encoder | |
| $5808_{hex}$ | Encoder Control | |
| $5810_{hex}$ | Encoder Load Prepare Value | |
| $5811_{hex}$ | Encoder Pulse Width | |
| $5820_{hex}$ | Measuring Integration Time | |
| $5821_{hex}$ | Measuring Low Limit | |
| $5822_{hex}$ | Measuring High Limit | |
| $5823_{hex}$ | Measuring Units Per Revolution | |
| $5824_{hex}$ | Encoder Measuring Divisor | |
| $5825_{hex}$ | Encoder Measuring Factor | |
| $5827_{hex}$ | Encoder Measuring Time Out | |
| $5830_{hex}$ | Encoder Measuring Value | |
| $5831_{hex}$ | Encoder Latch Value | |
| $5840_{hex}$ | SSI Diag Mapping | |
| **PWM** | | |
| $5901_{hex}$ | PWM Config | see chapter 11: Objects for Encoder modules (SSI, CNT) |
| $5902_{hex}$ | PWM Status | |
| $5903_{hex}$ | PWM Flags | |
| $5904_{hex}$ | PWM Diag | |
| $5908_{hex}$ | PWM Control | |
| $5910_{hex}$ | PWM Load Prepare Value | |
| $5913_{hex}$ | PWM Duty Cycle | |
| $5920_{hex}$ | PWM Period Duration | |
| $5931_{hex}$ | PWM Latch Value | |

**Address assignment**

In module-related objects, all modules of one module type within one BL××-station are counted and considered as one module group. The sub-index of the respective module defines the position of the module within this group (not the slot number).

**Example:**

A BL20-station contains 3 module type RS232:

To address the parameters of the **2.** RS232 module in the station, object $5600_{hex,}$ Sub-Index **2** zu wählen.

### 14.1.2 Slot-related manufacturer specific objects

The slot related objects ($3000_{hex}$ to $3FFF_{hex}$) are available for **each** module type in the BL××-product families. They do **not** depend on the respective device type.

The slot-related objects have to be used for certain device types for which no additional manufacturer specific device-type-related objects are defined (e.g. for parameterization).

The following table shows the slot-related objects for the BL××-system and defines for which modules they **have to be** used.

*Table 207: Slot-related manufacturer specific objects*

| Object no. | | To be used in module-type |
|---|---|---|
| **Input objects** | | |
| $3000_{hex}$ | XBlInputByte | |
| $3002_{hex}$ | XBlInputWord | |
| $3004_{hex}$ to $300B_{hex,}$ | XBlInputDWord0 to XBlInputDWord8 | |
| **Output objects** | | |
| $3020_{hex,}$ | XBlOutputByte | |
| $3022_{hex,}$ | XBlOutputByte | |
| $3024_{hex}$ to $02B_{hex}$ | XBlOutputDWord0 to XBlOutputDWord8 | |
| **Diagnosis objects** | | |
| $3040_{hex,}$ | XBlDiagByte | SWIRE, page 12-6 |
| $3042_{hex,}$ | XBlDiagWord | |
| $3044_{hex}$ to 3 $04B_{hex}$ | XBlDiagDWord0 to XBlDiagDWord8 | |
| **Parameter objects** | | |
| $3060_{hex,}$ | XBlParamByte | SWIRE, page 12-10 |
| $3062_{hex,}$ | XBlParamWord | |
| $3064_{hex}$ to $306B_{hex}$ | XBlParamDWord0 to XBlParamDWord8 | xDI-NAMUR, page 14-10<br>xDI-xDO-PD, page 5-2<br>xXSG-PD, page 5-2<br>SWIRE, page 12-6 |
| **Gateway objects** | | |
| $3081_{hex}$ | XBlReferenceModuleType | |
| $3084_{hex}$ | XBlReferenceInputSize | |
| $3085_{hex}$ | XBlReferenceOutputSize | |
| $3086_{hex}$ | XBlReferenceDiagSize | |
| $3087_{hex}$ | XBlReferenceParamSize | |
| $3090_{hex}$ | XBlCurrentModuleId | |

| | |
|---|---|
| $3091_{hex}$ | XBlCurrentModuleType |
| $3094_{hex}$ | XBlCurrentInputSize |
| $3095_{hex}$ | XBlCurrentOutputSize |
| $3096_{hex}$ | XBlCurrentDiagSize |
| $3097_{hex}$ | XBlCurrentParamSize |

**Address assignment**

In the slot-related objects, the sub-index defines the slot at which the respective module is located within the BL××-station.

### 14.1.3 Process input objects

**Object $3000_{hex}$ - XBlInputByte**

*Table 208: Object $3000_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | XBlInputByte |
| Object code | VAR |
| Data type | Unsigned8 |
| Access | ro |
| PDO mapping | |
| – Sub-index 0 | No |
| – Sub-index 1 and 2 | Yes |

**Object $3002_{hex}$ - XBlInputWord**

*Table 209: Object $3002_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | XBlInputWord |
| Object code | VAR |
| Data type | |
| – Sub-index 0 | Unsigned8 |
| – Sub-index 1 and 2 | Unsigned16 |
| Access | ro |
| PDO mapping | |
| – Sub-index 0 | No |
| – Sub-index 1 and 2 | Yes |

**Object $3004_{hex}$ - XBIInputDWord0 to Object $300B_{hex}$- XBIInputDWord8**

*Table 210: Object $3004_{hex}$ to $300B_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | XBIInputDWordx |
| Object code | VAR |
| Data type | |
| – Sub-index 0 | Unsigned8 |
| – Sub-index 1 and 2 | Unsigned32 |
| Access | ro |
| PDO mapping | |
| – Sub-index 0 | No |
| – Sub-index 1 and 2 | Yes |

### 14.1.4 Process output objects

**Object $3020_{hex}$- XBIOutputByte**

*Table 211: Object $3020_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | XBIOutputByte |
| Object code | VAR |
| Data type | Unsigned8 |
| Access | rww |
| PDO mapping | |
| – Sub-index 0 | No |
| – Sub-index 1 and 2 | Yes |

### Object $3022_{hex}$ - XBIOutputWord

*Table 212: Object $3022_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | XBIOutputWord |
| Object code | VAR |
| Data type | |
| – Sub-index 0 | Unsigned8 |
| – Sub-index 1 and 2 | Unsigned16 |
| Access | rww |
| PDO mapping | |
| – Sub-index 0 | No |
| – Sub-index 1 and 2 | Yes |

### Object $3024_{hex}$ - XBIOutputDWord0 to Object $302B_{hex}$- XBIOutputDWord8

*Table 213: Object $3024_{hex}$ to $302B_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | XBIOutputDWordx |
| Object code | VAR |
| Data type | |
| – Sub-index 0 | Unsigned8 |
| – Sub-index 1 and 2 | Unsigned32 |
| Access | rww |
| PDO mapping | |
| – Sub-index 0 | No |
| – Sub-index 1 and 2 | Yes |

### 14.1.5 Diagnosis objects

**object 3040$_{hex}$ – XBIDiag Byte**

The object reads the module's first diagnostic byte on the module bus. The sub-index 1 to 74 corresponds to the slot number of the module in the BL××-station.

*Table 214: Object 3040$_{hex}$*

| Feature | Description |
|---|---|
| Name | XBIDiagByte |
| Object code | ARRAY |
| PDO mapping | No |
| Data type | Unsigned8 |
| Access | ro |
| Default value | No |

**object 3042$_{hex}$ – XBIDiagWord**

The object reads the module's first diagnostic word on the module bus. The sub-index 1 to 74 corresponds to the slot number of the module in the BL××-station.

*Table 215: Object 3042$_{hex}$*

| Feature | Description |
|---|---|
| Name | XBIDiagWord |
| Object code | ARRAY |
| PDO mapping | No |
| Data type | Unsigned16 |
| Access | ro |
| Default value | No |

**Object 3044$_{hex}$ – XBIDiagDword to Object 3047$_{hex}$ – XBIDiagDword4**

The object XBIDiagDword reads the first diagnostic-Dword (byte 0...3) of a module on the module bus, XBIDiagDword2 reads the second diagnostic-Dword (byte 4...7) etc.. The sub-index 1 to 74 corresponds to the slot number of the module in the BL××-station.

*Table 216: Object 3044$_{hex}$ to 3047$_{hex}$*

| Feature | Description |
|---|---|
| Name | XBIDiagDwordx |
| Object code | ARRAY |
| PDO mapping | No |
| Data type | Unsigned32 |
| Access | ro |

| Table 216: Object $3044_{hex}$ to $3047_{hex}$ | Feature | Description |
|---|---|---|
| | Default value | No |

### 14.1.6 Parameter objects

Objects $3060_{hex}$ „XBIParamByte“ to 306Bhex „XBIParamDWord8“ are used to parameterize the BL××-modules byte-by-byte, word-by-word or Dword-by-Dword.

All modules which have according to the CANopen-profile no parameter objects, **have to be** parameterized using these objects.



**Attention**

The sub-index corresponds to the slot-number of the BL××-module within this BL××-station.



**Note**

Objects $3000_{hex}$ to $3097_{hex}$ allow direct access to the internal module bus of the station.

**Objekt $3060_{hex}$ - XBIParamByte**

| Table 217: Object $3060_{hex}$ | Feature | Description/ Value |
|---|---|---|
| | Name | XBIParamByte |
| | Object code | ARRAY |
| | Data type | Unsigned8 |
| | Access | rw |
| | Default value | No |
| | PDO mapping | No |

**Object $3062_{hex}$ - XBIParamWord**

| Table 218: Object $3062_{hex}$ | Feature | Description/ Value |
|---|---|---|
| | Name | XBIParamWord |
| | Object code | ARRAY |
| | Data type | Unsigned16 |
| | Access | rw |
| | Default value | No |
| | PDO mapping | No |

### object $3064_{hex}$ – XBIParamDword to object $3069_{hex}$ – XBIParamDword8

*Table 219: object $3064_{hex}$ to $3069_{hex}$*

| Feature | Description/ Value |
|---|---|
| Name | XBIParam Dword |
| Object code | ARRAY |
| Data type | Unsigned32 |
| Access | rw |
| Default value | No |
| PDO mapping | No |

The structure of the parameter data depends on the respective module.

In the following, the data structure is described for each module-type for which the parameterization using this object is necessary:

- **BLxx-4DI-NAMUR**

*Table 220: Parameters BLxx-4DI-NAMUR*

**A***Default-setting*

| Byte | Bit | Name | Description |
|---|---|---|---|
| 0 | 0 | Input filter x | 0 = deactivated (input filter 0,25 ms) **A**<br>1 = activated (input filter 2,5 ms) |
| | 1 | Digital input x | 0 = normal **A**<br>1 = inverted |
| | 2 | Short-circuit monitoring x | 0 = deactivated **A**<br>1 = activated |
| 1 | 3 | Short circuit diagnosis x | 0 = deactivated **A**<br>1 = activated |
| | 4 | Open circuit monitoring x | 0 = deactivated **A**<br>1 = activated |
| | 5 | Open circuit diagnosis x | 0 = deactivated **A**<br>1 = activated |
| 2 | 6 | Input on diagnostic x | 0 = output substitute value **A**<br>1 = hold current value |
| | 7 | Substitute value on diag x | 0 = off **A**<br>1 = on |

### 4DI-PD

*Table 221: Parameters 4DI-PD*

**A***Default-setting*

| Byte | Bit | Name | Description |
|---|---|---|---|
| 0 | 0 | Input filter 0 | 0 = deactivated (input filter 0,25 ms) **A**<br>1 = activated (input filter 2,5 ms) |
| | ... | ... | |
| | 3 | Input filter 3 | |
| 1 | 0 | Digital input 0 | 0 = normal<br>1 = inverted |
| | ... | ... | |
| | 3 | Digital input 3 | |
| 2 | 0 | Operation Mode Group A | 0 = normal **A**<br>1 = open-circuit monitoring |
| | 1 | Operation Mode Group B | |

### 4DI-PD

*Table 222: Parameters 4DI-PD*

**A***Default-setting*

| Byte | Bit | Name | Description |
|---|---|---|---|
| 0 | 0 | Input filter 0 | 0 = deactivated (input filter 0,25 ms) **A**<br>1 = activated (input filter 2,5 ms) |
| | ... | ... | |
| | 3 | Input filter 7 | |
| 1 | 0 | Digital input 0 | 0 = normal<br>1 = inverted |
| | ... | ... | |
| | 3 | Digital input 7 | |
| 2 | 0 | Operation Mode Group A | 0 = normal **A**<br>1 = open-circuit monitoring |
| | ... | ... | |
| | 3 | Operation Mode Group D | |

### 4DI4DO-PD

*Table 223: Parameters 4DI4DO-PD*

**A***Default-setting*

| Byte | Bit | Name | Description |
|---|---|---|---|
| 0 | 0 | Input filter 0 | 0 = deactivated (input filter 0,25 ms) **A**<br>1 = activated (input filter 2,5 ms) |
| | ... | ... | |
| | 3 | Input filter 3 | |
| 1 | 0 | Digital input 0 | 0 = normal **A**<br>1 = inverted |
| | ... | ... | |
| | 3 | Digital input 3 | |
| 2 | 0 | Output on overcurrent 0 | 0 = automatic recovery **A**<br>1 = controlled recovery |
| | ... | ... | |
| | 3 | Output on overcurrent 3 | |

### 8XSG-PD

*Table 224: Parameters 8XSG-PD*

**A***Default-setting*

| Byte | Bit | Name | Description |
|---|---|---|---|
| 0 | 0 | Input filter 0 | 0 = deactivated (input filter 0,25 ms) **A**<br>1 = activated (input filter 2,5 ms) |
| | ... | ... | |
| | 7 | Input filter 7 | |
| 1 | 0 | Digital input 0 | 0 = normal **A**<br>1 = inverted |
| | ... | ... | |
| | 3 | Digital input 7 | |
| 2 | 0 | Output on overcurrent 0 | 0 = automatic recovery **A**<br>1 = controlled recovery |
| | ... | ... | |
| | 7 | Output on overcurrent 7 | |
| 3 | 0 | Output 0 | 0 = deactivate **A**<br>1 = activate |
| | ... | ... | |
| | 7 | Output 7 | |

### SWIRE

see Representation of parameter data (page 12-10)

# 15 Diagnostics - Emergency Frames


**15.1 General ..... 2**

**15.2 Structure of the emergency frames ..... 2**

15.2.1 Emergency Error-Codes ..... 2
15.2.2 Error register ..... 4

**15.3 Emergency codes for module diagnostics ..... 5**

15.3.1 General module error codes ..... 5
15.3.2 Digital output modules ..... 5
15.3.3 Analog Input Modules ..... 6
– Analog input modules, current ..... 6
– Analog input modules, voltage ..... 6
– Analog input modules, PT/Ni ..... 7
– Analog Input modules, THERMO/ TC ..... 7
15.3.4 Technology modules ..... 8
– RS232/RS4××-modules ..... 8
– SSI-module ..... 8
– Counter/Encoder/PWM ..... 8
– SWIRE-module ..... 9

## 15.1 General

the state of the BL××-station, the communication through the internal modules bus, the communication with CANopen, the state of the gateway itself.

Diagnostics messages are indicated in two different ways:

- through the individual LEDs, see the additional documentation (page 1-2)
- through Emergency Frames in a CANopen configuration tool (software)

## 15.2 Structure of the emergency frames

BL×× CANopen supports Emergency Frames (EMCY) as standardized in CiA DS-301.

The COB-IDs for the EMCY telegrams are defined by the Predefined Master-Slave Connection Set:

**COB-ID = 129 - 1 + Node-ID**

In the event of a communication error, not only the Emergency Error Code but also the Error register (see Table 225: Bit assignment of the Error register) and additional information will be transmitted, so that the error can be more precisely identified.

Only a part of the 5 bytes is used for the additional information. The remaining bytes are then 0.

| **Byte** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| **Data contents** | Error code | | Error register | Additional information | | | | |

### 15.2.1 Emergency Error-Codes

| **Designation** | **0** | **1** | **2** | **3** | **4** | **5** | **Meaning** |
|---|---|---|---|---|---|---|---|
| | **Error code** (hex.) | | **Error register** (see also Table 225:) | **Additional information A, B** | | | |
| Error reset / no error | $0000_{hex}$ | | bit 0, bit 1 set | "0" or module no. **C** | Error code | | Error reset |
| Input current too high | $2110_{hex}$ | | bit 0, bit 1 set | module no. | channel no. | 0 | Input current too high |
| Input current too low | $2130_{hex}$ | | bit 0, bit 1 set | module no. | channel no. | 0 | Input current too low |
| Output current too high | $2310_{hex}$ | | bit 0, bit 1 set | module no. | channel no. | 0 | Output current too high |
| Output current out of range | $2323_{hex}$ | | bit 0, bit 1 set | module no. | channel no. | 0 | Output current below or above the permissible range |
| Load dump at outputs | $2330_{hex}$ | | bit 0, bit 1 set | module no. | channel no. | 0 | Output current too low |
| AI U voltage out of range | $3003_{hex}$ | | bit 0, bit 2 set | module no. | channel no. | 0 | Input voltage of an AI-module outside of the permissible |
| Mains voltage too high | $3110_{hex}$ | | bit 0, bit 2 set | 0 | channel no. | 0 | System voltage too high |

| Designation | | 0 | 1 | 2 | 3 | 4 | 5 | Meaning |
|---|---|---|---|---|---|---|---|---|
| | | Error code (hex.) | | Error register (see also Table 225:) | Additional information A, B | | | |
| Mains voltage too low | | $3120_{hex}$ | | bit 0, bit 2 set | module no. | channel no. | 0 | System voltage too low |
| Output voltage too low | | $3320_{hex}$ | | bit 0, bit 2 set | module no. | channel no. | 0 | Field voltage too low |
| Additional modules | SSI / RSxxx Error **D** | $7000_{hex}$ | | bit 0, bit 7 set | module no. | channel no. (always 1) | Module Error Code | Error in SSI or RS×××-module (∅ page 15-8 ff.) |
| Additional modules | General module error | $7010_{hex}$ | | bit 0, bit 7 set | 0 | 0 | 0 | (→ page 15-5 ff.) |
| Additional modules | Change of diagnosis | $7011_{hex}$ | | bit 0, bit 7 set | module no. | | | |
| Additional modules | Module list change ok | $707A_{hex}$ | | bit 0, bit 7 set | module no. | 0 | 0 | I/O module list, adaptable change, e.g. module removed |
| Additional modules | Module list change not ok | $707D_{hex}$ | | bit 0, bit 7 set | module no. | 0 | 0 | I/O module list, incompatible |
| Additional modules | Module list extended | $707E_{hex}$ | | bit 0, bit 7 set | module no. | 0 | 0 | I/O module list extended |
| Additional modules | Module list shortened | $707F_{hex}$ | | bit 0, bit 7 set | module no. | 0 | 0 | 1 module removed from I/O module list |
| Communication warning | | $8100_{hex}$ | | bit 0, bit 4 set | 2 | 0 | 0 | CAN communication faulty (at least one of the error counters for the CAN controller of the CANopen gateway has reached the value 96) |
| Communication transmit timeout | | $8100_{hex}$ | | bit 0, bit 4 set | 3 | 0 | 0 | The CANopen gateway was not able to transmit a frame within the specified time |
| Life guard / Heartbeat error | | $8130_{hex}$ | | bit 0, bit 4 set | 0 | 0 | 0 | The CANopen gateway has detected an error in the CANopen Guarding or Heartbeat Protocol, e.g. a time-out. |
| Recovered from Bus OFF | | $8140_{hex}$ | | bit 0, bit 4 set | 0 | 0 | 0 | The system has left the CAN-Bus Off state, i.e. the CAN controller for the CANopen gateway has moved out of this serious fault condition. |
| External Error | | $9009_{hex}$ | | bit 0, bit 4 set | 0 | 0 | 0 | Force Mode is active, i.e. the outputs of the station are not under the control of CANopen at the moment. |

**A** *In total, bytes 3 to 7 of the Emergency Frame can be used to display additional information. At the moment, only byte 3 to 5 are used (if necessary). Byte 6 and 7 are not listed.*

*Unused bytes of additional information are 0.*

**B***If the error message is related to the gateway itself (voltage module at the gateway), the value for module- and channel number is "0".*

**C***In this case, the module number is only displayed, if the previous and already reset error was an error with Error code $7011_{hex}$. For all other errors, this byte is "0".*

**D***An integration of this additional information is only possible, if the type of the respective module can be defined by means of the module number.*

**Note**

The Error code of the Emergency Frame can only be read out using a special analysis tool.

### 15.2.2 Error register

*Table 225: Bit assignment of the Error register*

**A***M = mandatory*
**B***O = optional*

| Error register | M/O | Meaning |
|---|---|---|
| Bit 0 | M **A** | Generate the error message |
| Bit 1 | O **B** | current error |
| Bit 2 | O | voltage error |
| Bit 3 | O | temperature error |
| Bit 4 | O | communication error (overrun, error state) |
| Bit 5 | O | device-profile-specific error |
| Bit 6 | O | reserved |
| Bit 7 | O | manufacturer-specific error |

## 15.3 Emergency codes for module diagnostics

### 15.3.1 General module error codes

| Designation<br>Meaning | byte 01 | Byte 2 | Byte 3 | Byte 4 |
|---|---|---|---|---|
| General module error<br>General module error | **Error code** | **Error register** | **Additional information** | Additional information |
| | $7010_{hex}$ | bit 0, bit 7 set (see also Table 225:) | reserved | channel number for multi-channel modules) |
| **Change of Dia.**<br>(change in byte 0 to 3 of the diagnostic data) | | | | |
| | $7011_{hex}$ | bit 0, bit 7 set (see also Table 225:) | Module number | channel number for multi-channel modules) |

### 15.3.2 Digital output modules

| Designation<br>Meaning | byte 01 | Byte 2 | Byte 3 | Byte 4 |
|---|---|---|---|---|
| **Output current too high:**<br>Current too high | **Error code** | **Error register** | **Additional information** | Additional information |
| | $2310_{hex}$ | bit 1 set (see also Table 225:) | Module number | channel number for multi-channel modules) |
| **Load dump at outputs:**<br>Open circuit or current too low (threshold: | | | | |
| | $2330_{hex}$ | bit 1 set (see also Table 225:) | Module number | channel number for multi-channel modules) |

**Note**

For the exact byte assignment of the Emergency frame, please see "Structure of the Emergency Frames", page 15-2.

### 15.3.3 Analog Input Modules

**Analog input modules, current**

| Designation<br>Meaning | byte 01 | Byte 2 | Byte 3 | Byte 4 |
|---|---|---|---|---|
| **Input current too high:**<br>The input current is outside of the permissible range. **A** | **Error code** | **Error register** | **Additional information** | Additional information |
| | $2110_{hex}$ | bit 1 set<br>(see also Table 225:) | Module number | channel number for multi-channel modules) |
| **Input current too low:**<br>Open circuit or input current (for the measuring range 4 to 20 mA) too low. The threshold is 3 mA. | | | | |
| | $2130_{hex}$ | bit 1 set<br>(see also Table 225:) | Module number | channel number for multi-channel modules) |

**A***The threshold for this error message is 1% over the upper measuring range value or 1 % below the lower measuring range value.*

**Note**

For the exact byte assignment of the Emergency frame, please see "Structure of the Emergency Frames", page 15-2.

**Analog input modules, voltage**

| Designation<br>Meaning | byte 01 | Byte 2 | Byte 3 | Byte 4 |
|---|---|---|---|---|
| **AI U voltage out of range:**<br>Wire break or the input voltage is outside of the permissible range. | **Error code** | **Error register** | **Additional information** | Additional information |
| | $3003_{hex}$ | bit 2 set<br>(see also Table 225:) | Module number | channel number for multi-channel modules) |

**Note**

For the exact byte assignment of the Emergency frame, please see "Structure of the Emergency Frames", page 15-2.

### Analog input modules, PT/Ni

| Designation<br>Meaning | byte 01 | Byte 2 | Byte 3 | Byte 4 |
|---|---|---|---|---|
| **Output current too high:**<br>Current too high (threshold: approx. 5 Ω; only with temperature measuring ranges) | **Error code** | **Error register** | **Additional information** | Additional information |
| | $2310_{hex}$ | bit 1 set (see also Table 225:) | Module number | channel number for multi-channel modules) |
| **Output current out of range:**<br>The input current is outside of the permissible range. **A** | | | | |
| | $2323_{hex}$ | bit 1 set (see also Table 225:) | Module number | channel number for multi-channel modules) |
| **Load dump at outputs:**<br>Open circuit or current too low (threshold: | | | | |
| | $2330_{hex}$ | bit 1 set (see also Table 225:) | Module number | channel number for multi-channel modules) |

**A***The threshold for this error message is 1% over the upper measuring range value or 1 % below the lower measuring range value.*

**Note**

For the exact byte assignment of the Emergency frame, please see "Structure of the Emergency Frames", page 15-2.

### Analog Input modules, THERMO/ TC

| Designation<br>Meaning | byte 01<br>Error code | Byte 2<br>Error register | Byte 3<br>Additional information | Byte 4<br>Additional information |
|---|---|---|---|---|
| **AI U voltage out of range:**<br>Wire break or the input voltage is outside of the permissible range. **A** | | | | |
| | $3003_{hex}$ | bit 2 set (see also Table 225:) | Module number | channel number for multi-channel modules) |

**Note**

For the exact byte assignment of the Emergency frame, please see "Structure of the Emergency Frames", page 15-2.

### 15.3.4 Technology modules

#### RS232/RS4××-modules

| Designation<br>Meaning | Byte 0 + 1:<br>+Error code | Byte 2<br>Error register | Byte 3 | Byte 4 | Byte 5 | Meaning |
|---|---|---|---|---|---|---|
| **Additional modules** | $7000_{hex}$ | Bit 8 set (see also Table 225:) | module no. | Channel number (always = 1) | $08_{hex}$ | Parameter error |
| | | | | | $10_{hex}$ | Hardware failure |
| | | | | | $20_{hex}$ | Handshake error |
| | | | | | $30_{hex}$ | Frame error |
| | | | | | $40_{hex}$ | RX buffer overflow |

**Note**

For the exact byte assignment of the Emergency frame, please see "Structure of the Emergency Frames", page 15-2.

#### SSI-module

| Designation<br>Meaning | Byte 0 + 1:<br>Error code | Byte 2<br>Error register | Byte 3 | Byte 4 | Byte 5 | Meaning |
|---|---|---|---|---|---|---|
| **Additional modules** | $7000_{hex}$ | Bit 8 set (see also Table 225:) | module no. | Channel number (always = 1) | $01_{hex}$ | SSI-Diag |
| | | | | | $02_{hex}$ | SSI-Error |
| | | | | | $04_{hex}$ | Overflow-Error |
| | | | | | $08_{hex}$ | Underflow-Error |
| | | | | | $10_{hex}$ | Parameter error |

#### Counter/Encoder/PWM

| Designation<br>Meaning | byte 01 | Byte 2 | Byte 3 | Byte 4 |
|---|---|---|---|---|
| **Output current too high:**<br>Current too high | **Error code** | **Error register** | **Additional information** | Additional information |
| | $2310_{hex}$ | bit 1 set (see also Table 225:) | Module number | channel number for multi-channel modules) |
| **Output current out of range:**<br>The input current is outside of the permissible range. **A** | | | | |
| | $2323_{hex}$ | bit 1 set (see also Table 225:) | Module number | channel number for multi-channel modules) |

### SWIRE-module

| Designation<br>Meaning | Byte 0 + 1:<br>Error code | Byte 3 | Byte 4 | Byte 5 | Byte 6 | Byte 7 |
|---|---|---|---|---|---|---|
| **Additional modules** | $7000_{hex}$ | module no. | Diag byte 0 | Diag byte 1 | Diag byte 2<br>\| 4 \| 6 | Diag byte 3<br>\| 5 \| 7 |

Bytes 6 and 7 of the Emergency frame contain the result of the bitwise OR relation of the mentioned diagnostic bytes (see Channel number (see page 12-6 ff.).

# 16 Index


**B**

Boot-up Message ..... 2-10

**C**

CANopen ..... 2-1
– Communication ..... 2-2
– EDS-file ..... 2-5
– Emergency object (Emcy) ..... 2-4
– Network Management Messages ..... 2-2
– Process Data Objects (PDOs) ..... 2-3
– Service Data Objects (SDOs) ..... 2-3
– Synchronization object ..... 2-4
– Time stamp object (Time) ..... 2-4
CANopen Standard Objects ..... 3-2
COB-ID ..... 2-8
Communication profile ..... 4-2

**D**

Default-mappings ..... 2-16
Default-PDOs ..... 2-16
Default-RPDOs ..... 2-18
Default-TPDOs ..... 2-18
Diagnostic ..... 13-1, 15-1

**E**

Emergency-Frame ..... 15-2
Error code ..... 15-2
Error register ..... 15-2, 15-4
Event Timer ..... 2-15

**I**

Identifier ..... 2-8
Inhibit Time ..... 2-15

**M**

maintenance ..... 1-2
Manufacturer specific objects ..... 4-39
Mappable Objects ..... 2-22
Mapping Objects ..... 2-16
Minimum Boot-up ..... 2-5

**N**

Node Guarding ..... 2-9
Node-ID ..... 2-8

**O**

Object ..... 4-5
Object dictionary ..... 3-1
objects
– 1000hex – Device Type ..... 4-5
– 1001hex – Error Register ..... 4-5
– 1005hex – SYNC COB-ID ..... 4-6
– 1008hex – Device Name ..... 4-7
– 1009hex - Manufacturer Hardware Version ..... 4-8
– 100Ahex – Manufacturer Software Version ..... 4-8
– 100Chex – Guard Time ..... 4-9
– 1011hex – Restore Default Parameters ..... 4-11
– 1014hex – Emcy COB-ID ..... 4-12
– 1F80hex – NMT Start-up ..... 4-32
– 1F81hex – Slave Assignment ..... 4-33
– 1F82hex – Request NMT ..... 4-35
– 1F83hex - Request Guarding ..... 4-37
– 2000hex – Serial Number ..... 4-39
– 2010hex – Node ResetModifiers ..... 4-40
– 2400hex - System Voltages ..... 4-43
– 2401hex - System Currents ..... 4-43
– 5420hex – Manu Spec Analog Input Range ..... 8-3
– 5440hex – Manu Spec Analog Output Range ..... 9-4
– 5600hex – RS232/RS4xx Parameters ..... 10-2
– 5601hex – RS232/RS4xx RxD ..... 10-4
– 5602hex – RS232/RS4xx TxD ..... 10-7
– 6000hex – Read Input 8 Bit ..... 5-4
– 6020hex – Read Input Bit (1 to 128) ..... 5-5
– 6021hex – Read Input Bit(129 to 256) ..... 5-5
– 6022hex – Read Input Bit (257 to 288) ..... 5-5
– 6100hex – Read Input 16 Bit ..... 5-6
– 6200hex – Write Output 8 Bit ..... 6-4
– 6206hex – Error Mode Output 8 Bit ..... 6-5
– 6207hex – Error State Output 8 Bit ..... 6-6
– 6220hex – Write Output Bit ..... 6-7
– 6221hex – Write Output Bit ..... 6-7
– 6222hex – Write Output Bit ..... 6-7
– 6250hex – Error Mode Output Bit (1 to 128) ..... 6-8
– 6251hex – Error Mode Output Bit (129 to 256) ..... 6-8
– 6252hex – Error Mode Output Bit (257 to 288) ..... 6-8
– 6260hex – Error State Output Bit (1 to 128) ..... 6-9
– 6261hex – Error State Output Bit (129 to 256) ..... 6-9
– 6262hex – Error State Output Bit (257 to 288) ..... 6-9
– 6401hex – Read Analog Input 16 Bit ..... 8-9
– 6411hex – Write Analog Output 16 Bit ..... 9-6
– 6421hex – Analog Input Interrupt Trigger Selection 8-10
– 6422hex – Analog Input Interrupt Source ..... 8-12
– 6423hex – Analog Input Global Interrupt Enable ..... 8-13
– 6424hex – Analog Input Interrupt Upper Limit Integer ..... 8-13
– 6426hex – Analog Input Interrupt Delta Unsigned ... 8-14
– 6427hex – Analog Input Interrupt Negative Delta Unsigned ..... 8-15
– 6428hex – Analog Input Interrupt Positive Delta Unsigned ..... 8-15
– 6443hex - Analog Output Error Mode ..... 9-7
– 6444hex – Analog Output Error State ..... 9-8
– 67FFh – Device Type ..... 4-38
– 6D00hex – Operating Status ..... 11-51
– 6D01hex – SingleTurn Resolution ..... 11-51
– 6D02hex – Number of Distinguishable Revolutions ..... 11-51
– 6FFFhex – Device Type ..... 4-39, 11-51
– Analog input modules ..... 8-1
– analog output modules ..... 9-1

–Digital combi modules .....7-1
–digital input modules .....5-1
–Digital output modules .....6-1
–general I/O objects ..... 14-1
–Manufacturer Hardware Version .....4-8
–RS××× modules ..... 10-1
–SSI modules ..... 11-1
operation, safe .....1-2

**P**
Parameter data .....11-10
Parameterization through SDOs ..... 2-10
PDO mapping ..... 2-23

**R**
Receive PDO-Mapping Parameters ..... 4-23

**S**
SDO
–Read .....2-9
–Write ..... 2-12
standard objects .....2-8
storage .....1-2
symbols .....1-3

**T**
Transmission Type ..... 2-14
Transmit PDO-Mapping Parameter ..... 4-29
Transmit PDO-Parameters ..... 4-26
transport, appropriate .....1-2

TURCK

Industrial
Automation



**www.turck.com**

D301230 0511

**Hans Turck GmbH & Co. KG**
45472 Mülheim an der Ruhr
Germany
Witzlebenstraße 7
Tel. +49 (0) 208 4952-0
Fax +49 (0) 208 4952-264
E-Mail more@turck.com
Internet www.turck.com

**Artisan Technology Group** is an independent supplier of quality pre-owned equipment

## Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

## We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

## Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP