CANopen Chip CoC-100

System Manual

Edition March 2020

Preliminary

This side was left blank intentionally.

In this manual are descriptions for copyrighted products which are not explicitly indicated as such. The absence of the trademark (©) symbol does not infer that a product is not protected. Additionally, registered patents and trademarks are similarly not expressly indicated in this manual

The information in this document has been carefully checked and is believed to be entirely reliable. However, SYS TEC electronic AG assumes no responsibility for any inaccuracies. SYS TEC electronic AG neither gives any guarantee nor accepts any liability whatsoever for consequential damages resulting from the use of this manual or its associated product. SYS TEC electronic AG reserves the right to alter the information contained herein without prior notification and accepts no responsibility for any damages which might result.

Additionally, SYS TEC electronic AG offers no guarantee nor accepts any liability for damages arising from the improper usage or improper installation of the hardware or software. SYS TEC electronic AG further reserves the right to alter the layout and/or design of the hardware without prior notification and accepts no liability for doing so.

Contact	**Direct**	**Your local distributor**
Address:	SYS TEC electronic AG Am Windrad 2 D-08468 Heinsdorfergrund GERMANY	Please find a list of our distributors under http://www.systec-electronic.com/distributors
Ordering Information:	+49 (3765) 38600 0 info@systec-electronic.com	
Technical Support:	+49 (3765) 38600 0 support@systec-electronic.com	
Fax:	+49 (3765) 38600 4100	
Web Site:	http://www.systec-electronic.com	

2th Edition 2020

This side was left blank intentionally.

Table of Content

This side was left blank intentionally.

Index of Figures

This side was left blank intentionally.

Index of Tables

1 Preface

This manual describes only the functions of the CANopen Chip CoC-100.

In this manual low active signals are denoted by a "/" in front of the signal name (i.e.: /RD). A "0" indicates a logic-zero or low-level signal, while a "1" represents a logic-one or high-level signal.

Hexadecimal numbers are represented in the document by the character H at the end of the number, e.g. 1234H.

Declaration of the Electro Magnetic Conformity for the CANopen Chip CoC-100

The CANopen Chip CoC-100 (henceforth product) was designed for installation in electrical appliances or as dedicated Evaluation Boards (i.e.: for use as a test and prototype platform for hardware/software development) in laboratory environments.

Note:
SYS TEC products lacking protective enclosures are subject to damage by Electro Static Discharge (ESD) and, hence, may only be unpacked, handled or operated in environments in which sufficient precautionary measures have been taken in respect to ESD dangers. It is also necessary that only appropriately trained personnel (such as electricians, technicians and engineers) handle and/or operate these products. Moreover, SYS TEC products should not be operated without protection circuitry if connections to the product's pin header rows are longer than 3 m.

SYS TEC products fulfill the norms of the European Union’s Directive for Electro Magnetic Conformity only in accordance to the descriptions and rules of usage indicated in this manual (particularly in respect to the pin header row connectors, power connector and serial interface to a host-PC).

Implementation of SYS TEC products into target devices, as well as user modifications and extensions of SYS TEC products, is subject to renewed establishment of conformity to, and certification of, Electro Magnetic Directives. Users should ensure conformance following any modifications to the products as well as implementation of the products into target systems.

The CANopen Chip CoC-100 is one of a series of SYS TEC DIPmodules that can be fitted with different controllers and, hence, offers various functions and configurations. SYS TEC supports all common 16- and 32-bit controllers in two ways:

(1) as the basis for Development Kits in which user-designed hardware can be implemented on a wrap-field around the controller and

(2) as insert-ready, fully functional ECUcore modules which can be directly embedded into the user's peripheral hardware design.

SYS TEC's microcontroller modules allow engineers to shorten development horizons, reduce design costs and speed project concepts from design to market.

2 Introduction to the CANopen Chip CoC-100

The CANopen Chip CoC-100 is available in various hardware- and firmware-versions:

Table 1: CANopen Chip CoC-100 versions

Item Number	Description
4003001	CANopen Chip CoC-100 standard version voltage 5.0V
4003002	CANopen Chip CoC-100 standard version voltage 3.3V
3301010	CANopen µChip CoC-100, singel chip version voltage 3.3V upt to 5.0V (see S32K Datasheet)

The differences and peculiarities of the both versions are described later in this document.

The CANopen Chip CoC-100 is a tiny yet highly cost-effective Single Board I/O device. In the size of a 40-pin DIP device, the module is designed for use as core component in a customer application design. The CANopen Chip CoC-100 features, besides the implemented standard CANopen firmware, digital inputs and outputs as well as analog input channels and PWM outputs. Using the various on-board configuration options, the module is adaptable to different applications.

All applicable controller signals extend to standard-width (2.54 mm.) pin header rows aligning two edges of the board, making the features and functionality of the Chip readily available to the user. To achieve the compact form factor of a 40-pin DIP device, small package types of the different components are used. The CANopen Chip CoC-100 operates with a single 5 V supply voltage according to the specifications of standard TTL devices or 3.3 V supply voltage.

The firmware implemented in the CANopen Chip CoC-100 has the complete functionality of a CANopen Slave device and has been certified by CiA® (CAN in Automation e.V.). The present version of the CANopen-Chip supports 11-Bit identifier (CAN 2.0B passive).

The firmware supports the standard Device Profile according to **CiA 401** and the Communication Profile according to **CiA 301 V4.01**. The CANopen Chip CoC-100 provides various on-board configuration options for both the CANopen network parameters and the selection of input und output characteristics (number and type of I/O's). The non volatile memory on the CANopen Chip CoC-100 stores the configuration data of the CANopen Slave during runtime. This provides the advantage that, in the event of a temporary loss of the power supply, configuration data are still valid in the non volatile memory.

2.1 Features

- single board CANopen I/O-node in 40-pin DIP dimensions (24x56mm) populated with NXP S32K142 32-bit microcontroller
- controller signals and ports extending to standard-width (2.54 mm.) pin rows lining the edges of the board
- CAN signals (CANH, CANL) in CAN 2.0B passive mode
- reference voltage pins for the on-chip A/D-converter
- non volatile storage of CANopen configuration parameters
- 5.0V on-board MCP2562 CAN transceiver (4003001)
- 3.3V on-board TCAN337GDR CAN transceiver (4003002)
- CAN signals also available for connection to an external, optically isolated CAN transceiver
- 8-position DIP-switch enables selection of:
  Node-ID (4-bit), baud rate (2-bit) and I/O configuration (2-bit),

3 Hardware Overview

3.1 Pin Layout

Please note that all module connections are not to exceed their expressed maximum voltage or current. Maximum signal input values are indicated in the corresponding controller User's Manual/Data Sheets located on the SYS TEC Products CD. As damage from improper connections varies according to use and application, it is the user's responsibility to take appropriate safety measures to ensure that the module connections are protected from overloading through connected peripherals. All controller signal extend to the pin header connector without any additional protection circuitry. External circuitry in which the CANopen Chip CoC-100 is implemented must follow the guidelines for EMC conformance. As *Figure 1* indicates, all controller signals extend to standard-width (2.54 mm / 0.10 in.) pin rows lining two sides the board (referred to as DIPmodul-connector). This allows the CANopen Chip CoC-100 to be plugged into any target application like a "big chip".

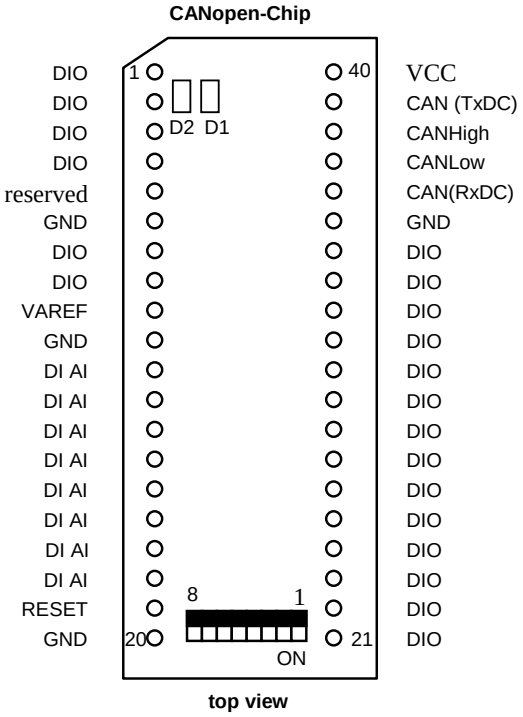

Figure 1: Pin Layout

A more detailed description of pin signals and functions is available in Table 2.

3.2 Pin Description Of The Board

Table 2: Pinout of the DIPmodul-connector

Pin Number	Function	I/O	Description
1	PTC8	I/O	Port Pin PTC8 of the microcontroller
2	PTC9	I/O	Port Pin PTC9 of the microcontroller
3	PTD1	I/O	Port Pin PTD1 of the microcontroller
4	PTD4	I/O	Port Pin PTD4 of the microcontroller
5	PTD0	I/O	Reserved for compatibility
6	GND	-	Ground 0V
7	PTD15	I/O	Port Pin PTD15 of the microcontroller
8	PTC15	I/O	Port Pin PTC15 of the microcontroller
9	VAREF	-	Reference voltage input of the on-chip A/D converter
10	GND	-	Analog Ground
11	PTA0	I	Port Pin PTA0 of the microcontroller
12	PTA1	I	Port Pin PTA1 of the microcontroller
13	PTA6	I	Port Pin PTA6 of the microcontroller
14	PTA7	I	Port Pin PTA7 of the microcontroller
15	PTB0	I	Port Pin PTB0 of the microcontroller
16	PTB1	I	Port Pin PTB1 of the microcontroller
17	PTB2	I	Port Pin PTB2 of the microcontroller
18	PTB3	I	Port Pin PTB3 of the microcontroller
19	RESIN	I	Reset input of the module, 0 - 1 transition triggers the RESET signal
20	GND	-	Ground 0V
21	PTE10	I	Port Pin PTE10of the microcontroller
22	PTE0	I	Port Pin PTE0 of the microcontroller
23	PTE1	I	Port Pin PTE1 of the microcontroller
24	PTE2	I	Port Pin PTE2 of the microcontroller
25	PTE3	I	Port Pin PTE3 of the microcontroller
26	PTE6	I	Port Pin PTE6 of the microcontroller
27	PTE7	I	Port Pin PTE7 of the microcontroller
28	PTE9	I	Port Pin PTE9 of the microcontroller
29	PTC16	I	Port Pin PTC16 of the microcontroller
30	PTC17	I	Port Pin PTC17 of the microcontroller
31	PTA2	I	Port Pin PTA2 of the microcontroller

Pin Number	Function	I/O	Description
32	PTA3	I	Port Pin PTA3 of the microcontroller
33	PTD5	I	Port Pin PTD5 of the microcontroller
34	PTD2	I	Port Pin PTD2 of the microcontroller
35	GND	-	Ground 0 V
36	RxDC	I	Receive line of the on-chip CAN controller
37	CANL	I/O	CANL in-/output of the CAN transceiver
38	CANH	I/O	CANH in-/output of the CAN transceiver
39	TxDC	O	Send line of the on-chip CAN controller
40	VCC	-	Power supply +5 V = or 3.3V

The assignment between applicable I/O lines and their specific function is provided in
.

The use of an external CAN driver or an external galvanic isolation of CAN interface (use of signals RxDC and TxDC instaed of CANL and CANH) demands the disassembling of resistors on J1 of the module. A simulaneously use of onboard CAN driver and external CAN driver is not possible.

3.3 Board Configuration

3.3.1 DIP-Switch

An 8-position DIP-switch is located on the topside of the CANopen Chip CoC-100. Four of these switches enable configuration of the Node-ID for CANopen, two set the CAN baud rate CAN bus, and the remaining two switches are used to enable I/O configuration.

Additional configurations are possible within CANopen (*refer to section 3.5*).

Figure 2 shows the DIP-switch pinout and functions. The switch position "OFF" corresponds to logic-zero or low-level signals while "ON" represents high level or logic-one. See *Figure* 1 for location and switch positions.

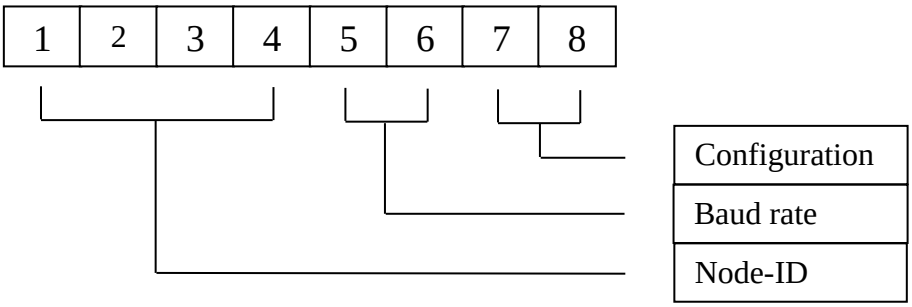

Figure 2: DIP-switch Pinout and Functions

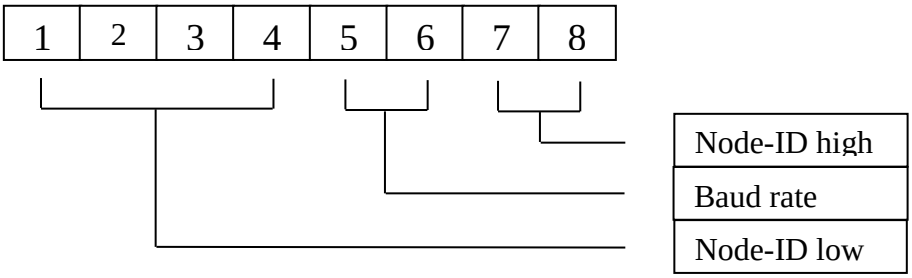

Figure 3: DIP-switch Pinout and Functions, version 3301002 only, currently not available

3.3.2 CAN Transceiver

The firmware operates with 11-bit identifier (Full CAN 2.0B passive). The selection of the CAN transceiver is made by selection of the corresponding pins of the CANopen Chip CoC-100. Pins 37 and 38 for use of the on-board or pins 36 and 39 for use of an external optically isolated CAN transceiver device.

3.4 Reset

A 100 nF capacitor is connected to the microcontroller‘s RESET input. This enables automatic release of a power-on reset. The capacitor is charged via a 4,7kOhm resistor when power is turned on and holds the RESET input at a low level for a duration of approximately 450 microseconds

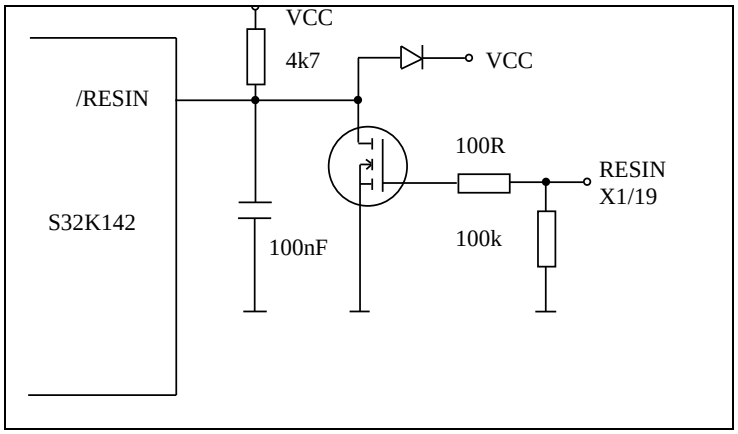

Figure 4: structure of /RESIN Line

3.5 Configuration of Communication Parameters

Node Address Configuration via DIP-switch :
The CANopen Chip CoC-100 is configured for a basic Node-ID of 40H. An additional, device-specific offset can be added to this base address and configured with the DIP-switch S1 DIP1 to DIP4. The resulting Node-ID is then calculated as follows:

Node-ID = 40H + DIP1*2^0 + DIP2*2^1 + DIP3*2^2 + DIP4*2^3

with DIPx = 1 if the corresponding switch is closed (position ON) and DIPx=0 if the switch is open (position OFF).

Example:
If both DIP-switches DIP1 and DIP4 are closed the following Node-ID will be configured:
Node-ID = 40H + 1*2^0 + 0*2^1 + 0*2^2 + 1*2^3
Node-ID = 40H + 1*1 + 1*8
Node-ID = 40H + 1 + 8
Node-ID = 49H

Note:
The configuration 0FH on DIP1...DIP4 is reserved and may not be used in an application. This setting resets the CANopen Chip CoC-100 into its factory default state.

The following table gives an overview of the possible configurations:

Table 3: Configuration of Node-ID

DIP1	DIP2	DIP3	DIP4	Node -ID
OFF	OFF	OFF	OFF	40H (default)
ON	OFF	OFF	OFF	41H
OFF	ON	OFF	OFF	42H
ON	ON	OFF	OFF	43H
OFF	OFF	ON	OFF	44H
ON	OFF	ON	OFF	45H
OFF	ON	ON	OFF	46H
ON	ON	ON	OFF	47H
OFF	OFF	OFF	ON	48H
ON	OFF	OFF	ON	49H

DIP1	DIP2	DIP3	DIP4	Node -ID
OFF	ON	OFF	ON	4AH
ON	ON	OFF	ON	4BH
OFF	OFF	ON	ON	4CH
ON	OFF	ON	ON	4DH
OFF	ON	ON	ON	4EH
ON	*ON*	*ON*	*ON*	*Reserved*

Node- Address Configuration via CANopen
Layer Setting Services (LSS) implemented in the CANopen protocol can be used to freely assign a Node-ID on the CANopen Chip CoC-100. This address can be in the range from 1 to 127 decimal (01H...7FH). After such LSS configuration the DIP-switch settings for the Node-ID are no longer valid.

Note:
Changing the Node-ID via CANopen LSS also results in resetting the Object Dictionary (OD) with default values or the values saved in the non volatile memory.

Bit Rate Configuration via DIP-switch:
Switches DIP5 and DIP6 of DIP-switch can be used to configure one of 4 pre-defined CAN bit rates available on the CANopen Chip CoC-100. The following bit rates are available:

Table 4: Configuration of CAN Bit Rate

DIP5	DIP6	Bit rates kBit/s
ON	OFF	20
OFF	OFF	125
OFF	ON	500
ON	ON	1000

Bit Rate Configuration via CANopen
Layer Setting Services (LSS) implemented in the CANopen protocol can be used to freely assign a desired bit rate on the CANopen Chip CoC-100. After such LSS configuration, the DIP-switch settings for the bit rate are no longer valid.

Table 5: Configuration of CAN Bit Rate over LSS

Index in LSS Timing Table	**Bit rate kBit/s**
0	1000
1	800
2	500
3	250
4	125
5	100
6	50
7	20
8	10

I/O Configuration via DIP-switch:
The CANopen Chip CoC-100 firmware features four pre-defined I/O configurations that can be configured using switches DIP7 and DIP8 of DIP-switch. The following I/O configurations are available:

Table 6: I/O Configuration

DIP7	DIP8	I/O Configuration
OFF	OFF	I/O Configuration 0
ON	OFF	I/O Configuration 1
OFF	ON	I/O Configuration 2
ON	ON	I/O Configuration 3

I/O Configuration via CANopen
The desired I/O configuration on the CANopen Chip CoC-100 can also be selected via an entry in the Object Dictionary (OD). The manufacturer-specific OD entry 2000H is provided for this purpose. The selection is done by writing the applicable I/O configuration number to this object. For example, when writing the value 5 to index 2000H the resulting I/O configuration will be configuration number 5. After such OD configuration, the DIP-switch settings for the I/O configuration are no longer valid.

Note:
Changing the Node-ID via CANopen OD entry also results in resetting the Object Dictionary (OD) with default values.

3.6 Default Configuration

At the time of delivery all DIP-switches on the CANopen Chip CoC-100 are open. This results in the following factory default settings:

- Node-ID = 40H
- bit rate = 125 kBit/s
- configuration = I/O configuration 0

3.7 Pin Assignments for Selected I/O Configurations CoC-100

Table 7: Number of I/Os Depending on the Selected Configuration CoC-100

Configuration	Digital Inputs	Digital Outputs	Analog Inputs	PWM Outputs
0	14	8	2	4
1	8	8	8	4
2	16	8	-	4
3	8	16	-	4
4	16	-	8	4
5	24	-	-	4
6	16	4	4	4

Table 8: Input/Output Configuration and I/O Pin Assignment CoC-100

Pin#	Config 0	Config 1	Config 2	Config 3	Config 4	Config 5	Config 6
1	DO 4	DO 0	DO 0	DO 0	DI 12	DI 20	DI 0
2	DO 5	DO 1	DO 1	DO 1	DI 13	DI 21	DI 1
3	DO 6	DO 2	DO 2	DO 2	DI 14	DI 22	DO 0
4	DO 7	DO 3	DO 3	DO 3	DI 15	DI 23	DO 1
5							
6							
7	PWM 0	PWM 0	PWM 0	PWM 0	PWM 0	PWM 0	PWM 0
8	PWM 1	PWM 1	PWM 1	PWM 1	PWM 1	PWM 1	PWM 1
9							
10							
11	AI 0	AI 0	DI 0	DI 0	AI 0	DI 0	AI 0
12	AI 1	AI 1	DI 1	DI 1	AI 1	DI 1	AI 1
13	DI 0	AI 2	DI 2	DI 2	AI 2	DI 2	AI 2
14	DI 1	AI 3	DI 3	DI 3	AI 3	DI 3	AI 3
15	DI 2	AI 4	DI 4	DI 4	AI 4	DI 4	DI 2
16	DI 3	AI 5	DI 5	DI 5	AI 5	DI 5	DI 3
17	DI 4	AI 6	DI 6	DI 6	AI 6	DI 6	DI 4
18	DI 13	AI 7	DI 7	DI 7	AI 7	DI 7	DI 5
19							
20							
21	DI 12	DO 4	DO 4	DO 4	DI 8	DI 16	DI 14
22	DI 11	DO 5	DO 5	DO 5	DI 9	DI 17	DI 15
23	DI 10	DI 0	DI 8	DO 8	DI 0	DI 8	DI 6
24	DI 9	DI 1	DI 9	DO 9	DI 1	DI 9	DI 7
25	DI 8	DI 2	DI 10	DO 10	DI 2	DI 10	DI 8
26	DI 7	DI 3	DI 11	DO 11	DI 3	DI 11	DI 9
27	DO 3	DI 4	DI 12	DO 12	DI 4	DI 12	DI 10
28	DO 2	DI 5	DI 13	DO 13	DI 5	DI 13	DI 11
29	DO 1	DI 6	DI 14	DO 14	DI 6	DI 14	DI 12
30	DO 0	DI 7	DI 15	DO 15	DI 7	DI 15	DI 13
31	DI 6	DO 6	DO 6	DO 6	DI 10	DI 18	DO 2
32	DI 5	DO 7	DO 7	DO 7	DI 11	DI 19	DO 3
33	PWM 2	PWM 2	PWM 2	PWM 2	PWM 2	PWM 2	PWM 2
34	PWM 3	PWM 3	PWM 3	PWM 3	PWM 3	PWM 3	PWM 3
35							
36							
37							
38							
39							
40							

Description of table entries:

DI - Digital Input
DO - Digital Output
AI - Analog Input
PWM - PWM Output

Note:
Configuration 0 is set up at time of delivery. Whenever changing the I/O configuration user must ensure that the circuitry connected to the applicable I/O pins meets the requirements of the specific I/O signal type. In appropriate signal connection could damage or destroy the CANopen Chip CoC-100. We recommend to disconnect the I/O pins when changing the I/O configuration.

3.8 reserved

Note:
Configuration 0 is set up at time of delivery. Whenever changing the I/O configuration user must ensure that the circuitry connected to the applicable I/O pins meets the requirements of the specific I/O signal type. In appropriate signal connection could damage or destroy the CANopen Chip CoC-100. We recommend to disconnect the I/O pins when changing the I/O configuration.

3.9 Technical Data

Environmental Conditions

Operational temperature:	-40°C to +85°C
Storage temperature:	-40°C

Mechanical Specifications

Dimensions:	24.0 mm * 58.7 mm., ± 0.3 mm
Weight:	approximately 10.5 grams
Connector Type:	40-pin dual–inline IC socket (2.54 mm pitch), pin diameter 0.47 mm, contact length 3.2 mm

These specifications describe the standard configuration of the CANopen Chip CoC-100 as of the printing of this manual.

3.9.1 Electrical Parameters at 3.3 V Range (4003002)

Operating Voltage:	3.0 V to 3.6 V
Power consumption :	typ. 25 mA, 48 MHz CPU clock, 25°C; max. 185mA depends on circuitry of I/Os and CAN
Clock generation :	external 8 MHz crystal

Output voltage of the I/O lines:

low level:	< 0.8 V
high level:	> V_{CC} - 0.8 V

Output current of the I/O lines:

low level 1 pin maximum current:	12mA
high level 1 pin maximum current:	14mA
high level all pins maximum current:	100mA

Input voltage of the I/O lines:

low level:	>VSS – 0.3V; <VCC * 0.3
high level:	> 0.7 * VCC; <VCC + 0.3V

Analog inputs:

Input voltage:	GND .... VREF
Resolution:	12-bit
Input capacity:	19 pF (typ.)
Reference voltage:	GND + 2.7V ... VCC

On-board CAN FD transceiver:

Type:	TI TCAN337GDR
Maximum bit rate:	1000 kBit/s / 5000 kBit/s
Number of nodes:	<100 (depending on all CAN transceivers in the CAN network)

3.9.2 Electrical Parameters at 5.0 V Range (4003001)

Operating Voltage:	4.5 V to 5.5 V
Power consumption :	typ. 25 mA, 48 MHz CPU clock, 25°C; max. 195mA depends on circuitry of I/Os and CAN
Clock generation :	external 8 MHz crystal

Output voltage of the I/O lines:

low level:	< 0.8 V
high level:	> V_{CC} - 0.8 V

Output current of the I/O lines:

low level 1 pin maximum current:	20mA
high level 1 pin maximum current:	20mA
high level all pins maximum current:	100mA

Input voltage of the I/O lines:

low level:	>VSS – 0.3V; <VCC * 0.3
high level:	> 0.7 * VCC; <VCC + 0.3V

Analog inputs:

Input voltage:	GND .... VREF
Resolution:	12-bit
Input capacity:	19 pF (typ.)
Reference voltage:	GND + 2.7V ... VCC

On-board CAN FD transceiver:

Type:	MCP2562
Maximum bit rate:	1000 kBit/s / 5000 kBit/s
Number of nodes:	<100, depending on all CAN transceivers in the CAN network

4 Setting up the CANopen Chip CoC-100

4.1 Power Supply

The CANopen Chip CoC-100 requires a power supply in a Range of 2.7V to 5.5V. Power can be supplied via pins 6 (GND) and pin 40 (V_{CC}. Additional there are pins 20 and 35 for GND connection. It is recommended to use a standard DIL-40 socket in order to integrate the CANopen Chip CoC-100 into a target hardware environment.

Table 2. Voltage and current operating requirements 1

Symbol	Description	Min.	Max.	Unit	Notes
V_{DD}[2]	Supply voltage	2.7[3]	5.5	V	4
V_{DD_OFF}	Voltage allowed to be developed on V_{DD} pin when it is not powered from any external power supply source.	0	0.1	V	
V_{DDA}	Analog supply voltage	2.7	5.5	V	4
$V_{DD} – V_{DDA}$	V_{DD}-to-V_{DDA} differential voltage	– 0.1	0.1	V	4
V_{REFH}	ADC reference voltage high	2.7	V_{DDA} + 0.1	V	5
V_{REFL}	ADC reference voltage low	-0.1	0.1	V	
V_{ODPU}	Open drain pullup voltage level	V_{DD}	V_{DD}	V	6
$I_{INJPAD\_DC\_OP}$[7]	Continuous DC input current (positive / negative) that can be injected into an I/O pin	-3	+3	mA	
$I_{INJSUM\_DC\_OP}$	Continuous total DC input current that can be injected across all I/O pins such that there's no degradation in accuracy of analog modules: ADC and ACMP (See section Analog Modules)	—	30	mA	

S32K1xx Data Sheet, Rev. 9, 09/2018

10 NXP Semiconductors

General

1. Typical conditions assumes V_{DD} = V_{DDA} = V_{REFH} = 5 V, temperature = 25 °C and typical silicon process unless otherwise stated.
2. As V_{DD} varies between the minimum value and the absolute maximum value the analog characteristics of the I/O and the ADC will both change. See section I/O parameters and ADC electrical specifications respectively for details.
3. S32K148 will operate from 2.7 V when executing from internal FIRC. When the PLL is engaged S32K148 is guaranteed to operate from 2.97 V. All other S32K family devices operate from 2.7 V in all modes.
4. V_{DD} and V_{DDA} must be shorted to a common source on PCB. The differential voltage between V_{DD} and V_{DDA} is for RF-AC only. Appropriate decoupling capacitors to be used to filter noise on the supplies. See application note AN5032 for reference supply design for SAR ADC.
5. V_{REFH} should always be equal to or less than V_{DDA} + 0.1 V and V_{DD} + 0.1 V
6. Open drain outputs must be pulled to V_{DD}.
7. When input pad voltage levels are close to V_{DD} or V_{SS}, practically no current injection is possible.

Figure 5: Voltage and current operating requirements

4.2 CAN Interface

The CANopen Chip CoC-100 is populated with the NXP FS32K142 microcontroller, featuring Full 2.0B on-chip CAN, and an on-board CAN transceiver (MCP2562). The following CAN signals are available at header pins aligning two edges of the board:

CAN_HIGH	Pin 38
CAN_LOW	Pin 37
CAN_GND	Pin 6, 20 or 35

The potentials CAN_GND and GND (general board-level Ground) are identical.

These CAN signals can be routed as follows to a DB-9 socket, according to the CiA 301 Communication Profile standard

CAN_HIGH	Pin 7
CAN_LOW	Pin 2
CAN_GND	Pin 3 and/or Pin 6

The CAN signals can be optically isolated from the CANopen Chip CoC-100 using an external transceiver IC. The maximum transmission rate is 5MBit/s. The maximum CAN bus load should not exceed 50%.

In electromagnetic disturbed environments we recommend to use a common mode choke for CAN_HIGH / CAN_LOW like the B82789C0513N from TDK (Epcos)

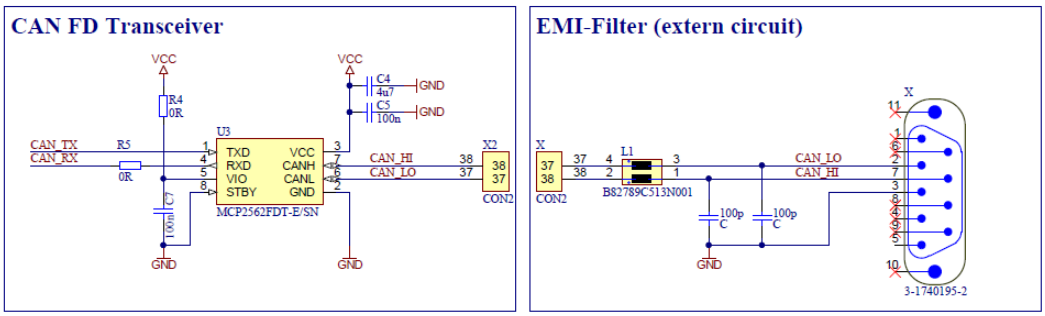

Figure 6: Design for common mode choke, recommendation

5 QuickStart

This section describes basic start-up of the CANopen Chip CoC-100. It assumes basic knowledge of CANopen networks. It also requires, that the CANopen Chip CoC-100 is properly connected to the CAN bus and power is supplied to the CANopen Chip CoC-100. Please *refer to sections **6*** and ***7*** for basic description of CAN and CANopen.

5.1 Start-Up of the CANopen Chip CoC-100

All values in the Object Dictionary (OD) are pre-configured with default-values. Hence, start-up configuration of the CANopen Chip CoC-100 is not necessary. The CANopen Chip CoC-100 supports the CANopen Minimum Boot-Up. Following reset and internal initialization, the board is in PRE-OPERATIONAL state (*refer to section 8.3 PRE-OPERATIONAL*). Upon receipt of a single message (*Start_Remote_Node*) the board switches to OPERATIONAL state (*refer to section 8.4 OPERATIONAL*).

11-bit CAN Identifier	2 Byte Data	
0	*01H*	*Node_ID*

The first data byte contains the Start command, while the second byte contains the node address (*Node_ID*). If you specify the value 00H then all nodes in the network (Broadcast) are started.

5.2 Shut-Down of the CANopen Chip CoC-100

All node activity can be stopped by receipt of the *Enter_Pre_Operational_State* message. This message consists of the following:

11-bit CAN Identifier	2 Byte Status Data	
0	*80H*	*Node_ID*

The *Node-ID* identifies the node-addresses. *Node-ID = 00H* addresses all devices on a network (Broadcast).

When in “PRE-OPERATIONAL“ state, SDO-Transfer is still possible. All configuration parameters are then captured and “frozen” as they were in their most recent active state.

Note:
CANopen configuration tools, such as ProCANopen from Vector or CDM and CCM from Port always use SDO-Transfer when accessing a CANopen node. Due to this fact, these tools also work in “PRE-OPERATIONAL“ state.

5.3 CAN Message and Identifier

According to **CiA 301**, a specific CAN identifier is assigned to each CAN message containing process data (Process-Data-Object - PDO). The CAN identifier for input and output data is derived from the node address.

Table 11: CAN ID for Different PDO Types

CAN Identifier (hex)	Data Type
180H + node address	1st Tx PDO
280H + node address	2nd Tx PDO
380H + node address	3rd Tx PDO
480H + node address	4th Tx PDO
200H + node address	1st Rx PDO
300H + node address	2nd Rx PDO
400H + node address	3rd Rx PDO
500H + node address	4th Rx PDO

5.4 PDO Mapping for I/O’s

The PDO mapping of the available I/O's depends on the selected I/O configuration (refer to

).

The CANopen Chip CoC-100 also supports variable PDO mapping. This allows for free mapping of inputs to Tx PDOs and Rx PDOs to output lines. Such free mapping settings can be saved in the on-board non volatile memory by writing to index 1010H.

5.4.1 Default Mapping CANopen Chip CoC-100

In the default mapping, the 4th Tx PDO and the 2nd Rx PDO and the 4th Rx PDO are invalid. This results in the following arrangement of I/Os and PDOs:

Table 12: PDO Mapping for I/O's CoC-100

Config	1st Tx PDO	2nd Tx PDO	3rd Tx PDO	1st Rx PDO	3rd Rx PDO
0	DI 0...DI 7 DI 8...DI 13	AI 0...AI 1	invalid	DO 0...DO7	duty cycle PWM 0...PWM 3
1	DI 0...DI 7	AI 0...AI 3	AI 4...AI 7	DO 0...DO7	duty cycle PWM 0...PWM 3
2	DI 0...DI 7 DI 8...DI 15	invalid	invalid	DO 0...DO7	duty cycle PWM 0...PWM 3
3	DI 0...DI 7	invalid	invalid	DO 0...DO7 DO8...DO15	duty cycle PWM 0...PWM 3
4	DI 0...DI 7 DI 8...DI 15	AI 0...AI 3	AI 4...AI 7	invalid	duty cycle PWM 0...PWM 3
5	DI 0...DI 7 DI 8...DI 15 DI 16...DI 23	invalid	invalid	invalid	duty cycle PWM 0...PWM 3
6	DI 0...DI 7 DI 8...DI 15	AI 0...AI 3	invalid	DO 0...DO3	duty cycle PWM 0...PWM 3

5.5 Board Reset

Following each board reset, the CANopen Chip CoC-100 transmits an Boot-up message without data content. Temporary suspension of CANopen Chip CoC-100 activity and subsequent restart can be recognized without Nodeguarding *(refer to section 5.6 Node Guarding)*. The transmitter of this message will be detected by the CAN identifier.

11-bit CAN Identifier	1 Byte Data
700H+ Node_ID	*00H*

5.6 Node Guarding

Node Guarding and *life guarding* functions monitor operation of the CANopen network. Distributed peripheral CAN devices are monitored via Node Guarding, while the Life Guarding function supervises the guarding master. To realize Node Guarding, the master requests a

cyclic status message from the slave nodes. This status request is initiated with a remote frame message that contains only the status data. The RTR-Bit (Remote Transmit Request Bit) is set for this reason.

11-bit CAN Identifier	1 Byte Data
700H + Node_ID	*Node Guarding*

Following transmission of the remote frame message, the slave nodes responds with a status message consisting of 1 byte of service data.

11-bit CAN Identifier	1 Byte Data
700H+ Node_ID	*Guarding Protocol*

The data bytes within status message further contain a toggle bit that is supposed to change after each message. Should the status and toggle bits not correspond to the message pattern expected by the Master, or should no response to a message follow, the Master assumes a Slave malfunction. If the Master requests cyclic guard messages, a Slave node can recognize shut-down of the Master. This is the case if the Slave does not receive a message request from the Master within the pre-configured *Life Time*. The Slave then assumes failure of the Master, sets its inputs into Error state, transmits an Emergency message and switches into *Pre-Operational* state (condition index [67FEH] or [1029H] = 0).

The *Life Time Factor* is configured within the Object [100DH] and is multiplied by the *Guard Time* [100CH]. This results in the *Life Time* of the "Node Guarding Protocol". The time base of these cycles is 1 ms.
The *Guard Time* specifies how much time must elapse between two *Node Guarding* messages. The *Life Time Factor* indicates how many times the *Guard Time* can elapse before an error is generated.

Default Values:

Life Time Factor	0
Guard Time	0 ms.
Life Time	0 sec.

Example Values:

Life Time Factor	3
Guard Time	1000 ms.
Life Time	3 sec.

This side was left blank intentionally.

6 Controller Area Network – CAN

6.1 Communication with CANopen®

The Controller Area Network (the CAN bus) is a serial data communications bus for real-time applications. CAN was originally developed by the German company Robert Bosch for use in the automotive industry. It is a two-wire bus system that provides a cost-effective communication bus alternative to expensive and cumbersome harness wiring. CAN operates at data rates of up to 1 Mbit per second and has excellent error detection and confinement capabilities. On account of its proven reliability and robustness, CAN is being used in many other automation and industrial applications. CAN is now an international standard and is documented in ISO 11898 (for high-speed applications).

CANopen is a higher-layer network protocol based on the CAN serial bus system, specifically, it is a software level protocol standard for industrial communication between automated devices. CANopen is authorized by the User and Manufacturers' Group "CAN in Automation e.V." (CiA®) and adheres to ISO/OSI standards.

CANopen unleashes the full power of CAN by allowing direct peer to peer data exchange between nodes in an organized, hierarchical manner. The network management functions specified in CANopen simplify project design, implementation and diagnosis by providing standard mechanisms for network start-up and error management.

CANopen supports both cyclic and event-driven communication. This makes it possible to reduce the bus load to a minimum, while still maintaining extremely short reaction times. High communication performance can be achieved at relatively low baud rates, thus reducing EMC problems and minimizing cable costs. CANopen is the ideal networking system for all types of automated machinery. One of the distinguishing features of CANopen is its support for data exchange at the supervisory control level as well as accommodating the integration of very small sensors and actuators on the same

physical network. This avoids the unnecessary expense of gateways linking sensor/actuator bus systems with higher communication networks and makes CANopen particularly attractive to original equipment manufacturers.

CANopen Advantages

- Vendor-independent open-source structure
- Universal standards
- Supports inter-operability of different devices
- High speed real-time capability
- Modular - covers simple to complex devices
- User-friendly - wide variety of support tools available
- Real-Time-capable communication for process data without protocol overhead;
- a modular, configurable structure that can be tailored to the needs of the user and his or her networked application

CANopen Features

- Auto configuration of the network
- Easy access to all device parameters
- Device synchronization
- Cyclic and event-driven data transfer
- Synchronous reading or setting of inputs, outputs or parameters

In addition to its designation as a physical CAN layer standard, CANopen is a “layer-7 protocol” implementation and is defined by the CANopen Communications Profile in CiA 301.

6.2 CANopen – Open Industrial Communication

The following Special Interest and Working Groups have developed CANopen communication profile:

- SIG Distributed I/O
- SIG Motion Control

and the Working Group (WG)

- WG Higher Layer Protocols

The CiA 301 CANopen standard derived from the results of the ASPIC ESPRIT project. The communication profile describes in detail how data are exchanged over the CAN bus. This data can be sorted into two main types:

- Process data
- Service data

Process data is real-time data generated by a networked device. This data is transmitted via a Process Data Object (PDO). The CANopen communication profile defines which protocol is used for data transmission in PDO. PDOs can be used simultaneously by multiple networked devices, hence enabling broadcast operations.

Service data are used to configure and establish parameters for networked devices. Service data directly communicate to the Object Dictionary of each device and are transmitted using Service Data Objects (SDO).

An SDO can only be used between two networked devices, typically a configuration Master and another device that is to be configured. The SDO-Transfer is also capable of confirmation of message receipt.

Each individual networked device provides several PDOs and SDOs. This enables configuration of multi-master networks, in addition to typical single Master / multiple Slave networks.

In addition to data classes, CANopen defines the communication classes that describe:

- Synchronized communication
- Event processing
- Communication initialization

CANopen also defines device profiles that describe the basic functions of networked devices. These device profiles consist of the following two primary components:

- Functional Description
- Operational Description

The ***Functional Description*** of a device is represented by functional blocks and data flows. Descriptive parameters are stored in the Object Dictionary. Each Object Dictionary has a pre-defined structure. Hence, parameters for networked devices of a certain type (for instance I/O modules or drives) are always located in the same place within an Object Dictionary. Parameters can be classified as mandatory, optional and manufacturer-specific.

The ***Operational Description*** of a device is described by state flow diagram (*refer to Figure 7 in Section 7.9*).

Device Profiles are standardized for:

- Generic I/O Modules CiA 401
  digital I/O's
  analog I/O's
- Drives and Motion Control CiA 402
  Servo drivers,
  Step motors and
  Frequency transformers
- Measurement Devices and
  Closed Loop Controllers CiA 405
- IEC61131-3 Programmable
  Devices CiA 405
- Encoder CiA 406
- Inclinometer CiA 410

Please refer to the CAN in Automation homepage www.can-cia.org for up-to-date information of available device profiles. All device profiles correspond to the DRIVECOM Profile with CAN-specific modifications to enable multi-master capability.

Software for CANopen Slave functions is based on services for data exchange and network management as defined in CiA 301 standard.

6.3 Network topology parameters

The wiring topology of a CAN network should be as close as possible to a single line structure in order to avoid cable-reflected waves. Network topology parameters shall be in accordance with the following table.

Table 14: recommended cable length

Cable Length **[m]**	**Max. Bitrate** **[kBit/s]**	**Length-related resistance** **[mOhm/m]**	**Wire Size** **[mm²]**
0-40	1000	70	0.25 – 0.34
40-100	500	<60	0.34 – 0.60
100-500	100	<40	0.50 – 0.60
500-1000	20	<26	0.75 – 0.80

7 CANopen Communication

7.1 CANopen Fundamentals

Open fieldbus systems enable design of distributed network systems by connecting components from multiple vendors while minimizing the effort required for interfacing. To achieve an open networking system, it is necessary to standardize the various layers of communication used.

CANopen uses the international CAN standard, ISO 11898 as the basis for communication. This standard covers the lower two layers of communication specified by the OSI model. Based on this, the CANopen profile family specifies standardized communication mechanisms and device functionality for CAN-based systems. The profile family, which is available and maintained by CAN in Automation e.V. (CiA) consists of the Application layer and communication profile (CiA 301), various frameworks and recommendations (CiA 30x) and various device profiles (CiA 40x).

The network management functions specified in CANopen simplify project design, implementation and diagnosis by providing standard mechanisms for network start-up and error management.

CANopen is the ideal networking system for all types of automated machinery. One of the distinguishing features of CANopen is its support for data exchange at the supervisory control level as well as accommodating the integration of very small sensors and actuators on the same physical network. This avoids the unnecessary expense of gateways linking sensor/actuator bus systems with higher communication networks and makes CANopen particularly attractive to original equipment manufacturers.

7.2 CANopen Device Profiles

CANopen profiles are defined for communication in CiA 301, for I/O Modules in CiA 401, for Drives and Motion Control in CiA 402 and for Encoder in CiA 406. Other profiles are in preparation.

The profiles of a CANopen device are stored in the Object Dictionary (OD) in a defined manner. The Object Dictionary manages the objects using a 16-bit index. This index can be further subdivided with an 8-bit sub-index. All entries are summarized within groups.

For example, the Communication profile is located at index 1000H to 1FFFH.

Certain types of object entries are mandatory; others are optional or manufacturer-specific. The following types of objects are available:

- Domain a variable number of data
- Deftyp a definition entry, such as unsigned16
- Defstruct record type, such as PDO mapping
- Var an individual variable
- Array a multiple data field, whereby each individual data field is a simple variable of the same type
- Record a multiple data field, whereby the data fields are any combination of simple variables

With structured entries, subindex 0 indicates the number of following subindices.

7.3 Communication Profile

The interface between application and CANopen device is clearly defined by a uniform communication profile. The CANopen communication protocol defines several methods for transmission and receipt of messages over the CAN bus, including transfer of synchronous and asynchronous messages. Coordinated data exchange across an entire network is possible by means of synchronous message transmission. Synchronous data transfer allows network wide coordinated data exchange. Pre-defined communication objects, i.e. SYNC Objects transmitted on a cyclic time period and Time Stamp objects support synchronous transfers. Asynchronous or event messages may be transmitted at any time and allow a device to immediately notify another device without having to wait for the next synchronous data transfer cycle.

7.4 Service Data Objects

Network management controls communication and device profiles of all networked devices. For this type of access service data objects (SDO) are used. In CANopen devices, all parameters and variables that are accessible via CAN are clearly arranged in the Object Dictionary.

All objects in the Object Dictionary can be read and/or written via SDOs. SDO represent a peer-to-peer communication between networked nodes. This access occurs according to the Multiplexed Domain protocol, whereby the index and subindex of the addressed objects are used as a multiplexor. This protocol is based on handshaking.

Individual parameters are addressed using a 16-bit index and an 8-bit subindex addressing mechanism. In this mode data packages may be larger than 8 bytes using multiple CAN messages. Messages smaller than 5 bytes can be transferred with a transmission acknowledgement. The owner of the Object Dictionary is the server of the domain. Read and write accesses via SDOs are supervised by the CANopen server and are checked for validity.

A variety of access restrictions must be taken into account, such as; *Read only*, *Write only* and *No PDO mapping*. Error messages provide detailed information on any access conflicts. Service Data Objects (SDOs) are normally used for device configuration such as setting device parameters. They are also used to define the type and format of information communicated using the Process Data Objects.

7.5 Process Data Objects

A Process Data Object (PDO) is a CAN message whose data contents, identifier, inhibit time, transmission type and CMS priority are configurable via entries in the Object Dictionary. PDO format and data content of the message may be fixed or dynamically configured using SDO data transfers. PDOs do not contain any explicit protocol overhead, hence enabling very fast and flexible exchange of data between applications running on each node. Hence, PDO transfers are typically used for high speed, high priority data exchange. Data size in a PDO message is limited to 8 bytes or less. PDOs can be transmitted directly from any device on the network simultaneously to any number of other devices. Data exchange across a CANopen network does not require a bus Master. This multicast capability is one of the unique features of CAN and is fully exploited in CANopen.

PDO entries start at index 1400H for receipt objects and at 1800H for transmission objects. CANopen permits cyclic and event-controlled communication. The type of transfer indicates the manner of the reaction to the SYNC message; while the inhibit time is the minimum time that must elapse between two transmissions of the PDO. PDOs reduce the bus load to a minimum, achieve a high information flow-rate and can be accessed via remote frames.

A simple CANopen device usually supports four PDOs. These are initialized with preset identifiers. Additional PDOs can be designated, yet to avoid message collision they may be set invalid (deactivated). This deactivation is configured by setting the MSB (bit 31) in the identifier of the PDO.

The message identifier can be found in the Object Dictionary under the entry for communication parameter in subindex 1. Bit 30 indicates

if remote request for this PDO is enabled (bit 30 = 0) or not. Bit 29 configures the CAN frame format, bit 29 = 0 indicates 11-bit identifier.

Table 15: COB-Identifier (Communication Target Object Identifier)

Bit	**31**	**30**	**29**	**28 – 11**	**10 - 0**
11-bit-ID	0/1	0/1	0	000000000000 000000	11-bit Identifier
29-bit-ID	0/1	0/1	1	29-bit Identifier	

The transmission types in subindex 2 can be configured within a range of 0 to 255. The values 0 to 240 define that the transfer of the PDO is in relation to the SYNC message. The value 0 indicates that current input values are only transmitted upon arrival of a SYNC message and if the requested input value has changed. Values between 1 and 240 indicate that the PDO is transmitted upon arrival of a corresponding number of SYNC messages. The values 241 to 251 are reserved. The values 252 and 253 are intended only for remote objects. For value 252, data is updated but not transmitted upon receipt of the SYNC message. The value 253 updates data upon receipt of the remote request. Values 254 and 255 are used for asynchronous PDOs. The release of these asynchronous PDOs is manufacturer or Device Profile-specific.

The inhibit time is stored in multiples of 100 µs as unsigned16 values at subindex 3.

Subindex 04H is reserved.
Depending on the supported subindices, subindex 0 must be set to the applicable value.
PDO settings must correspond to the I/O profile rules:

- the first transmit and receipt PDO is used for exchange of ***digital*** data;
- the second transmit and receipt PDO is used for exchange of ***analog*** data.

If a CANopen device does not support digital inputs or outputs, it is recommended that the first transmit and receipt PDO remains unused.

If a CANopen device does not support analog signals, it is recommended that the second transmit and receipt PDO remains unused.

7.6 PDO-Mapping

A unique mapping entry exists for each communication parameter entry of a PDO. This mapping entry is located in the Object Dictionary 200H above the corresponding communication parameter entry for this PDO. This mapping table corresponds to PDO data contents. The requirement for PDO mapping is the presence of variables in the Object Dictionary that are capable of mapping. For example, digital outputs at index 6200H and digital inputs at index 6000H can be mapped. These values can also be set and read out via SDO. However, in order to use the benefits of the CAN bus, the variables of a CANopen device are put in PDOs.

The mapping of variables is organized as follows:
All mapping entries are 4 bytes in size. The number of objects to be mapped is written to subindex 0. Each following subindex contains a reference to the index and subindex of variables and their length stored in “Bit“. For example:

```
                    60000108H
• reference to index 6000H
• subindex              01
• length                  08 bit
```

In this example the value of the digital input is indicated by the first byte of a transmit PDO. For most CANopen devices, mapping occurs with a granularity of eight (8). This means that a maximum of eight entries per byte is possible for a mapping table.
In special cases mapping of bit objects can be supported. It is also advisable to sometimes exclude areas from mapping. For example, a CANopen device might evaluate only the fifth byte of a PDO. In this case, 2 unsigned16 dummy objects are inserted in the mapping identity, if supported by the CANopen device. A mapping table can be used to appropriately configure communication parameters to encode a PDO for transmission or to decode a received PDO.

7.7 Error Handling

Each node in the network is able to signal error states as far as they are detected by the hardware and software. Error Handling is enabled by Emergency Objects. Internal fatal error states are encoded in error codes and sent only once to the other nodes. If other errors occur, the node remains in error state and transmits a new Emergency Object. If the error is recovered, the node then transmits an error message with the code *No error*. The Emergency message consists of 8 bytes, whereby the first and second bytes contain additional information that is found in the device profiles. The third byte contains the contents of the error register; while the remaining five bytes contain manufacturer-specific information. The Emergency Error code is stored in object [1003H], the *Pre-Defined Error Field*. This creates an error log that chronologically sorts errors. The oldest error is situated at the highest subindex.

Table 16: Emergency-Message Contents

Byte	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Content	Emergency Error Code		Error Register, Object [1001H]	Manufacturer Specific Error Field				

7.8 Network Services

In addition to services enabling configuration and data exchange, various CAN network services also support monitoring of networked devices. NMT (network management) services require a node in the network that assumes the functions of the NMT-Masters. The NMT-Master services include initialization of NMT-Slaves, distribution of the identifiers, the node monitoring and network booting.

7.8.1 Life Guarding

Optional node monitoring is achieved by "*Life Guarding*". The NMT-Master periodically transmits a lifeguard message to the Slave. The Slave responds to the Lifeguard message with a return message indicating its present status and a bit that toggles between two

messages. Should the Slave not respond or indicate an unexpected status, the NMT-Master application is informed by means of a status message. Moreover, the Slave can detect failure of the Master. *Life Guarding* is started with the transmission of the initial message from the Master.

7.8.2 Heartbeat

Similar to life guarding, the Heartbeat function is an additional network supervisory service. But unlike the life guarding, the Heartbeat does not require a NMT-Master. Only CANopen Slaves are able to function as Heartbeat Producer and Consumer because they provide an Object Dictionary in order to store the Heartbeat times.

7.8.3 Heartbeat Producer

The Heartbeat Producer cyclically sends a Heartbeat message. The configured Producer Heartbeat time (16-bit – value in ms), located at index 1017H, will be used as an interval time. If this interval time expires, a message with the following contents will be sent:

Table 17: Heartbeat Message Structure

Byte	**0**	**1...7**
Content	Producer State	reserved

The COB-ID that is used is 0700H + the node address.
The Heartbeat Producer gives its status, which can be any of the following values, in the first byte of the message:

00H BOOTUP
04H STOPPED
05H OPERATIONAL
7FH PRE-OPERATIONAL

The Heartbeat Producer is deactivated when the producer Heartbeat time is set to 0.

7.8.4 Heartbeat Consumer

The Heartbeat Consumer analyzes Heartbeat messages sent from the producer. In order to monitor the Producer, the Consumer requires every producers' node address, as well as the consumer Heartbeat time.

For every monitored Producer, there is a corresponding sub-entry that has the following contents:

Table 18: Structure of a Consumer Heartbeat Time Entry

	MSB		**LSB**
Bit	31-24	23-16	15-0
Value	00H	Node-ID	Consumer Heartbeat Time

The Consumer is activated when a Heartbeat message has been received and a corresponding entry is configured in the OD. If one of the activated Heartbeat times expires during an active Heartbeat consumer without receipt of a corresponding Heartbeat message, then the consumer for this producer is deactivated.

The Heartbeat consumer is completely deactivated when the consumer Heartbeat time is given a value of 0.

7.9 Network Boot-Up

The NMT-Master is responsible for booting of the network. The boot procedure takes place over several steps. According to the type of networked CANopen device, the identifier defaults to pre-defined values (for minimum CANopen devices) or is configured via SDO services. The pre-defined configuration for the identifier values include Emergency Objects, PDOs and SDOs. These are calculated according to node addresses, which can be located between 1 and 128 and are added to a base COB-identifier that determines the function of an object.

Table 19: Calculation of the COB-identifier from the Node Addresses

Bit	**10**			**7**	**6**						**0**
COB-Identifier											
	Function Code				Node-ID						

This base COB- identifier (accoding to CiA DS301 Pre Defined Connection Set) is determined as follows:

Table 20: Base COB-identifier

Object	**Resulting COB-ID [hex]**	**Resulting COB-ID [decimal]**	**Communication Parameter at Index**
EMERGENCY	80H + Node-ID	129 – 255	
PDO1 (tx)	180H + Node-ID	385 – 511	1800H
PDO1 (rx)	200H + Node-ID	513 – 639	1400H
PDO2 (tx)	280H + Node-ID	641 – 767	1801H
PDO2 (rx)	300H + Node-ID	769 – 895	1401H
PDO3 (tx)	380H + Node-ID	896 – 1022	1802H
PDO3 (rx)	400H + Node-ID	1025 – 1050	1402H
PDO4 (tx)	480H + Node-ID	1152 – 1278	1803H
PDO4 (rx)	500H + Node-ID	1281 – 1406	1403H
SDO (tx)	580H + Node-ID	1409 – 1535	
SDO (rx)	600H + Node-ID	1537 – 1663	
Nodeguarding	700H + Node-ID	1793 – 1919	(100EH)

Configuration data can be loaded on Slave devices via the pre-defined SDO. PDOs can be transmitted after nodes are set from

Pre_Operational to *Operational* state by the NMT service *Start_Remote_Node*. Minimum CANopen devices also support the *Stop_Remote_Node*, *Enter_Pre-Operational_State*, *Reset_Node*, *Reset_Communication* services. As indicated in Figure 3, networked nodes automatically enter *Pre_Operational* following boot-up and initialization. The *Reset_Node* service completely resets target nodes. *Reset_Communication* resets communication parameters.

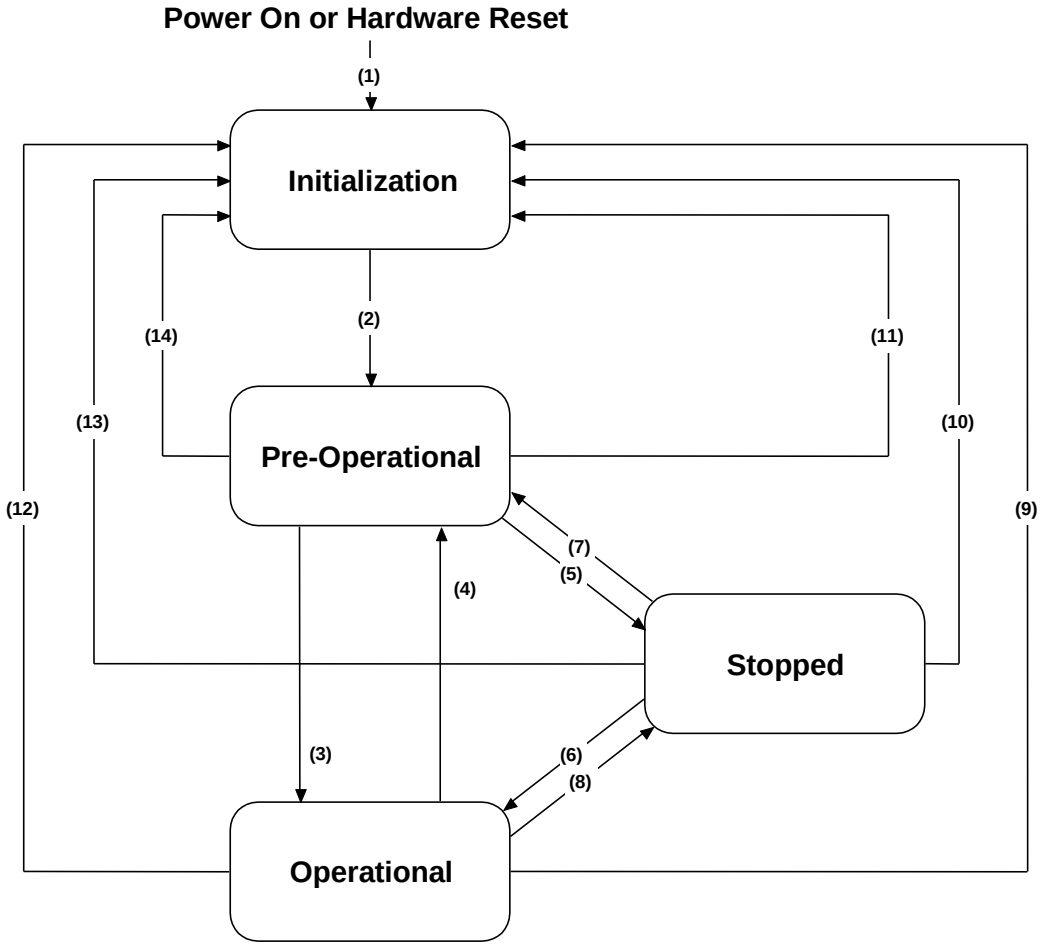

Figure 7: State Diagram of a CANopen Device

Table 21: Description of State Flow Diagram Symbols

State transition	Action required
(1)	following "Power On", automatically switches into "Initialization" state
(2)	"Initialization" finished, automatically switches into "Pre-Operational" state
(3),(6)	NMT service "Start_Remote_Node"
(4),(7)	NMT service "Enter_Pre-Operational_State"

(5),(8)	NMT service "Stop_Remote_Node"
(9),(10),(11)	NMT service "Reset_Node"
(12),(13),(14)	NMT service "Reset_Communication"

7.10 Object Dictionary Entries

Beside the parameters for the PDOs, a number of additional entries in the Object Dictionary belong to the data that specify a CANopen device. The communication profile contains such information as:

- the device type at index [1000H];
- the error register at index [1001H];
- the Pre-Defined Error Field at [1003H];
- the identifier of the SYNC message at [1005H];
- the device name at [1008H],
- the hardware and software version of the manufacturer at [1009H] and [100A];
- the node address at [100BH];
- the parameter Guard Time at [100CH] and
- the parameter Life Time Factor at [100DH].

In the device type, information about the implemented device profile and the capabilities of the device is encoded. The error register gives information about internal errors of the device; the pre-defined error field provides an error log. In case the Guard Time and Life Time Factor are unequal to Zero, the multiplied values result in the Life Time of the CANopen device for the node monitoring protocol.

7.11 PDO Mapping Example

All network variables can be transferred by PDOs, which can transmit a maximum of 8 bytes of information. The allocation of variables to PDOs is defined by mapping tables. These variables are addressable via the Object Dictionary. Reading and writing of entries to the Object Dictionary occurs by means of Service Data Objects (SDO), which are used to configure the network by means of a special configuration tool.

This process is illustrated below in
Table 22. Inputs 2 and 3 of device A are to be transferred to the outputs 1 and 3 of device B. Both devices support complete mapping.

Table 22: PDO Mapping Example

Device A:

1000H	Device Type
.....	
6000H,1	Input 1, 8 Bit
6000H,2	Input 2, 8 Bit
6000H,3	Input 3, 8 Bit
....	

Transmit PDO Mapping Parameter

1A00H,0	# of Entries	2
1A00H,1	1.Map Object	60000208H
1A00H,2	2.Map Object	60000308H

Transmit PDO Communication Parameter:

1800H,0	# of Entries	2
1800H,1	COB-ID	501
1800H,2	Transm. Type	255
....		

Resulting PDO:

COB-ID	DATA	
501	Output 1	Output 3

Transmit and receive PDOs utilize the same CAN identifier 501. Thus device B automatically receives the PDO transmitted by device A. The recipient, device B, interprets the data in accordance with its mapping scheme; it passes the first byte at output 1 and the second byte at output 3. These correspond to inputs 2 and 3, respectively of the transmitting device A.

7.12 Input/Output Assignment to Object Dictionary Entries

The CANopen Chip CoC-100 allows an easy configuration for a specific CANopen application. The fixed number of inputs and outputs on the CANopen Chip CoC-100 makes easy configuration of Process Data Objects (PDOs) possible. Both digital and analog inputs, as well as the digital outputs, are configured in accordance with CiA standards. Configuration of Object Dictionary Input/Output entries for the CANopen Chip CoC-100 **,** according to data type, is shown in *Table 23* .

Table 23: Object Dictionary Input/Output Entries CoC-100

Data Type	Index / Subindex	Size
Digital Input		
DI0 ... DI7	6000H / 1	BYTE
DI8 ... DI15	6000H / 2	BYTE
DI16 ... DI23	6000H / 3	BYTE
Digital Outputs		
DO0 ... DO4	6200H / 1	BYTE
DO8 ... DO15	6200H / 2	BYTE
Analog Inputs		
AI0	6401H / 1	WORD
AI1	6401H / 2	WORD
AI2	6401H / 3	WORD
AI3	6401H / 4	WORD
AI4	6401H / 5	WORD
AI5	6401H / 6	WORD
AI6	6401H / 7	WORD
AI7	6401H / 8	WORD
PWM Outputs		
PWM0	6500H / 1; 6510H / 1	WORD
PWM1	6500H / 2; 6510H / 2	WORD
PWM2	6500H / 3; 6510H / 3	WORD
PWM3	6500H / 4; 6510H / 4	WORD

Note:
After boot-up of the CANopen Chip CoC-100, objects can be accessed via SDOs. If the node is in state *Operational***,** objects can be accessed via PDOs. The default mapping parameters applies for Object Dictionary Input/Output entries. Any modification of mapping parameters can be done via SDO with the help of a network configuration tool.

8 CANopen Chip CoC-100 Operation

8.1 CANopen State Transitions

The structure of messages that changes the state of a CANopen node is as follows:

11-bit CAN Identifier	2 Byte Data	
0	*cs*	*NODE_ID*

Node_ID Node address; Node_ID = 0 to address all devices (Broadcast)
cs Command

Table 25 summarizes all NMT-Master messages used for status control:

Table 25: NMT-Master Messages for Status Control

Command (cs)	Description	Function	State after Execution
1 (01H)	Start_Remote_ Node	Starts the CANopen device and PDO transmission, activates outputs	OPERATIONAL
2 (02H)	Stop_Remote_ Node	Stops PDO transmission, renders outputs in error state	STOPPED or PREPARED
128 (80H)	Enter_Pre_ Operational_ State	Stops PDO transmission, SDO remains active	PRE-OPERATIONAL
129 (81H)	Reset_Node	Executes a system Reset; Initial Start-up, resets all settings to default values	PRE-OPERATIONAL
130 (82H)	Reset_ Communication	Resets all communication parameters to default values	PRE-OPERATIONAL

8.2 Power On

After “Power-On”, the CANopen Chip CoC-100 executes required initialization routines and switches into *Pre_Operational* state.

8.3 PRE-OPERATIONAL

Process Data Objects (PDOs) are not active in *Pre_Operational* state. The default identifier for Service Data Objects (SDOs) is available and all necessary network configurations can be executed via SDO. At the end of the configuration process, the CANopen device can be rendered into *Operational* state. This can be done by the network Master or by the user with the help of a network configuration tool.

8.4 OPERATIONAL

All Process Data Objects (PDOs) can be exchanged in *Operational* state. Access via SDO is also possible.

8.5 STOPPED

Network communication is suspended in state *STOPPED*. This does not affect the Node Guarding and the “Heartbeat“, if this was enabled before. This state can be used to render the application into a “Safety State“. In *STOPPED* state PDO, SDO, SYNC and Emergency communication are **NOT** functioning. Leaving this state is only possible with a NMT message.

8.6 Restart Following Reset / Power-On

Each Reset of the CANopen Chip CoC-100 transmits an Emergency message without data contents. Temporary operational failure of the CANopen Chip CoC-100 and subsequent power-up of the device are detected without Node Guarding (*refer to Section 5.6*), as the sending device can be determined by the message identifier.

The CANopen Chip CoC-100 distinguishes between *"Load"*_Start and *"Save"*_ Start. *"Load"*_Start is necessary:

- for initial operation of the CANopen Chip CoC-100 after its delivery
- if the device parameters (Object Dictionary entries in RAM) should be overwritten by default values

With *"Load"*_Start, all default CANopen Chip CoC-100 Object Dictionary entries are copied to RAM after Reset/Power On (manufacturer default values).

The string "save" must be written to object [1010H] at subindex 1 in order to carry out the *"Save"*_ Start routine. With *"Save"*_ Start all Object Dictionary entries are copied from non volatile memory to RAM after a Reset/Power-On using the saved user-specific values. If the bus Master or the user, by means of a network configuration tool, modifies Object Dictionary entries, then the modifications are only active as of the next RESTART if *"Save"* is written to object [1010H] in subindex 1. This means that only the stored values are valid after the Reset/Power-On of the CANopen Chip CoC-100. These values are stored then in the non volatile memory and, in the event of power-down, are not lost. A *"Save"*_ Start can take up to 1.5 seconds, because the entire OD is read from the non volatile memory and written into RAM.

All device parameters can be stored in the non volatile memory using object [1010H] in subindex 1. In order to prevent unintended storage of parameters in the non volatile memorydevice, a special *"Save"* signature must be written to subindex 1. This 32-bit signature (in hex format) appears as follows:

MSB LSB

'e'	'v'	'a'	's'
65H	76H	61H	73H

All device parameters can be reset to manufacturer default values according to CiA 301 or CiA 401 standards via the object [1011H] in subindex 1. In order to prevent an unintended reset following a store

instruction with the *"Save"* signature, the *"Load"* signature must be written to subindex 1. This 32-bit signature (in hex format) appears as follows:

MSB LSB

'd'	'a'	'o'	'l'
64H	61H	6fH	6cH

In order to set the default values, a Reset/Power-On must be subsequently executed.

8.7 NMT-Boot-Configuration

The CANopen Chip CoC-100 can be configured, that it works as a NMT-Boot-Master for all CANopen-Nodes in the network. The configurations is made in the Object Dictionary at Index [2001H].
Subindex 1 is the entry for NMT-Boot-Enable:

- 0 (default), Node is no Boot-Master
- 1 Node is Boot-Master.

Subindex 2 is the entry for NMT-Start-Time: It is the delay-time [ms] after this the node sends the NMT-Boot-Message (default 500ms, max. 65s).
The entries for NMT-Boot-Configuration are saved into non volatile memory when they are written. They are not used at save and recover the OD *(see also section 8.6*). So the entry has to written directly with the recommended values to configure the CANopen Chip CoC-100 as Boot-Master or not.
The configuration will be active at the next reset of the CANopen Chip CoC-100.

Example for SDO messages for write NMT-Boot-Enable und NMT-Start-Time:

Table 26: SDOs zum Setzen der NMT-Boot-Konfiguration

Action	CAN ID for Node 40H	DLC	SDO Cmd	Index		Sub-index	Value			
set NMT-Boot-Enable	640	8	2F	01	20	01	01	00	00	00
clear NMT-Boot-Enable	640	8	2F	01	20	01	00	00	00	00
set NMT-Start-Time to 3E8H (1000 dec)	640	8	2B	01	20	02	E8	03	00	00

The used CAN-Identifier is 600H + Node-ID. The values are according to CANopen-Standard LSB first.

8.8 Analog Input Operation

8.8.1 Handling Analog Values

This section provides general information on data storage of analog values in a CANopen frame.

The CANopen Standard CiA 401 defines that all analog values till 15 bit have to be stored as 16-bit value aligned left with a sign bit. On the CANopen Chip CoC-100 all A/D-conversion values are stored with 12-bit data. Consequently, for each analog channel, two data bytes must be transmitted.

These data bytes are stored and transmitted on the CAN bus as shown in *Table 27* and

Table 28.

Table 27: Storage of Analog Values Byte 2

Byte 2							
Sign	14	13	12	11	10	9	8
+/-	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5

Table 28: Storage of Analog Values Byte 1

Byte 1							
7	6	5	4	3	2	1	0
Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0	0	0

On the CAN bus, first byte 1 and then byte 2, is transmitted.

8.8.2 Calibrate Analog Values

The CANopen Chip supports the calibration of the analog values. The calibration takes effect to the 16 bit value stored in byte 1 and 2.

The offset could be changed via object 642EH. The default value is '0.0'. This value is added to the result of the A/D-conversion after the gain (pre-scaling value) takes effect (see formula). The values after the decimal point have no effect.

The gain (pre-scaling value) could be changed via object 642FH. The default value is always '1.0'. This value is multiplied with the result of the A/D-conversion (see formula).

$$calibratedResult_{/\square ex} = (result_{/\square ex} \bullet gain_{/float}) + \text{offset}_{/float}$$

The result of the calibration is limited to upper limit (2^{15} - 1) and lower limit 0. Therefore the result can't get negative.

8.8.3 Formula for Calculating the Analog Input Value

The formula listed below is used to calculate a voltage value of an analog input from the A/D-conversion result:

$$AIn_{/V} = \frac{resultA/Dconversion_{/\square ex} \bullet voltagerange_{/V}}{2^{resolutionADC}}$$

The following example will explain the use of this formula in more detail:

over CAN transmitted A/D-value	= 012A0H (4768 dec)
voltage range	= 5V (standard supply on pins VAREF and GND)
logical resolution ADC	= 15 Bit
analog input value (AI)	= 0.727V

For the lowest quantization of the A/D-value the real resolution of the ADC has to be used.

A/D-value	= 01H
voltage range	= 5V (standard supply on pins VAREF and GND)
real resolution ADC	= 12 Bit
lowest resolution	= 1.22 mV/Digit

8.8.4 Selecting the Interrupt Trigger

This object entry determines which event can release an interrupt. For this purpose the object [6421H] "Interrupt_Trigger_Selection" is available. If the "Global_Interrupt_Enable" [6423H] is activated, the release of an interrupt transmits the TX-PDO for analog inputs. A specific subindex is available for each analog input channel. This allows precise configuration of the interrupt event for each channel.

The following values are available:

Table 29: Interrupt Trigger Bits

Bit Number	Interrupt Trigger
0	Upper limiting value exceeded
1	Lower limiting value exceeded
2	Input value fluctuates more than *DELTA [6426H]*
3	*Not supported!*
4	*Not supported!*
5 to 7	Reserved

Example:
6421H,1 = 04H means: the first analog input must fluctuate by more than *DELTA* in order to send the PDO.

Note:
The default values for all analog inputs are set to 04H.

8.8.5 Interrupt Source

This object entry stores which analog input caused the interrupt. The object [6422H] "Analog_Input_Interrupt_Source" is available for this purpose. Every single bit refers to the corresponding analog input channel. These bits will be reset automatically if the entry has been read by a SDO or the object entry was transmitted with a PDO.

The following convention is used:
"1" : Channel caused an interrupt,
"0" : Channel caused no interrupt.

Example:
6422H,1 = 01H means: analog input channel 0 caused an interrupt.

8.8.6 Interrupt Enable

All interrupts can be enabled or disabled using the object entry [6423H] "Analog_Input_Global_Interrupt_Enable". The default value is "0", indicating interrupt execution is disabled. To enable the interrupt execution, the value "1" must be written to the object entry (*also refer to section 8.8.4*).

8.8.7 Interrupt Upper and Lower Limit

An interrupt is released, if the analog input value is higher or lower than the specified limiting value in the applicable subindex. The upper limit is specified in object [6424H], the lower limit in [6425H]. To release an interrupt, the OD entry [6423H] must be set to "1".

Each analog input value will be transmitted as long the trigger condition is given. This assumes that no other trigger condition, such as the Delta Function, is enabled. The limit values must be specified as 32-bit value aligned left.

For this purpose, the objects:
- [6424H] "Analog_Input_Interrupt_Upper_Limit_Integer" and
- [6425H] "Analog_Input_Interrupt_Lower_Limit_Integer"

are available.

Note:
The default value in both entries for all analog inputs is "0".

Example:
6423H = 1, 6421H,1 = 05 and **6424H,1 = 2000H**:

The analog input #1 releases an interrupt if the value exceeds the limit of 2000H, and then the value fluctuates by more than specified in the Delta function (see following section).

8.8.8 Delta Function

The delta function allows configuration of the extent to which an analog input value can fluctuate since the most recent transmission. Only if the fluctuation on the analog input exceeds the value specified in the delta function transmission of the corresponding PDO on the CAN bus is initiated. This configuration can be done using the object [6426H] *Analog_Input_Interrupt_Delta*. Entries specify the number of digits in the conversion result that are allowed to fluctuate. The default value for all four analog inputs is 5. This means that the A/D-conversion result may change by up to 5 digits before a PDO is transmitted. The value must be specified as aligned left and assumes 10-bit resolution.

Note:
The default value for delta for all analog inputs is "A0H".

8.8.9 Example for Trigger Conditions

In the following figure the combination of objects 6424H to 6426H is shown.

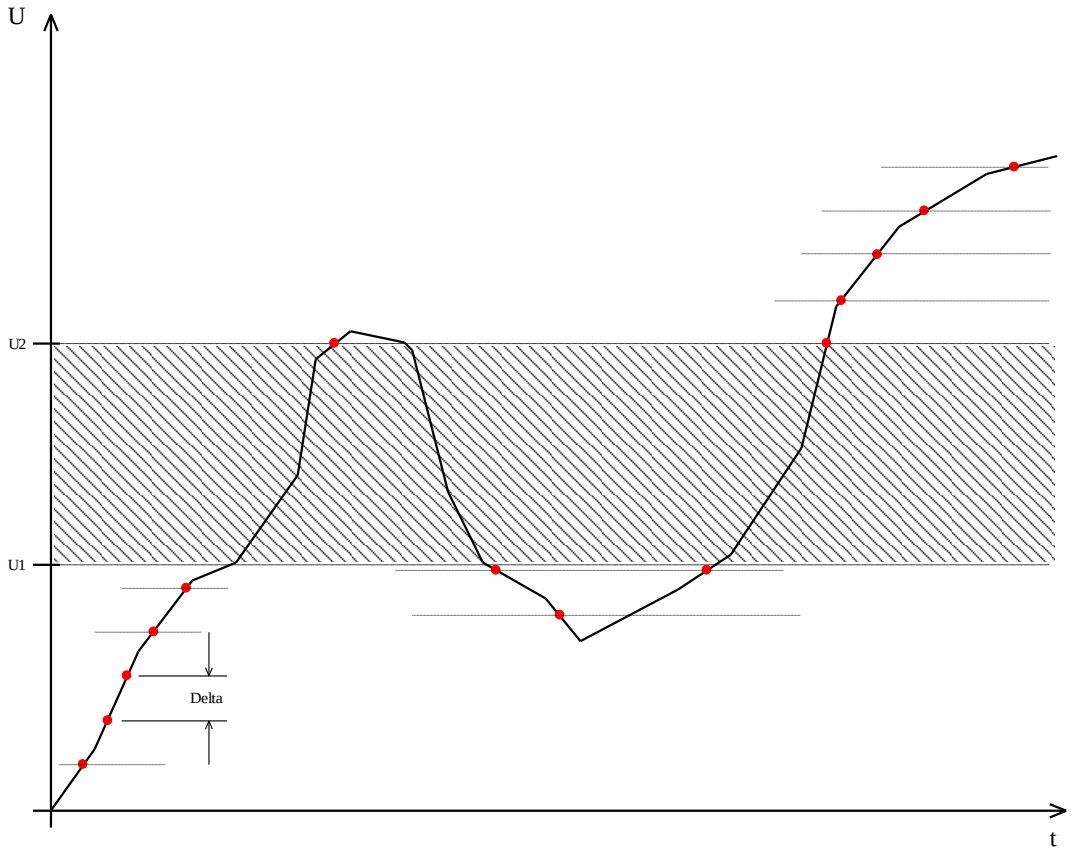

Figure 8: Example Trigger Conditions

The value *U1* was entered at index [6424H] - Lower Limit and *U2* at index [6425H] - Upper Limit in OD. Furthermore the value *Delta* was entered at index [6426H] - Delta. The shown voltage trace is located at an analogue input. At the moments, what are marked with λ , a corresponding PDO will be transmitted from CANopen Chip CoC-100. Is the analogue input vale between U1 and U2 (the hatched area), no PDO will be transmitted.

8.9 Functionality of PWM Outputs

The CANopen Chip CoC-100 can generate PWM-signals. For every Output it exists one OD-entry for period (index [6510H]) and one for duty cycle (Index [6500H]).
Both parameter have the format unsigned 16.

The duty cycle is declared in percent. That means, the value 0H – 0% and 0FFFFH – 100% are corresponding.

The default value for all PWM-Outputs is 0.

The period is declared as a multiple of 1µs. The value of 1000 corresponds with a frequency of 1kHz.

The default value for all PWM-Outputs is 1000.

The lowest adjustable value is 33.

8.10 Emergency Message

In the event of an error, the status of the CANopen Chip CoC-100 is transmitted via a high-priority Emergency Message. These messages consist of 8 data bytes and contain error information. The Emergency Message is transferred as soon as one of the specified errors occurs. A specific Error Message is only transmitted once, even if the recent error is not resolved for a longer period of time. If all error causes are eliminated, then an Error Message with contents “0“ (error eliminated) is transmitted. The structure of the 8-byte Emergency Message is depicted below:

Table 30: Emergency Message

BYTE 0	BYTE 1	BYTE 2	BYTE 3	BYTE 4	BYTE 5	BYTE 6	BYTE 7
Error Code		Error-Register [1001H]	Manufacturer-specific Error Code				

8.10.1 Error Code

The Error Code (byte field 0+1, LSB, MSB) indicates whether an error is present or whether the error has already been eliminated (no error). The following error codes are valid:

- 0000H: no error
- 1000H: global error
- 5000H Hardware Error (Hardware reset was detected)
- 6100H Software Error (Software reset was detected)
- 6101H Software Error (Watchdog reset was detected)
- 6102H Software Error (Stack overrun of microcontroller)
- 8110H CAN-Message lost (busload too high)
- 8120H Device is in Error passive mode
- 8130H: Lifeguard or Heartbeat Error
  In case of a Heartbeat Consumer error, the Node-ID of the failing node is transmitted in the manufacturer-specific error code field.
- 8140H Recovery from CAN-BUSOFF
- 8210H The PDO was not processed due a length error.

8.10.2 Error Register

The Error Register (byte field 2) can contain the following values:

- 81H: Occurrence of a manufacturer-specific error
- 11H: CAN communication error
- 01H: Occurrence of a common error
- 00H: Error has been eliminated - error reset

8.11 Display State at Run and Error LED

The current state of the CANopen Chip CoC-100 is displayed at the both state-LEDs D1 und D2. The functionality of both LED's is defined in standard **CiA 303-3 V1.0** .

8.11.1 Run LED

The green Run LED (D2) displays the NMT State of the device. The *Table 31* shows the different states and their meaning.

Table 31: Run LED States

RUN LED	state	meaning
On	OPERATIONAL	CANopen Chip CoC-100 is in state OPERATIONAL
Blinking relation 50:50	PREOPERATIONAL	CANopen Chip CoC-100 is in state PREOPERATIONAL
Single Flash	STOPPED	CANopen Chip CoC-100 is in state STOPPED
Triple Flash synchronous with the Error LED	Stack overrun	The software caused a stack overrun of the microcontrollers.
Blinking by turns with Error LED	access with LSS	A LSS service is running.
Synchronous fast Blinking with Error LED	Reset to default values	At DIP-Switch is the reset to default values set.

8.11.2 Error LED

The red Error LED (D1) displays the current error state of the CANopen Chip CoC-100. The
Table 32 shows the different states and their meaning.

Table 32: Error Led States

ERROR LED	State	Description
Off	no error	no error on device detected
Single Flash	Warning Limit reached	The Warning Limit in CAN Controller was reached (too much Error Frames on CAN Bus).
Blinking by turns with Run LED	access with LSS	A LSS service is running.
Double Flash	Error Control Event	An error at life guarding, node guarding or Heartbeat was detected.
Triple Flash synchronous with the Run LED	Stack overrun	The software caused a stack overrun of the microcontrollers.
On	Bus Off	The CAN controller is in state "Bus Off".
Synchronous fast Blinking with Run LED	Reset to default values	At DIP-Switch is the reset to default values set.

This side was left blank intentionally.

9 Operation in the Event of Errors

9.1 State of the CANopen Chip CoC-100 in the Event of Errors

The object dictionary entry "Error Behavior" at index [1029H] for CoC-100 can be used to define which state the CANopen Chip CoC-100 should transfer to in case of an error.

The following entries are possible:

0: change state to PRE-OPERATIONAL
1: no change of state
2: change state to STOPPED

These settings over all possible error sources described in *sections 8.10.1*. The entries Output Error (subindex 2) and Input Error (subindex 3) are not supported.
The error code 6102H sets the outputs into error state. This error is critical and it can leave only by reset.

9.2 Output Handling in the Event of Errors

The user can determine how each output is supposed to behave in the event of an error. The outputs are only changed, when at index “Error Behavior” a state change was activated. All errors that are not lead to a state change do not change the outputs.

9.2.1 Digital Outputs

On digital outputs, error handling can be pre-defined via the objects [6206H] (*“Error_Mode_Output_8-Bit“*) and [6207H] (“*Error_Value_Output_8-Bit”*). These entries can be configured by means of a network configuration tool. In the default configuration, the outputs do not change their states in the event of an error.

A value of “1“ at the bit position for an applicable output in the object [6206H] results in writing the bit value (“0“ or “1“) located in the object [6207H] to the corresponding output.

Example for digital outputs:

Table 33: Example for Error Handling digital outputs

Index	Subindex	DO 3	DO 2	DO 1	DO 0	Description
6206H	1	0	0	1	1	Error Mode Output 8-bit
6207H	1	X	X	0	1	Error Value Output 8-b

In the event of an error, the digital output DO0 is set to 1 while DO1 is set to 0. The status of the outputs OUT2 and OUT3 remain unchanged.

9.2.2 PWM Outputs

On PWM outputs, error handling can be pre-defined via the objects [6543H] (*“PWM_Output_Error_Mode“*) and [6544H] (*”PWM_Output_Error_Value”*). These entries can be configured by means of a network configuration tool. In the default configuration, the outputs do not change their states in the event of an error.

A value of “1“ in the object [6543H] results in writing the value located in the object [6544H] to the corresponding output.

Example for PWM outputs:

Table 34: Example for Error Handling PWM outputs

Index	Subindex	PWM0 Case 1	PWM0 Case 2	Description
6543H	1	0	1	PWM Output Error Mode
6544H	1 = 4000H	no change	4000H, 25%	PWM Output Error Value

In the event of an error, in case 1 the PWM output 0 is not changing. In Case 2 the duty cycle changes to 25%.

9.3 Changing from Error State to Normal Operation

In the event of an error, the outputs retain their active values until overwritten (by means of PDO/SDO) by new output values. This requires that the error, such as “Bus Off” or “Life Guarding” error, is eliminated and the CANopen Chip CoC-100 be switched into *Operational* state by a Master *“Start_Remote_Node”* message.

This side was left blank intentionally.

10 Object Dictionary CANopen Chip CoC-100

Table 35: Object Dictionary of the CANopen Chip CoC-100

Index [hex]	Object type	Name	Data type	in PDO mappable
1000	Var	Device Type	Unsigned32	-
1001	Var	Error Register	Unsigned8	-
1003	Array	Error Message	Unsigned32	-
1005	Var	Identifier SYNC-message	Unsigned32	-
1007	Var	SYNC window length	Unsigned32	-
1008	Var	Device name	String	-
1009	Var	Hardware Version	String	-
100A	Var	Software Version	String	-
100C	Var	Guard Time	Unsigned16	-
100D	Var	Life Time Factor	Unsigned8	-
1010	Array	User-Parameter save	Unsigned32	-
1011	Array	Default-Parameter reload	Unsigned32	-
1014	Var	Identifier Emergency	Unsigned32	-
1016	Array	Consumer Heartbeat Time	Unsigned32	-
1017	Var	Producer Heartbeat Time	Unsigned16	-
1018	Record	Identity Object	Identity	-
1029	Array	Error Behavior	Unsigned8	-
1200	Record	1st Server SDO Parameter	SDO Parameter	-
1201	Record	2nd Server SDO Parameter	SDO Parameter	-
1400	Record	RxPDO1 Communication parameter	PDOComPar	-
1401	Record	RxPDO2 Communication parameter	PDOComPar	-
1402	Record	RxPDO3 Communication parameter	PDOComPar	-
1403	Record	RxPDO4 Communication parameter	PDOComPar	-
1600	Record	RxPDO1 Mapping parameter	PDOMapping	-
1601	Record	RxPDO2 Mapping parameter	PDOMapping	-
1602	Record	RxPDO3 Mapping parameter	PDOMapping	-
1603	Record	RxPDO4 Mapping parameter	PDOMapping	-
1800	Record	TxPDO1 Communication parameter	PDOComPar	-
1801	Record	TxPDO2 Communication parameter	PDOComPar	-

Index [hex]	Object type	Name	Data type	in PDO mappable
1802	Record	TxPDO3 Communication parameter	PDOComPar	-
1803	Record	TxPDO4 Communication parameter	PDOComPar	-
1A00	Record	TxPD01 Mapping parameter	PDOMapping	-
1A01	Record	TxPD02 Mapping parameter	PDOMapping	-
1A02	Record	TxPD03 Mapping parameter	PDOMapping	-
1A03	Record	TxPD04 Mapping parameter	PDOMapping	-
2000	Var	I/O Configuration	Unsigned8	-
2001	Var	NMT-Boot-Configuration	Unsigned8	-
2500	Record	for production only		-
6000	Array	PDO Digital Input	Unsigned8	x
6200	Array	PDO Digital Output	Unsigned8	x
6206	Array	Error Mode Digital Output	Unsigned8	-
6207	Array	Error State Digital Output	Unsigned8	-
6401	Record	PDO Analog Input	Integer16	x
6421	Array	Interrupt Trigger Selection	Unsigned8	-
6422	Array	Interrupt Source	Unsigned32	x
6423	Var	Global Interrupt Enable	Boolean	-
6424	Array	Interrupt upper Limit	Integer32	-
6425	Array	Interrupt lower Limit	Integer32	-
6426	Record	Input Interrupt Delta	Unsigned32	-
642E	Array	Analog Input offset float	float	-
642F	Array	Analog Input pre-scaling float	float	-
6500	Array	PWM Pulse	Unsigned16	x
6510	Array	PWM Period	Unsigned16	x
6543	Array	PWM Output Error Mode	Unsigned8	-
6544	Array	PWM Output Error Value	Unsigned16	-

11 Open Issues

none

12 Revision History of this Document

Date	Manual Version	Changes
13/11/2019	L-2365e_01	Initial version
02/24/2020	L-2365e_02	Add Objects 642EH and 642FH Add 4003002 3.3V Version of CoC-100
03/09/2020	L-2365e_03	

This side was left blank intentionally.

Index

This side was left blank intentionally.

Document:	CANopen Chip CoC-100
Document number:	L-2365e_03, Edition March 2020

How would you improve this manual?

Did you find any mistakes in this manual? page

Submitted by:

Customer number: _____

Name: _____

Company: _____

Address: _____

Return to:

SYS TEC electronic AG
Am Windrad 2
D-08468 Heinsdorfergrund
GERMANY
Fax : +49-3765-38600-4100

Published by

SYS TEC ELECTRONIC

Ordering No. L-2365e_03
Printed in Germany