compatible with you

CAN 300 PRO Application Example DeviceNet Slave

CAN Communication Module for S7-300

Application Example for DeviceNet Slave Handling Blocks

Manual

Edition 1 / 11.03.2010

Systeme Helmholz **GmbH** ● Hannberger Weg 2 ● 91091 Grossenseebach ● Germany
Phone: +49 9135 7380-0 ● Fax: +49 9135 7380-110 l E-mail: info@helmholz.de ● Internet: www.helmholz.de

Systeme Helmholz **GmbH**

Hannberger Weg 2, 91091 Grossenseebach, Germany

Note:

We have checked the content of this manual for conformity with the hardware and software described. Nevertheless, because deviations cannot be ruled out, we cannot accept any liability for complete conformity. The information in this manual is regularly updated. When using purchased products, please heed the latest version of the manual, which can be viewed in the Internet at www.helmholz.de, from where it can also be downloaded.

Our customers are important to us. We are always glad to receive suggestions for improvement and ideas.

Revision history of this document:

Edition	Date	Revision
1	11.03.2010	1st version

Contents

1 Overview

1.1 Application and function description

This manual describes the application example (data handling block) of a CAN 300 PRO as a DeviceNet Slave on a S7-300.

This is for use in conjunction with the manual for the CAN 300 PRO module. It is assumed that the reader is familiar with the content of this manual.

The handling block FB 99 described here provides the following DeviceNet functions:

- Allocate/Release
- Get/Set Attribute
- Polled connection
- Cyclic connection (with and without Ack)

1.1.1 Get Attribute

	C/I/A
Max Instance of Class ID 0x01	1/0/2 = 1
Vendor ID	1/1/1 = 999
Device type	1/1/2 = 12
Product code	1/1/3 = 700
Release	1/1/4 = 1.0
Produced connection size (polled)	5/2/7 = length of produced data (ANY)
Consumed connection size (polled)	5/2/8 = length of consumed data (ANY)
Produced connection size (cyclic)	5/4/7 = length of produced data (ANY)
Consumed connection size (cyclic)	5/4/7 = length of consumed data (ANY)

1.1.2 Set Attribute

	C/I/A
Expected packet rate	5/1/9
Expected packet rate (polled)	5/2/9
Expected packet rate (cyclic)	5/4/9
Acknowledge handler object	2B/1/1

Note:

The set attribute jobs are acknowledged positively and not used any further.

1.1.3 Approval

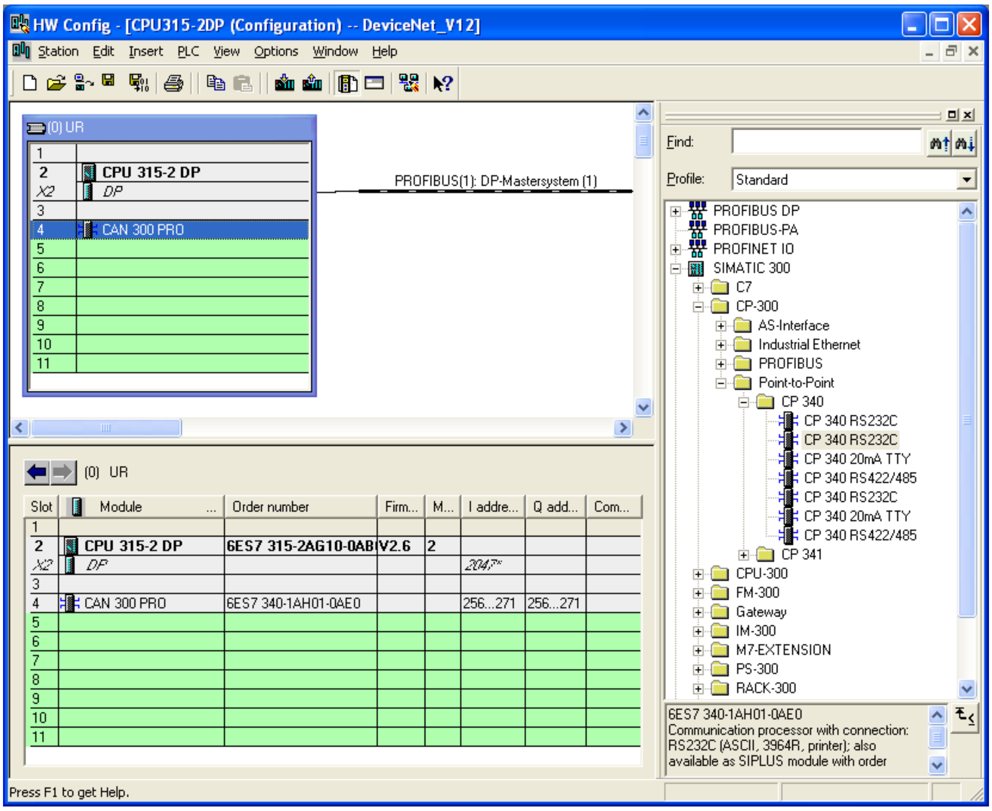
The handling blocks were tested on an Allen-Bradley (scanner) and on an Omron scanner.

They have also been commissioned in an Adept and in a Kuka robot.

If you operate a device other than the scanners named above, we cannot guarantee proper functioning but will be glad to support you with initial startup.

1.2 PLC configuration

In the application example, an S7-300 CPU315-2DP and a CAN 300 PRO module are used.



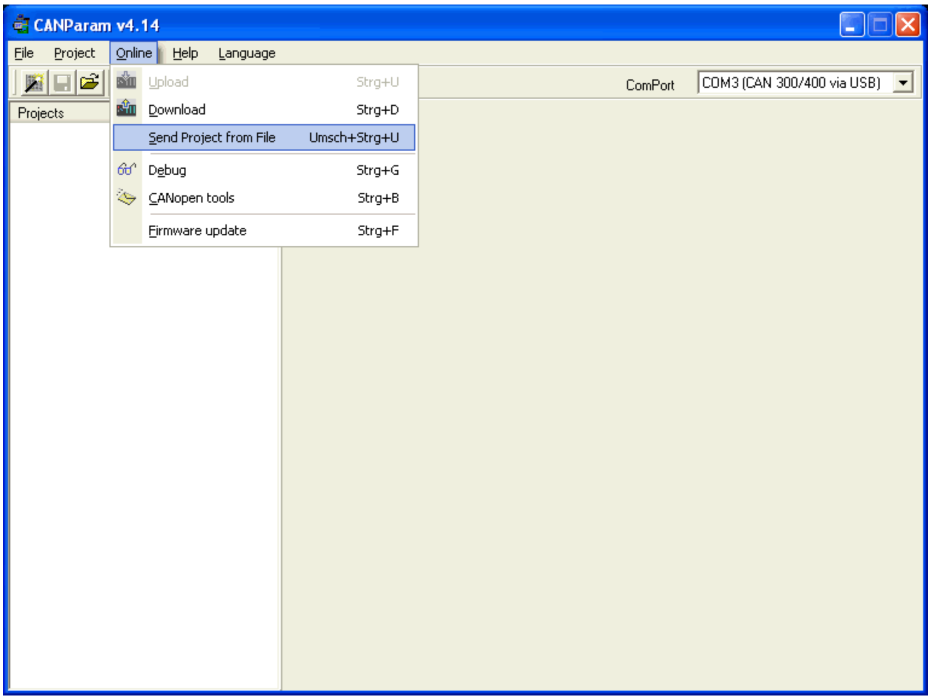
(0) UR

Slot	Module	Order number	Firm...	M...	I addre...	Q add...	Com...
1							
2	CPU 315-2 DP	6ES7 315-2AG10-0AB	V2.6	2			
X2	DP				2047*		
3							
4	CAN 300 PRO	6ES7 340-1AH01-0AE0			256...271	256...271	
5							
6							
7							
8							
9							
10							
11							

6ES7 340-1AH01-0AE0
Communication processor with connection: RS232C (ASCII, 3964R, printer); also available as SIPLUS module with order

2 Configuring the CAN Module

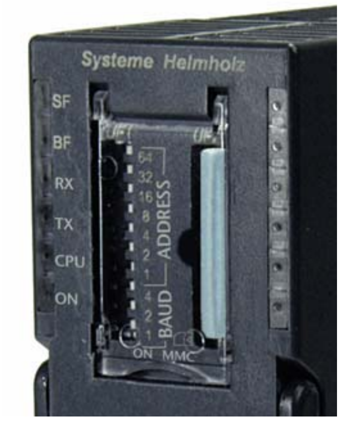
To use the example program, the supplied CAN project “DeviceNet_Slave.PAR” with the CANParam software (Version 4.1x and higher) must be transferred to the CAN 300 PRO module.



The CAN project sets the CAN 300 PRO module to DeviceNet mode.

The address of the DeviceNet slave and the baud rate can now be set at the DIP switch. The address must be between 0-63. The same address must also be set
on the data handling block.

DeviceNet slave address	2^6	+ 64
	2^5	+ 32
	2^4	+ 16
	2^3	+ 8
	2^2	+ 4
	2^1	+ 2
	2^0	+ 1
Baud rate	2^2	+ 4
	2^1	+ 2
	2^0	+ 1



Baud rates:

0	1	2	3	4	5	6	7
10K	50K	100K	125K	250K	500K	800K	1M

3 Programming in the PLC

3.1 Overview

The example contains a data handling block FB 99 that performs the entire DeviceNet protocol processing.

FB 99 must be called in the cycle. Each time the FB is called, frames are fetched from the CAN 300 PRO module, processed and, if applicable, a response sent.

Depending on the length of the PLC cycle and frame frequency on the CAN bus, the FB can process several frames in succession. The maximum number of frames to be processed can be set with the parameter MaxComRequests.

3.2 FB 99 “DN Adapter”

Parameter	Direction	Type	Example
Init	IN	BOOL	FALSE
EnableFB	IN	BOOL	TRUE
Base	IN	INT	256
MacID	IN	INT	10
ProducedData	IN	ANY	P#A 50.0 BYTE 32
ConsumedData	IN	ANY	P#E 50.0 BYTE 32
Cyclic	IN	BOOL	FALSE
CycAck	IN	BOOL	FALSE
ExpectedPacketRate	IN	TIME	T#1s
MaxComRequests	IN	INT	20
MasterRun	OUT	BOOL	M 50.0
DataExchange	OUT	BOOL	M 50.1
Allocate	OUT	BOOL	M 50.2
SendBusy	OUT	BOOL	M 50.3
Done	OUT	BOOL	M 50.4
RcvBusy	OUT	BOOL	M 50.5
NewData	OUT	BOOL	M 50.6
Error	OUT	BOOL	M 50.7
RetVal	OUT	INT	MW 52

Init — Call with TRUE during startup (OB 100) to perform basic initialization; always call with FALSE during the cycle

EnableFB — Enable flag for activating the block

Base — Address of the CAN 300 PRO module

MacID — DeviceNet address of slave, must always correspond to the DIP switch setting.

ProducedData — Any pointer to the receive data

ConsumedData — Any pointer to the transmit data

Cyclic — FALSE: Polled operation / TRUE: Cyclic operation

CycAck — Cyclic connection acknowledged

ExpectedPacketRate — for cyclic operation and Timeout monitoring

MaxComRequests — Number of processed CAN frames for each FB 99 call

MasterRun	Master is running and transmitting data
DataExchange	Communication active
Allocate	Connection established (for one cycle only)
SendBusy	Transmit active
Done	Transmit completed
RcvBusy	Receive active
NewData	Receive completed
Error	One error has occurred (pending for one cycle only)
RetVal	Error number, see Section 3.5

Note:

Data handling block FB 99 calls FCs 65, 66, 67 unconditionally. The FC numbers can therefore only be modified if FB 99 is adapted.

3.3 Example OB 100

In start-up OB 100, FB 99 must be called once with parameter Init = TRUE to perform basic initialization.

Example of call:

```
CALL  FB    99 , DB99
 Init              :=TRUE
 EnableFB          :=TRUE
 Base              :=256
 MacID             :=10
 ProducedData      :=
 ConsumedData      :=
 Cyclic            :=
 CycAck            :=
 ExpectedPacketRate:=
 MaxComRequests    :=
 MasterRun         :=
 DataExchange      :=
 Allocate          :=
 SendBusy          :=
 Done              :=
 RcvBusy           :=
 NewData           :=
 Error             :=
 RetVal            :=
```

3.4 Example OB 1

Example of call:

```
      CALL  FB    99 , DB99
       Init              :=FALSE
       EnableFB          :=TRUE
       Base              :=256
       MacID             :=10
       ProducedData      :=P#I 50.0 BYTE 32
       ConsumedData      :=P#Q 50.0 BYTE 32
       Cyclic            :=FALSE
       CycAck            :=FALSE
       ExpectedPacketRate:=T#1S
       MaxComRequests    :=20
       MasterRun         :=M50.0
       DataExchange      :=M50.1
       Allocate          :=M50.2
       SendBusy          :=M50.3
       Done              :=M50.4
       RcvBusy           :=M50.5
       NewData           :=M50.6
       Error             :=M50.7
       RetVal            :=MW52

//save allocate
      A     M     50.2
      S     M     51.2
//save error
      A     M     50.7
      S     M     51.7
//save retval
      L     MW    52
      L     0
      ==I
      JC    M000
      L     MW    52
      T     MW    54
M000: NOP   0
```

3.5 Return parameter RETVAL

The return parameter RETVAL of the function block can contain both function-specific errors or error numbers of the Siemens system function blocks SFC 58, SFC 59, and SFC 20.

Error codes of DeviceNet handling:

80D1: Receive frame with length > 8

80D5: Scanner attempts to read non-implemented attribute

80D6: Scanner attempts to write non-implemented attribute

80DA: Timeout (3*ExpectedPacketRate), scanner does not respond (poll)

4 Diagnostics

4.1 Process image in the PLC

The CAN 300 PRO module occupies 16 bytes in the input and output process image. The content of the output process image is not used.

The content of the input process image can be used for information purposes by the user in the application.

Byte	Meaning
0	Module status generally, CAN group error display
1	CAN controller status (register of the CAN controller)
2	FIFO status bits (send & receive)
3	CAN controller: TX error counter
4	CAN controller: RX error counter
5	CANopen: Masterstatus
6	CANopen: Assignment of the SDO request mailboxes
7	CANopen: Number of nodes in operational
8	Node ID on use of the bit filter or of the master
9	*reserved*
10	*reserved*
11…15	*used internally*

The input image can only be accessed with the I/O direct access commands: L PIB, L PIW

4.1.1 Byte 0: Module status

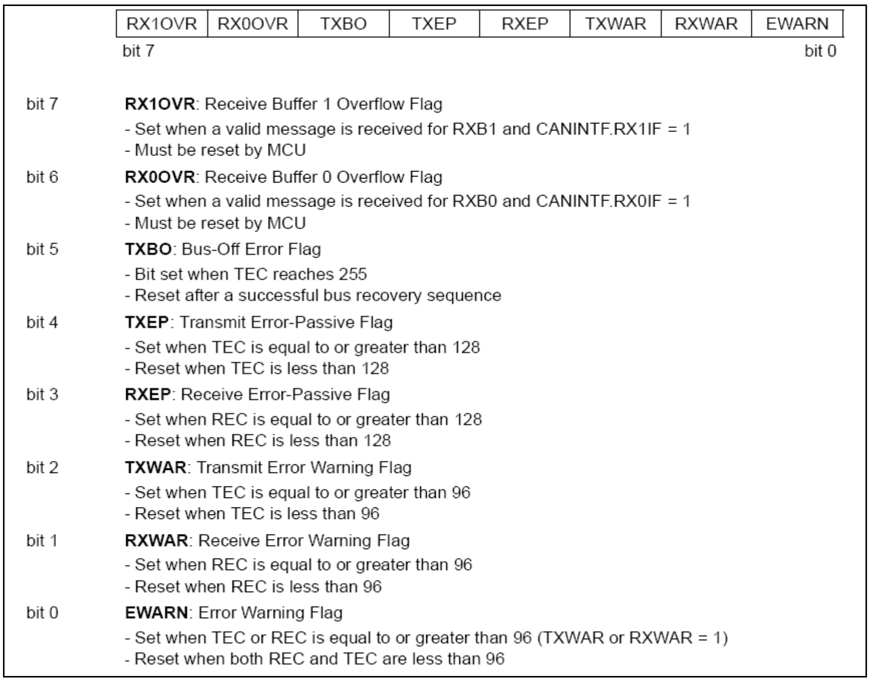
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CAN controller group error	Module is CAN 300 PRO						Module parameterized and running

Bit 0: The CAN 300 PRO module has processed the configuration and is ready for operation.

Bit 6: This bit is always 1 in order to detect the CAN 300 PRO.

Bit 7: Group error bit for errors on the CAN controller, more precise information about the cause of error can be found in byte 1.

4.1.2 Byte 1: Error status (EFLG) of the CAN controller



RX1OVR	RX0OVR	TXBO	TXEP	RXEP	TXWAR	RXWAR	EWARN
bit 7							bit 0

bit 7 **RX1OVR**: Receive Buffer 1 Overflow Flag
- Set when a valid message is received for RXB1 and CANINTF.RX1IF = 1
- Must be reset by MCU

bit 6 **RX0OVR**: Receive Buffer 0 Overflow Flag
- Set when a valid message is received for RXB0 and CANINTF.RX0IF = 1
- Must be reset by MCU

bit 5 **TXBO**: Bus-Off Error Flag
- Bit set when TEC reaches 255
- Reset after a successful bus recovery sequence

bit 4 **TXEP**: Transmit Error-Passive Flag
- Set when TEC is equal to or greater than 128
- Reset when TEC is less than 128

bit 3 **RXEP**: Receive Error-Passive Flag
- Set when REC is equal to or greater than 128
- Reset when REC is less than 128

bit 2 **TXWAR**: Transmit Error Warning Flag
- Set when TEC is equal to or greater than 96
- Reset when TEC is less than 96

bit 1 **RXWAR**: Receive Error Warning Flag
- Set when REC is equal to or greater than 96
- Reset when REC is less than 96

bit 0 **EWARN**: Error Warning Flag
- Set when TEC or REC is equal to or greater than 96 (TXWAR or RXWAR = 1)
- Reset when both REC and TEC are less than 96

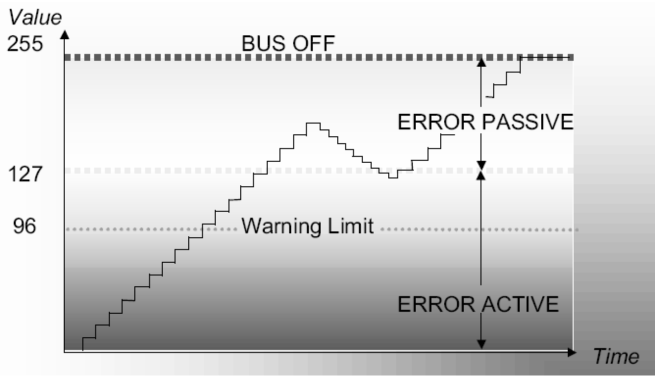
4.1.3 Byte 2: FIFO status bits

Bit 7	Bit 6	Bit 5	Bit 4
Send-FIFO (high) half full	Send-FIFO (high or low) overflow	Send-FIFO (low) half full	Send-FIFOs (high & low) completely empty

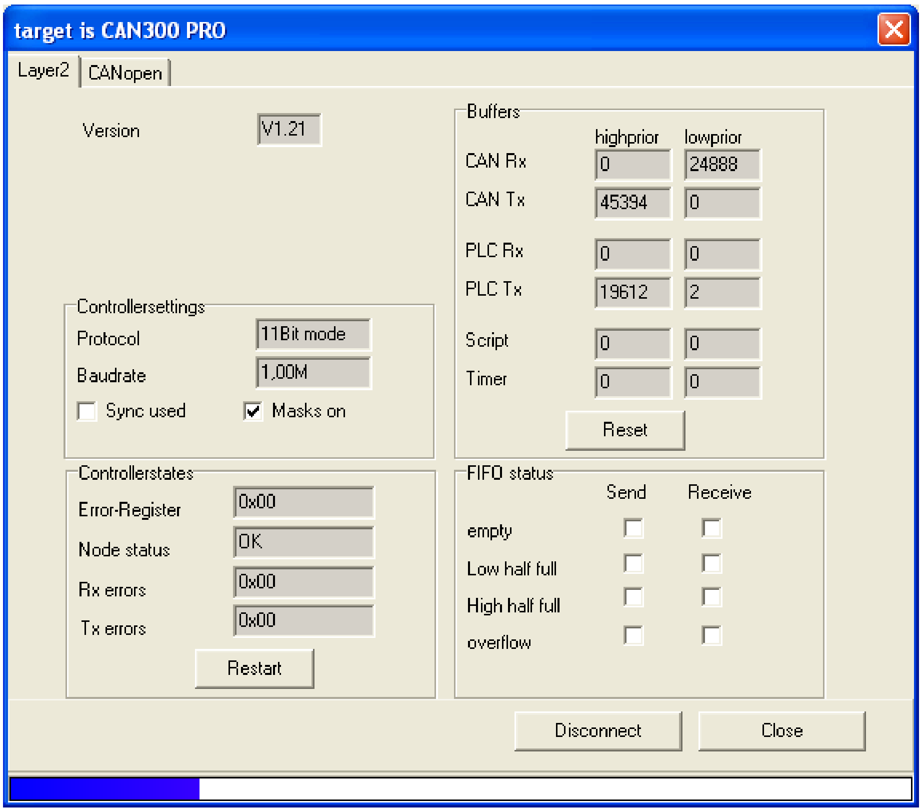
Bit 3	Bit 2	Bit 1	Bit 0
Receive-FIFO (high) half full	Receive-FIFO (high or low) overflow	Receive-FIFO (low) half full	Receive-FIFOs (high & low) completely empty

4.1.4 Byte 3/4: CAN controller Tx/Rx error counter

The error counter is incremented on every CAN frame transmitted or received with an error. If a CAN frame has been correctly transmitted, the error counter is decremented again. If the counter is greater than 96, the CAN controller goes into “warning” mode. If the error counter exceeds 127, the CAN controller goes into “error passive.”



4.2 CANParam debug screen



Version V1.21

Buffers

	highprior	lowprior
CAN Rx	0	24888
CAN Tx	45394	0
PLC Rx	0	0
PLC Tx	19612	2
Script	0	0
Timer	0	0

Controllersettings

Protocol 11Bit mode

Baudrate 1,00M

☐ Sync used ☑ Masks on

Controllerstates

Error-Register 0x00

Node status OK

Rx errors 0x00

Tx errors 0x00

FIFO status

	Send	Receive
empty	☐	☐
Low half full	☐	☐
High half full	☐	☐
overflow	☐	☐

The "Connect" button activates monitoring mode. If you press the button again, the link will be disconnected again.

The debug dialog box provides the following information:

Version Version number of the operating system

Protocol Configured CAN protocol (11bit/29bit)

Baud rate Active CAN baud rate

Controller status Content of the CAN status register:

!

Node status should always be "OK" to ensure fault-free CAN data transmission.

Error register Content of the CAN error register EFLG:

Node status Content of the CAN status register (see above): "OK," "Warning," "Passive," "Bus Off"

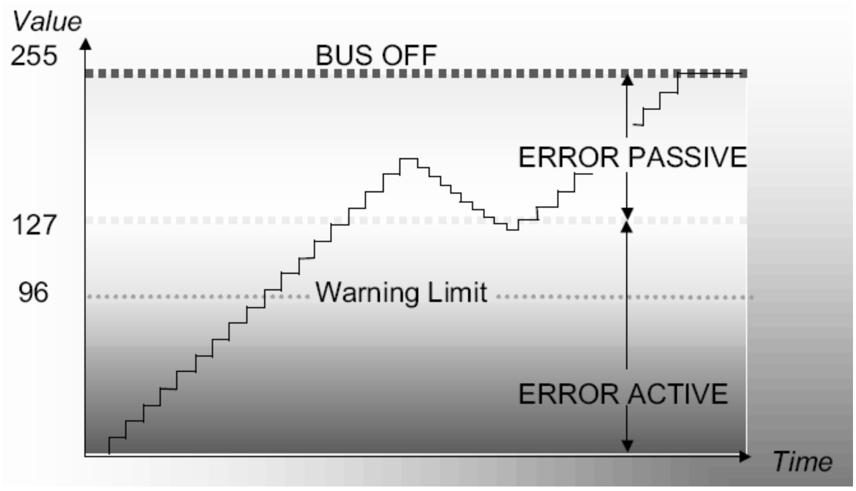
Rx error counter Error counter CAN reception

Tx error counter Error counter CAN transmission

!

The error counters must be "0"; otherwise data transmission on the CAN bus is faulty.

Note: The transmit and receive error counters are incremented by the can controller if transmission or receipt of a frame has failed. As soon as a frame has been correctly sent or received, the corresponding counter is decremented again. These counters should always be 0 if the CAN bus is functioning correctly!



!

The information about the buffers and FIFOs are only relevant in layer 2. In CANopen Master mode, the firmware performs control of the buffers.

Buffer Display of the number of received and transmitted CAN frames in the buffers on the CAN bus, to the PLC and internally

Note: The CAN 300 PRO module has receive and transmit buffers of 400 frames (low priority) and 20 frames (high priority). The counters show how many frames have been processed.

There should never be a big difference between the Rx and Tx counter pairs. However, if this does occur, the CAN frames are not being fetched from the PLC fast enough or are being transmitted to the PLC too fast.

If, in the case of a full FIFO, further frames are received or transmitted, the error bit FIFO overflow is set and the oldest frame is deleted from the FIFO.

FIFO status Display of the filling level of the FIFOs

In the PLC, the FIFO status can be evaluated via the peripheral byte 2. aThe FIFOs can be deleted with data handling block FC 67 CANCTRL.