DP/CAN Coupler

Profibus DP to CAN-Bus Coupler

700-650-CAN01

Manual

Edition 2 / 27.03.2006 HW1 & FW 1.07 and higher

Manual order number: 900-650-CAN01/en

Systeme Helmholz **GmbH** ● Gewerbegebiet Ost 36 ● D-91085 Weisendorf

Tel.: +49 9135 7380-0 ● Fax: +49 9135 7380-50 ● E-mail: info@helmholz.de ● Internet: www.helmholz.de

Systeme Helmholz GmbH

Gewerbegebiet Ost 36, 91085 Weisendorf, 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 data in this manual have been checked regularly and any necessary corrections will be included in subsequent editions. We always welcome suggestions for improvement.

Contents

1 Safety Information

Please observe the safety information given for your own and other people's safety. The safety information indicates possible hazards and provides information about how you can avoid hazardous situations.

The following symbols are used in this manual.

Caution, indicates hazards and sources of error

gives information

hazard, general or specific

*danger of **electric shock***

1.1 General

The DP/CAN coupler is only used as part of a complete system.

The operator of a machine system is responsible for observing all safety and accident prevention regulations applicable to the application in question.

During configuration, safety and accident prevention rules specific to the application must be observed.

Emergency OFF facilities according to EN 60204 / IEC 204 must remain active in all modes of the machine system. The system must not enter an undefined restart.

Faults occurring in the machine system that can cause damage to property or injury to persons must be prevented by additional external equipment. Such equipment must also ensure entry into a safe state in the event of a fault. Such equipment includes electromechanical safety buttons, mechanical interlocks, etc. (see EN 954-1, risk estimation).

Never execute or initiate safety-related functions using the operator terminal.

1.2 Restriction of access

Only authorized persons must have access to the modules!

The modules are open equipment and must only be installed in electrical equipment rooms, cabinets, or housings. Access to the electrical equipment rooms, barriers, or housings must only be possible using a tool or key and only permitted to personnel having received instruction or authorization. See also Chapter 1.5.

1.3 Information for the user

This manual is addressed to anyone wishing to configure or install the DP/CAN coupler.

It is intended for use as a programming manual and reference work by the configuring engineer. It provides the installing technician with all the necessary data.

The DP/CAN coupler is intended for use with a Profibus DP network only. For that reason, the configuring engineer, user, and installing technician must observe the standards, safety and accident prevention rules applicable in the particular application. The operator of the automation system is responsible for observing these rules.

1.4 Use as intended

The DP/CAN coupler must only be used as a communication system as described in the manual.

1.5 Avoiding use not as intended!

Safety-related functions must not be controlled via the DP/CAN coupler alone.

2 Installation and Mounting

The DP/CAN 300 coupler must be installed according to VDE 0100 IEC 364. Because it is an “OPEN type” module, you must install it in a (switching) cabinet. Ambient temperature: 40 °C – 60 °C.

Before you start installation work, all system components must be disconnected from their power source.

*Danger of **electric shock!***

During installation, application-specific safety and accident prevention rules must be observed.

2.1 Vertical and horizontal mounting

The modules can be mounted either vertically or horizontally.

Permissible ambient temperature:

0 to 60 °C

3 System Overview

3.1 Application and function description

The DP/CAN coupler from System Helmholz GmbH allows you to connect any CAN stations to the Profibus DP.

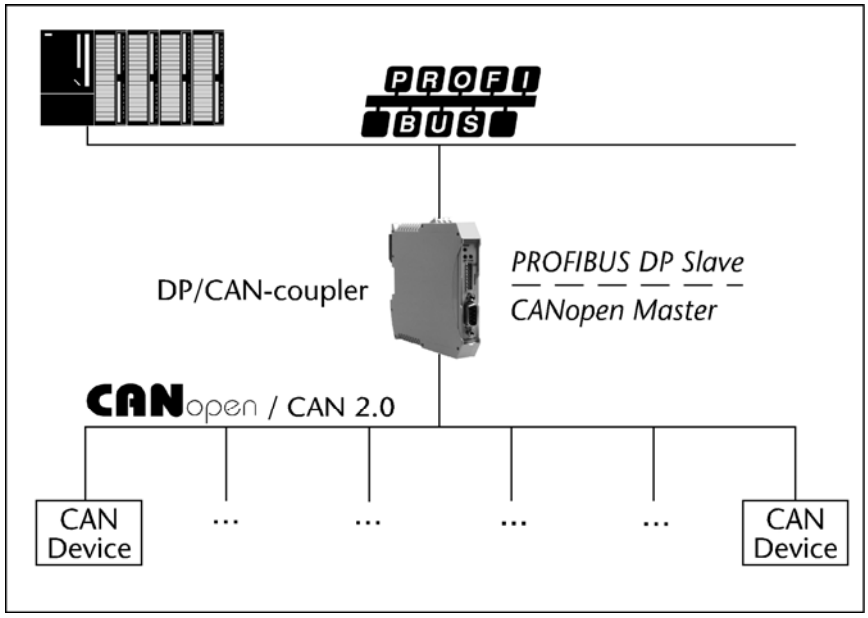
The DP/CAN coupler must be parameterized as a Profibus station in the Hardware Configurator. The necessary GSD files are supplied with the device.

The PROFIBUS side is configured as a DP slave. The interfaces meet EN 50170 and are electrically isolated. The baud rate of 9.6kBaud to 12Mbaud is detected automatically. The maximum volume of input and output information is 320 bytes (which corresponds to 10 CAN modules with 2 PDOs of 8 bytes each in both the transmit and receive direction).

Die CANopen side is configured as an independent master that can be controlled via the PROFIBUS. This allows up to 15 CANopen slave modules to be operated in one CANopen network according to CiA standard DS-301 Version 3.0. All parameterized modules are detected, started up, and monitored for their operating status by the CANopen master. Up to 5 transmit PDOs and 5 receive PDOs can be managed per node for data exchange. Emergency frames from the CANopen nodes are processed by the DP/CAN coupler and forwarded to the Profibus master. Any SDOs can be transmitted or received to parameterize the CANopen slave modules.

Alternatively, the DP/CAN coupler can also be used on Layer 2. The CAN messages that are displayed in the Profibus are freely selectable.

The CAN bus interface meets ISO/DIN 11898-2 and is electrically isolated.



3.2 Connections

Profibus 9-way Sub-D socket:

Pin	Profibus DP
1	-
2	
3	Data line B
4	-
5	GND
6	VP (power supply for terminating resistors)
7	-
8	Data line A
9	-

3-way CAN connector (no terminating resistor):

1	CAN High
2	CAN-GND
3	CAN-Low

3-way power supply:

1	GND
2	V-
3	V+

3.3 LED displays

The three LEDs on the front of the module inform you about its operating state.

LED Power (green): Continuous light indicates the Profibus is running and the CPU is in run. Slow blinking indicates the CPU is in stop.

LED DP (green): A parameterization error on the Profibus has occurred.

LED CAN (yellow): CAN frames are being received from the CAN bus.

3.4 DIP switches

Changing the DIP switches is only effecting after power on!

The 8-switch DIP switch on the housing front is used for setting the Profibus address of the device and the CAN operating mode.

The switches are counted from bottom to top.

"Layer 2" mode is currently not supported!

Switch	Function		
8	Mode	ON=CANopen	OFF=Layer 2
7	Profibus address	2^6	+ 64
6		2^5	+ 32
5		2^4	+ 16
4		2^3	+ 8
3		2^2	+ 4
2		2^1	+ 2
1		2^0	+ 1

3.5 Scope of supply

DP/CAN Coupler	700-650-CAN01

incl. 2 x 3-pin connector for CAN bus and 24V power supply

CD with GSD file examples and instructions

3.6 Accessories

Manual, German/English	900-650-CAN01
CAN bus plug connector	700-690-0BA11
CAN bus plug connector with cable connector	700-690-0BB11
CAN bus connector axial	700-690-0CA11

4 Configuration (CANopen Master)

The DP/CAN coupler allows you to convert data from a PROFIBUS DP circuit to a CAN circuit and vice versa.

Before data can be exchanged between the PROFIBUS and the CAN bus, the start-up phase of the Profibus of the DP/CAN coupler must be parameterized. This is done automatically by the Profibus master (CPU) when the bus starts up.

The configuration of the CANopen bus (baud rate, number of nodes, distribution of PDOs) must be stored as a Profibus node in the parameter set of the DP/CAN coupler when the DP/CAN coupler is defined.

When parameterization is complete and the CPU has started, the CAN modules are started up and monitored. The status of the CAN modules is displayed in the diagnostics area of the process image. The emergency frames from the CAN bus are also displayed here.

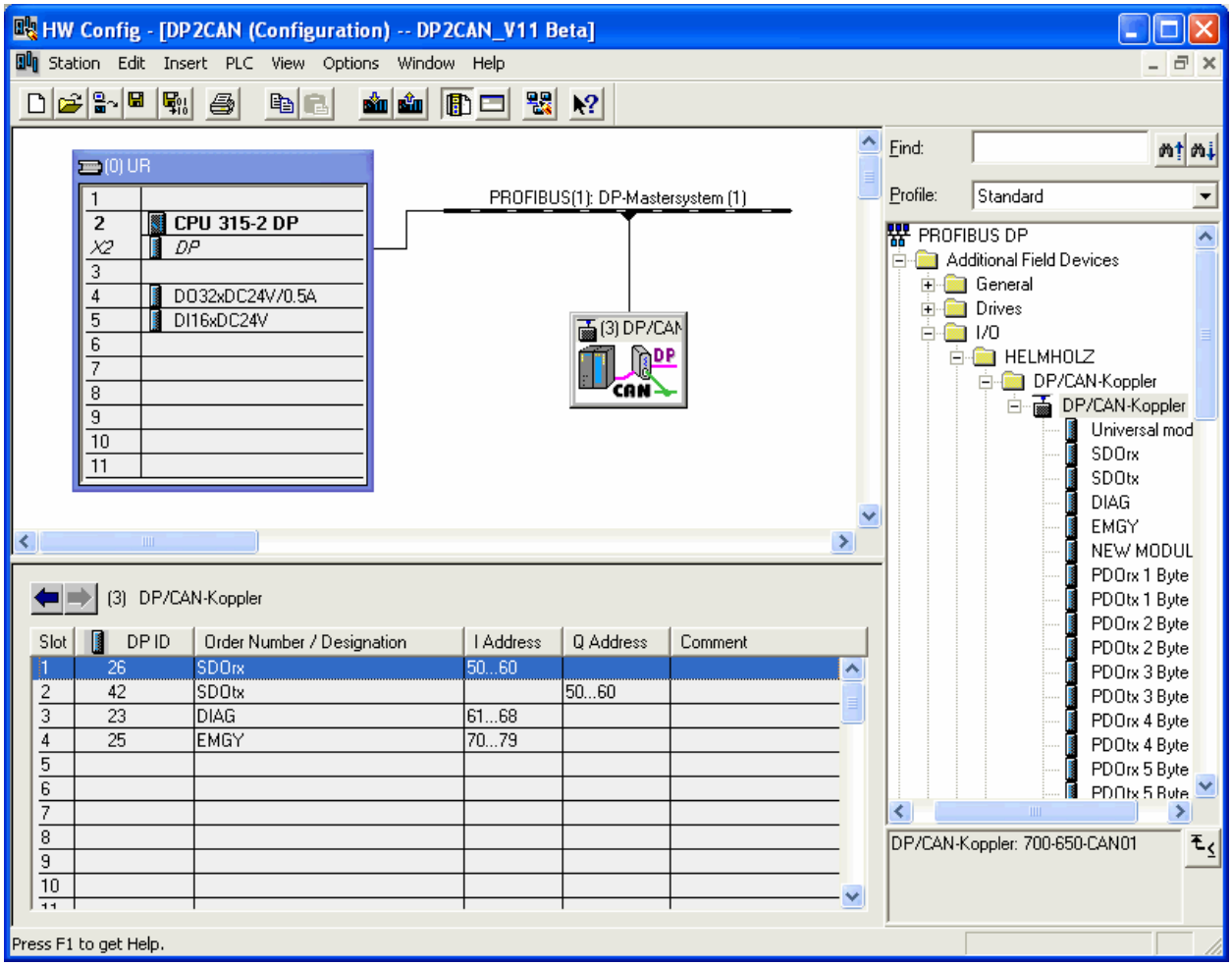
CANopen modules can be parameterized with SDO frames in an SDO window of the process image. The input and output data are also updated on each configured guarding cycle (request from PDOs).

4.1 Install and parametrize the device

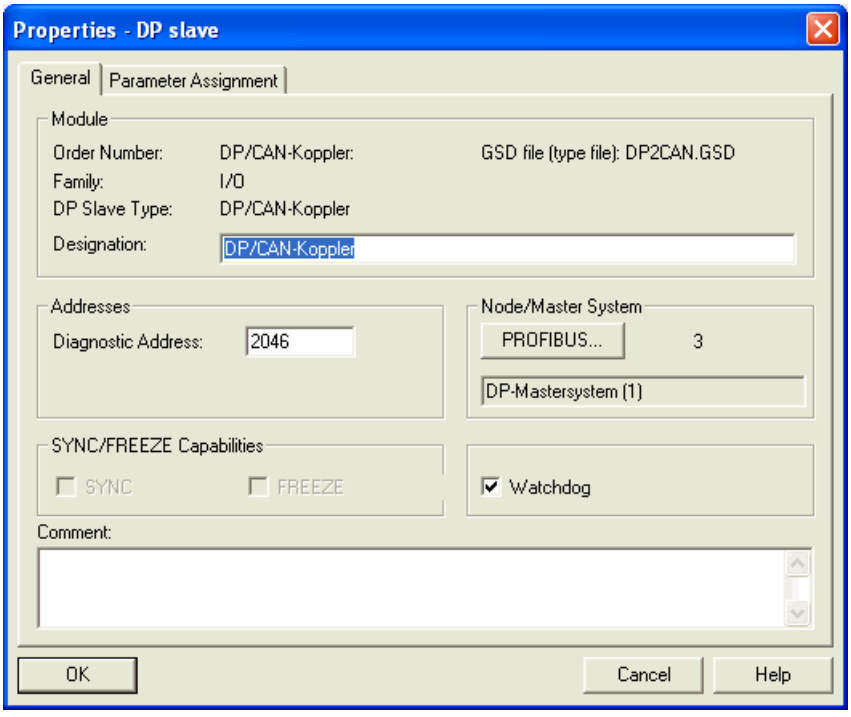
Before you can use the DP/CAN coupler in the Hardware Configurator you must install the supplied GSD file “DP2CAN.GSD”. You can do this in the Hardware Configurator under menu item “Options / Install GSD Files”.

Having done that you will find the DP/CAN coupler in the hardware catalog under ‘Additional FIELD DEVICES / IO / Helmholz’.

You can now drag and drop the “DP/CAN coupler” onto a Profibus network you have already set up.



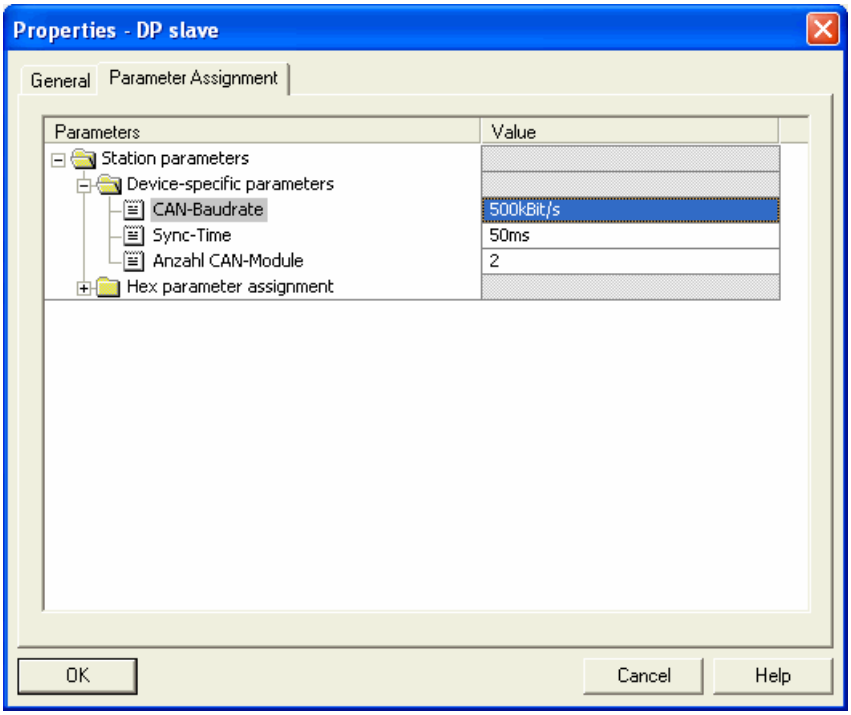
Then assign a suitable station address to the slave.



The DP/CAN coupler is supplied with the necessary information about the structure of the CANopen bus from the master via the parameterization frame during start-up.

The following CAN parameters are defined here:

- CAN baud rate
- Transmission time for the SYNC frame
- Number of used CANopen nodes in the CAN circuit



CAN baud rate:

Possible baud rates: 1 Mbps, 500 Kbps, 250 Kbps, 125 Kbps, 100 Kbps, 50 Kbps, 20 Kbps, 10 Kbps

Sync time:

The transmission time of a Sync frame (COB-ID: 80) on the CAN bus is set here in units of 1 ms (10 ms to 64 ms possible),

Number of CAN modules:

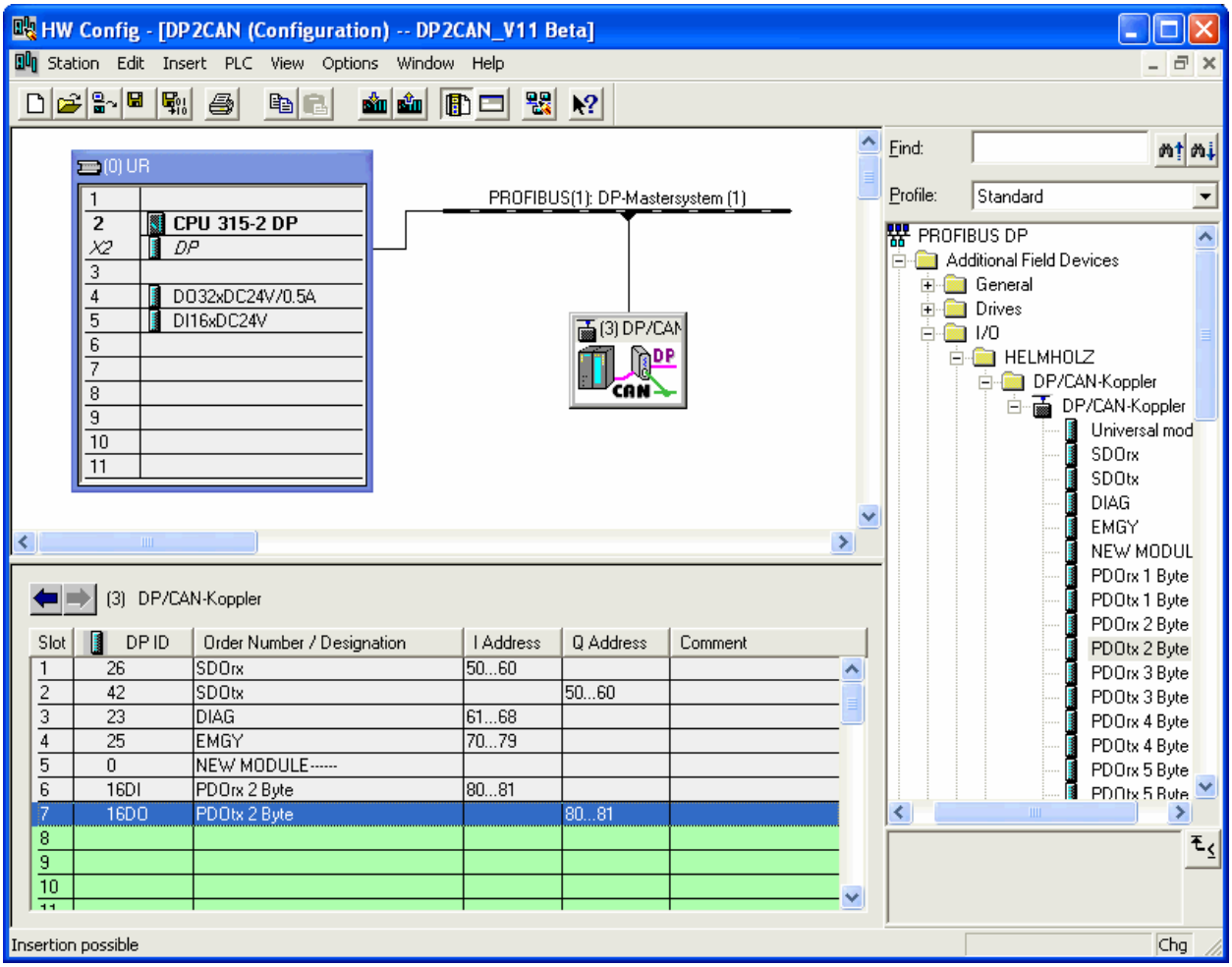
Specifies the number of CAN nodes in the CAN circuit (value range: 1..15).

The number of CAN modules must match the number of defined "NEW MODULE------" entries!

4.2 Defining the I/O address area in the PLC

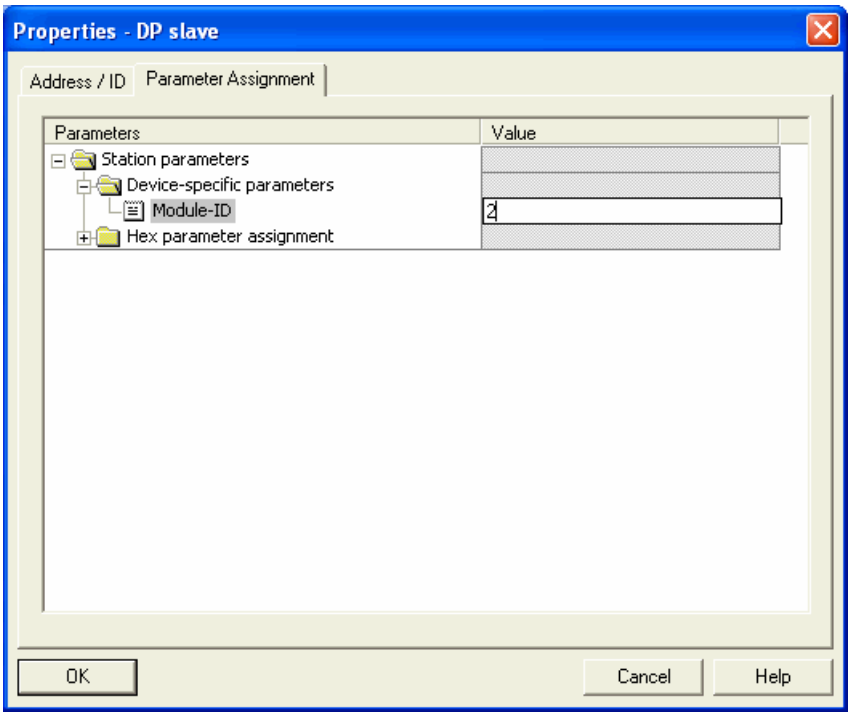
Once the basic parameters of the CAN bus have been defined all data elements must be shown in the I/O area of the CPU.

Elements “SDOrx”, “SDOtx”, “DIAG”, “EMGY” must always be at the beginning of the list. The I/O addresses are freely selectable.



4.3 Parametrization of modules

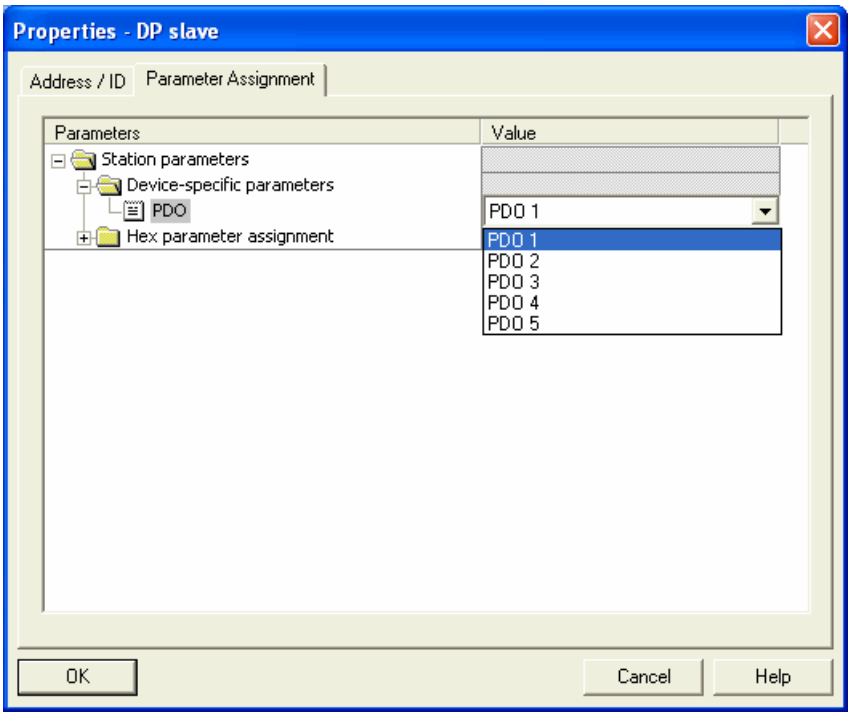
Now the CAN modules and the PDOs belonging to the module can be defined on the remaining slots of the DP/CAN coupler. Entry “NEW MODULE----”.



The number of modules (“NEW MODULE” entries) must correspond to the number defined in the device parameterization.

Incorrect definition of the number of modules results in data corruption or bus start-up errors!

This is followed by the PDO elements. Here, too, the I/O addresses are freely selectable.



5 Programming (CANopen Master)

5.1 Data exchange

When the master has detected that parameterization and configuration is successfully completed without errors at the end of the start-up phase, it starts transmitting data exchange frames. For that, the PROFIBUS master transmits all the data of the parameterized transmit identifier to the DP/CAN coupler every cycle.

Whenever a data item is changed (on the CANopen side or DP side) it transmitted immediately in the next cycle. In addition, the input data of the CANopen modules are updated every guarding cycle (default 1s).

5.2 CAN network start-up procedure

Start-up is initiated by a Stop → Run state transition of the PLC and is executed as follows:

1. Read in the guarding COB-ID (object 0x100E, 0x00) of all modules (1 to 127)
2. Read in the emergency COB-ID (object 0x100E, 0x00) of all parameterized modules
3. Read in the PDO Rx COB-IDs (object 0x1400ff., 0x01) of all parameterized modules and PDOs
4. Read in the PDO Tx COB-IDs (object 0x1800ff., 0x01) of all parameterized modules and PDOs
5. Read in the PDO Rx transmission type (object 0x1400ff., 0x02) only when PDO valid
6. Read in the PDO Tx transmission type (object 0x1800ff., 0x02) only when PDO valid
7. Write the guard time (object 0x100C, 0x00) to the active modules (default: 1 s)
8. Write the lifetime factor (object 0x100D, 0x00) to the active modules (default: 2)
9. Wait for 100 ms
10. Transmit NMT start frame (COB-ID: 000, data: 0x01, 0x00)
11. Node guarding of modules starts
12. Transmit SYNC frames (if parameterized)
13. In each guarding cycle, in addition to the guarding request the current status of the inputs (if available) is queried and the outputs (if available) are updated.

After a power-off, the DP/CAN Coupler always requires a PLC Stop → Run transition before it can start up again!

If a CAN module signs on again after a failure it is automatically parameterized and included in cyclic operation again.

5.3 Diagnostics area

The diagnostics area consists of 8 bytes. It states which area in the process image is invalid and which CAN module is not in the operational state.

Diagnostics area:

Byte	Bit	Function
0	**0**	Entry 1 in process image invalid
	1	Entry 2 in process image invalid
	2	Entry 3 in process image invalid
	3	Entry 4 in process image invalid
	4	Entry 5 in process image invalid
	5	Entry 6 in process image invalid
	6	Entry 7 in process image invalid
	7	Entry 8 in process image invalid
…		…
5	**0**	Entry 41 in process image invalid
	1	Entry 43 in process image invalid
	2	Entry 44 in process image invalid
	3	Entry 45 in process image invalid
	4	Entry 46 in process image invalid
	5	Entry 47 in process image invalid
	6	Entry 48 in process image invalid
	7	Entry 49 in process image invalid
6	**0**	*Reserved*
	1	CAN module 1 not operational
	2	CAN module 2 not operational
	3	CAN module 3 not operational
	4	CAN module 4 not operational
	5	CAN module 5 not operational
	6	CAN module 6 not operational
	7	CAN module 7 not operational
7	**0**	CAN module 8 not operational
	1	CAN module 9 not operational
	2	CAN module 10 not operational
	3	CAN module 11 not operational
	4	CAN module 12 not operational
	5	CAN module 13 not operational
	6	CAN module 14 not operational
	7	CAN module 15 not operational

5.4 Receiving emergency frames

The EMERGENCY frames received from the CAN bus are displayed in the Emergency receive mailbox. The first EMERGENCY frame to be received from the CAN bus is copied directly to the Emergency receive mailbox, all other EMERGENCY frames are temporarily stored in a circular buffer in the DP/CAN coupler and only written to the Emergency receive mailbox when the previous EMERGENCY frame has been read by the user. That is why handshaking is necessary.

5.4.1 Emergency receive mailbox

The Emergency receive mailbox is structured as follows:

Byte	Function
0	CAN module node ID
1	Date length of Emcy frame (0 to 8 bytes)
2	Data byte 1
3	Data byte 2
4	Data byte 3
5	Data byte 4
6	Data byte 5
7	Data byte 6
8	Data byte 7
9	Data byte 8

5.4.2 Handshaking for emergency frames

The handshake bits are located in the high byte of byte 0 in the SDO frame.

Bit 4 (in byte 0 receive SDO) set → new emergency frame in emergency area.

Bit 5 (in byte 0 receive SDO) set → data in emergency area are valid.

Bit 6 (in byte 0 transmit SDO) set → data have been processed by PLC program.

Sequence of operation:

1. If Bit 4 = 0 and Bit 6 = 0 and new EMERGENCY frames have been received, the first EMERGENCY frame is written to the emergency area.
2. Bit 4 and bit 5 are set
3. If the PLC has processed the emergency frame, bit 6 = 1 must be set and the frame acknowledged.
4. If bit 6 = 1, the DP/CAN coupler clears the emergency area and resets bit 4 and bit 5.
5. The PLC waits for reset bits 4 and 5 and then resets bit 6

5.5 Parameterizing CAN modules (SDO transfer)

SDO communication (SDO = Service Data Object) is a confirmed and acknowledged service of the CAN open protocol. Every SDO request is acknowledged with a response from the addressed module. If the module does not respond a timeout message is sent by the DP/CAN coupler to the PLC.

SDOs are usually needed for setting parameters (write request SDO) or querying (read request SDO) a module (CANopen slave) and are not suitable for fast process data transmission that is processed via PDOs (Process Data Objects).

The entire SDO communication is processed by the DP/CAN coupler by means of an 11-byte window in the process image.

5.5.1 Expedited SDO transfers (up to 4 bytes of data)

New requests are written to the I/O area of **SDOtx**. The responses can be read from the **SDOrx** area.

It is only possible to process one request at a time. A response must be awaited after each request.

Only expedited SDO transfer with up to 4 bytes is supported at present.
Support of SDO transfers with more than 4 bytes is available on request.

CAL WRITE REQ (transmit SDO Write):

SDOtx	Function
0	Handshake byte
1	Frame type: 36
2	Node ID (1..127)
3	SDO index (high byte)
4	SDO index (low byte)
5	SDO subindex
6	Number of valid data bytes (size of the SDO)
7	Data byte 1
8	Data byte 2
9	Data byte 3
10	Data byte 4

CAL WRITE REQ (transmit SDO Read):

SDOtx	Function
0	Handshake byte
1	Frame type: 39
2	Node ID (1..127)
3	SDO index (high byte)
4	SDO index (low byte)
5	SDO subindex

The response of the slave must be awaited after each request! SDO transfers can only be processed one after the other!

CAL_READ_CNF_P (SDO Read Response):

SDOrx	Function
0	Handshake byte
1	Frame type: 40
2	Node ID (1..127)
3	Data length 4-7 bytes
4	SDO index (high byte)
5	SDO index (low byte)
6	SDO subindex
7	Data byte 1
8	Data byte 2
9	Data byte 3
10	Data byte 4

CAL_READ_CNF_N (SDO Read negative Response):

SDOrx	Function
0	Handshake byte
1	Frame type: 41
2	Node ID (1..127)
3	Data length 7 bytes
4	SDO index (high byte)
5	SDO index (low byte)
6	SDO subindex
7	Error_Class
8	Error_Class
9	Additional _Class (high byte)
10	Additional _Class (low byte)

CAL_WRITE_CNF_P (SDO Write Response):

SDOrx	Function
0	Handshake byte
1	Frame type: 37
2	Node ID (1..127)
3	Data length 3 bytes
4	SDO index (high byte)
5	SDO index (low byte)
6	SDO subindex

CAL_WRITE_CNF_N (SDO Write negative Response):

SDOrx	Function
0	Handshake byte
1	Frame type: 38
2	Node ID (1..127)
3	Data length 7 bytes
4	SDO index (high byte)
5	SDO index (low byte)
6	SDO subindex
7	Error_Class
8	Error_Class
9	Additional _Class (high byte)
10	Additional _Class (low byte)

5.5.2 SDO timeout

This frame appears in the receive area (SDOrx) if no response is received from the slave after 200 ms.

SDOrx	Function
0	Handshake byte
1	Frame type: 240
2	Node ID (1..127)

5.5.3 Handshaking SDOtx (transmit SDO)

The handshake for sending SDO requests is performed in the first byte of the **SDOtx** area. The PLC program sets and resets these bits.

Bit 0 set: Data are valid, send SDO frame

Bit 0 reset: Data are invalid, wait for next SDO frame. This bit must be set to 0 between two SDO frames.

5.5.4 Handshaking SDOrx (receive SDO)

The handshake for receiving SDO responses to current SDO requests is performed in the first byte of the **SDOrx** area. The PLC program evaluates these bits.

Bit 0 set: SDO transfer is active

Bit 1 set: valid data in receive buffer

Bit 2 set: SDO frame sent

Once bit 1 has been set the receive frame can be processed. Then bit 0 of the SDOtx handshake byte must be reset so that a new request can be started.

6 Configuration (CAN Layer 2)

In Layer 2 mode, the DP/CAN coupler can transmit and receive any CAN messages (CAN 2.0A, 11 bits).

There are two different transmission methods. In the first method, the identifier and size of the CAN message is permanently parameterized in predefined transmit and receive objects and only the data are transmitted via the Profibus. Each predefined transmit and receive object therefore corresponds to one particular CAN frame.

In the second method, a variable receive object can receive several messages filtered by a parameterizable acceptance mask. With this method, not only the data but also the identifier and the length of the CAN frame are transmitted via the Profibus to the PLC application.

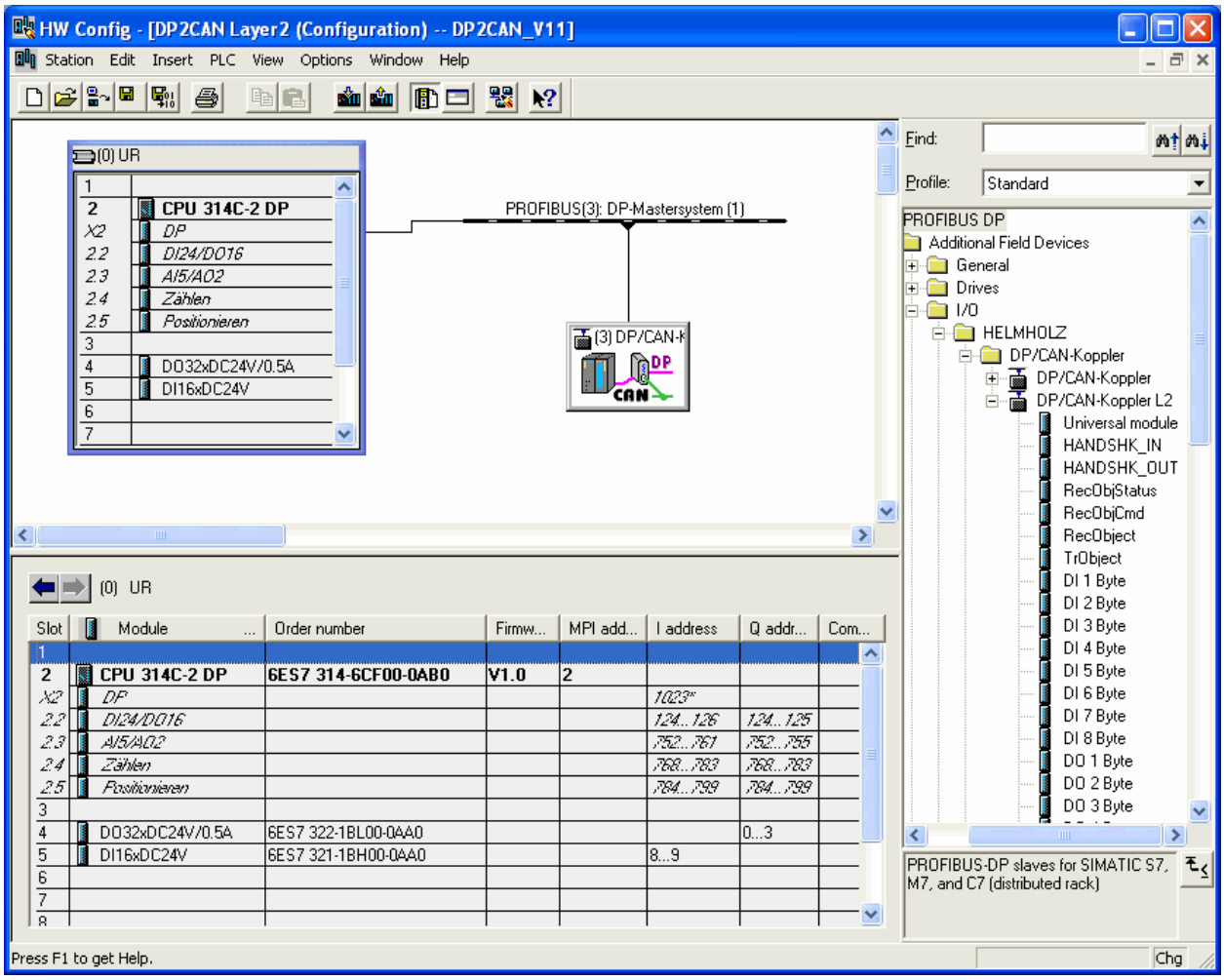
Any number of CAN messages can be sent in a variable transmit object. Moreover, a variable transmit object can also transmit the message cyclically at fixed intervals.

6.1 Installing and parameterizing the device

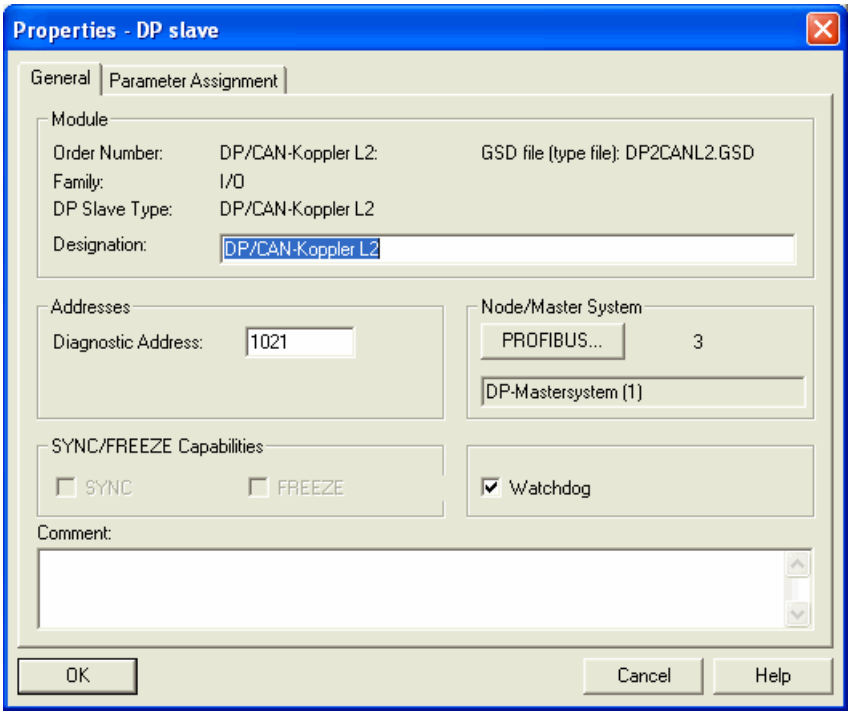
Before you can use the DP/CAN coupler with Layer 2 in the Hardware Configurator you must install the GSD file “DP2CANL2.GSD” supplied. You can do this in the Hardware Configurator under menu item “Options / Install GSD Files”.

Having done that you will find the DP/CAN coupler in the hardware catalog under ‘Additional FIELD DEVICES / IO / Helmholz’.

You can now drag and drop “DP/CAN coupler L2” onto a Profibus network you have already set up.



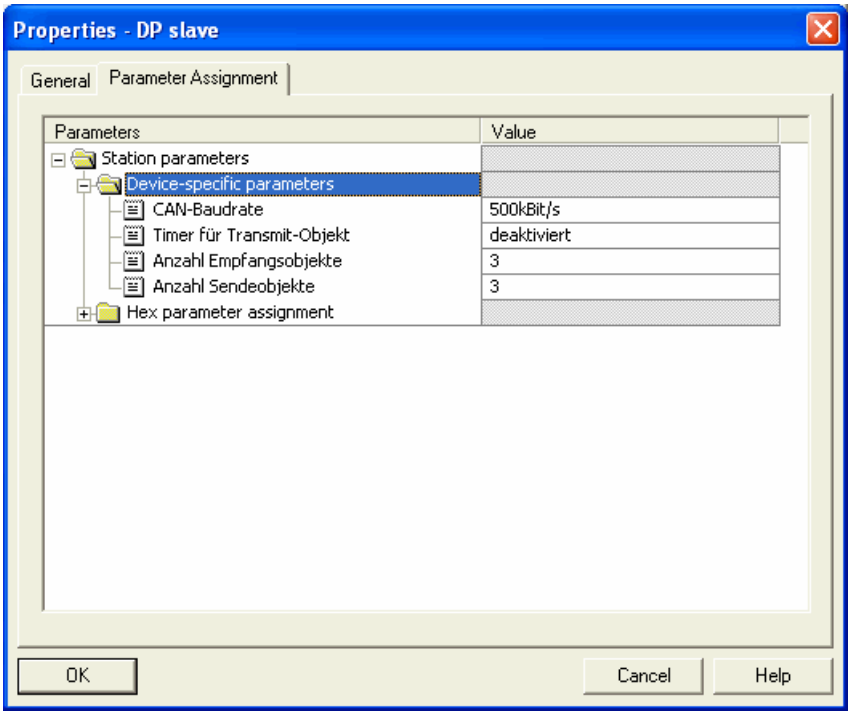
Then assign a suitable station address to the slave.



In the parameterization frame, the master provides the DP/CAN coupler on start-up with the necessary information about the CAN frames to be processed.

The following CAN parameters are defined here:

- CAN baud rate
- Cyclic transmission time of the variable transmit object (if required)
- Number of predefined receive and transmit messages



CAN baud rate:

Possible baud rates: 1 Mbps, 500 Kbps, 250 Kbps, 125 Kbps, 100 Kbps, 50 Kbps, 20 Kbps, 10 Kbps

Timer for variable transmit object:

This is where you set the time for cyclic transmission of the variable transmit object, in 1 ms units (10 ms to 64 ms possible). If you select the “Deactivated” option, the variable transmit object will always be sent immediately.

Incorrect definition of the number of modules results in data corruption or bus start-up errors!

Number of predefined receive objects:

The number of predefined receive objects (DI) used.

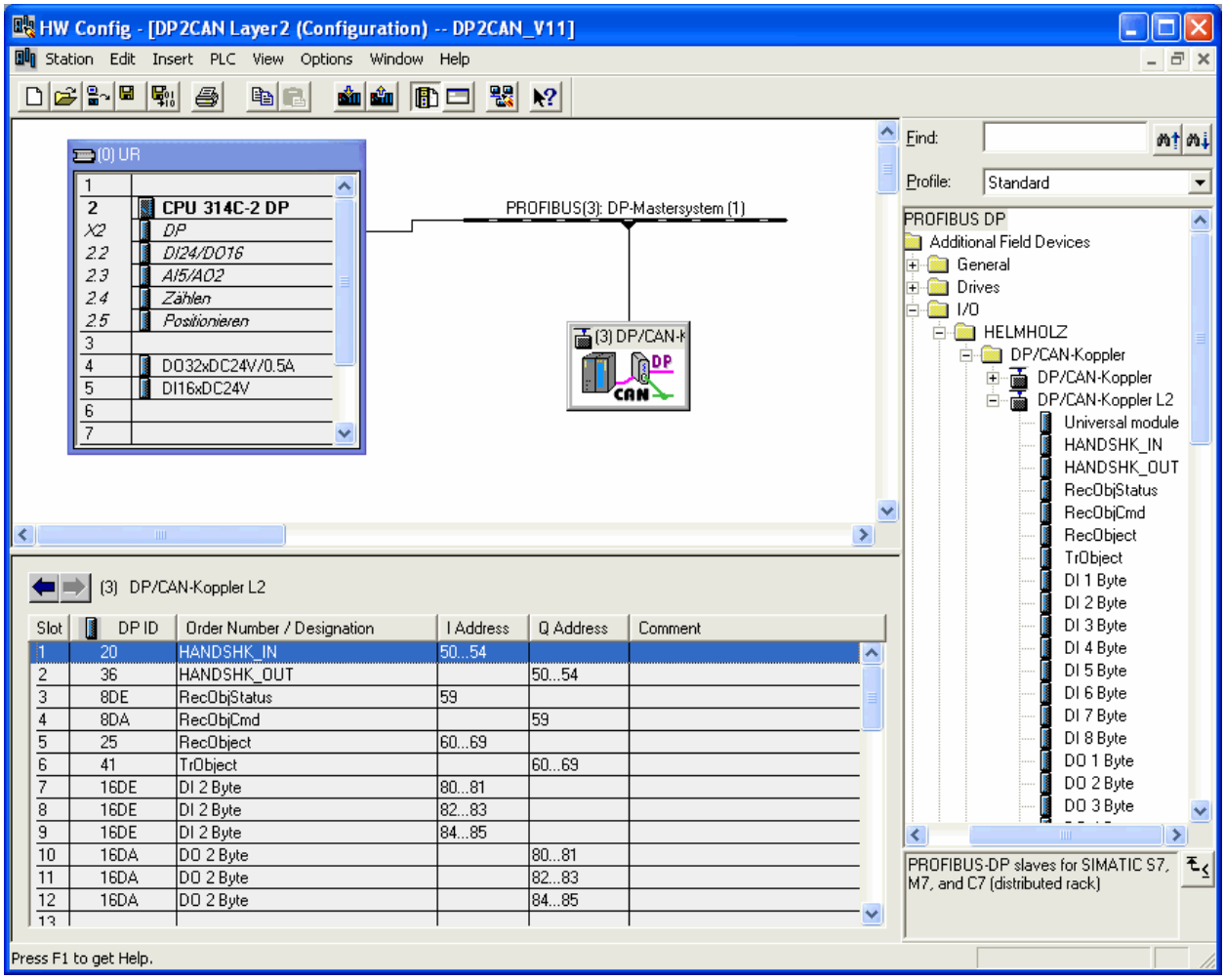
Number of predefined transmit objects:

The number of transmit objects (DO) used.

6.2 Defining the I/O address area in the PLC

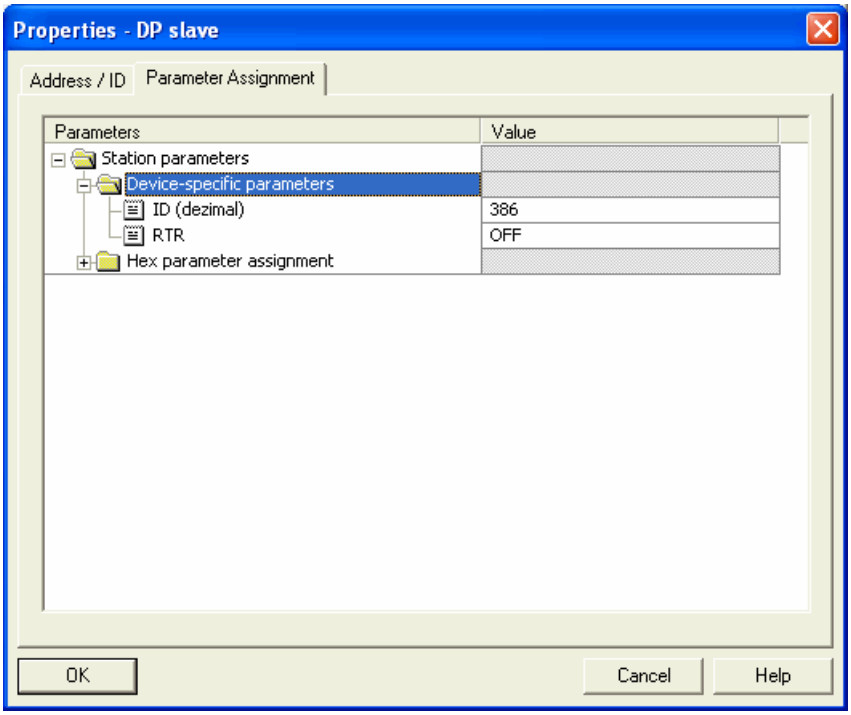
Once the basic parameters of the CAN bus have been defined all data elements must be shown in the I/O area of the CPU.

The first 6 elements must always be at the beginning of the list in the defined sequence. The I/O addresses are freely selectable.



6.3 Parameterizing transmit and receive messages

Now the transmit and receive messages (“DI x byte”, “DO x byte”) can be defined on the remaining slots of the DP/CAN coupler.



Each predefined transmit or receive object can be defined for just one particular CAN message. The length of the CAN message corresponds to the size of the DI/DO object. The CAN ID is defined in the parameter set.

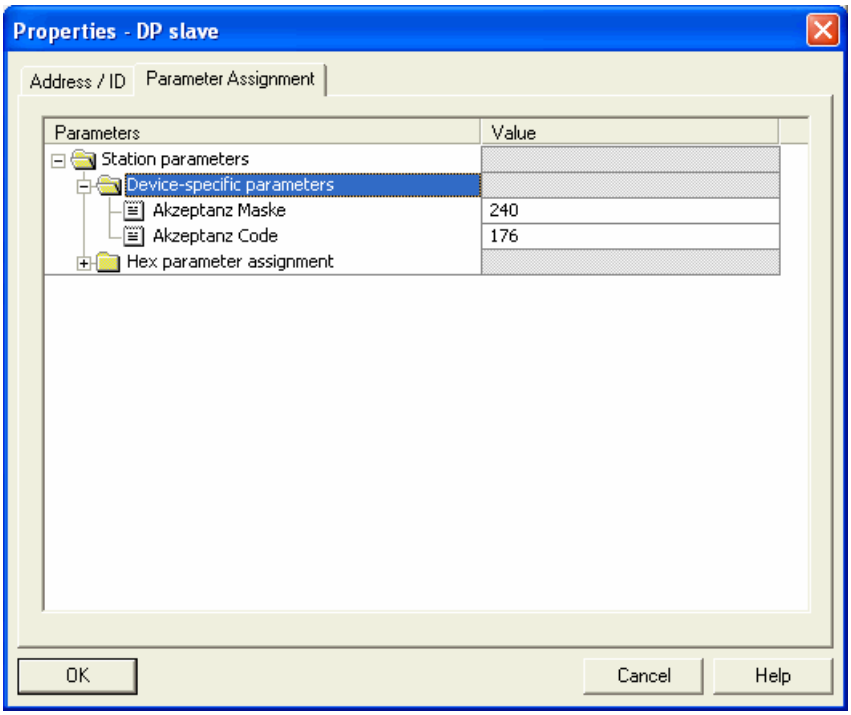
6.4 Parameterizing the variable receive object

The variable receive object must also be parameterized if it is to be used.

To receive any CAN frames with the variable receive object, the upper 8 bits of the CAN identifier are first filtered with a mask (acceptance mask) and then compared with a predefined value (acceptance code). If this comparison is positive, the CAN frame is entered in the receive FIFO and the SPs are released.

10	9	8	7	6	5	4	3	2	1	0
Received CAN identifier										
AcceptanceMask (e.g. 11110000)								*ignored*		
AcceptanceCode (e.g. 1011xxxx)										

Acceptance mask 11110000 (= 240) filters out the top 4 bits. Acceptance code 1011xxxx (= 176) defines which frames are accepted after filtering. In this example, these are the CAN frames with identifiers 0x580 to 0x5FF.



7 Programming (CAN Layer 2)

7.1 Data exchange

When the master has detected that parameterization and configuration is successfully completed without errors at the end of the start-up phase and the PLC has been started, the DP/CAN coupler can transmit and receive frames via CAN.

7.2 Handshake bits

The 5 bytes of the HANDSHK_IN area indicate receipt of CAN messages via the variable receive object and the DI objects. The bits are inverted each time a new message is received (toggle bit).

Byte	Bit	Function
0	**0**	New frame variable receive object
	1	New frame predefined receive object 1
	2	New frame predefined receive object 2
	3	New frame predefined receive object 3
	4	New frame predefined receive object 4
	5	New frame predefined receive object 5
	6	New frame predefined receive object 6
	7	New frame predefined receive object 7
…		…
4	**0**	New frame predefined receive object 32
	1	New frame predefined receive object 33
	2	New frame predefined receive object 34
	3	New frame predefined receive object 35
	4	New frame predefined receive object 36
	5	New frame predefined receive object 37
	6	New frame predefined receive object 38
	7	New frame predefined receive object 39

Do not forget the objects „RecObjStatus” and “RecObjCmd” when interpreting the variable receive object.

The 5 bytes of the HANDSHK_OUT area are used to transmit the predefined transmit objects and the variable transmit object. The bits always initiate transmission of the message when the bit is inverted (toggle bit).

Byte	Bit	Function
0	**0**	Send frame for variable transmit object
	1	Send frame for predefined transmit object 1
	2	Send frame for predefined transmit object 2
	3	Send frame for predefined transmit object 3
	4	Send frame for predefined transmit object 4
	5	Send frame for predefined transmit object 5
	6	Send frame for predefined transmit object 6
	7	Send frame for predefined transmit object 7
…		…
4	**0**	Send frame for predefined transmit object 32
	1	Send frame for predefined transmit object 33
	2	Send frame for predefined transmit object 34
	3	Send frame for predefined transmit object 35
	4	Send frame for predefined transmit object 36
	5	Send frame for predefined transmit object 37
	6	Send frame for predefined transmit object 38
	7	Send frame for predefined transmit object 39

7.3 Predefined transmit and receive objects

Depending on the parameterized size of the object, the predefined transmit and receive objects (“DI x bytes”, “DO x bytes”) always contain the data of the last CAN frame received with the corresponding CAN identifier.

7.4 Variable receive object

If a frame that matches the parameterized acceptance mask of the variable receive object is received, the frame is transferred to the variable receive object area.

Byte	Content of variable receive object
0	HighByte CAN identifier
1	LowByte CAN identifier + RTR + Len
2	Data byte 1
3	Data byte 2
4	Data byte 3
5	Data byte 4
6	Data byte 5
7	Data byte 6
8	Data byte 7
9	Data byte 8

The entire CAN message header is stored in the first two bytes (0+1).

Byte 0								Byte 1							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAN identifier (11 bit)											RTR	Data length			

The receipt of a new frame is recognized by the inversion of bit 0 in byte 0 of the **HANDSHK_IN** area.

Once the message has been processed by the PLC program this must be acknowledged to the DP/CAN coupler. Acknowledgment is processed via the **RecObjCmd** byte.

Here again, all bits must be used as toggle bits, i.e. the function is executed when the bit is inverted.

Bit	Function ReceiveObjCmd
0	Acknowledge last variable receive object frame
1	*reserved*
2	Delete RecObjStatus overflow error bit
3	Reset variable receive object FIFO
4	*reserved*
5	*reserved*
6	*reserved*
7	*reserved*

A FIFO which can accept up to 24 messages has been implemented in the DP/CAN coupler for the variable receive object. If more than 24 messages are received without being fetched by the PLC program, the oldest frames are removed and an overflow error is displayed in the RecObjStatus byte. The bits of the ReceiveObjStatus byte must be processed as status displays.

Bit	Function ReceiveObjStatus
0	FIFI Ok
1	Overflow status currently inactive
2	Overflow status error flag
3	Number of frames still in FIFO
4	
5	
6	
7	

Bit 2 of the **RecObjStatus** can be reset with bit 2 in the **RecObjCmd**.

7.5 Variable transmit object

Any number of messages can be sent via the variable transmit object (**TrObjekt**).

Byte	Content of variable transmit object
0	HighByte CAN identifier
1	LowByte CAN identifier + RTR + Len
2	Data byte 1
3	Data byte 2
4	Data byte 3
5	Data byte 4
6	Data byte 5
7	Data byte 6
8	Data byte 7
9	Data byte 8

The entire CAN message header is stored in the first two bytes (0+1):

Byte 0								Byte 1							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAN identifier (11 bit)											RTR	Data length			

Transmission of the message is initiated by inverting bit 0 in byte 0 of the **HANDSHK_OUT** area.

The variable transmit object can be transmitted via the device parameters of the DP/CAN coupler cyclically by means of a timer.

8 CANopen Communication

8.1 General

CIA = CAN in Automation e.V., Am Weichselgarten 26, 91085 Erlangen, Germany

The CANopen protocol is a layer 7 protocol (application layer) based on the CAN bus (ISO 11898). Layer 1 and 2 (physical layer and data link layer) are not affected by the CAN bus.

The CANopen communication profiles for the various applications are managed by the CIA.

The services elements provided by the application layer permit implementation of an application distributed over the network. These service elements are described in “CAN Application Layer (CAL) for Industrial Applications”.

CANopen always works with CAN 2.0A (11 bits).
This must be taken into account in configuration of the module with CANparam.

The 11 bit identifier and the 8 data bytes of a CAN layer 2 frame have a fixed meaning.

Each devices in a CANopen network has a fixed node ID (module number, 1-127).

8.2 Objects

Data exchange with a CANopen slave is performed either using permanently defined service data objects (SDO) or using freely configurable process data objects (PDO).

Each CANopen slave has a fixed list of SDOs that are addressed by and object number (16 bits) and an index (8 bits).

Example: Object 0x1000/ Index 0 = Device Type, 32Bit Unsigned

SDOs with a width of 8/16/32 bits can be read and written with a CANopen frame. SDOs that are longer are transmitted in more than one frame. For very large volumes of data, SDO block transmission is possible.

SDOs can be processed as soon as a CANopen slave is ready for operation. For the SDOs, only the COB ID functions “SDO request” or “SDO response” are available. The object number, access mode, and type are stored in the first 4 bytes of the CAN frame.

The last 4 bytes of the CAN frame then contain the value for the SDO.

PDOs contain the “working values” of a CANopen slave for cyclic process operation. Each CANopen slave can manage several PDOs (normally up to 4 for transmitting and 4 for receiving).

Each CANopen slave should have a directory containing the objects it supports.

Each of the existing PDOs has its own COB-ID. It is possible to map any information of the CANopen slave to the 8 data bytes of the frame for reading and writing. These can be both existing SDOs and updated values of the slaves (e.g. analog value or an input).

The PDOs are automatically mapped from most CANopen slaves on startup. The assignment can be changed using certain SDOs.

8.3 Functions

The CANopen functions are subdivided into the three basic groups:

Reading and writing SDO

Reading and writing PDO

Network management

The function code is stored in the upper 4 bits of the identifier. Together with the node ID this makes up the COB identifier.

COB identifier (COB-ID):

10	9	8	7	6	5	4	3	2	1	0
Function				Node ID						

It is possible to change some COB-IDs to other values using special service data objects (SDOs).
This is NOT supported by the CANopen data handling!

Broadcast functions:

Function	Function code (binary)	Resulting COB-ID
NMT	0000	0h
SYNC	0001	80h
TIME STAMP	0010	100h

Node functions:

"Tx" = is transmitted by the slave
"Rx" = is transmitted by the slave

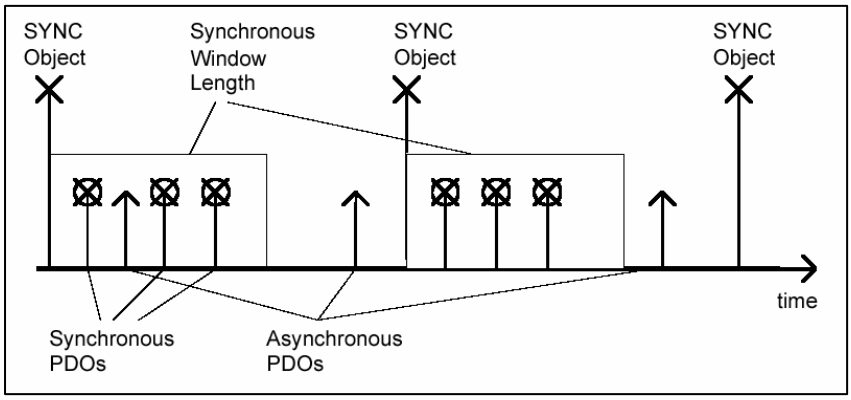
Function	Function code (binary)	Resulting COB-ID
EMERGENCY	0001	81h –FFh
PDO1 (tx)	0011	181h – 1FFh
PDO1 (rx)	0100	201h – 27Fh
PDO2 (tx)	0101	281h – 2FFh
PDO2 (rx)	0110	301h – 37Fh
PDO3 (tx)	0111	381h – 3FFh
PDO3 (rx)	1000	401h – 47Fh
PDO4 (tx)	1001	481h – 4FFh
PDO4 (rx)	1010	501h – 57Fh
SDO (tx)	1011	581h – 5FFh
SDO (rx)	1100	601h – 67Fh
NMT Error Control	1110	701h – 77Fh

8.4 Network management

The SYNC frame can be implemented using a timer with the CAN 300 module.

SYNC:

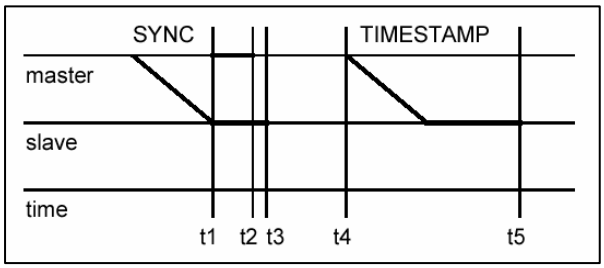
The SYNC frame is a cyclic “broadcast” frame and sets the basic bus clock. To ensure isosynchronism, the SYNC frame has a high priority.
[COB-ID: 80h]



The time stamp frame can be implemented using a timer with the CAN 300 module.

Time Stamp:

The time stamp frame is a cyclic “broadcast” frame and provides the system time. The time stamp frame is usually transmitted directly after a SYNC frame and then provides the system time of the SYNC frame.



To ensure a precise transmission, the time stamp frame has a high priority.
[COB-ID: 100h]

Nodeguarding:

With the Nodeguarding function, the master monitors the CANOpen slave modules by transmitting frames cyclically to each slave. Each CANopen slave must respond to the Nodeguarding frame with a status frame.

The control can detect failure of a CANopen slave using Nodeguarding.
[COB-ID: 700h + Node-ID]

Lifeguarding:

In Lifeguarding, each CANopen slave continuously monitors whether the master is performing Nodeguarding once it has been started within certain time limits.

If the Nodeguarding frame of the master fails, the distributed I/O module can detect that using Lifeguarding and, for example, put all outputs into the safe state.

Heartbeat:

Heartbeat monitoring is equivalent to Nodeguarding although no request frames are generated by CANopen master. The heartbeat frame is transmitted automatically by the node and can be evaluated in the master.

Some CANopen slave modules generate special emergency messages on switch-on or switch-off.

Emergency message:

If a fault occurs on a CANopen slave, for example, the Lifeguarding timer elapses, it transmits an emergency message on the bus.
[COB-ID: 80h + Node-ID]

All stations can perform an emergency stop on receiving an emergency frame, for example.

BootUp message:

CANopen slaves generate a BootUp message after switch-on that the master can recognize to initialize this new station.
[COB-ID: 700h + node ID + 1 byte data: 00h]

9 Appendix

9.1 Technical data

Order number	DP/CAN Coupler	700-650-CAN01
Dimensions	114 x 18 x 108 mm (LxWxH)	
Weight	Approx. 120 g	

CAN interface

Type:	ISO/DIN 11898-2, CAN high speed physical layer
Transmission rate:	10 kbps to 1Mbps
Protocol:	CANopen master CAN 2.0A (11bit)
Pin:	3-way screw-type terminal

PROFIBUS DP interface

Type:	Profibus DP to EN 50 170
Transmission rate:	19.2 kbps to 12Mbps
Pin:	Sub-D connector, 9-way

Power supply

Voltage:	+24V DC
Current consumption:	180 mA (type)

Permissible ambient conditions

Operating temperature:	0°C … 60°C
Storage temperature:	-25°C … 75°C
Degree of protection:	IP 20

Special features

Quality assurance:	according to ISO 9001:2000
Maintenance:	Maintenance-free (no battery, rechargeable or non-rechargeable)

9.2 Pin assignment

Pin	Profibus DP
1	-
2	
3	Data line B
4	-
5	GND
6	VP (power supply for terminating resistors)
7	-
8	Data line A
9	-

9.3 Further documentation

Internet: www.can-cia.org

CAN Specification 2.0, Part A & Part B

High Layer Protocol CANopen

Holger Zeltwanger: “CANopen”, VDE Verlag, ISBN 3-8007-2448-0

Notes