CAN - CSC595/2

CAN - PLC Interface Module
for
S5-90U, S5-95U and S5-100U

Hardware Manual

Document file:	I:\texte\Doku\MANUALS\CAN\CSC595.2\Englisch\CSC0514H.en9
Date of print:	16.11.98

PCB version:	CSC505-2

Changes in the chapters

The changes in the user's manual listed below affect changes in the hardware as well as changes in the description of the facts only.

Chapter	Changes versus previous version
-	First English version.
-	

Technical details are subject to change without notice.

N O T E

The information in this document has been carefully checked and is believed to be entirely reliable. **esd** makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors that may appear in this document. **esd** reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance or design.

esd assumes no responsibility for the use of any circuitry other than circuitry which is part of a product of **esd** gmbh.

esd does not convey to the purchaser of the product described herein any license under the patent rights of **esd** gmbh nor the rights of others.

esd electronic system design gmbh
Vahrenwalder Str. 207
30165 Hannover
Germany

Phone: +49-511-372 98-0
Fax: +49-511-372 98-68
E-mail: info@esd-electronics.com
Internet: www.esd-electronics.com

USA / Canada:
esd electronics Inc.
12 Elm Street
Hatfield, MA 01038-0048
USA

Phone: +1-800-732-8006
Fax: +1-800-732-8093
E-mail: us-sales@esd-electronics.com
Internet: www.esd-electronics.us

Contents

Page

This page is intentionally left blank.

1. Overview

1.1 Module Description

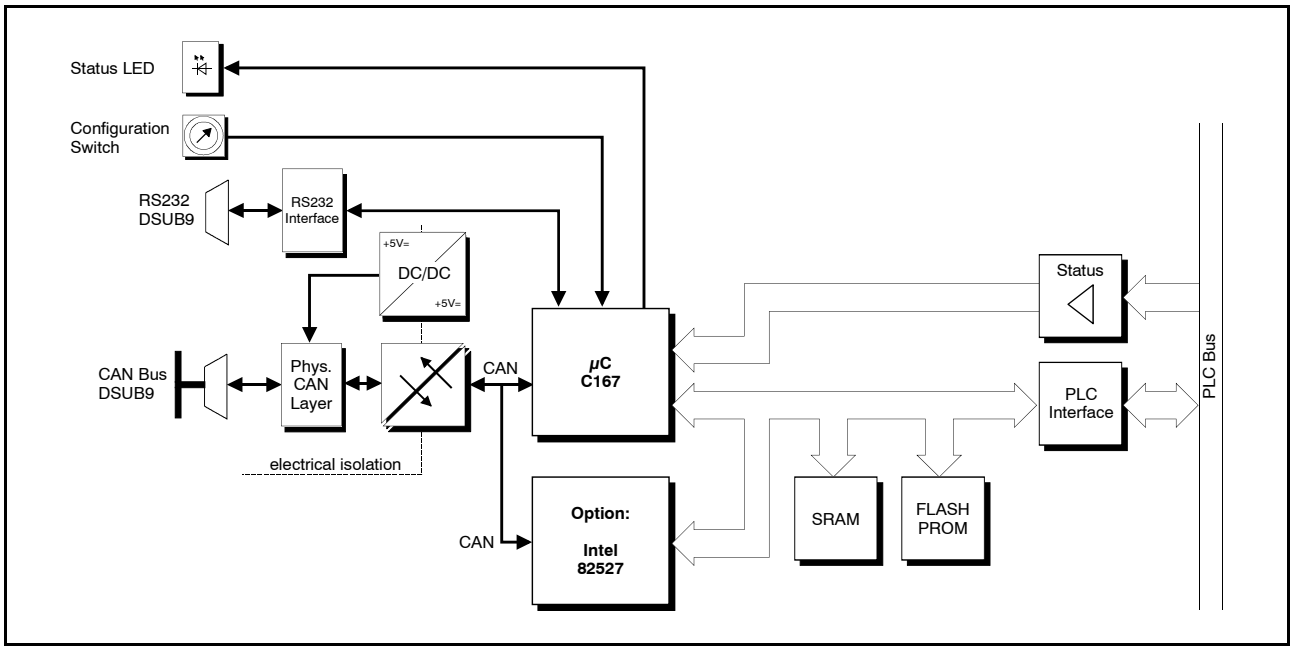


Fig. 1.1.1: Block-circuit diagram of the CSC595/2

By means of the communication processor CAN-CSC595/2 SIEMENS PLCs of S5-90U, S5-95U or S5-100U and esd-CAN-I/O modules or other CAN participants can be directly linked.

The module guarantees complete transparency of process data to the PLC programmer. No further function or data components are required so that PLC programs can be run as usual.

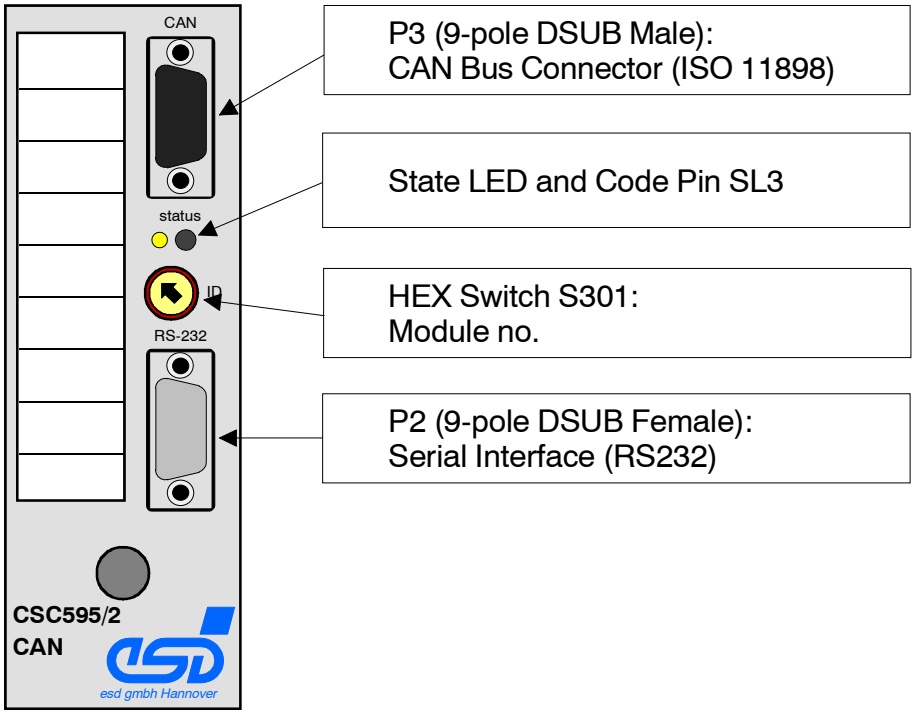
The CAN-CSC595/2 uses the high performance microcontroller C167C with integrated CAN-controller and guarantees a bit rate of 1 Mbit/s without data loss even when the C167C is running as a high-level-protocol master.

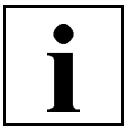
The physical CAN-layer corresponds to ISO 11898. Like all CAN-identifiers, the bit rate can be set via the local RS-232 interface by means of the software.

This and other modules can be configured via the RS-232 interface. An automatic configuration of other modules (after cold start) is also possible. The settings are stored into the local EEPROM.

The module is shipped in a plastic case which is compatible to SIEMENS S5 devices.

1.2 Front-Panel View With LEDs- and Connectors





1.3 Summary of Technical Data

1.3.1 General Technical Data

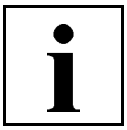
RS-232 interface	RS-232C interface at 9-pin female DSUB as input and configuration interface and for loading new S-records for software updates of the FLASH EPROM
Temperature range	0…50°C ambient temperature
Humidity	max. 90%, non-condensing
Power supply CAN-module	fed via PLC bus, nominal voltage 9V ±10%, current (typical, at 20°C): ca. 200 mA (without CAN-controller 82527)
Connectors	P301 - PLC connection P2 (DSUB9/female) - RS-232 interface P3 (DSUB9/male) - CAN bus interface
Case	Siemens PLC-module case, compatible to SIMATIC S5 bus module
Weight	ca. 250 g

Table 1.3.1: General data of the CSC595/2

1.3.2 CAN-Interfaces of the CSC595/2

Number of CAN-interfaces	one interface at connector P3
Controller components	C167 and 82527(option)
Use of the optional second controller	reception and evaluation of RTR frames
CAN-identifiers	programmable via CAN- or RS232 interface
esd-module No.	can be set via coding switch in front panel or programmed via CAN- or RS-232 interface
I²C-EEPROM	for storing the parameters
Physical layer	physical layer in accordance with ISO 11898, transmission rate programmable from 10 kbit/s to 1 Mbit/s
Electrical insulation of the CAN-interfaces from other units	insulation via optical couplers and DC/DC-converters in accordance with German VDE regulation 0110b §8, isolation group C and installation into cubicle): 300 V(DC), 250 V(AC)

Table 1.3.2: CAN-interfaces of the CSC595/2



1.3.3 PLC Unit

PLC link	compatible PLC units: SIEMENS S5-90U SIEMENS S5-95U SIEMENS S5-100U SIEMENS S5-102U SIEMENS S5-103U SIEMENS ET-100
Monitoring mode	monitoring the PLC bus: transmission of all PLC data to the CAN bus

Table 1.3.3: PLC unit

1.4 Software Support

The complete EPROM-resident CAN-communication firmware for operating the CSC595/2 module is contained in the product package. The software will be explained in the second part of the manual.

This page is intentionally left blank.

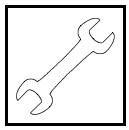
1.5 Order Information

Type	Features	Order No.
CAN-CSC595-2	interface CAN/Siemens S5-SPS product package: device with CAN-controller C167 (82527 not mounted) with plastic case, coding pin for bootstrap loader, software and hardware manual	C.2902.02
CAN-CSC595/2-SDS	option: SDS master firmware	C.2902.50
CAN-CSC595/2-CoS	option: CANopen slave firmware	C.2902.52
CAN-CSC595/2-CoM	option: CANopen master firmware	C.2902.54
CAN-CSC595/2-MD	German manual 1*)	C.2902.20
CAN-CSC595/2-ME	English manual 1*)	C.2902.21

1*) If ordered together with the module, the manual is free of charge.

Table 1.5.1: Order information

This page is intentionally left blank.



2. Installation Notes

2.1 Installing the CSC595/2 Module into a SIMATIC-Automation Device

Please read the instructions in the SIMATIC-S5 manual carefully before taking the SIMATIC-S5 automation device into operation! The following steps relate only to the installation of the CSC595/2 module.

Way of procedure:

1. Switch off (disconnect) the power supply of the SIMATIC central extension devices and of the signal feeder and signal receiver.

2. Select a free stack in the central device, plug CSC595/2 to board carrier of the SIMATIC and fix by means of the recess screw accessible in the front panel.

3. Connect CAN-interface.
   The CAN-interface is connected via the 9-pin DSUB-connector in the front panel. Notes on wiring the CAN-network can be taken from the chapter 'Correctly Wiring Electrically Insulated Networks' at the end of this manual.

4. Connect terminal to RS232-interface.
   You can either use a normal terminal (such as WYSE, FALCO) or a PC or Laptop with a terminal program. The connection will be described separately in the following chapter 'Connecting a terminal'.

5. Switch on central device, switch on the other CAN bus participants, switch on terminal (the sequence is arbitrarily)

6. If the driver software is already in the local Flash EPROM (default status when module is shipped), the status LED of controller C167 (next to the coding connector) has to flash: green for 500 ms and red for 100 ms. Doing this, the LED signalizes that the module status is OK and that the module is operating by using the default parameters.

7. Now the CSC595/2 module can be configured via a terminal. During the configuration various parameters (such as bit rate, identifiers) can be changed. All configuration parameters can be stored into the local EEPROM. The changed and stored parameters will only become active after a RESET.

The configuration of the module will be described in the software manual. The you will also find a complete list of default parameters with which the module is operating after being shipped.

2.2 Connecting a Terminal

The terminal is required to configure the CSC595/2 module. You can either use a normal terminal or a PC with a terminal program. If users want to install new software updates themselves, a PC is absolutely necessary.

The setting parameters of the interface (bit rate, etc.) Will be described in the chapter 'Specification of the serial Interface', starting on page 21.

2.2.1 Terminal

During wiring the terminal should be switched off. The terminal is connected via the 9-pin female DSUB connector (P2) in the front panel. The signal assignment has been chosen in a way that a terminal can be directly connected without a null modem.

2.2.2 PC or Laptop with Terminal Program

During wiring the PC or Laptop should be switch off.

The port to which the module is connected during operation (configuration) depends on the terminal program that is used. Normally, various ports are supported.

When connecting to a 9-pin mouse port a *null modem* has to be connected to the supply. If the PC or Laptop is connected to a 25-pin DSUB-connector a null modem is not required. The signal assignment of suitable connection lines will be listed in the appendix.

esd also offers manufactured connection cables with 9-pin connectors for which no null modem is required.

3. Component Print, Jumpers and Coding Switches

3.1 Component Print

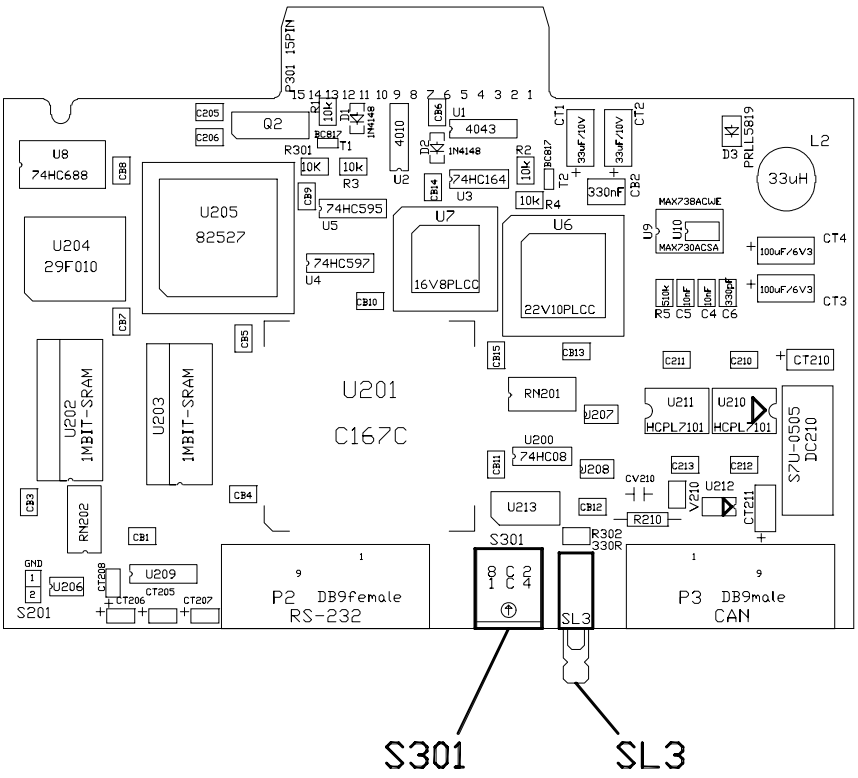


Fig. 3.1.1: Position of configuration elements on the component layer of the PCB

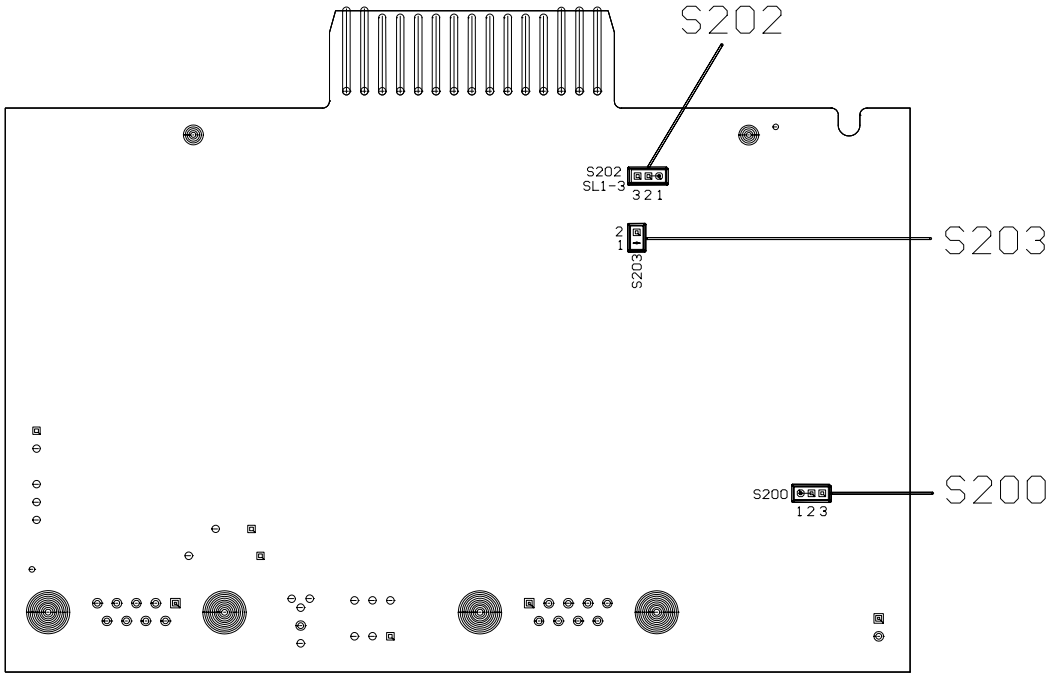
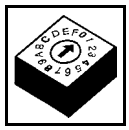


Fig. 3.1.2: Position of solder bridges on the bottom layer of the PCB



3.2 Default Setting of Bridges and Coding Switches

The respective default setting of bridges, coding switches and of the plug contact at the time the board is shipped, will be listed in the following figures.

Please refer to figure 3.1.1 for the position of the components on the top layer of the PCB. In the following descriptions the components will be described as seen by the user with the board in a position where the CAN bus connectors are pointing to the left.

The position of the solder bridges can be taken from figure 3.1.2. In the following descriptions the solder bridges will be described as seen by the user with the board in a position where the CAN bus connectors are pointing to the right (bottom layer view).

Summary of default settings when the module is shipped:

Solder bridge	Function	Setting
S200	memory capacity of SRAMs	256 kByte (2 x 128 kByte)
S202	operation of 82527 with 10 MHz or 20 MHz	82527 is pulsed with 20 MHz
S203	Tx-signal of the 82527 to CAN interface	board without 82527: solder bridge open board with 82527: bridge closed, i.e. Tx-signal is connected to the CAN bus interface

Note: Solder bridge S200 will not be described again below, because the position of the bridge depends from the SMD-memory components (SRAMs) used. The SMD memories used are mounted at the factory and cannot be changed afterwards. Therefore the user must not change the position of solder bridge S 200!

Plug contact/ coding switch	Function	Setting
Plug contact SL3	activate bootstrap loader	not set, i.e. bootstrap loader is inactive
Coding switch S301	module No.	the module No. has always to be adjusted to an available CAN network by the user, therefore, there is no defined default setting

Table 3.2.1: Default setting of bridges and coding switches

3.3 Description of Bridges and Coding Switches

3.3.1 Operation of the 82527 with 10 MHz or 20 MHz (S202)

By means of this solder bridge the pulse frequency of the 82527 controller can be set to 10 MHz or 20 MHz. When the module is shipped, the bridge is set to 20 MHz.

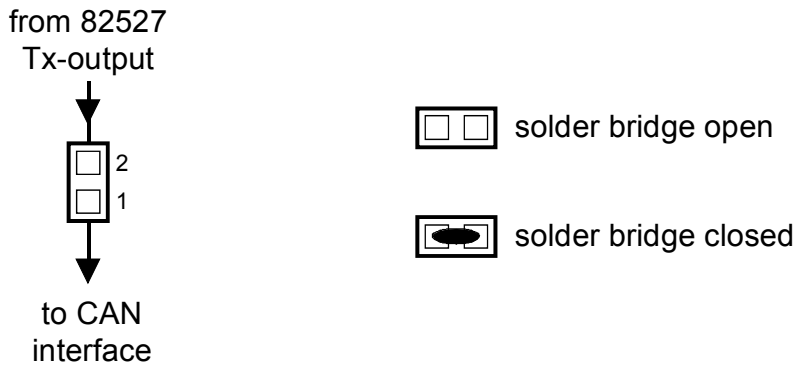
The position of this solder bridge is not to be changed by the user.

Example above: Setting of solder bridge for 20 MHz operation of controller 82527

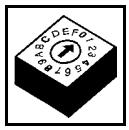
3.3.2 Connection of Tx-signal of CAN controller 82527 to CAN interface (S203)

By means of this solder bridge the Tx-signal of the CAN controller 82527 is connected to the CAN interface. The solder bridge is open, if the controller is not equipped.

The position of this solder bridge is not to be changed by the user.



Example above: CAN controller 82527 not equipped



3.3.3 Activating the Bootstrap Loader (SL3)

In order to be able to download a software update via the serial interface into the local memory, the bootstrap loader has to be enabled. It has been locked to prevent the local program code from being overwritten accidentally.

In order to enable the bootstrap loader the coding pin, which is included in the product package, has to be plugged into the socket SL3. The pin closes an internal contact and by doing so enables the bootstrap loader. Now the loading procedure can be started via the operation software.

The coding pin makes a correct contact, when it is inserted as far as possible.

3.3.4 Setting the Module No. via Coding Switch S301

The module No. with which the CSC595/2 module is selected via the CAN bus when operating by means of the default parameters, consists of 8 bits. The module No. is required for the firmware to identify the module.

By means of the four pin coding switch S301 in the front panel bits 0 to 3 of the CAN-module No. are set. Bits 4 to 7 of the module No. have been fixed to ‘0'.

The assignment of coding switch position to module No. is therefore as follows:

Coding switch position	Module-No. bit [HEX]
0 (*)	00 (*)
1	01
2	02
:	:
E	0E
F	0F

Table 3.3.4: Assignment of coding switch position to module No.

(*) If the coding switch is set to ‘00' and a RESET is triggered (via Power Down), the module keeps on operating by means of the default parameters after being switched on again. All previously changed parameters are lost, even if they had been stored into the local I²C-EEPROM.

When the module is operating via the default parameters, the module No. which has been set at the coding switch is active. The complete 8-bit module No. can be freely programmed via the firmware. The programmed module No. replaces the module No. set via the coding switch immediately.

Programming the module No. will be described in the software manual of this module.

Attention: It is not possible to set and save new parameters of the module, while the coding switch is set to ‘0’! This happens because, after a reset, that is necessary after the programming of the module the new parameters will be overwriten by the default parameters.

The setting of the coding switch at the moment the module is shipped has bot been determined, because the user has to synchronize it with other module numbers in the CAN-network.

4. Description of the Units

4.1 PLC-Bus Interface

The CSC595/2 module has a PLC-interface which has been designed for the connection to Siemens SIMATIC-S5 units. The interface is controlled by programmable logical components. The transmit and receive data is buffered into SRAM memories. The following figure represents the structure of the interface control.

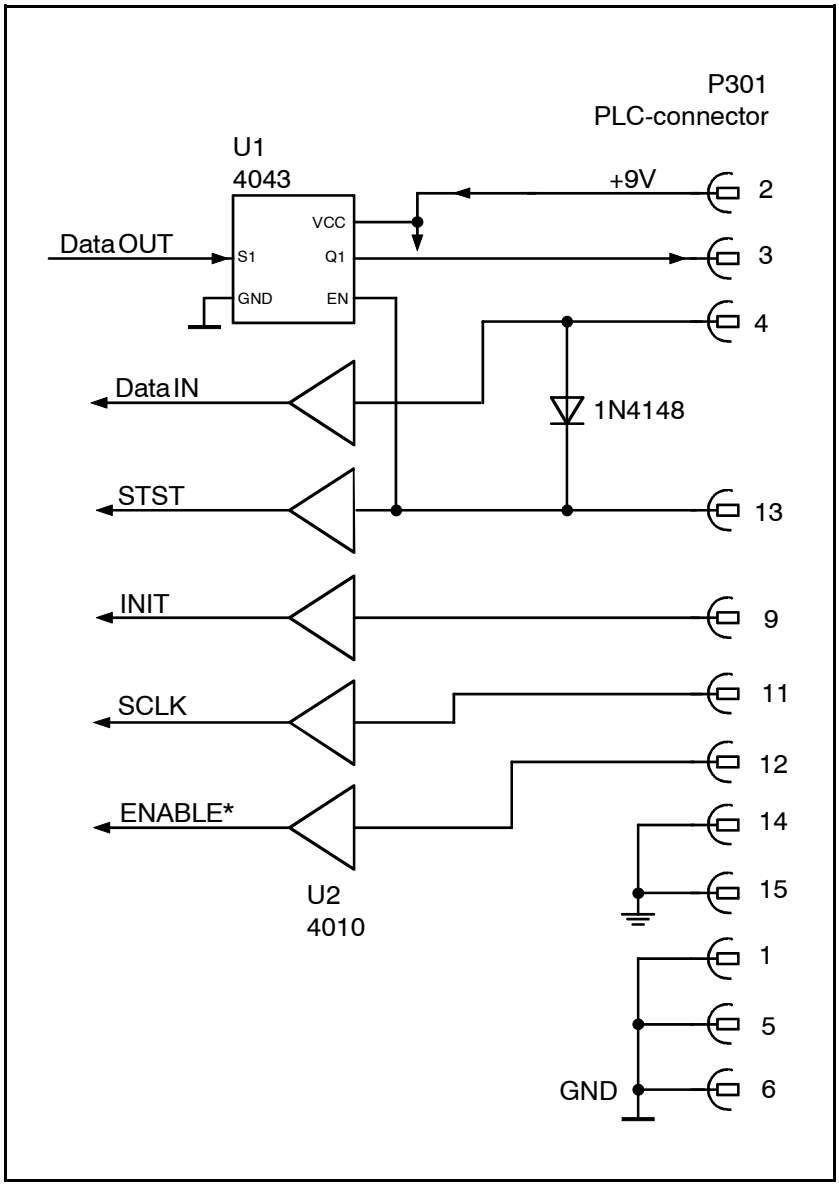


Fig. 4.2.1: Block-circuit diagram of the PLC-interface control

4.2 CAN Bus Interface

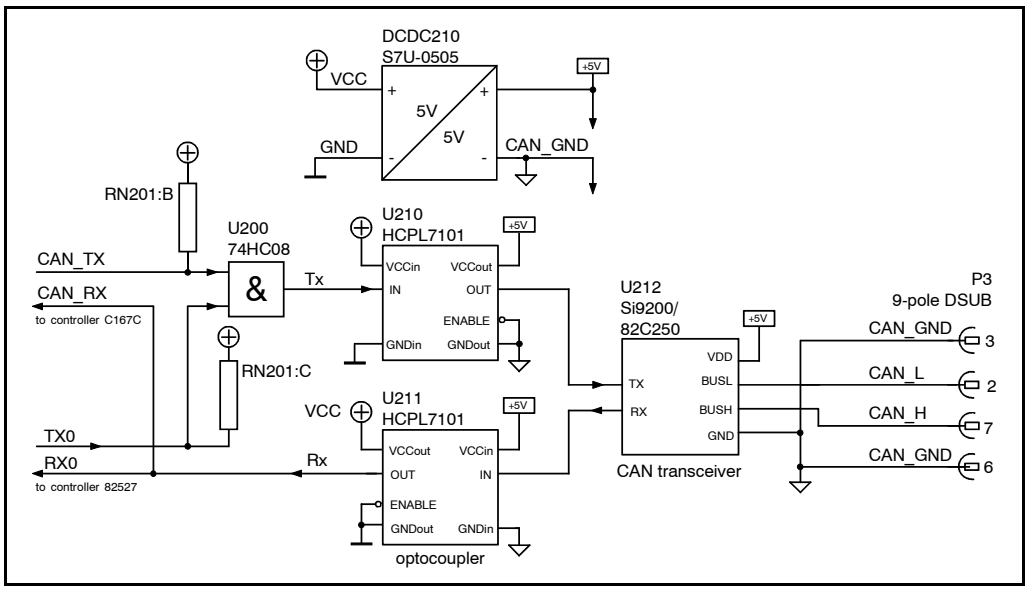
4.2.1 Bit Rate

The transmission speed of the CAN-interface can be varied between 10 kbit/s and 1.0 Mbit/s. The bit rate is set by means of the local software. Further information on this can be taken from the software manual of this module.

4.2.2 Transmit and Receive Circuit of the CAN-Interface (Physical Layer)

The C167C is used as a CAN-controller. The physical interface of the CAN bus is in accordance with the ISO 11898 norm. The Si9200 or the 82C250 are used as CAN bus transceivers in the module. The CAN-interface is supplied with power from the local +5V supply voltage by a DC/DC-converter. The signals to the CAN bus are electrically insulated by optical couplers.

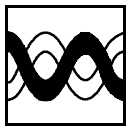
Notes on the wiring of the CAN-network:
The reference potential of the CAN bus (CAN_GND) has to be connected to the earth potential at exactly *one* point in the CAN-network.



+5V, VCC, GND, CANGND…	local supply voltages
CAN_TX, CAN_RX…	signals of CAN-controller C167C (U201)
TX0, RX0…	signals of the optional CAN-controller 82527 (U205)
CAN_L, CAN_H…	CAN bus signal lines

Fig. 4.3.1: Functional circuit diagram of the CAN bus interface when using the interface components Si9200 or 82C250

The connector assignment can be taken from the appendix.



4.3 Specification of the Serial Interface

The RS-232 interface is specified as follows:

Parameters	Settings
baud rate	19200 baud
data bits	8 bits/character
stop bits	1 stop bit
parity bit	no parity
handshake	no handshake (or, if this cannot be set: XON/XOFF)

Table 4.4.1: Parameters of the serial interface

The PC or terminal connected has to be set to the values specified above.

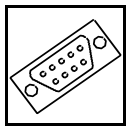
The connector assignment of the interface at P2 can be taken from the appendix.

4.4 LED Display

The status LED is next to the bootstrap plug.

LED status	Meaning of the LED status		
	module is in RESET status	module is in bootstrap mode	module is in 'normal' operation
constantly green	-	-	General status of the nodule is OK
LED flashes: short green, short red (approx. 100 ms green, 100 ms red)	-	-	EEPROM error
LED flashes: short green, short red (approx. 200 ms green, 200 ms red)	-	Displaying the bootstrap mode	-
LED flashes: long green, short red (approx. 500 ms green, 100 ms red)	-	-	Default status (module is operating via default parameters)
LED flashes: short green, short red, short off (330 ms green, 330 ms red, 330 ms off)	-	-	Module has been configured as a CANopen master, but the CANopen network has not yet started
LED off	Module is in RESET	-	-

-… This combination of operating status of the module and LED status does not exist.



5. Appendix

5.1 Connector Assignments

5.1.1 PLC-Bus Connector P301

Pin	Signal
1	GND
2	+9V
3	Data OUT
4	Data IN
5	GND
6	GND
7	-
8	-
9	INIT
10	-
11	SCLK
12	ENABLE*
13	STST-Bus
14	PE
15	PE

+9V, GND…… voltage supply

Data IN,
Data OUT…… data lines

STST-Bus…… start/stop signal of control unit

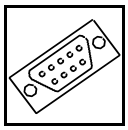
INIT……. signal for initialisation sequence

SCLK……. synchronous shift pulse

ENABLE*……. enable input for bus

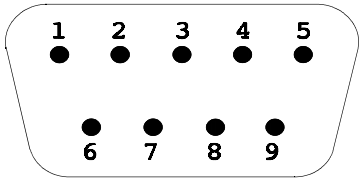
PE…… protection earth connection

'-'……. not connected



5.1.2 Connector of CAN Bus Interface P3 (9-pin DSUB Male)

Pin Position:



Pin Assignment:

Signal	Pin		Signal
		1	reserved
CAN_GND	6		
		2	CAN_L
CAN_H	7		
		3	CAN_GND
reserved	8		
		4	reserved
reserved	9		
		5	reserved

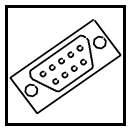
9-pin male DSUB connector

Signal Description:

CAN_L, CAN_H ... CAN-signal lines

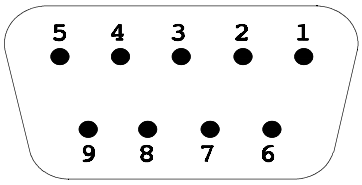
CAN_GND ... reference potential of the local CAN-physical layer

reserved ... pins which are reserved for future applications



5.1.3 Serial Interface RS-232 at P2 (9-pin DSUB Female)

Pin Position:



Pin Assignment:

Signal	Pin		Signal
-	1	6	-
RxD	2	7	-
TxD	3	8	-
DTR	4	9	-
GND	5		

9-pin female DSUB connector

Signal Description:

TxD … signal line: data output of the CSC595/2

RxD … signal lines: data input of the CSC595/2

DTR … handshake signal (output)

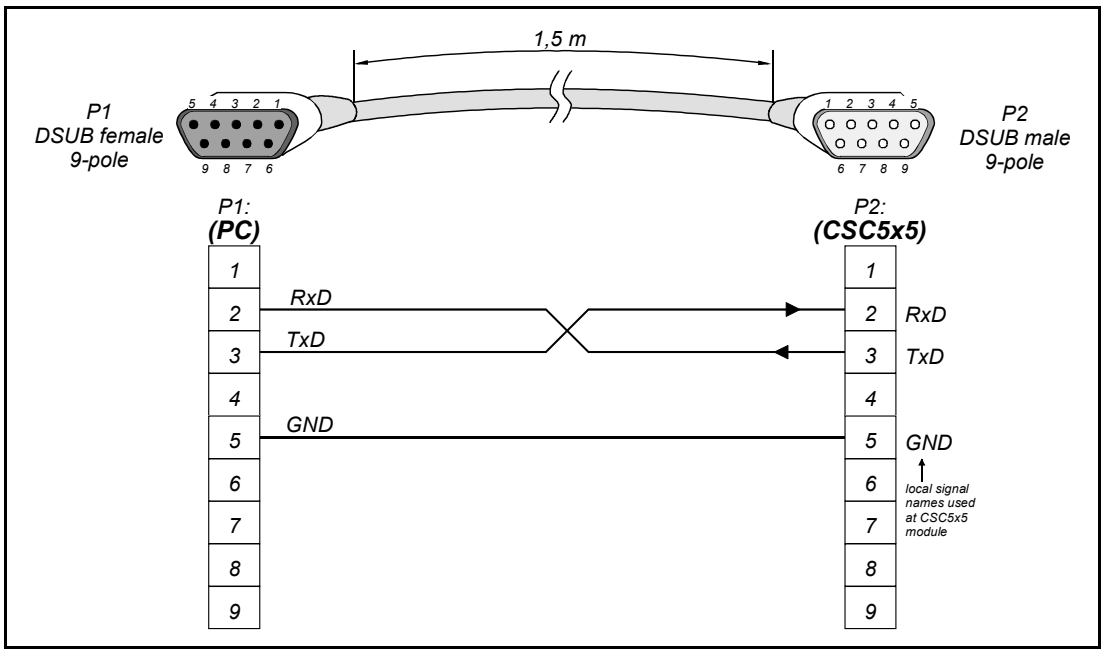
GND … reference potential

'-' … not connected

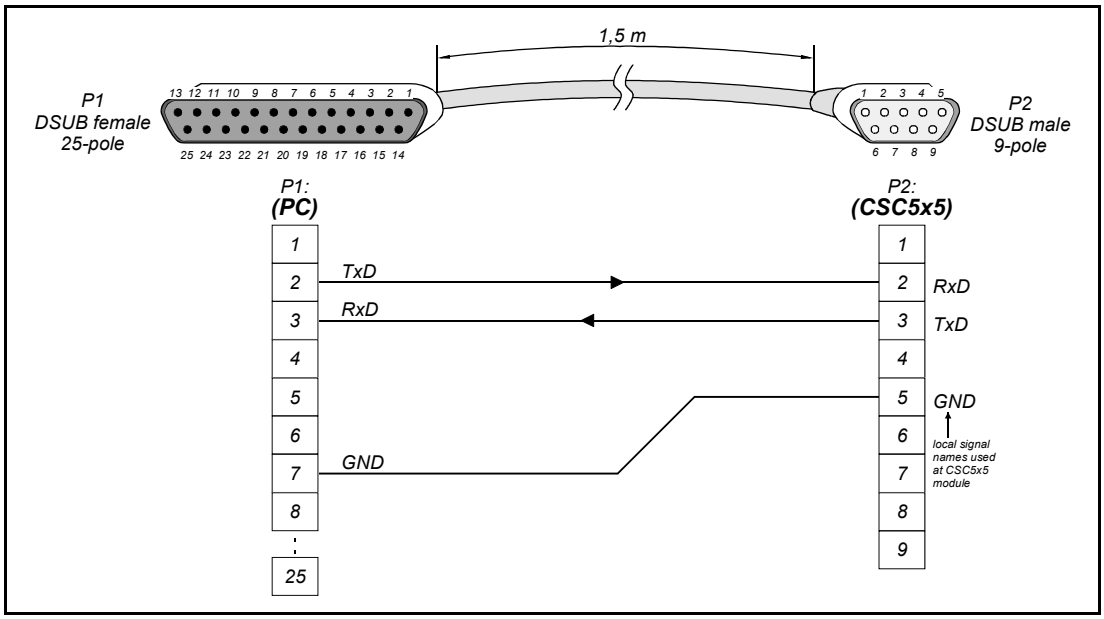
5.1.4 Connection Lines for CSC595/2 to PC (RS-232 Interface)

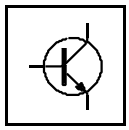
The following two figures show the required assignment for two RS-232 connection lines between PC and CSC595/2.

Adapter cable 9-pin DSUB female to 9-pin DSUB male



Adapter cable 25-pin DSUB female to 9-pin DSUB male





5.2 Circuit Diagrams

This page is intentionally left blank.

6. Correctly Wiring Electrically Insulated CAN Networks

Generally all instructions applying for wiring regarding an electromagnetic compatible installation, wiring, cross sections of wires, material to be used, minimum distances, lightning protection, etc. have to be followed.

The following **general rules** for the CAN wiring must be followed:

1.	A CAN net must not branch (exception: short dead-end feeders) and has to be terminated by the wave impedance of the wire (generally 120 W ±10%) at both ends (between the signals CAN_L and CAN_H and **not** at GND)!
2.	A CAN data wire requires **two twisted** wires and a wire to conduct the reference potential (CAN_GND)! For this the shield of the wire should be used!
3.	The reference potential CAN_GND has to be connected to the earth potential (PE) at **one** point. Exactly **one** connection to earth has to be established!
4.	The bit rate has to be adapted to the wire length.
5.	Dead-end feeders have to kept as short as possible (l < 0.3 m)!
6.	When using double shielded wires the external shield has to be connected to the earth potential (PE) at **one** point. There must be not more than **one** connection to earth.
7.	A suitable type of wire (wave impedance ca. 120 Ω ±10%) has to be used and the voltage loss in the wire has to be considered!
8.	CAN wires should not be laid directly next to disturbing sources. If this cannot be avoided, double shielded wires are preferable.

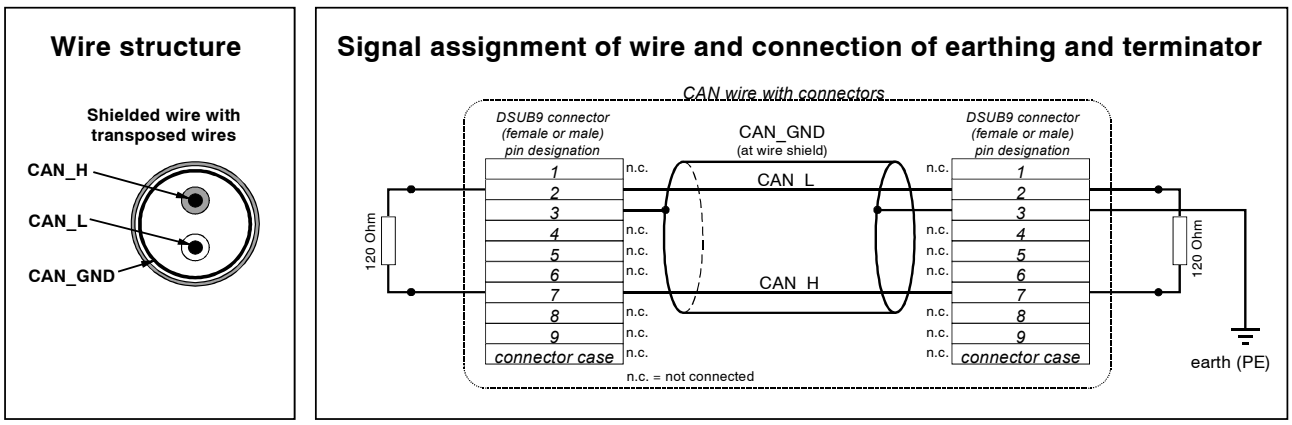


Figure: Structure and connection of wire

Cabling

❍ for devices which have only one CAN connector per net use T-connector and dead-end feeder (shorter than 0.3 m) (available as accessory)

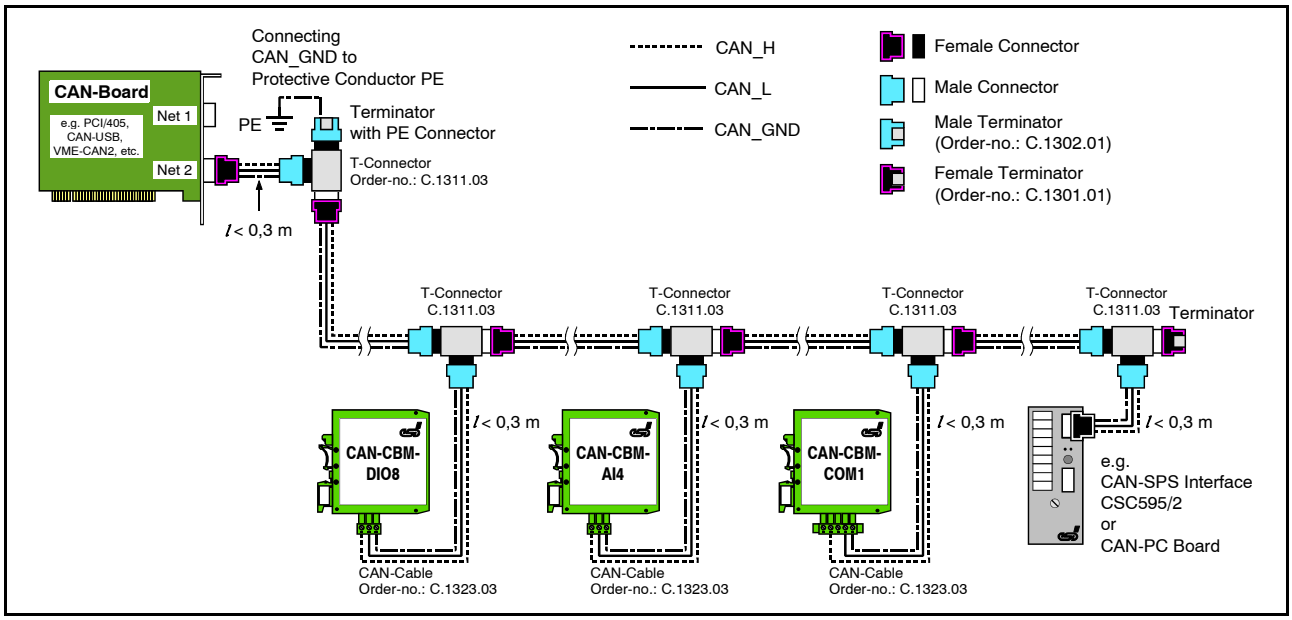


Figure: Example for correct wiring (when using single shielded wires)

Terminal Resistance

❍ use **external** terminator, because this CAN later be found again more easily!

❍ 9-pin DSUB-terminator with male and female contacts and earth terminal are available as accessories

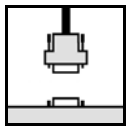
Earthing

❍ CAN_GND has to be conducted in the CAN wire, because the individual esd modules are electrically isolated from each other!

❍ CAN_GND has to be connected to the earth potential (PE) at **exactly one** point in the net!

❍ each CAN user without electrically isolated interface works as an earthing, therefore: do not connect more than one user without potential separation!

❍ Earthing CAN e.g. be made at a connector



Wire Length

❍ Optical couplers are delaying the CAN signals. By using fast optical couplers and testing each board at 1 Mbit/s, however, esd CAN guarantee a reachable length of 37 m at 1 Mbit/s for most esd CAN modules within a closed net without impedance disturbances like e.g. longer dead-end feeders. (Exception: CAN-CBM-DIO8, -AI4 and AO4 (these modules work only up to 10 m with 1 Mbit/s))

Bit rate [Kbit/s]	Typical values of reachable wire length **with esd interface** l_{max} [m]	**CiA recommendations** (07/95) for reachable wire lengths l_{min} [m]
1000	37	25
800	59	50
666.6	80	-
500	130	100
333.3	180	-
250	270	250
166	420	-
125	570	500
100	710	650
66.6	1000	-
50	1400	1000
33.3	2000	-
20	3600	2500
12.5	5400	-
10	7300	5000

Table: Reachable wire lengths depending on the bit rate when using esd-CAN interfaces

Examples for CAN Wires

Manufacturer	Type of wire
U.I. LAPP GmbH Schulze-Delitzsch-Straße 25 70565 Stuttgart Germany www.lappkabel.de	e.g. UNITRONIC ®-BUS CAN UL/CSA (UL/CSA approved) UNITRONIC ®-BUS-FD P CAN UL/CSA (UL/CSA approved)
ConCab GmbH Äußerer Eichwald 74535 Mainhardt Germany www.concab.de	e.g. BUS-PVC-C (1 x 2 x 0,22 mm²) Order No.: 93 022 016 (UL appr.) BUS-Schleppflex-PUR-C (1 x 2 x 0,25 mm²) Order No.: 94 025 016 (UL appr.)
SAB Bröckskes GmbH&Co. KG Grefrather Straße 204-212b 41749 Viersen Germany www.sab-brockskes.de	e.g. SABIX® CB 620 (1 x 2 x 0,25 mm²) Order No.: 56202251 CB 627 (1 x 2 x 0,25 mm²) Order No.: 06272251 (UL appr.)

Note: Completely configured CAN wires can be ordered from **esd**.

CAN - CSC595/2

CAN - PLC Interface Module
for
S5-90U, S5-95U and S5-100U

DeviceNet

Software Manual

N O T E

The information in this document has been carefully checked and is believed to be entirely reliable. **esd** makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors that may appear in this document. **esd** reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance or design.

esd assumes no responsibility for the use of any circuitry other than circuitry which is part of a product of **esd** gmbh.

esd does not convey to the purchaser of the product described herein any license under the patent rights of **esd** gmbh nor the rights of others.

esd electronic system design gmbh
Vahrenwalder Str. 205
D-30165 Hannover
Germany

Tel: +49-511-372-980
Fax: +49-511-633-650
Email: pm@esd.h.eunet.de

Manual File:	I:\TEXTE\DOKU\MANUALS\CAN\CSC595.2\SC0512SD.EN6
Date of Print:	28/10/97

Described Software:	DeviceNet driver for CSC595/2	CSC595/2 Kernel
Revision/Date:	V0.1f	V0.2b

Changes in the chapters

The changes in the user's manual listed below affect changes in the software, as well as changes in the description of the facts only.

Firmware Manual Version	Chapter	Alterations as compared to version 1.0
CSC595/2 V1.2	1.3	Commands DM, SM, and SY deleted in document, because they are designed for service and programming only.
	4.2	PLC address in example changed.
	-	-

Technical details are subject to change without notice.

1. Configuration Commands via RS-232

1.1 Introduction

The RS-232 interface is used for configurating and debugging. It can be used by a terminal or a PC with a terminal program (i.e. Windows terminal) with the following parameters:

Bit rate: 19200 baud
Data: 8 bit
Parity: no parity
Stop bit: 1 stop bit
Handshake: no handshake

All written parameters are interpreted as HEX-values !!!! (In this document HEX values are marked with an '$' in front of the given value.)

The configuration parameters are only valid after storing them into the internal EEPROM ('CS') and resetting the module (e.g. by command 'RS' or by power_up).

An individual help to each command is printed by pressing the command with a following '?'.

1.2 Common Commands

1.2.1 Save Configuration

Name:	Save configuration
Input command syntax:	CS
Input parameter syntax:	> ok > error

Description:
All actual configuration parameters are stored into the internal EEPROM. The module answers with an 'ok', if well done, or with 'error', if an error occurred. The parameters are only valid until the next reset of the module.

1.2.2 Set Bit Rate, Display Bit Rate

Name:	Set bit rate
Input command syntax:	SB
Input parameter syntax:	< index(0-$F)/register-format

Name:	Display bit rate
Input command syntax:	DB
Output parameter syntax:	> index(0-$F)/register-format

The default bit rate is set by the coding switch at the front panel:

Setting of coding switch	Default-bit rate [Kbit/s]
0	125
1	250
2	500

Table 1.2.1: Setting of default bit rate

This bit rate can be overwritten by the command ‘SB’, with any value except $FFFF. The new value can be displayed by calling the command ‘DB’.

The bit rate can be given as index (0...$F) or as the direct register format (intel CAN controller, 20 MHz):

Index [HEX]	Register [HEX]	Bit rate [bit/s]
0	0x1600	1 M
1	0x1b00	666 K
2	0x2f00	500 K
3	0x1b01	333 K
4	0x2f01	250 K
5	0x2f02	166 K
6	0x1c04	125 K
7	0x2f04	100 K
8	0x1b09	66 K
9	0x2f09	50 K
A	0x2f0e	33 K
B	0x2f18	20 K
C	0x2f27	12.5 K
D	0x2f31	10 K
E	0x7f7f	6 K
F	0x1200	1.6 M

Table 1.2.2: Setting bit rate by command

1.2.3 Wakeup Time

Name:	Wakeup time
Input command syntax:	WU
I/O parameter syntax:	<> time(ms)

Description:
After wakeup the module waits for the duration of the given wakeup time before is starts with any action at the CAN.
Acceptable values are 0...$7FFF, according to 0...32767 ms. If the command is called without a parameter value, the actual value of the parameter is displayed at the monitor.

1.2.4 Module Number

Name:	Module number
Input command syntax:	MN
I/O parameter syntax:	<> number(1-$3F)

Description:
The MAC_ID of the module is set by this command. If the command is called without a parameter value, the actual value of the MAC_ID is displayed at the monitor.

1.2.5 Reset Module

Name:	Reset module
Input command syntax:	RS
I/O parameter syntax:	No parameters

Description:
This commands is an easy way to reset the module (instead of a power_up).

1.2.6 CAN Default

Name:	CAN default
Input command syntax:	CD
I/O parameter syntax:	No parameters

Description:
With this command, all parameter are defaulted (but not actually stored in the EEPROM).

1.3 Additional Common Commands

1.3.1 Help

Name:	Help
Input command syntax:	??
Output parameter syntax:	> commands

Description:
This command returns a list of all possible commands (without following supervisor commands).

1.4 Configuring the DeviceNet interface

1.4.1 Scanner Parameter

Name:	Scanner parameter
Input command syntax:	SC
I/O parameter syntax:	<> scan_nr MAC_ID rxlen txlen rate

Parameter description:
scan_nr: table index (0...$1F)
MAC_ID: MAC_ID of the module. that should be scanned
rxlen: Length of the produced data of that module (0...$FE)
txlen: Length of the consumed data of that module (0...$FE)
rate: Expected package rate of the connection (0 - $FFFF)

Command description:
This command installs a module to the internal scanner. The ‘scan_nr’ means the table number of this entry. The scanner polls all modules written in this table up to the first not written entry. If an entry should be deleted, just type ‘SC nr -’ and the scanner stops at this point. The ‘rxlen’ and ‘txlen’ parameters are checked against the remote module and if a difference is recognized, a configuration error is shown to the PLC. The expected package rate is written to the remote module and the module is scanned by the half rate to reduce CAN access.

If the command is called without parameters, the actual values of the parameters are displayed at the monitor.

1.4.2 Installing COMM_ID (with PAM-scanner only)

Name:	COMM_ ID install
Input command syntax:	CI
Input parameter syntax: Output parameter syntax:	< scan_nr R/T COMM_ID < scan_nr >RX_COMM_ID TX_COMM_ID

Parameter description:
can_nr: The entry-number from the scanner table above
R/T: ‘R’ for a receiving COMM_ID,
‘T’ for a transmitting COMM_ID
COMM_ID: COMM_ID (0...$FFFE)

Command description:
This command installs a COMM_ID for a given scanner entry (refer to 1.5.1). If at least one COMM_ID is given for a scanner entry, the data length of this entry means the pure net data length without any COMM_ID. The scanner switches automatically to a special COMM_ID handler, who builds the correct transmit-frames with COMM_ID rsp. extract raw data from received frames.

COMM_ID can only be added to a given scanner entry, but overwriting a scanner entry clears installed COMM_ID's of the old entry.

1.4.3 Slave Parameter

Name:	Slave parameter
Input command syntax:	SL
I/O parameter syntax:	<> rxlen txlen

Parameter description:
rxlen: Consumed_Data_Length (0...$FE)
txlen: Produced_Data_Length (0...$FE)

Command description:
The module can also be accessed by another scanner. You just have to define the Consumed_Data_Length (rxlen) and the Produced_Data_Length (txlen) with this command.
If the command is called without a parameter value, the actual value of the parameters are displayed at the monitor.

1.5 Configuration of the PLC Interface

The interface between the module and the PLC is located in the peripheral address range of the PLC. The absolute address of the module depends on the slotnumber of the module. Every slot occupies only 8 byte data of the PLC. To raise the performance of this interface, the module is able to simulate more than one slot. Due to this simulation the addresses of the following modules on the PLC bus are shifted to slots behind the last simulated slot.

There are generally two ways to access the data of DeviceNet:

1. Direct access through I/O-addresses is an easy and quick way to write and read data. It is just necessary to make a mapping table between I/O-addresses and DeviceNet data. The disadvantage is the small I/O data area of the PLC and the larger PLC cycle time simulating more data.

2. Indirect access through a command and a data window.
   The command window is fixed in the first 4 bytes of the first simulated slot, input and output. So the module must be inserted in an ‘analog slot’, i.e. one of the first 8 slots!!! The data window is definable in address and range.

1.5.1 Slotnumber

Name:	Slot number
Input command syntax:	SN
I/O parameter syntax:	<> slot

Parameter description:
slot: slotnumber (0...7)

Command description:
Calling this command stores the slotnumber of the module. The module needs this number, to recalculate the relative address positions from absolute addresses. The absolute addresses are set by the commands described in the following chapters.
The slot number is necessary to communicate with the PLC.

1.5.2 Data Window

Name:	Window
Input command syntax:	WD
I/O parameter syntax:	< I/O addr len > I addr len O addr len

Parameter description:
I: Input Window, PLC can read data from DeviceNet
O: Output Window, PLC can write data to DeviceNet
addr: Absolute address of the window
len: Length of the window (0,2,4,8)

Command description:
This command defines the address and the range of the data window. Using this data window and additional the command window the PLC can access to the whole data area of the DeviceNet interface (chapter 2).
If the command is called without parameter values, the actual values of the parameters are displayed at the monitor.

1.5.3 PLC - DeviceNet Mapping Table

Name:	PLC parameter
Input command syntax:	PP
I/O parameter syntax:	< nr M P I/O M/I addr len MAC_ID offs < nr M P I/O M/I addr len MAC_ID offs COMM_ID < nr S P I/O M/I addr len offs < nr E > nr M/S/E P I/O M/I addr len [MAC_ID] offs [COMM_ID] | | | | | | | | MOTOROLA / INTEL notation | | Input / Output | Peripheral area access to Master / Slave, or table-End

Parameter description:

nr:	Table index (0...$3F)
addr:	Absolute PLC address of this entry
len:	Data length of this entry
MAC_ID:	DeviceNet MAC_ID of the module, that should be accessed (Master only)
offs:	Offset inside the data stream of the device
COMM_ID:	COMM_ID of the data object (only on devices with special COMM_ID handling)

Command description:
This command installs a data mapping table between PLC and DeviceNet to perform direct data exchange. The link is made between PLC peripheral input or output (‘I/O’) address ‘addr’ with data length ‘len’ and DeviceNet data from the remote module ‘MAC_ID’ (via Master ‘M’) or from the own module (via Slave interface ‘S’) after position ‘offs’ of the data stream depending on the direction (I/O).
If it is a remote device with the special COMM_ID handling, the COMM_ID in this entry is necessary to distinguish between several objects.

SPECIAL:
A special feature is made with the ‘M/I’ letter: The PLC accesses to words in MOTOROLA notation, means a high-byte/low-byte orientation inside the PLC. Because most DeviceNet data is orientated in INTEL notation (low-byte/high-byte), it is possible to change this orientation with the ‘I’ letter. This is only possible with a even data length, otherwise it is switched back automatically.

The table ends at a special entry written with ‘PP nr E’. If no parameter is written, this command shows the complete table.

2. The Command Page

The easiest and quickest data access is done with the mapping table (refer to 1.5.1). But it maybe possible that not all data is directly accessible due to the small peripheral address range of the PLC. Therefore it is possible to perform an indirect access with a command and a data window.

The data window is definable with the ‘WD’ command (refer to 1.5.2). The command window occupies the first 4 bytes of the module, input and output.

In input direction it holds some state information of the module. In output direction the PLC program can send a command with several specifier to the module. The module recognize a new command only, if at least one bit of this window is changed. So, if nobody writes to the command window, no command is executed and the old command has not to be reset.

2.1 Module State

The command window shows in input direction the state of the module. The memory layout with ‘n’ as the base address of the module is:

IB n:	state bits
IB n+1:	error MAC_ID
IB n+2 \	
IB n+3 - IW n+2:	special information, command dependent

The state bits in the first input byte are:

I n.7 : can_off
I n.6 : can_warn
I n.5 : scan_ok
I n.4 : slave_ok
I n.3 : config_error
I n.2 : future use
I n.1 : success
I n.0 : toggle

The bits ‘can_off’ and ‘can_warn’ show the actual state of the CAN chip.

‘scan_ok’ shows, if the internal scanner is connected to all slaves of the scan-table.

The bit ‘slave_ok’ means, that the internal slave interface is connected by another scanner. If the data length of the slave is zero, this bit is always set.

The bit ‘config_error’ shows a configuration error like ‘Duplicate_MACID’ or ‘wrong data size’ in scan-table.

The bits ‘success’ and ‘toggle’ belongs directly to the command interface:
‘toggle’ toggles after each finished command. ‘success’ shows, if this command was finished successfully.

The second input byte shows the MAC_ID of the module, which caused the last error like ‘timeout’ or ‘configuration error’.
The default value is $0xFF.

The second word of the command window is reserved for command dependent return values (future use).

2.2 Command Interface

Via the command window the PLC can send special commands to the module. A new command is only recognized by the module, if at least one bit of the command window is changed. So, if nobody writes to the command window, no command is executed, and the last written command has not to be reset. A recognized command is finished by set or reset the ‘success’ bit and by an inverse ‘toggle’ bit. The memory layout with ‘n’ as the base address of the module is:

QB n: command
QB n+1: subcmmd
QB n+2 : \
QB n+3: -QW n+2 : command dependent information

Following commands are implemented:

command	subcmmd	QB n+2 / QW n+2	QB n+3	description
0x00	x	x		No command is executed.
0x01	x	x		Resets error: The error MACID (IB n+1) is reset to his default value $0xFF.
0x02	MACID	len	off	Reads 'len' data bytes from device 'MACID' after data offset 'off'. If 'MACID' > $0x3F, data is read from the internal slave interface.
+0x08= 0x0A				Converts INTEL to MOTOROLA notation (refer to 1.5.3).
0x03	MACID	len	off	Writes 'len' data bytes to device 'MACID' after data offset 'off' and updates mirror_image to process image. If 'MACID' > $0x3F, data is written to the internal slave interface.
+0x08= 0x0B				Converts INTEL to MOTOROLA notation (refer to 1.5.3).
+0x10= 0x13				Writes only to mirror_image to have consistent data with the following writes.
+0x08 +0x10= 0x1B				Converts INTEL to MOTOROLA notation and writes only to mirror_image.
0x04	MACID +off	COMM_ID		Reads data from object 'COMM_ID' from PAM device 'MACID'. The data size is always equal to the whole data window size, the 2 MSB's in 'MACID+off' determine the offset in steps of the data-window size. Example: Is the read-data window size 8 bytes and the 2 MSB's are '01' (=1), the command reads from offset 8.
+0x08= 0x0C				Converts INTEL to MOTOROLA notation (refer to 1.5.3).
0x05	MACID +off	COMM_ID		Writes data to object 'COMM_ID' from PAM device 'MACID' and updates mirror_image to process_image. The data size is always equal to the whole data window size, the 2 MSB's in 'MACID+off' determine the offset in steps of the data-window size.
+0x08= 0x0D				Converts INTEL to MOTOROLA notation (refer to 1.5.3).
+0x10 0x15				Writes only to mirror_image to have consistent data with the following writes.
+0x08 +0x10= 0x1D				Converts INTEL to MOTOROLA notation and writes only to mirror_image.

3. Meaning of the CAN/DvN-LED in Front of the Module

At start-up the LED indicates the wakeup procedure by changing the colour periodically. During the wakeup time the module is passive on the CAN. The wakeup time can be set by the command ‘WU’.

LED State	Meaning
LED off	The LED is turned off while sending the duplicate MAC-ID request.
LED green flashing	After wakeup and duplicate MAC-ID request the LED turns to flashing green. The communication between DeviceNet participants has not yet been established.
LED lights continuously green	There is no error and communication has been established
LED red flashing	A time-out has been detected in the communication between one DeviceNet participant and the CSC595/2.
LED lights continuously red	A fatal error has occurred. Please check the settings and wiring of attached DeviceNet nodes.

Table 3.1.1: LED states

4. Examples

4.1 Configuration Of A *FESTO* Pneumatic Valves Unit As A DeviceNet Slave

- Set the CAN bit rate of the CSC595 to the same value as all other CAN participants at this CAN net.
- Additionally it is necessary to define a MAC-ID for each device.

Use the according manuals for the above required settings .

For the following example the MAC_ID of the *FESTO* pneumatic valves has to be set to 5. All numbers in the following example has to be set as HEX values.

All commands used in this example are described in the DeviceNet software manuals of these devices

---

1. Use the command ‘SC’, to setup the slaves that shall be scanned by the master with the following parameters:

SC	**0**	**5**	**3**	**2**	**64**	
	*	*	*	*	)	expected package rate is 100 ms (=$0x64)
	*	*	*	*		-> polling cycle 50 ms
	*	*	*	*		
	*	*	*	)		the master shall send 2 bytes to the module *)
	*	*	*			
	*	*	)			the module shall answer with 3 bytes *)
	*	*				
	*	)				the MAC-ID of the device has to be set to 5
	*					
	)					indicates the first entry of the scan table

*) The number of bytes that has to be transferred is device-specific.

The end of the table has to be defined by the same command but without any parameters:

SC	**1**	**-**	
	*	)	end of table
	*		
	)		2nd entry in the scan table

2. Set bit rate to 500 kbit/s

SB	**2**	
	)	bit rate index (2 = 500 kbit/s)

3. Use the command ‘PP’ to create the following PLC mapping table (two entries):

```
PP  0  M  P  O  I  4C 2  5  0
    *  *  *  *  *  *  *  *  )
    *  *  *  *  *  *  *  *
    *  *  *  *  *  *  *  )
    *  *  *  *  *  *  *
    *  *  *  *  *  *  )
    *  *  *  *  *  *
    *  *  *  *  *  )
    *  *  *  *  *
    *  *  *  *  )
    *  *  *  *
    *  *  *  )
    *  *  *
    *  *  *
    *  *  )
    *  *
    *  )
    *
    )
```

0 byte offset

MAC-ID of slave

length = 2 (data)

PLC address $0x4C = decimal 76 (AW 76) intel format

output (view from PLC)

P-area

master (scanner)

first table entry

```
PP  1  M  P  I  I  4C 2  5  0
    *  *  *  *  *  *  *  *  )
    *  *  *  *  *  *  *  *
    *  *  *  *  *  *  *  )
    *  *  *  *  *  *  *
    *  *  *  *  *  *  )
    *  *  *  *  *  *
    *  *  *  *  *  )
    *  *  *  *  *
    *  *  *  *  )
    *  *  *  *
    *  *  *  )
    *  *  *
    *  *  *
    *  *  )
    *  *
    *  )
    *
    )
```

0 byte offset

MAC-ID of slave

length = 2 (data)

PLC address $0x4C = decimal 76 (EW 76) intel format

input (view from PLC)

P-area

master (scanner)

2. table entry

4. If you wish to determine the state of the pneumatic valves, you have to set the PLC data mapping table to the following values:

```
PP    2     M     P     I     M     4E    1     5     2
      *     *     *     *     *     *     *     *     )     0 byte offset
      *     *     *     *     *     *     *     *
      *     *     *     *     *     *     *     )           MAC-ID of slave
      *     *     *     *     *     *     *
      *     *     *     *     *     *     )                 length = 1 (status)
      *     *     *     *     *     *
      *     *     *     *     *     )                       PLC address $0x4C =
      *     *     *     *     *                             decimal 78 (EB 78)
      *     *     *     *     )                             motorola format
      *     *     *     *
      *     *     *     )                                   input
      *     *     *                                         (view from PLC)
      *     *     *
      *     *     )                                         P-area
      *     *
      *     )                                               master (scanner)
      *
      )                                                     3. table entry

PP    3     E
      *     )                 end of table
      *
      )                       4. entry in PLC data mapping table
```

5. The MAC-ID of the CSC595 shall be set to 17 in this example with the following command:

MN 11
*) MAC-ID of CSC595 ($0x11 = decimal 17)
*
) module number

Important: Do not miss to save all settings with the command ‘CS’ and to trigger a RESET with the command ‘RS’ or with the PLC at the CSC595!

Additionally you must consider, in which slot the CSC595 is inserted:

SN 0

The module is inserted in slot 1 --> base address of this module is 64 (EW und AW)

If more PLC modules shall be inserted in the following slots of the PLC, the simulated I/O data length of the CSC595 has to be considered (unless, if the following modules are additional CSC595 modules or ‘standard‘ PLC devices).
Does the first CSC595 simulate a maximum of 8 data bytes in the slot, the slot number for a second CSC595 has to be set to

SN 1

If the first CSC595 extends the data length of one slot, i.e. if it simulates more than 8 bytes, the slot number for a second CSC595 has to be set to

SN 2

This is necessary because the PLC must be able to address each data byte, even if it is one of a standard PLC device or if it is a simulated data byte of the CSC595. With each set of 8 simulated data bytes the slot number of the following PLC devices has to be incremented by ‘1’:

If the first CSC595 is inserted in slot one and uses the data length of 2 slots, the second CSC595 must use the data bytes of the third slot.

Example for direct access:
As an option it is possible to access data directly without using the above listed table. The data bytes that are used with the direct access must be different to those which are used in the PLC data mapping table!

WD	**I**	**44**	**8**	data window EW 68 - EW 74 for common reading of data
WD	**O**	**44**	**8**	data window AW 68 - EW 74 for writing data (outside the data mapping table !)

4.2 Configuration Of A *KUKA* Robot As A DeviceNet Master

The robot acts as a master. The data of the robot shall be received using the PLC's peripheral data input bytes 80 to 85. Data that shall be transmitted to the robot using the PLC's peripheral data output bytes 80 to 83.

```
PP   00   S   P   I   M   50   6   0
     *    *   *   *   *   *    *   *
     *    *   *   *   *   *    *   *
     *    *   *   *   *   *    *   )      0 byte offset
     *    *   *   *   *   *    *
     *    *   *   *   *   *    )          length = 6 bytes
     *    *   *   *   *   *
     *    *   *   *   *   )                PLC address $0x50 =
     *    *   *   *   *                    decimal 80 (EY 80)
     *    *   *   *   )                    motorola format
     *    *   *   *                        (no data swapping)
     *    *   *   )                        input
     *    *   *                            (view from PLC)
     *    *   *
     *    *   )                            P-area
     *    *
     *    )                                slave
     *
     )                                     1. table entry
```

```
PP   01   S   P   O   M   50   4   0
     *    *   *   *   *   *    *   *
     *    *   *   *   *   *    *   *
     *    *   *   *   *   *    *   )      0 byte offset
     *    *   *   *   *   *    *
     *    *   *   *   *   *    )          length = 4 bytes
     *    *   *   *   *   *
     *    *   *   *   *   )                PLC address   $0x50 =
     *    *   *   *   *                    decimal 80 (AY 80)
     *    *   *   *   )                    motorola format
     *    *   *   *                        input
     *    *   *   )                        (view from PLC)
     *    *   *                            P-area
     *    *
     *    )                                slave
     *
     )                                     2. table entry
```

PP	**2**	**E**	
	*	)	end of table
	*		
	)		3. entry of data mapping table

Setting the MAC-ID (module number) of the CSC595.

MN	**11**		
*	)		MAC-ID of the CSC595 ($0x11 = decimal 17)
*			
)			module number

Setting the slave parameters of the CSC595:

SL	**6**	**4**	
	*	)	produced data length of the internal slave (CSC595 ->robot)
	*		
	)		consumed data length of the internal slave (robot -> CSC595)

Setting the scanner parameters of the CSC595:

SC	**0**	**-**	no scan table, no master functionality

Now just save the data at the CSC595 with the command ‘CS’ and trigger RESET using the command ‘RS’.