PMA Prozeß- und Maschinen-Automation GmbH

KS 800
Multi-temperature-controller

KS800

DeviceNet

KS

Interface Protocol

DeviceNet

9499 040 58611

gültig ab: 8363

A publication of PMA Prozeß- und Maschinen-Automation GmbH

Subject to alteration without notice.

PMA Prozeß- und Maschinen-Automation GmbH
P.O. Box 31 02 29
D 34058 Kassel
Germany

Warranty restriction:

No warranty is taken for the complete correctness of the contents, since errors cannot be precluded completely despite utmost care. Any hints will be gratefully accepted.

Contents

1 Introduction

According to the DeviceNet instrument classification, KS800-DN is “Generic Device”, in particular, a “Group 2 Only Server”, i.e. services of the ”Predefined Master/Slave Connection Set” instead of UCMM (Unconnected Message Manager) support are used. For parameter or configuration data access, ”Explicit Messaging” is used. Process data exchange is by ”I/O polling” (”Bit-Strobe” and ”Change-Of-State” are not supported).
For details, see “DeviceNet Specifications Volume I, Release 2.0, Chapter 7”.

The access to the KS 800 functionality via the CAN bus is described on the following pages (DeviceNet). As a prerequisite, a tool for sending CAN messages to KS800 via the CAN bus is required.

The KS800-DN controller is provided with two (bus) interfaces as standard:
CAN bus for control and configuration
Connection of the general KS800 engineering tools (TTL level)

For integrating KS800-DN into a programmer/configuration environment (e.g. Rockwell's RSNetWorx), an EDS file (Electronic Data Sheet) is available. This file can be loaded from PMA homepage http://www.pma-online.de.

2 DeviceNet protocol stack

The description of the DeviceNet protocol stack interface does not explain the operating principle of the KS800-DN controller. Only the access mechanisms via the CAN bus are described. DeviceNet knowledge is an advantage.

2.1 Data

The DeviceNet protocol stack permits the access to the control parameters or process data of KS800-DN only via defined classes. There are two types of classes. Classes specified in DeviceNet and manufacturer-specific classes. However, the difference is not in the access type. The classes specified by DeviceNet mainly describe the communication, the manufacturer-specific classes describe the controller and its parameters.
For addressing individual attributes (values, parameters, etc.), the “address” is required. This address comprises three components. Class number (Class Id), instance number (instance Id) and attribute number (Attrib Id). The controller provides the following classes:

DeviveNet Classes			Manufacturer-specific classes		
Class Id (hex)	**max. numb. of instan-ces**	**Name**	**Class Id (hex)**	**max. numb.of instan-ces**	**Name (CANopen Index)**
01	1	Identity	64	1	atManProfile0 (0x2000 - 0x204F)
02	1	Message Router	65	8	atManProfile1 (0x2100 – 0x214F)
03	1	DeviceNet	66	8	atManProfile2 (0x2200 – 0x22AF)
04	1	Assembly	67	8	atManProfile3 (0x2300 – 0x234F)
05	2	Connection	68	8	atStdProfileSection0 digital Input (0x6000 - 0x6015)
			69	8	atStdProfileSection1 digital Output (0x6200 – 0x6215)
			6A	8	atStdProfileSection2 Controller Function Block (0x7000 - 0x705F)
			6B	8	atStdProfileSection3 (0x7100 - 0x715F)
			6C	8	atStdProfileSection4 (0x7800 - 0x782F)
			6D	8	atStdProfileSection5 (0x7900 - 0x702F)

Classes implemented in the KS800-DN controller

The class function or the signification of their attributes are not explained in this description. For DeviceNet classes, this information is given in the specification. The manufacturer-specific classes are given in the functional description.
The structure of manufacturer-specific classes is similar to the structure of the CANopen object directory. The class of the index area is given in Table 1. The sub-index (instance) is determined from the addressed control channel (1 to 8), or is 1 for all device data (see Class 0x64). The attribute is determined by the last two digits of the index. E.g. for access, under DeviceNet, to the same value as with CANopen under index 0x7803, sub-index 2, the address is as follows: Class 0x6C, instance 0x02, attribute 0x03. Accesses, under CANopen, to sub-index 0 and thus to instance 0 with DeviceNet are not permitted.

Exchange of configuration data is by Explicit Messages.
Exchanging process data via the CAN bus is done as a byte stream with the following structure (by polling).

Total 49-byte produce data (1byte + 6 bytes * 8 channels)
48-byte consume data (6 bytes * 8 channels)

The terms Produce/Consume are selected from the KS800-DN view, whilst the ”bus” view is always used in the DeviceNet specification.

Byte (produce)	Desciption	Byte (consume)	Description
Byte 0	Device status	Byte (n*6)+0	Wvol (LSB)
Byte (n*6)+1	Xeff (LSB)	Byte (n*6)+1	Wvol (MSB)
Byte (n*6)+2	Xeff (MSB)	Byte (n*6)+2	Yman (LSB)
Byte (n*6)+3	Channel status (LSB)	Byte (n*6)+3	Yman (MSB)
Byte (n*6)+4	Channel status (MSB)	Byte (n*6)+4	Control Byte
Byte (n*6)+5	Ypid (LSB	Byte (n*6)+5	Update Byte
Byte (n*6)+6	Ypid (MSB)		

Process data description (Polling)

n = channel number ($0 \leq n \leq 7$)

2.2 Algorithms and sequences

After connecting the controller to supply voltage and CAN bus, the controller outputs Duplicate MAC check. For addressing the controller via the CAN bus subsequently, the following steps are necessary.
A master must be assigned to the instrument and, simultaneously, the communication channel for configuration data and/or process data must be opened.

Now, configuration data can be exchanged with the controller via Explicit Messages. With the KS800-DN controller, process data exchange is exclusively via polling. After completing, the communication channels must be re-closed.
The following example is given for communication between a master (MAC Id 0) and a KS800-DN (MAC Id 1). All values in columns ID, DLC and data bytes in the table are hexadecimal.

ID	DLC	Data bytes								Description
		0	1	2	3	4	5	6	7	
40E	6	0	4B	3	1	1	0			Allocate Explicit Message (Byte 4 = 1).
40C	7	0	10	5	1	C	3	0		Set Attribute. class 5, instance 1, attribute 12, value = 3 This ensures that the Explicit Message goes timeout only, when no other communications (e.g. IO polling) exist any more.
40E	6	0	4B	3	1	2	0			Allocate Polling (byte 4 = 2). Opens the communication channel for process data communication.
40C	7	0	10	5	2	9	0	0		Sets the expected packet rate. Writing this value activates polling and sets the timeout for this communication. Writing 0 (byte 5 and 6) as in this example deactivates the timeout.
										Start of Fragmented Poll-Request-Sequence
40D	8	0	1	2	3	4	5	6	7	Sends process data 01 - 07 (bytes 1 to 7). The process data structure is given in Table 2.
40D	8	41	8	9	A	B	C	D	E	Sends process data 08 - 14 (bytes 1 to 7).
40D	8	42	F	10	11	12	13	14	15	Sends process data 15 - 21 (bytes 1 to 7).
...	.	.	.	.	.	.	.	.	.	...
40D	8	45	24	25	26	27	28	29	2A	Sends proces data 36 – 42 (bytes 1 to 7).
40D	7	86	2B	2C	2D	2E	2F	30		Sends process data 43 – 48 (bytes 1 to 6). After receiving the last telegram, the controller will reply with 49 bytes.
										End of Fragmented Poll-Request-Sequence
40E	5	0	4C	3	1	2				Release Polling (Byte 4 = 2). Deletes the communication channel for the process data.
40E	5	0	4C	3	1	1				Release Explicit (Byte 4 = 1). Deletes the communication channel for configuration data.

Example for communication via the bus

2.3 Telegram structure

Each of the following tables describes the structure of a CAN message comprising CAN identifier (11 bits) and data bytes (max. 8 bytes). The fields marked X in the data bytes are not transmitted.

2.3.1 Duplicate MAC check

Identifier bits											Data bytes							
10	9	8	7	6	5	4	3	2	1	0	0	1	2	3	4	5	6	7
1	0	MAC ID						1	1	1	00	20	02	Serial number				X

Duplicate MAC check request

Sent 2x by the KS800-DN controller during boot-up . Data length code = 7.
MAC Id: KS800-DN device address
Series number: KS800-DN series number (8 last digits of E²PROM code number)

2.3.2 Opening the communication channels

Allocate Explicit Message and Allocate Polled

Identifier bits											Data bytes							
10	9	8	7	6	5	4	3	2	1	0	0	1	2	3	4	5	6	7
1	0	MAC ID						1	1	1	Source MAC ID	4B	3	1	Alloc	Source MAC ID	X	X

Allocate Master/Slave Connection set

Must be sent to KS800-DN by the master. Data length code = 6
Alloc provides information, whether only Explicit Message or polling is opened. 1 ≤ Alloc ≤ 3.
Explict ⇒ Alloc = 1; Polling ⇒ Alloc = 2; Explicit and Polling ⇒ Alloc = 3.
MAC Id: KS800-DN device address
Source MAC Id: master device address (e.g. scanner, CANanalyzer)

2.3.3 Closing the communication channels

Release Explicit Message and Polled

Identifier bits											Data bytes							
10	9	8	7	6	5	4	3	2	1	0	0	1	2	3	4	5	6	7
1	0	MAC ID						1	1	0	Source MAC ID	4C	3	1	Release	X	X	X

Release Master/Slave Connection set

Must be sent to KS800-DN by the master. Data length code = 5.
Release provides information, whether Explicit Message or Polling is closed. 1 ≤ Release ≤ 3.
Explict ⇒ Release = 1; Polling ⇒ Release = 2; Explicit and Polling ⇒ Release = 3.
MAC Id: KS800-DN device address
Source MAC Id: master device address (e.g. scanner, CANanalyzer)

2.3.4 Parameter reading

For reading out an individual parameter from a class, the following request must be sent to KS800-DN.

Identifier bits											Data bytes							
10	9	8	7	6	5	4	3	2	1	0	0	1	2	3	4	5	6	7
1	0	MAC ID						1	0	0	Source MAC ID	0E	Class	Instance	Release	X	X	X

Get Attribute Single

Must be sent to KS800-DN by the master. Data length code = 5.
Class: Addressed class
Instance: Addressed instance of a class
Attributes: Addressed attribute of a class
MAC Id: KS800-DN device address
Source MAC Id: master device address (e.g. scanner, CANanalyzer)
As a reply, the KS800-DN controller sends a CAN message with DLC between 5 and 8, whereby the value of the requested parameter is from data byte 4. Byte 4 contains the (LSB).

2.3.5 Parameter writing

Identifier bits											Data bytes							
10	9	8	7	6	5	4	3	2	1	0	0	1	2	3	4	5	6	7
1	0	MAC ID						1	0	0	Source MAC ID	10	Class	Instance	Instance	X	X	X

Set Attribute Single

Must be sent to KS800-DN by the master. Data length code between 6 and 8, dependent of the value to be written (8, 16 or 24 bits).
Class: Addressed class
Instance: Addressed instance of a class
Attributes: Addressed attribute of a class
MAC Id: KS800-DN device address
Source MAC Id: Master device address (e.g. scanner, CANanalyzer)
Value: Parameter value 1 to 3 bytes
As a reply, the KS800-DN controller sends a CAN message with DLC 5

2.3.6 Requesting or sending process data

Since exclusively polling is supported with KS800-DN, new process data must always be sent when requesting process data from KS800-DN. Request is based on the 48-byte Consume process data with a sequence of CAN messages (fragmented polling).

1st Telegram

Identifier bits											Data bytes							
10	9	8	7	6	5	4	3	2	1	0	0	1	2	3	4	5	6	7
1	0	MAC ID						1	0	1	n	Byte 0 bis 6 der Prozeßdaten						X

Masters I/O Poll Message

The CAN messages must be sent to KS800-DN by the master. The data length code is always 8 data bytes except the last telegram.
n: current number indicating the actual telegram number. $0 \leq n \leq 63$. Thereby, maximum 63 fragments each with 7 bytes can be transmitted under DeviceNet. This corresponds to max. 512 bytes of process data. According to Table 2: Process data description (Polling), however, KS800-DN requires only 48 bytes (Consume). I.e. 6 messages each with 7 bytes of process data + 1 message with 6 bytes process data must be transmitted. The first message is identified n=0 (fragment-type: first fragment, fragment count: 0). With the other messages, bit 6 is set (fragment type: middle, i.e. n=0x41,0x42,..). To identify the last message, the uppermost bit in the 1st data byte is simply set (fragment type: final). For KS800-DN, this means that n extends from 0x00, 0x41 to 0x45 and the last telegram is transmitted with n = 0x86.
MAC Id: KS800-DN device address
As a reply, the KS800-DN controller sends a sequence of CAN messages with a structure similar to the ones described above, however, 49 bytes of Produce process data (7 * 7) are sent. Only the CAN identifier is different.

Identifier bits											Data bytes							
10	9	8	7	6	5	4	3	2	1	0	0	1	2	3	4	5	6	7
1	0				MAC ID						n	Byte 0 bis 6 der Prozeßdaten						X

Slave's I/O Poll Response Message

n: see above
MAC Id: KS800-DN device address

Further details on the message formats are given in the DeviceNet specifications volume I and II, release 2.0. The Predefined Master/Slave Connection Set is given in chapter 7 (volume I), for an example of Fragmented Poll, see
7-6.4, page 7-32.

3 Object directory / polling access PRODUCE DATA

The following data can be read by KS800 in polling mode:

Byte	Channel	Value	Data Type
1	1	Device status information	unsigned8
2/3	1	Xeff	FixedPoint1
4/5	1	Channel status information	Unsigned16
6/7	1	Ypid	FixedPoint1
8/9	2	Xeff	FixedPoint1
10/11	2	Channel status information	Unsigned16
12/13	2	Ypid	FixedPoint1
14/15	3	Xeff	FixedPoint1
16/17	3	Channel status information	Unsigned16
18/19	3	Ypid	FixedPoint1
20/21	4	Xeff	FixedPoint1
22/23	4	Channel status information	Unsigned16
24/25	4	Ypid	FixedPoint1
26/27	5	Xeff	FixedPoint1
28/29	5	Channel status information	Unsigned16
30/31	5	Ypid	FixedPoint1
32/33	6	Xeff	FixedPoint1
34/35	6	Channel status information	Unsigned16
36/37	6	Ypid	FixedPoint1
38/39	7	Xeff	FixedPoint1
40/41	7	Channel status information	Unsigned16
42/43	7	Ypid	FixedPoint1
44/45	8	Xeff	FixedPoint1
46/47	8	Channel status information	Unsigned16
48/49	8	Ypid	FixedPoint1

Used data-formats:
Unsigned8: 8-bit-value, without sign, range 0 ... 256
Unsigned 16: 16-bit-value, without sign, range 0 ... 65535
Fixedpoint1: 16 bit-integer-value with 1 fixed digit behind the decimal point, within –32767 ... +32767. When interpreting these values, the last digit is cosidered as a digit behind the decimal point, e.g. 12345 means 1234,5 (°C), or 873 are 87,3 (%).

Note: The data signification is described in the KS800 function manual.

- **Device status information:**

  Bit 0 *) - offline[0], online [1]
  Bit 1 *) - DO1...12 fail
  Bit 2 *) - DO13...16 fail
  Bit 3 *) - heating current short circuit
  Bit 4 - di1 ⎫ "Input State" signalling,
  Bit 5 - di2 ⎪ if configred as an input,
  Bit 6 - di3 ⎪ otherwise "0".
  Bit 7 - di4 ⎭

- **Channel status information:**

  Bit 0 *) - alarm HH
  Bit 1 *) - alarm H
  Bit 2 *) - alarm L
  Bit 3 *) - alarm LL
  Bit 4 *) - sensor fail alarm
  Bit 5 *) - heating current alarm
  Bit 6 *) - leakage current alarm
  Bit 7 *) - alarm DO_x
  Bit 8 *) - W2 active
  Bit 9 *) - Wint active
  Bit 10*) - Wanfahr active
  Bit 11*) - self-tuning active
  Bit 12*) - self-tuning error
  Bit 13*) - controller A/M
  Bit 14*) - controller switched off (coff)

4 Object directory / polling access CONSUME DATA

The following data can be written into KS800 in polling mode:

Byte	Channel	Value	Data Type
1/2	1	Set-point Wvol	FixedPoint1
3/4	1	Correcting variable Yman	FixedPoint1
5	1	Control information	Unsigned8
6	1	Update information	Unsigned8
7/8	2	Set-point Wvol	FixedPoint1
9/10	2	Correcting variable Yman	FixedPoint1
11	2	Control information	Unsigned8
12	2	Update information	Unsigned8
13/14	3	Set-point Wvol	FixedPoint1
15/16	3	Correcting variable Yman	FixedPoint1
17	3	Control information	Unsigned8
18	3	Update information	Unsigned8
19/20	4	Set-point Wvol	FixedPoint1
21/22	4	Correcting variable Yman	FixedPoint1
23	4	Control information	Unsigned8
24	4	Update information	Unsigned8
25/26	5	Set-point Wvol	FixedPoint1
27/28	5	Correcting variable Yman	FixedPoint1
29	5	Control information	Unsigned8
30	5	Update information	Unsigned8
31/32	6	Set-point Wvol	FixedPoint1
33/34	6	Correcting variable Yman	FixedPoint1
35	6	Control information	Unsigned8
36	6	Update information	Unsigned8
37/38	7	Set-point Wvol	FixedPoint1
39/40	7	Correcting variable Yman	FixedPoint1
41	7	Control information	Unsigned8
42	7	Update information	Unsigned8
43/44	8	Set-point Wvol	FixedPoint1
45/46	8	Correcting variable Yman	FixedPoint1
47	8	Control information	Unsigned8
48	8	Update information	Unsigned8

- **Control information:**

Bit 0	-	A[0] / M [0/1]
Bit 1	-	COFF [1]
Bit 2	-	W/W2 [0/1]
Bit 3	-	Wext/Wint [0/1]
Bit 4	-	Ostart [1]

- **Update information:**

Bit 0	-	A/M [0/1]
Bit 1	-	COFF [1]
Bit 2	-	W/W2 [0/1]
Bit 3	-	Wext/Wint [0/1]
Bit 4	-	Ostart [1]
Bit 6	-	Yman [1]
Bit 7	-	Wvol [1]

The update information can be used to select, which data are taken over from the control record by KS800. If one of the bits is set, the relevant KS800 data are updated.

5 Object directory / EXPLICITE ACCESS

Class/-Attr.	Obj. (Symb. Name)	Name	Type	read/-write
FB Device				
CLS 64		***Process-data Fktnr.0***		
1	VAR	Status 1 Unit-State1	Unsigned8	ro
2	VAR	Basic hardware options HWbas	Unsigned16	ro
3	VAR	SW options SWopt	Unsigned16	ro
4	VAR	SW code no. SWCode	Unsigned16	ro
5	VAR	SW version SWVersion	Unsigned16	ro
6	VAR	Operating version OPVers	Unsigned16	ro
7	VAR	Version of EEPROMS EEPVers	Unsigned16	ro
8	VAR	Configuration mode switch-over OPMod	Unsigned8	rw
9	VAR	Self-tuning stop/start of all group controllers OStartg	Unsigned8	rw
A	VAR	Reset of local data change flag	Unsigned8	rw
CLS64		***Parameters and configuration data fct. no. 0***		
B	VAR	Baudrate COM 1 C900	Unsigned16	rw*
C	VAR	Decive address Adr1	Unsigned16	rw*
D	VAR	Mains frequency 50/60	Unsigned8	rw*
E	VAR	Baudrate COM 2 C900 (CAN Baudrate)	Unsigned16	rw*
F	VAR	Device address Adr2 (CAN node ID)	Unsigned16	rw*
10	VAR	Release of cooling function for water cooling	FixedPoint	rw
11	VAR	Heating-current-reset /- quicktest	Unsigned8	rw
CLS64		***Process data fct. no. 2***		
20	VAR	Status alarm outputs State_alarm_out	Unsigned8	ro
21	VAR	Status dig. inputs/outputs State_dio	Unsigned8	ro
CLS64		***Parameters and configuration data fct. no. 2***		
22	VAR	Main config. C500	Unsigned16	rw*
23	VAR	Main config. C530	Unsigned16	rw*
24	VAR	Allocation HC/leakage current C151	Unsigned16	rw*
25	VAR	Heating current cycle Hccycl	Unsigned16	rw*
26	VAR	Span end for HC input HC 100	FixedPoint1	rw*
27	VAR	Forced digital output OUT1...OUT8	Unsigned8	rw
28	VAR	Forced digital output OUT9...OUT16	Unsigned8	rw

29	VAR	Forced digital output OUT17...OUT19 + Calibrating-relais	Unsigned8	rw
FB Input				
CLS 65		***Process data fct. no. 0***		
0	ARRAY	Signal input fail Input_X_Failed	Unsigned8	ro
1	ARRAY	Main variable x1	FixedPoint1	ro
2	ARRAY	Raw measurement value before measurement value correction INP1	FixedPoint1	ro
CLS 65		***Parameters and configuration data fct. no. 1***		
10	ARRAY	Measurement value correction X1 input X1in	FixedPoint1	rw
11	ARRAY	Measurement value correction X1 output X1out	FixedPoint1	rw
12	ARRAY	Measurement value correction X2 X2in	FixedPoint1	rw
13	ARRAY	Measurement value correction X2 X2out	FixedPoint1	rw
14	ARRAY	Sensor type (T,H) C200	Unsigned16	rw*
15	ARRAY	Fail: sensor break (T) C205	Unsigned16	rw*
16	ARRAY	Phys. value at 0% X0	FixedPoint1	rw*
17	ARRAY	Phys. value at 100% X100	FixedPoint1	rw*
18	ARRAY	Substitute value at sensor fail XFail	FixedPoint1	rw*
19	ARRAY	Filter time constant measurement value correction Tfm	FixedPoint1	rw*
1A	ARRAY	Ext. cold junction reference temperature Tkref	FixedPoint1	rw*
1B	ARRAY	Digital signal allocation C190	Unsigned16	rw*
FB Controller				
CLS 66		***Process data fct. no. 0***		
0	ARRAY	Status 1	Unsigned8	ro
1	ARRAY	Eff. set-point Weff	FixedPoint1	ro
2	ARRAY	Effective process value Xeff	FixedPoint1	ro
3	ARRAY	Effective correcting variable Ypid	FixedPoint1	ro
4	ARRAY	Control deviation xw	FixedPoint1	ro
5	ARRAY	Automatic/manual switch-over	Unsigned8	rw
6	ARRAY	Self-tuning start OStart	Unsigned8	rw
7	ARRAY	Wext/Wint switch-over	Unsigned8	rw
8	ARRAY	W/W2 switch-over	Unsigned8	rw
9	ARRAY	Controller on/off Coff	Unsigned8	rw

CLS66		***Parameters and configuration data fct. no. 0***		
A	ARRAY	Main configuration 1, control C100	Unsigned16	rw*
B	ARRAY	Main configuration 2, control C101	Unsigned16	rw*
C	ARRAY	Configuration tuning C700	Unsigned16	rw*
D	ARRAY	Signal allocation anal. C180	Unsigned16	rw*
CLS66		***Process data fct. no. 1***		
10	ARRAY	Set-point status WState	Unsigned8	ro
11	ARRAY	Effective internal set-point Wint	FixedPoint1	ro
12	ARRAY	Int. set-point, non-volatile Wnvol	FixedPoint1	rw
13	ARRAY	Int. set-point, volatile Wvol	FixedPoint1	rw
CLS66		***Parameters and configuration data fct. no. 1***		
14	ARRAY	Min. set-point limit for Weff W0	FixedPoint1	rw
15	ARRAY	Max. set-point limit for Weff W100	FixedPoint1	rw
16	ARRAY	Additional set-point W2	FixedPoint1	rw
17	ARRAY	Set-point gradient plus Grw+	FixedPoint1	rw
18	ARRAY	Set-point gradient minus Grw-	FixedPoint1	rw
19	ARRAY	Set-point gradient W2 Grw2	FixedPoint1	rw
1A	ARRAY	Loop-alarm on/off C102	Unsigned16	rw*
CLS66		***Parameters and configuration data fct. no. 3***		
30	ARRAY	Neutral zone Xsh	FixedPoint1	rw
31	ARRAY	Min. pulse length Tpuls	FixedPoint1	rw
32	ARRAY	Actuator travel time Tm	FixedPoint1	rw
33	ARRAY	Switching difference signaller Xsd1	FixedPoint1	rw
34	ARRAY	Trigger point separation additional contact LW	FixedPoint1	rw
35	ARRAY	Switching difference additional contact Xsd2	FixedPoint1	rw
36	ARRAY	Neutral zone Xsh1	FixedPoint1	rw
37	ARRAY	Neutral zone Xsh2	FixedPoint1	rw
CLS66		***Process data fct. no. 4***		
40	ARRAY	Difference correcting variable dYman	FixedPoint1	rw
41	ARRAY	Absolute correcting variable Yman	FixedPoint1	rw
42	ARRAY	Increm. correcting variable adjustment Yinc	Unsigned8	rw
43	ARRAY	Decrem. correcting variable adjustment Ydec	Unsigned8	rw
44	ARRAY	Rate of increm. and decrem. correcting variable adjustment Ygrw_is	Unsigned8	rw

CLS66		***Parameters and configuration data fct. no. 4***		
45	ARRAY	Min. correcting variable limiting Ymin	FixedPoint1	rw
46	ARRAY	Max. correcting variable limiting Ymax	FixedPoint1	rw
47	ARRAY	Working point for correcting variable Y0	FixedPoint1	rw
48	ARRAY	Max. mean value of correcting variable Yhm	FixedPoint1	rw
49	ARRAY	Limit for mean value formation LYh	FixedPoint1	rw
CLS66		***Process data fct. no. 5***		
50	ARRAY	Status Tuning State_Tune1	Unsigned8	ro
51	ARRAY	Eff. additional parameter number ParNeff	Unsigned8	ro
52	ARRAY	Effective additional parameter number ParNr	Unsigned8	rw
53	ARRAY	Delay time heating Tu1	FixedPoint1	ro
54	ARRAY	Rate of increase heating Vmax1	FixedPoint1	ro
55	ARRAY	Process gain heating Kp1	FixedPoint1	ro
56	ARRAY	Self-tuning error code heating MSG1	Unsigned8	ro
57	ARRAY	Delay time cooling Tu2	FixedPoint1	ro
58	ARRAY	Rate of increase cooling Vmax	FixedPoint1	ro
59	ARRAY	Process gain cooling Kp2	FixedPoint1	ro
5A	ARRAY	Self-tuning error code cooling MSG2	Unsigned8	ro
CLS66		***Parameters and configuration data fct. no. 5***		
5B	ARRAY	Correcting variable during process at rest YOptm	FixedPoint1	rw
5C	ARRAY	Step height at identification dYopt	FixedPoint1	rw
5D	ARRAY	Parameter set to be optimized POpt	Unsigned8	rw
5E	ARRAY	Hysteresis at parameter switch-over OXsd	FixedPoint1	rw
5F	ARRAY	Trigger point 1 Trig1	FixedPoint1	rw
CLS66		***Parameters and configuration data fct. no. 6***		
60	ARRAY	Proportional band 1 Xp1_1	FixedPoint1	rw
61	ARRAY	Integral time 1 Tn1_1	FixedPoint1	rw
62	ARRAY	Derivative time 1 Tv1_1	FixedPoint1	rw
63	ARRAY	Min. cycle time 1 T1_1	FixedPoint1	rw
64	ARRAY	Proportional band 2 Xp2_1	FixedPoint1	rw
65	ARRAY	Integral time 2 Tn2_1	FixedPoint1	rw
66	ARRAY	Derivative time 2 Tv2_1	FixedPoint1	rw
67	ARRAY	Min. cycle time 2 T2_1	FixedPoint1	rw

CLS66		***Parameters and configuration data fct. no. 7***		
70	ARRAY	Proportional band 1 Xp1_2	FixedPoint1	rw
71	ARRAY	Integral time 1 Tn1_2	FixedPoint1	rw
72	ARRAY	Derivative time 1 Tv1_2	FixedPoint1	rw
73	ARRAY	Min. cycle time 1 T1_2	FixedPoint1	rw
74	ARRAY	Proportional band 2 Xp2_2	FixedPoint1	rw
75	ARRAY	Integral time 2 Tn2_2	FixedPoint1	rw
76	ARRAY	Derivative time 2 Tv2_2	FixedPoint1	rw
77	ARRAY	Min. cycle time 2 T2_2	FixedPoint1	rw
CLS66		***Parameters and configuration data fct. no. 10***		
7A	ARRAY	Max. correcting value Ya	FixedPoint1	rw
7B	ARRAY	Start-up set-point Wa	FixedPoint1	rw
7C	ARRAY	Start-up holding time TPa	FixedPoint1	rw
		FB Alarm		
CLS 67		***Process data fct. no. 0***		
0	ARRAY	Alarm status 1 Status_AL1	Unsigned8	ro
1	ARRAY	Alarm status 2 Status_AL2	Unsigned8	ro
2	ARRAY	Heating current measurement value HC	FixedPoint1	ro
CLS67		***Parameters and configuration data fct. no. 0***		
3	ARRAY	Low alarm LimL	FixedPoint1	rw
4	ARRAY	High alarm LimH	FixedPoint1	rw
5	ARRAY	Switching difference low/high alarms xsd_2	FixedPoint1	rw
6	ARRAY	Low low alarm LimLL	FixedPoint1	rw
7	ARRAY	High high alarm LimHH	FixedPoint1	rw
8	ARRAY	Heating current limit value LimHC	FixedPoint1	rw
9	ARRAY	Src: signal source (T,H) C600 Fnc:function (Z)	Unsigned16	rw*
A	ARRAY	Alarm target C601	Unsigned16	rw*

*) These data can be changed only in configuration mode.

6 DeviceNet application notes:

6.1 General hints for DeviceNet applications with KS800-DN and Rockwell scanners

(The following Rockwell HW/SW combination can be used as a reference for the hints: SLC 5/03 CPU with 1747-SDN scanner module under RSNetWorx)

Scanner settings (default):

- interscan delay (10msec) — for I/O poll rate, must be determined application-specifically (if too low, it might be a problem for KS800 polled I/O)
- foreground-to-background poll ratio (1) — if >1, the node configured accordingly is polled only each nth time (e.g. in case there is even faster I/O polling apart from KS800)
- advanced settings (caution: **will affect the network functionality !!!**)
  - expected packet rate (75msec) — 4 x expected packet rate = timeout for I/O-msg. (expl. msg. without timeout), separate for each communication. I.e. I/O transfer must be completed after 300msec (otherwise timeout => new communication build-up)
  - transmit retries (1) — Repetition rate for faulty sending (effect on application (reaction time, etc.))

6.2 Scanner configuration (start/stop/reset)

These settings are <u>not</u> possible via RSNetWorx (only via bits in output word 0) of the scanner (via RSLogix ...). The details are given in the 1747-SDN installation instructions.

Command Word 0:S.0

Bit 4: 1 = disable network	scanner stops communication with devices on the network
Bit 6: 1 = halt scanner	all scanner operations stop when this command is issued
Bit 7: 1 = reboot	causes the scanner to reset as though power had been cycled

Important: During operation <u>without</u> scanner (for first tests during commissioning, e.g. with CANalyzer), the system must be reset" first, because all modules on the scanlist go/are in timeout and are not visible any more at the network (valid for all group 2 modules, e.g. KS800-DN).

6.3 General configuration values for use of KS800 in a DeviceNet

- *Do not select a too short Interscan delay* (< 20msec critical), the KS800-DN cycle time is 65msec, i.e. a shorter delay does not lead to new I/O infos
- *alternatively, select background polling* adjust background poll ratio accordingly

• *Put KS800 at the beginning of the MAC-Id area*, to avoid too long I/O cycles (the large I/O size will lead to unacceptable cycle times, because ready-to-send frames cannot be transmitted).

The following rule is applicable for all DeviceNet applications:
Set slave with highest I/O to lowest MAC-Id, independent of background polling !

6.4 Checklist for commissioning/trouble shooting of DeviceNet networks

Network: topology [DN-Spec] terminations [DN-Spec] voltage (24V, min. 11V) [DN spec.]

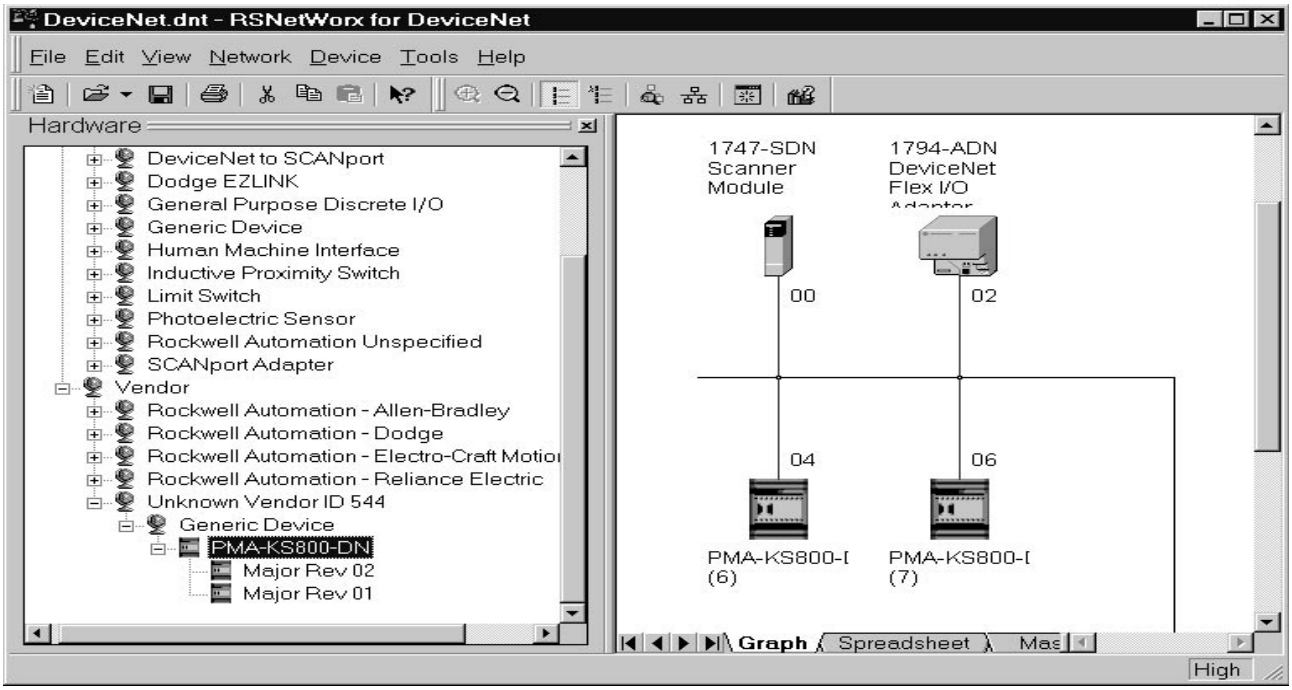
Settings: baudrate (flex-I/O autobaud) run-time problems (scanning time) oscillator (100ppm stability => quartz oscillator) [DN spec.]

without scanner: start testing only using analyzer (to preclude a possible error source) don't forget system reset (see above) !!!

Network scan sequence (CPU, RSLinx, RSNetWorx):
2 x UCMM (analyzer shows: "open expl msg com ..."), as not supported => 2 x timeout
2 x "Alloc Master/Slave Cl3(Dnet) ..." [cyclic repetition of this sequence]

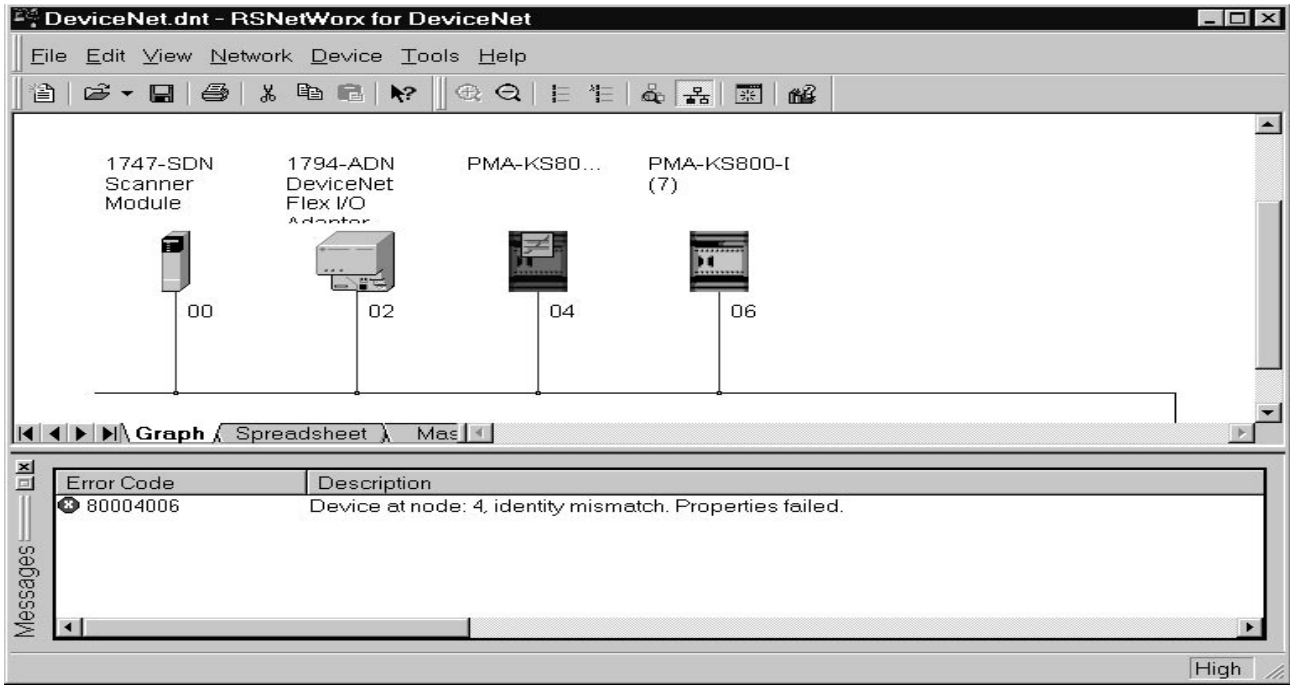
6.5 Version handling of KS800-DN EDS files under Rockwell's RSNetWorx

Software versions of DeviceNet units are specified with "Major.Minor" revision in the identity object. I.e. with a version 1.4, major=1, minor=4. According to DeviceNet specification, a new EDS file in the RSNetWorx database is only created with major revision changes. This is selectable in the hardware menu (for offline projecting).



Selection of various (software/EDS) major revisions

A mismatch of projected (offline) and connected KS800-DN identity object is detected during "Browsing Network" with the following structure.

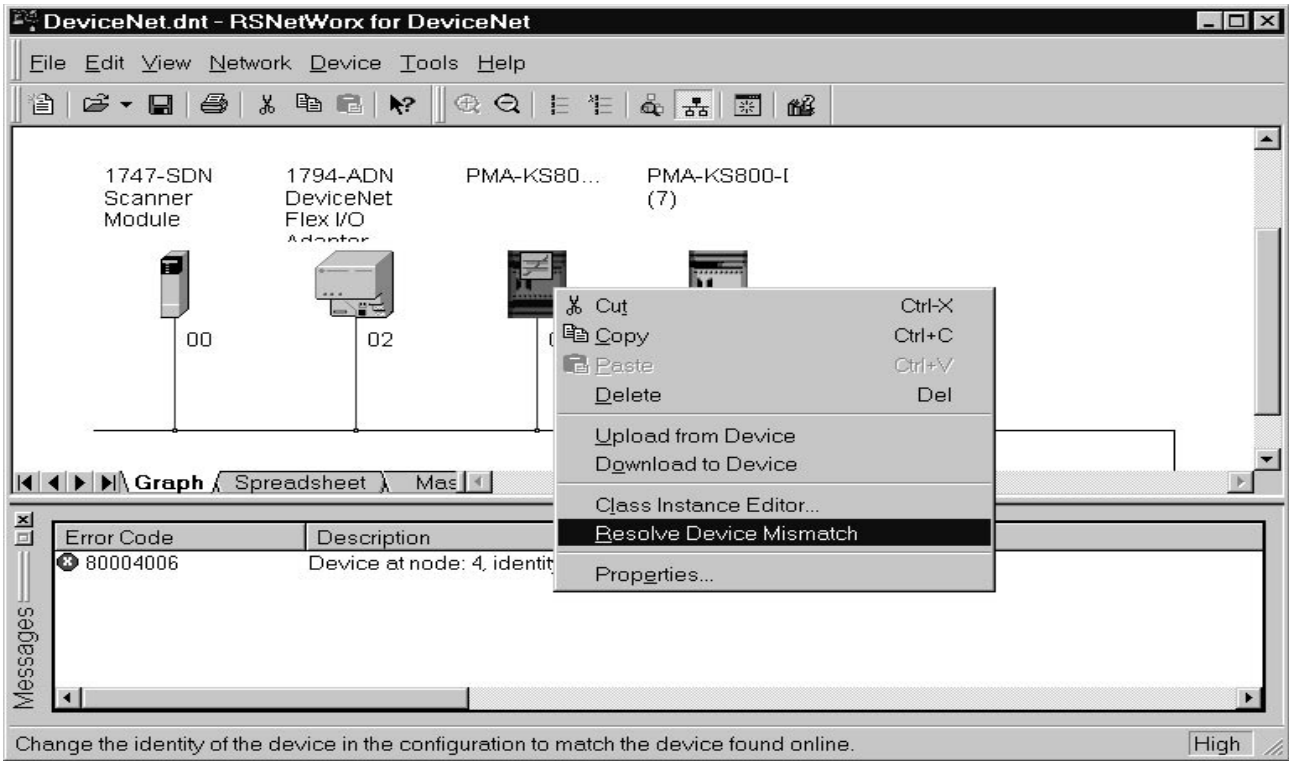


Software version mismatch structure

By double-clicking on the “mismatched” unit, windows “Properties” as shown is opened. Cards “Device Parameters” and “EDS I/O default” are not provided.

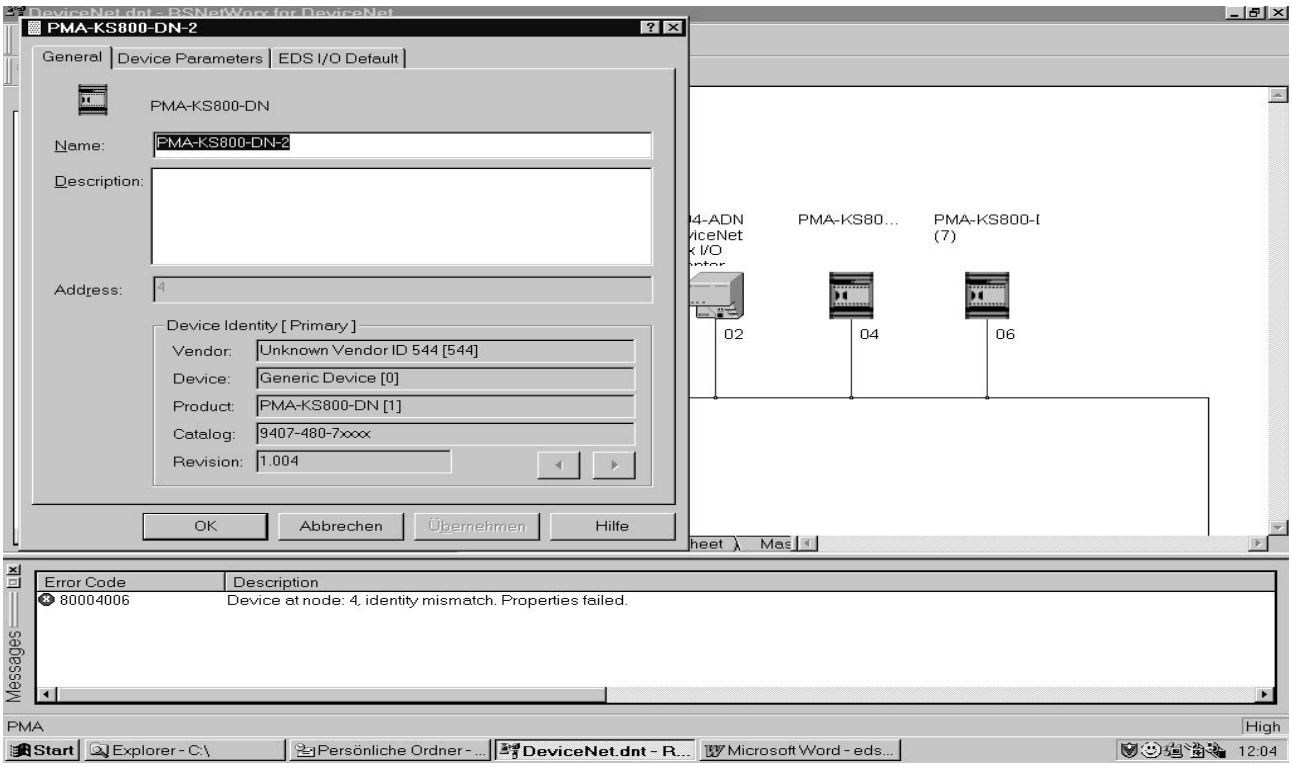
"Mismatched" Device

When activating the context menu (right mouse key), a new selection item “Resolve Device Mismatch” is provided. It can be used for updating the configuration with the identity information from the unit.



Identity information update via context menu

When double-clicking on KS800 for the next time, the correct version (provided from the unit) is displayed. Now, access to parameters and I/O default is also possible.



Update completed (full device access)

The procedure described above permits handling of various versions in one project. Major changes of the unit must always be marked “Major Revision”. This will always lead to creation of another EDS file and relevant selection possibility under “Hardware”. “Minor” changes will cause overwriting of the old file within the EDS database of RSNetWorx during installation of the relevant EDS file.

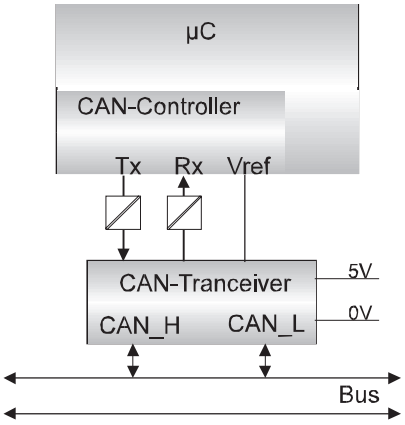
When making changes to KS800-DN, increase minor or major revision according to the DeviceNet Spec. This must be marked on the type label of the unit. The current KS800-DN EDS file(s) are available on PMA homepage http://www.pma-online.de in the Internet.

7 CAN Physical Layer

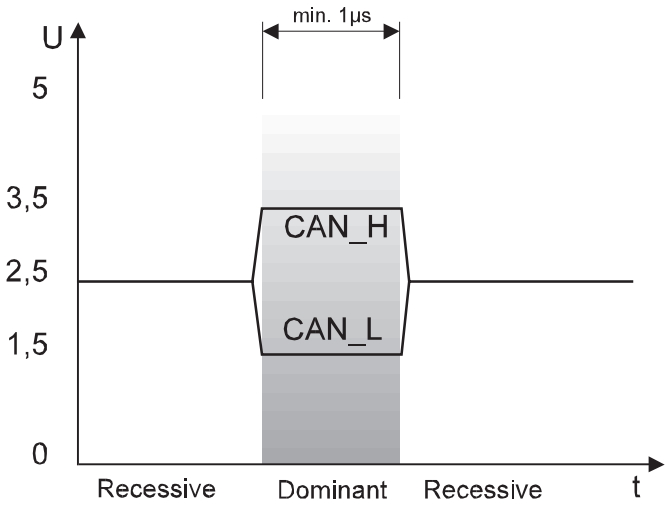
Related to the CAN Physical Layer, there are a number of standards. The most important standard for general applications is the "CAN High-Speed Standard ISO 11898-2". The recommendations given below are based primarily on this standard and are valid independent of the used CAN protocol (CANopen / DeviceNet).

7.1 ISO 11898-2 nodes:

A node compliant with ISO 11898-2 comprises an _C with CAN controller (which may or may not be integrated), which is connected with a CAN transceiver via Rx and Tx line. The transceiver is connected to the differential CAN-H and CAN-L lines at the CAN bus. With KS800, this (transceiver) connection is galvanically isolated



With CAN bus, the nominal CAN bus levels are described as _Recessive_ (nominal voltage 2,5V for CAN-H and CAN-L) and _Dominant_ (nominal 3,5V for CAN-H and 1,5V for CAN-L).



7.2 Bit rates and bus lengths

The maximum useful bus length in a CAN network is determined by a variety of physical effects, in particular:

- The delay time of the connected bus nodes (with/without opto-couplers) and the delay time of the bus cable (propagation delays),
- various scanning times within a CAN bit cell due to the oscillator tolerances of bus nodes,
- signal amplitude attenuation due to the DC resistance of the bus cable and the input resistances of bus nodes.

When using ISO 11898-2-compliant transceivers, the bus mentioned below can be reached with standard bus cables. Nevertheless, the bus lengths may be considerably shorter with the high bit rates (1 Mbit / 800 kbit) due to the number/speed of any opto-couplers (galvanic isolation)!

7.3 Practical bus lengths

CAN-Profil(s)	Baud-rate	Bus-length	Nom. Bit-time
CANopen	1 MBd	30 m	1 µs
CANopen	800 kBd	50 m	1,25 µs
CANopen/DeviceNet	500 kBd	100 m	2 µs
CANopen/DeviceNet	250 kBd	200 m	4 µs
CANopen/DeviceNet	125 kBd	500 m	8 µs
CANopen	50 kBd	1000 m *)	20 µs
CANopen	20 kBd	2500 m *)	50 µs
CANopen	10 kBd	5000 m *)	100 µs

(*) With very long cables, using galvanic isolation and repeaters is indispensable.

For further information on bus lengths, see also standards CiA _DS-102_ (CANopen) or ODVA _DeviceNet Specifications Volume I, Release 2.0_, in particular, Appendix A and B.

7.4 Cable parameters

ISO 11898-2 defines some DC or AC parameters for the cables which can be used in CAN bus networks (typically, pairwisely twisted cables with defined electrical properties are used). The important AC parameters are 120 Ohm cable impedance and a nominal "propagation delay" of 5 ns/m ! Recommendations for the bus cables and terminating resistors are given in the following table:

Bus-length	Bus-cable (Z: 120 Ohm, tp: 5ns/m)		Terminating-resistor	Max. Bit-Rate
	Spec. resistance	Cable cross section		
0 ... 40 m	70 mOhm/m	0,25mm□, 0,34mm□ AWG 23, AWG 22	124 Ohm, 1%	1 MBd @ 40m
40 m ... 300 m	<60 mOhm/m	0,34mm□, 0,6mm□ AWG 22, AWG 20	127 Ohm, 1% *)	> 500 kBd @ 40m
300 m ... 600 m	<40 mOhm/m	0,5mm□, 0,6mm□ AWG 20	127 Ohm, 1% *)	> 100 kBd @ 40m
600 m ... 1 km	<26 mOhm/m	0,75mm□, 0,8mm□ AWG 18	127 Ohm, 1% *)	> 50 kBd @ 40m

*) With very long cables, a higher value for the terminating resistor (150 .. 300 Ohm) is useful for reducing the attenuation.

Further recommendations for CAN networks (especially large ones):

- Galvanic isolation is necessary with very long bus cables (e.g. 400m). A separate ground line is purposeful.
- The voltage drop (potential difference) between the transceiver ground potentials should be low (smaller than 2 V, supply voltage from power supply in the middle of the cable).
- The total input resistance of bus nodes should be > 500 Ohm.
- Any tap lines should be as short as possible to prevent/reduce reflections, e.g. 6m @ 500kbit (DeviceNet). < 1 m with higher bit rates !

For further information, refer to ODVA (DeviceNet), CiA (CANopen), various chip manufacturers and Internet.