System Specifications

3

6

2

Standard Interfaces

Physical External Interface

Summary

This document specifies the mechanical, electrical and logical functionality of the standard Physical External Interface (PEI).

Version 01.01.02 is a KNX Approved Standard.

This document is part of the KNX Specifications v2.1.

Document updates

Version	Date	Modifications
0.1	2000.11.13	SDB: Document creation
0.2	2000.12.01	SDB: Prepared for Release for Voting
0.3	2001.05.23	Preparation for Final Voting. Inclusion of comments from RfV.
1.0	2001.12.20	Preparation of the Approved Standard.
1.1	2009.02.09	• AN113 “Use of mechanical PEI” integrated. • Preparation for inclusion in the KNX Specifications v2.0.
01.01.01	2013.10.28	Editorial updates for the publication of KNX Specifications 2.1.
01.01.02	2013.12.11	Final editorial review in view of publication of the KNX Specifications v2.1.

References

[01] Chapter 3/6/3 “External Message Interface”

[02] Part 9/1 “Cables and Connectors”

[03] Part 9/4 “BCUs and BIMs”

Filename: 03_06_02 Physical External Interface v01.01.02 AS.docx
Version: 01.01.02
Status: Approved Standard
Savedate: 2013.12.11
Number of pages: 29

Contents

1 Overview

The Physical External Interface (PEI) is the standard interface situated in a KNX device between the Bus Access Unit and the Application Module. The Bus Access Unit contains all the KNX protocol layers plus the optional internal user application.

The Application Module either contains input and output values accessible in parallel by the internal user application via the parallel PEI I/O interface. Or there is a serial communication between Application Module and Bus Access Unit, i.e. via the serial PEI interface. Serial PEI-communication with restricted PEI I/O communication is also possible.

For pure serial PEI-communication three typical examples can be given:

- The Application Module contains a serially readable and writeable 8 bit shift register.
- The Application Module contains an own microcontroller that runs the external user application. That means that the whole KNX device is a serially communication two-processor system with non-shared memory.
- The Application Module is a PC which makes asynchronous peer-to-peer communication via its serial interface. In that case the external user application is mostly some kind of tool software, e.g. the ETS or another PC software of the KNX Association.

The PEI is the KNX standard way to couple Bus Access Unit and Application Module. In both the PEI consists of a mechanical/electrical and a software part. The software part is optional for the Application Module. The mechanical/electrical part exists in two versions: a 10 pin hardware interface and a 12 pin hardware interface. Resistors of certain defined values that are connected between pin 5 and 6 of both hardware interface versions at the Application Module allow encoding 21 different PEI-types.

The 21 PEI-types can be grouped in 4 different categories:

1. Special purpose PEI-types 0, 1, 13, 15 and 20

PEI-type 0:	for applications intended for use without an adapter present (i.e. no resistor between pin 5 and 6).
PEI-type 1:	By convention, PEI-type 1 adapters shall not be implemented. This type is reserved as an application-PEI-type that can be set to ensure that the application will not be started.
PEI-type 13 and 15:	These PEI-type shall encode an implementation specific PEI-type. See 5.4.
PEI-type 20:	allows a manufacturer to download initial settings to the BCU. The behaviour is implementation specific. Refer to [03] for closer descriptions.

2. PEI-types 3, 5, 7, 9, 11 and 18

   These PEI-types are reserved for future extensions of the PEI standard. No application programs or hardware adapters shall use these types.

3. PEI-types 2, 4, 6, 8, 17, and 19

   These PEI-types shall allow for parallel PEI I/O communication. The parallel PEI I/O interface implements a digital read/write I/O interface to the Application Module with the capability to read analogue values too. The Application Module's binary values are to be read and written by the internal KNX user application.

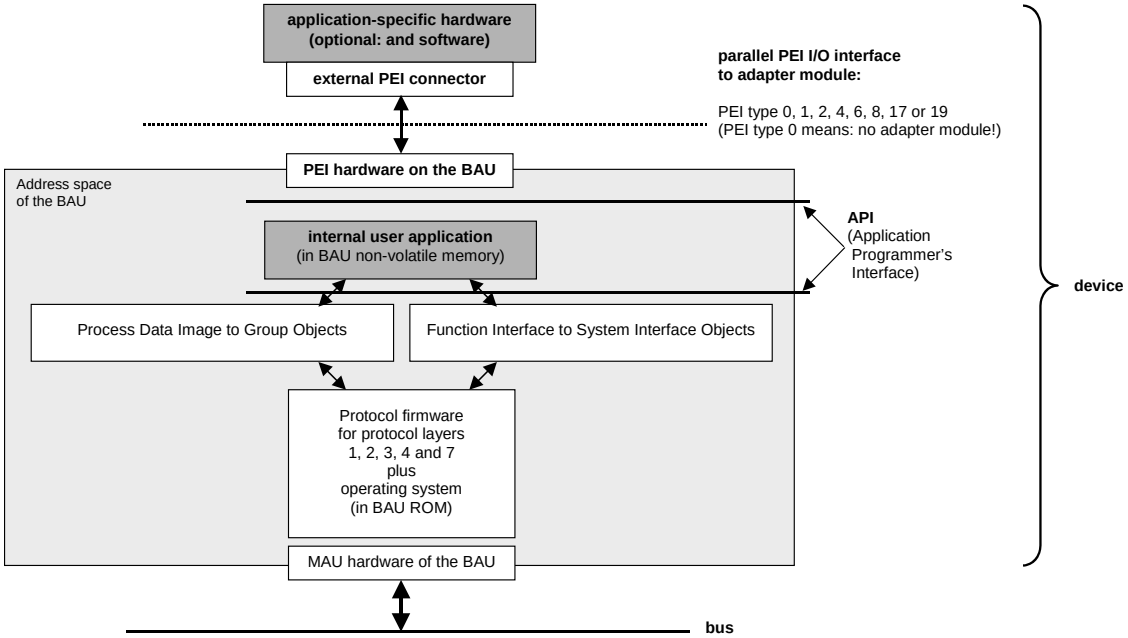


Figure 1 - A typical one-processor device with parallel PEI I/O communication

4. PEI-types 10, 12, 14 and 16:

   These PEI-types shall allow for serial PEI-communication. The serial PEI implements a serial interface to the Application Module. The Application Module has to run the serial protocol requested by the PEI of the Bus Access Unit. Communication between the external user application run by the external processor and the processor which runs the Bus Access Unit with its (optional) internal user application is via the External Message Interface. The External Message Interface is a representation of the Internal Message Interface. The mapping between both interfaces is done by the serial protocol between Bus Access Unit and Application Module. For more details about the External Message Interface see [01].

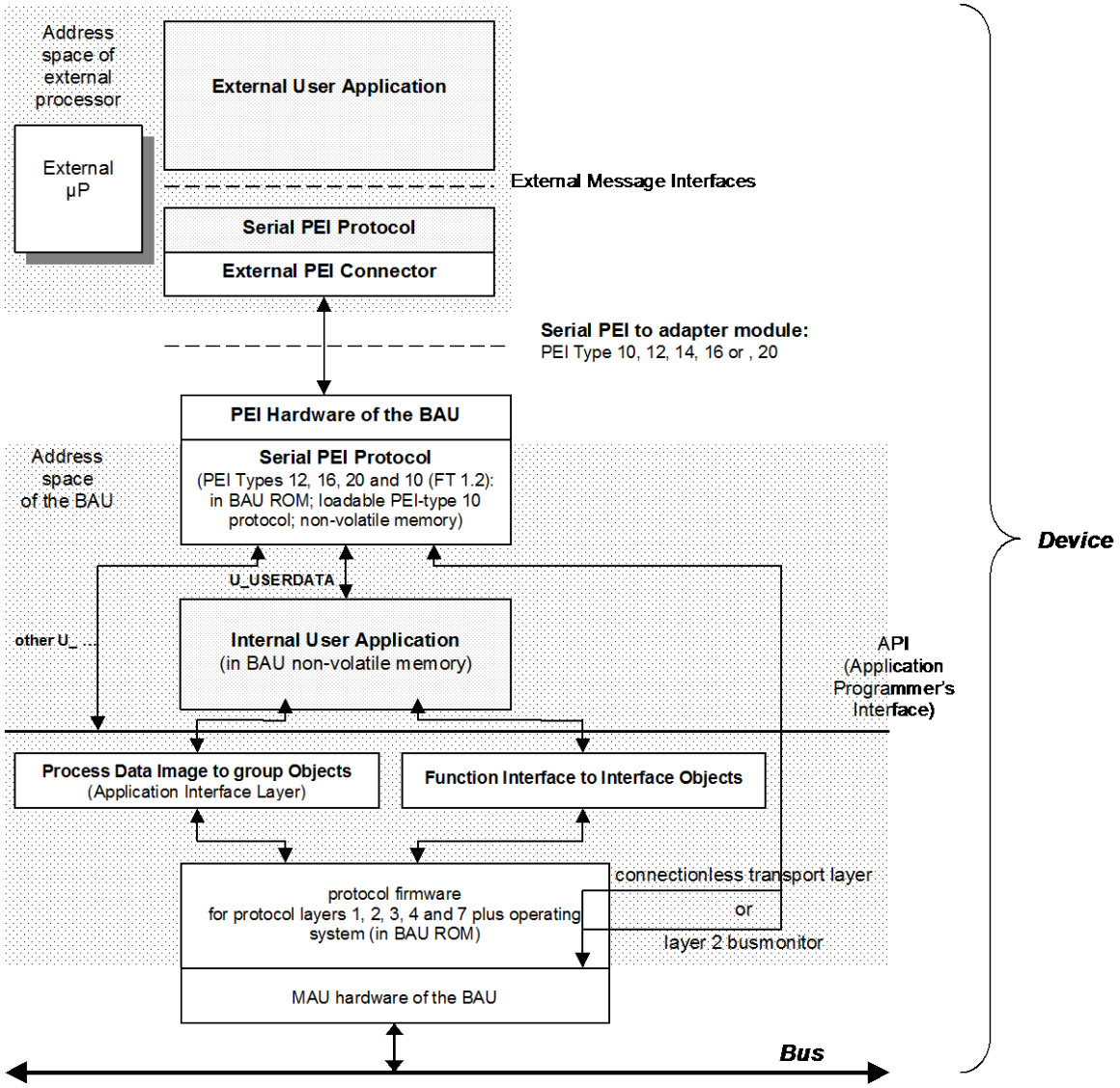


Figure 2 - A typical device communicating via Serial PEI, PEI-type 14

Each of the PEI-types 10, 12, 14 and 16 stands for a different serial protocol support:

- Types 10 & 16 serial asynchronous
- Type 12 serial synchronous with message interface
- Type 14 serial synchronous with data block interface
- Type 10 additionally allows defining a serial protocol of its own between Application Module and Bus Access Unit. In this case the Application Module's serial protocol's counterpart must be downloaded to the Bus Access Unit.

Devices need not to have the Physical External Interface: other non-standard ways of coupling Bus Access Unit and periphery are also possible, e.g. by a Dual-Ported RAM or by a private point-to-point protocol which is not based on the PEI hardware specification. Finally devices can also be totally integrated, i.e. without a separation of Bus Access Unit and Application Module.

2 PEI-type management and handling

The PEI-type shall be encoded by the system parameter EE_PEI-Type in the Bus Access Unit (software type) and by means of a resistor in the Application Module (hardware type).

The EEPROM-variable EE_PEI_Type shall be a system variable of the Bus Access Unit. It must be set by the Management Client during download of the internal user application into the Bus Access Unit. EE_PEI_Type shall be set to the hexadecimal value of the PEI-type required by the internal user application. The Bus Access Unit's operating system, which is a part of the Bus Access Unit's firmware, shall recognize and validate the EE_PEI_Type according to the following algorithm.

- At the PEI-type line the resistor value (hardware type) shall be measured cyclically. The cycle time is implementation specific.
- The measured value shall be transformed to a PEI-type value.
- The PEI-type value shall be compared with the value of the system variable EE_PEI_Type.

1. The internal user application in the Bus Access Unit shall only run if hard- and software PEI-type are equal.
2. The internal user application in the Bus Access Unit shall not run if hard- and software PEI-type are different.

   The software type is to be set to PEI-type 1, that is reserved for this purpose.

   If an application programmer wants to have serial communication over the PEI, the Bus Access Unit shall set its serial interface according to the detected hardware type 10, 12, 14 or 16 (see clause 6 “Serial PEI-communication”).

   If in this case hardware - and software type do however fit, there shall be an application program running in the Bus Access Unit (should at least exist of an RTS-instruction). Replies to external messages will be sent to this internal application. So, an external application will get no response to e.g. an A_FLAGS_READ- or an A_VALUE_WRITE-message. Neither the A_EVENT_INDICATION-messages shall be forwarded to the external application. In this case the internal user application has to handle and/or forward these messages.

3 PEI electrical signal specification

3.1 Wiring diagram

For the constructional specifications of the Physical External Interface, please refer to [02].

Figure 3 shows the wiring diagram of the Bus Access Unit's PEI connector in case of digital I/O. For more details see also the data sheet of the corresponding Bus Access Unit's microcontroller.

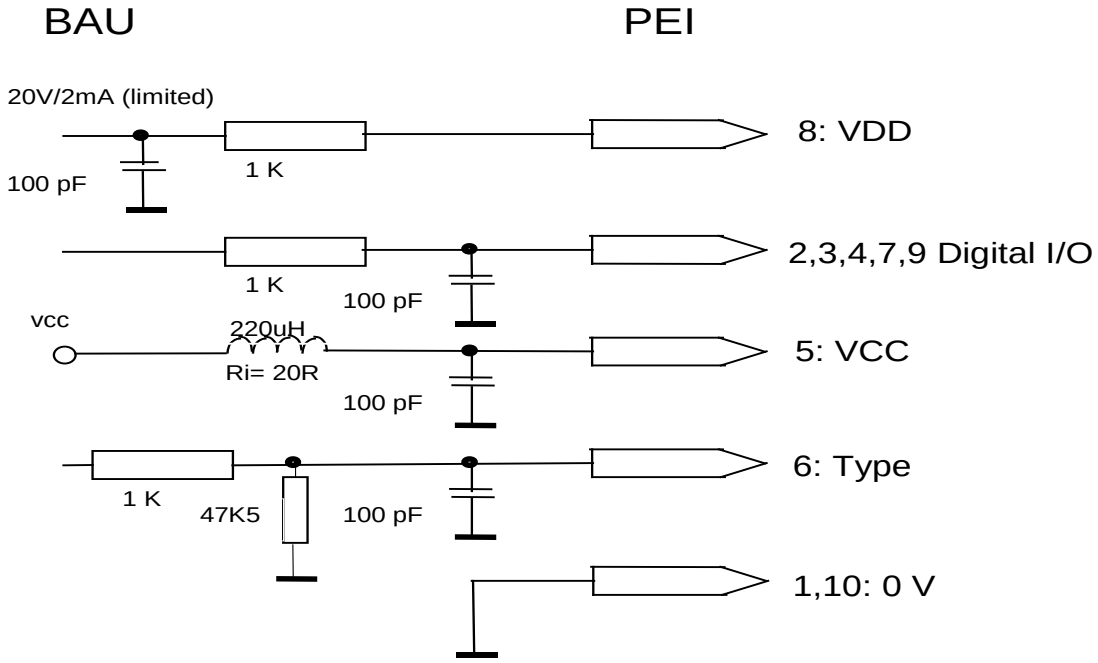


Figure 3 - Example of the wiring diagram of the PEI-connector

According to Figure 6 and Figure 7 the PEI shall have the following signal lines:

- 0 V (pin 1 and 10), +5 V DC (pin 5) and +24 V DC (pin 8).
- PEI-type (pin 6)
- I/O 1, 2, 3, 4 and 5 (pins 3, 2, 4, 7 and 9) or alternatively SCLK, RxD/RDI, TxD/TDO, CTS and RTS
- PLMB and C7 (pins 5a and 6a) in case of a 12-lines connector.

The following clauses define the electrical signal of each line.

3.1.1 PEI 0 V and power supply lines

At the Bus Access Unit's PEI-connector 0 V shall be at pin 1 and pin 10. The internal resistance of both lines shall be Ri=10 kΩ.

The output of the Bus Access Unit's power supply lines shall be:

+ pin 8: 24^{+6}_{-4} V DC; 2 mA max.

+ pin 5: (5 ±0,4) V DC; 10 mA max.

The maximum common load for the PEI power supply lines of 5 V DC and 24 V DC shall be 50 mW.

Example:

1. +5 V DC, max. 10 mA and +24 V DC not connected: ⇒ 50 mW

or

2. +5 V DC, 5 mA and (at the same time) +24 V DC, max. 1 mA ⇒ 49 mW

3.1.2 PEI-type line

The PEI-type line's wiring (pin 6) shall be in the way as specified in Figure 4.

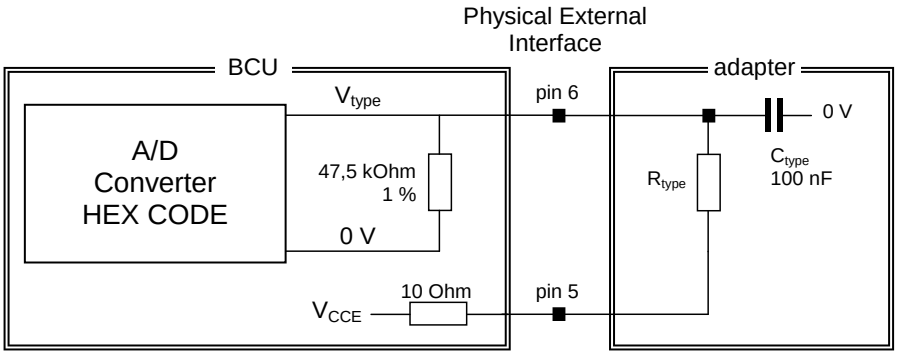


Figure 4 - PEI-type line's wiring diagram

The recommended resistance values for the different PEI-types are:

PEI-Type Nr.	Description of the communication type functional type	Resistor value [kΩ] + tolerance
0	No adapter	no R-type
1	"Illegal adapter": stop user application	910 5%
2	4 inputs, +1 output (LED)	430 5%
3	reserved	255 1%
4	2 inputs / 2 outputs, +1 output (LED)	187 1%
5	reserved	140 1%
6	3 inputs / 1 output, +1 output (LED)	107 1%
7	reserved	84,5 1%
8	reserved	66,5 1%
9	reserved	54,9 1%
10	Default: message protocol on top of FT 1.2 protocol Option: protocol on top of loadable serial (synchronous or asynchronous) protocol	45,3 1%
11	reserved	37,4 1%
12	Serial synchronous interface, message protocol	30,1 1%
13	reserved	24,3 1%
14	Serial synchronous interface, data block protocol	19,1 1%
15	reserved	14,7 1%
16	Serial asynchronous interface, message protocol	11,0 1%
17	Programmable I/O	7,50 1%
18	reserved	4,53 1%
19	4 outputs, +1 output (LED)	2,00 1%
20	Download	0

Figure 5 - Resistor values used for PEI-type encoding

3.1.3 Parallel I/O signal lines

At pins 3, 2, 4, 7, 9 and 6a the TTL I/O signals 1, 2, 3, 4, 5 and 6 shall be connected. At pin 5a the PWM2 output signal shall be connected. Unused output signals shall be not connected; unused input signal shall be connected to TTL high level.

3.1.4 Serial protocol signal lines

At pins 3, 2, 4, 9 and 7 the serial protocol signals SCLK, RxD/RDI, TxD/TDO, RTS and CTS shall be connected. CTS and RxD/RDI shall be inputs to the Bus Access Unit, TxD/TDO, RTS and SCLK shall be outputs. All signal lines except for the SCLK line shall be relevant for both synchronous - and asynchronous protocols; the output SCLK shall be for synchronous protocols only. For more details see the data sheet of the Bus Access Unit's microcontroller.

Unused output signals shall be not connected; unused input signals shall be connected to TTL high level.

4 PEI logical specification

4.1 Logical specification

The PEI standard interface between the Application Module and the Bus Access Unit is designed as a plug in unit.

In case of a Bus Access Unit mounted in a wall box ("flush-mounted") the Application Module's connector shall be male and shall have 10 pins. If the device uses the parallel PEI I/O interface then the logical meaning of the 10 pins shall be according to Figure 6, otherwise according to Figure 7.

NOTE In case of parallel PEI I/O communication the Application Modules are called PEI adapters.

In case of a device mounted at the (wall) surface ("surface-mounted device") the Application Module's connector shall also male but shall have 12 pins. If the device uses the parallel PEI I/O interface then the logical meaning of the 12 pins shall be according to Figure 6, otherwise according to Figure 7.

In case of a device mounted at the DIN rail ("DIN rail-mounted device") the Application Module's connector shall be 10 pin male if the PEI interface is at the side of the Bus Access Unit 10-pole female if the PEI interface is at the top of the Bus Access Unit. If the device uses the parallel PEI I/O interface then the logical meaning of the ten lines shall be according to Figure 6, otherwise according to Figure 7.

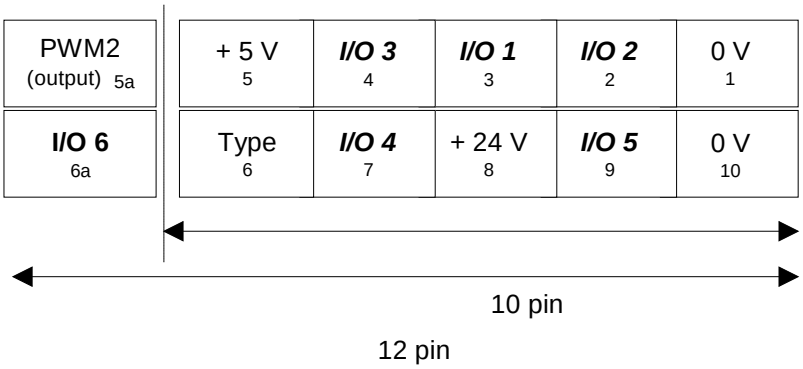


Figure 6 - Logical specification of the parallel PEI I/O lines

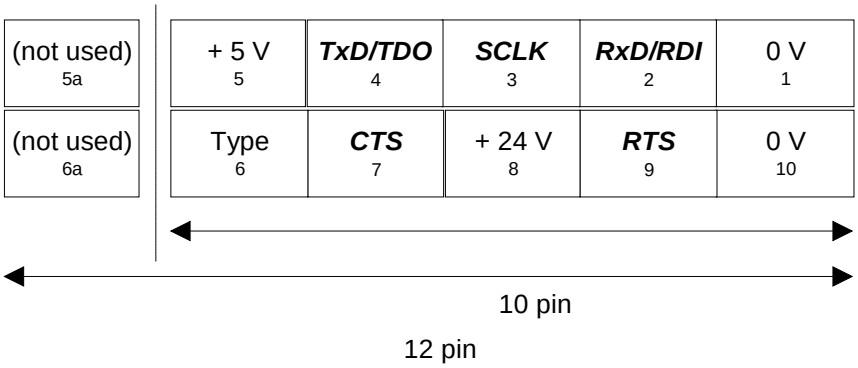


Figure 7 - Logical specification of the Serial PEI lines

4.2 Summary: PEI-type-dependent logical properties of the PEI connector lines

The electrical and logical properties of the PEI connector lines shall depend on the PEI-type given by the Application Module. For the various applications, different functional types of physical external interfaces shall be available. The PEI-lines for 24 V (pin 8), 5 V (pin 5), 0 V (pin 1 and pin 10) and PEI-type selection (pin 6) shall be the same for all PEI-types.

Figure 8 shows the logical specification of the other PEI lines, dependent on the PEI-type. The column headers give the logical names in relation to Figure 6 and Figure 7, distinguished in parallel and serial line usage.

		PEI pin nr						
		2	3	4	5a	6a	7	9
		PEI pin function						
		I/O 2	I/O 1	I/O 3	PWM2	I/O6	I/O 4	I/O 5
PEI-Type	**Functional description**	RxD	SCLK (syn)	TxD	none	none	CTS	RTS
0	No adaptor							
1	illegal adaptor							
2	4 inputs, 1 output (LED)	input	input	input	output	output	input	output
3	reserved							
4	2 inputs & 2 outputs + 1 output (LED)	output	output	input	output	output	input	output
5	reserved							
6	3 inputs & 1 output +1 output (LED)	input	output	input	output	output	input	output
7	reserved							
8	5 inputs	input	input	input	output	output	input	input
9	reserved							
10	default: FT1.2 protocol	ser. input: RxD	output	ser. output: TxD	output	output	input	output
10	loadable serial protocol	def. by user	def. by user	def. by user	def. by user	def. by user	def. by user	def. by user
11	reserved							
12	serial synchronous interface message protocol	ser. input RDI	output: SCLK	ser. output: TDO	output	output	CTS	RTS
13	reserved							
14	Serial synchronous interface data block protocol	ser.input: RDI	output: SCLK	ser. output: TDO	output	output	CTS	RTS
15	reserved							
16	Serial asynchronous interface, message protocol	ser.input: RxD	output	ser. output: TxD	output	output	CTS	RTS
17	programmable I/O	def. by user	def. by user	def. by user	output	output	def. by user	def. by user
18	reserved							
19	4 outputs, 1 output (LED)	output	output	output	output	output	output	output
20	Download	ser. input: RxD	output	ser. output: TxD	output	output	CTS	RTS

Figure 8 - PEI-type-dependent specification of PEI-lines

5 Parallel PEI I/O communication

5.1 PEI-types for parallel I/O communication

The PEI-types 0, 1, 2, 4, 6, 8, 17 and 19 define parallel I/O communication or are special purpose PEI-types with a parallel I/O background. The PEI-type table in clause 4.2 gives an overview which PEI-lines are relevant for each of the PEI-types.

5.2 Meaning of PEI-type 0

The BAU shall recognize PEI-type 0 if the PEI-type line is not connected, respectively if the adapter module is removed. As a consequence the BAU shall stop the internal user application, because the PEI-type expected, which shall be contained in the application, does not correspond to the PEI-type measured at the PEI-type line.

5.3 Meaning of PEI-type 1

The PEI-type 1 shall be reserved for applications that shall run without an application program.

5.4 Meaning of PEI-types 13 and 15

5.4.1 Use and conditions

PEI-types 13 and 15 shall be used in case of use of an implementation specific, non-standard PEI.

The marking of a non-standard PEI is mandatory on BAU and AM.

Both sides (AM and BAU) shall foresee a means to avoid damage on either side if AMs or BAUs are used with an implementation specific PEI in combination with BAUs or respectively AMs with standard PEI. In decreasing order of preference, the following is possible:

1. electrical protection

   NOTE It is very unlikely that the full 100 % damage prevention can be guaranteed.

2. Mechanical protection (optional)
3. other methods to prevent damage (software …)

   It is not allowed to foresee no protection at all.

The product description (for BAU and for AM) shall properly highlight the deviation from the standard PEI.

5.4.2 Protection requirements, measures and examples

5.4.2.1 Electrical protection

5.4.2.1.1 Need

In case there is no mechanical protection (or in case of insertion by force), some electrical protection is required to avoid damage of BAU and AM.

5.4.2.1.2 Standard PEI pinning

5	4 (In1)	3 (Out 1)	2 (Out 2)	1
5 V	**PortC.3 TxD/ AD5 I/O 3**	**PortC.4 CLK/ AD6 I/O 1**	**PortC.2 RxD/ AD7 I/O 2**	**Ground**
PEI-Type	**PortC.6 CTS/ AD3 I/O 4**	**+20V**	**PortC.5 RTS/ AD2 I/O 5**	**Ground**
6	7 (In 2)	8	9	10

5.4.2.1.3 Electrical requirements on PEI

pin	function	Requirement
pin 5	5 V	BAU: Shall not provide a higher voltage than specified on either pin (5 V DC, resp. 31 V DC). I.e. in the worst case there is no voltage drop towards the bus. Lower voltages are allowed. A current limitation shall be included on the BAU side according to Figure 3. No other pins shall be used for power feeding. No other pin than pin 8 may provide a voltage above 5 V DC. AM: Protection of the AM against voltages above 5 V DC (e.g. by foreseeing current limitation circuitry) is mandatory for pin 8 if no mechanical protection is provided. Otherwise, it is optional.
pin 8	24 V	
pin 1	GND	This pin shall always be used for GND.
pin 10	GND	This pin can be used for GND, but also for I/O and power feeding, both under the conditions of electrical protection as specified before in the document.
pin 6	Type	BAU, AM: This pin shall always be used for hardware PEI-type detection.
other	I/O	Next to this electrical protection, there shall be a **software protection** to change these pins to high impedance inputs in case an incompatible AM is detected.

5.4.2.1.4 Example of a manufacturer specific implementation

"AM Manufacturer specific" PEI pinning

Pin	Basic PIR AM	Enhanced PIR AM
1	GND	GND
2	IR sensor (Analog Output)	IR sensor (Analog Output)
3	Light sensor (Analog Output)	Light sensor (Analog Output)
4	Light setting (Analog Output)	Light setting (Analog Output)
5	3,3 V	3,3 V
6	PEI-type 13 (manufacturer specific)	PEI-type 15 (manufacturer specific)
7	Time setting (Analog Output)	Time setting (Analog Output)
8	/	/
9	/	Led (Binary Input)
10	GND	GND

Example of possible combination "BAU Manufacturer specific" PEI pinning

Pin	BAU
1	GND
2	I /O (binary, Analog)
3	I /O (binary, Analog)
4	I /O (binary, Analog)
5	3,3 V
6	I /O PEI-Type / P6.4
7	I /O (binary, Analog)
8	20 V
9	I /O (binary, Analog)
10	GND

"Non standard" AM on a standard BAU:

- Normally when plugging AM with PEI-types 13 or 15, the BAU will not work, because these PEI-types are reserved.
- The Vcc at 5 V is not a problem for the AMs (there are integrated circuits in the AMs that can all be supplied at 5 V). Electrical protection is necessary against over voltage (Zener for example).

Wrong AM on the "Non standard" BAU:

- In all cases, the Vcc is 3,3 V: if the AMs are normally supplied with 3,3 V or more (standard 5 V), there will be no risk of hardware damage, because the power voltage is lower. Risk of malfunction due to improper microcontroller start.
- AM with unknown PEI-type (other than 13, 15, 16 or 17) it is recommended that all port of the BAU to be in input state.
- AM with PEI-Type 13 or 15 : in this case, the current is limited by a serial resistor.
    - There are series resistors on every signal in the BAU and in the AM, so no risk of damage.

5.4.2.2 Mechanical protection

An optional mechanical protection can be integrated in the BAU to avoid the mounting of standard AM.

An optional mechanical protection can be integrated in the AM avoid the mounting of standard BAU.

Figure 9 shows an example of mechanical protection to avoid standard AM to be plugged in a "manufacturer specific BAU" . In this case, it is not possible (or very hard) to plug a standard AM.

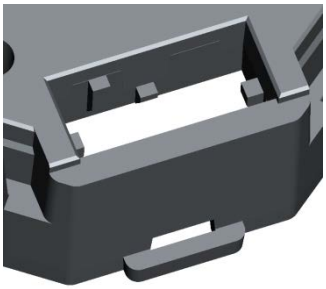


Figure 9 – Example of a mechanical protection

5.5 PEI-types 2, 4, 6, 8, 17 and 19

PEI-types 2, 4, 6 and 8 are for parallel PEI I/O communication. See the PEI-type table in clause 4.2 and the other explanations in [02].

5.5.1 Programmable I/O Configuration with PEI-type 17

This system variable EE_PortCDDR17 shall contain the direction bit register contents for port C. The operating system shall enter the value of EE_PortCDDR17 in the port C data direction register of the bus access module processor only under the following conditions.

1. The bus access module recognizes a PEI-type 17 at the PEI-type line, and
2. the system variable EE_PEI_Type is also 17 and
3. the EEPROM error flag of system variable EE_RunError (bit 2) is 1 (i.e. no EEPROM error).

For the specific location of the system variable EE_PortCDDR17, please refer to [03].

Port C

bit #	**7** **I/O 6**	**6** **I/O 4**	**5** **I/O 5**	**4** **I/O 1**	**3** **I/O 3**	**2** **I/O 2**	**1** **-**	**0** **-**
meaning	I/0-flags 0 = input 1 = output							

6 Serial PEI-communication

6.1 Overview

The PEI-types 10, 12, 14, 16 and 20 shall define serial PEI-communication. See the PEI-type table in 4.2 for information about the meaning of the PEI-pins with respect to the serial PEI protocol.

At PEI-type 14 there shall run a synchronous protocol that shall serve to transfer data blocks of information from the internal user application to the external user application and vice versa. The data block format shall be as specified in 6.2; the synchronous protocol shall be as specified in 6.3.1.2.

In distinction to the serial PEI-type 14 protocol the PEI-type 10, 12, 16 and 20 protocols shall serve to transfer messages from a specific communication instance to the external user application and vice versa. The messages shall depend on the communication instance that communicates to the external user application. See [01] for a description of the data format and the contents of the exchanged messages.

At PEI-type 12 there shall run the same synchronous protocol as for PEI-type 14; therefore see also 6.3.1.2 for a detailed communication protocol specification.

At PEI-types 16 and - 20 there shall run an asynchronous protocol that is specified in the following.

At PEI-type 10 there shall run either the FT1.2 protocol as the default (asynchronous) protocol, or a synchronous or asynchronous protocol of which the BAU protocol counterpart can be manufacturer-defined. During device programming time the BAU protocol counterpart shall be downloaded to the BAU to use it as the loadable PEI-type 10 protocol.

6.2 Synchronous PEI-type 14 communication

6.2.1 Data format

The data block format shall comply with the specification of Figure 10. A data block of length n shall be transmitted transparent to the user.

Figure 10 - Data Block Format for synchronous PEI-type 14 communication

6.2.1.1 Block data

The contents of the block data octets shall be totally left to the application programmer.

6.2.2 Protocol specification

The protocol shall be as specified in 6.3.2, concerning communication request and data exchange, but there shall be no exchange of the length octet.

6.3 Synchronous PEI-type 12 and asynchronous PEI-type 16 communication

6.3.1 Data format

The following figure shows the message format. A message of length n shall be composed of 1 octet for the length and n-1 octets for the message data.

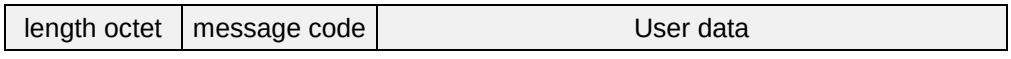


Figure 11 - Message format for synchronous PEI-type 12 communication

6.3.1.1 Length octet

The length octet shall contain the number of subsequent block data octets. Figure 12 shows the encoding of the length octet. The most significant bit shall be the even parity bit for the whole octet. The second-most and third-most significant bits shall build the length octet tag, which shall always be the bit sequence '01'. The other five bits shall allow encoding the number of octets following the length octet, i.e. the allowed value shall range for the number of following octets is 0 to 31.

P	0	1	X	X	X	X	X

Figure 12 - Encoding of the length octet

6.3.1.2 Message Code and Userdata

For definition of message codes and Userdata see [01].

6.3.2 Protocol Description

Both the synchronous - and the asynchronous serial PEI protocol for message transmission (i.e. the protocols at PEI-types 12, 16 and 20) shall serve to transfer messages between the external user application and the BAUs communication stack. The message exchange shall consist of 4 phases.

1. communication request (hardware handshake)
2. transfer of the length octet (software handshake)
3. data exchange
4. pause

6.3.2.1 Hardware handshake - communication request

The handling shall be the same, whatever microcontroller wants to communicate. A hardware handshake shall take place on each octet transfer. It shall be a protocol of request/answer on the lines RTS (request to send) and CTS (clear to send). The communication initiator shall reset its RTS line and shall poll its CTS line, see Figure 15. If CTS = 0 then the handshake is okay, what shall be interpreted as a positive communication request.

6.3.2.2 Software handshake – transfer of the length octet

The first data exchange shall be a bidirectional transmission of the length octet. If a controller has nothing to send, it shall put FFh as the length octet; otherwise it shall put the length of the data block it has to transmit. In case of simultaneous requests of the external user application and the BAU, the BAU controller shall be considered as the master. The external protocol instance has to request a new data transfer after the complete reception of the message of the BAU controller.

This software handshake shall take place on the first octet exchange. This means, that after this the communication direction is defined until the complete message is transferred from one microcontroller to the other one.

6.3.2.3 Data exchange

After the communication relationship is established, the communication initiator shall send the data octets. The other protocol instance shall responds in parallel by octets of value 00h.

6.3.2.4 Pause

After a complete message transfer a new transfer shall wait 3 ms.

6.3.2.5 Error handling

In case of errors, protocol errors or time-outs, the BAU shall reset the serial port. Then the BAU shall set RTS high and shall poll CTS until CTS is also high. If high the BAU controller shall wait 10 ms before it requests a new data transfer. Otherwise it shall consider the request of the application controller. The time-out used for a data block transfer shall be about 130 ms.

6.3.2.6 Initialisation

After a hardware or software reset the BAU shall tries to send an LM_Reset.ind-message, but only once. If a communication error occurs, the transmission shall not be repeated.

The LM_Reset.ind message is a single octet A0h. This is a length octet with length 0.

P	0	1	L	L	L	L	L	General format of a length octet
1	0	1	0	0	0	0	0	LM_Reset.ind message

6.3.2.7 Protocol and handshake specification

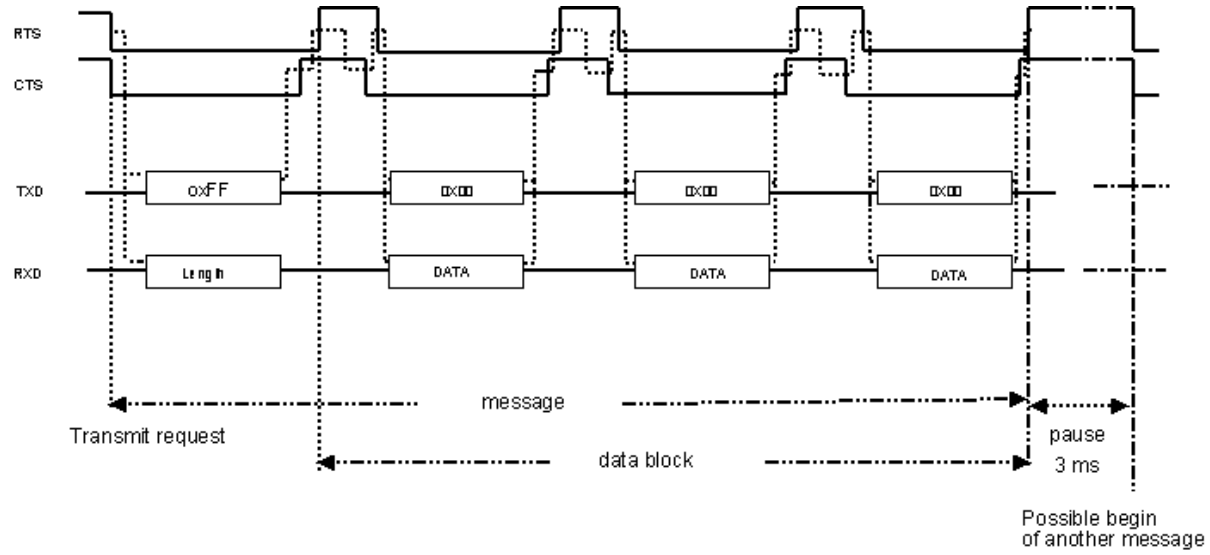


Figure 13 - Protocol if the BAU controller is receiver

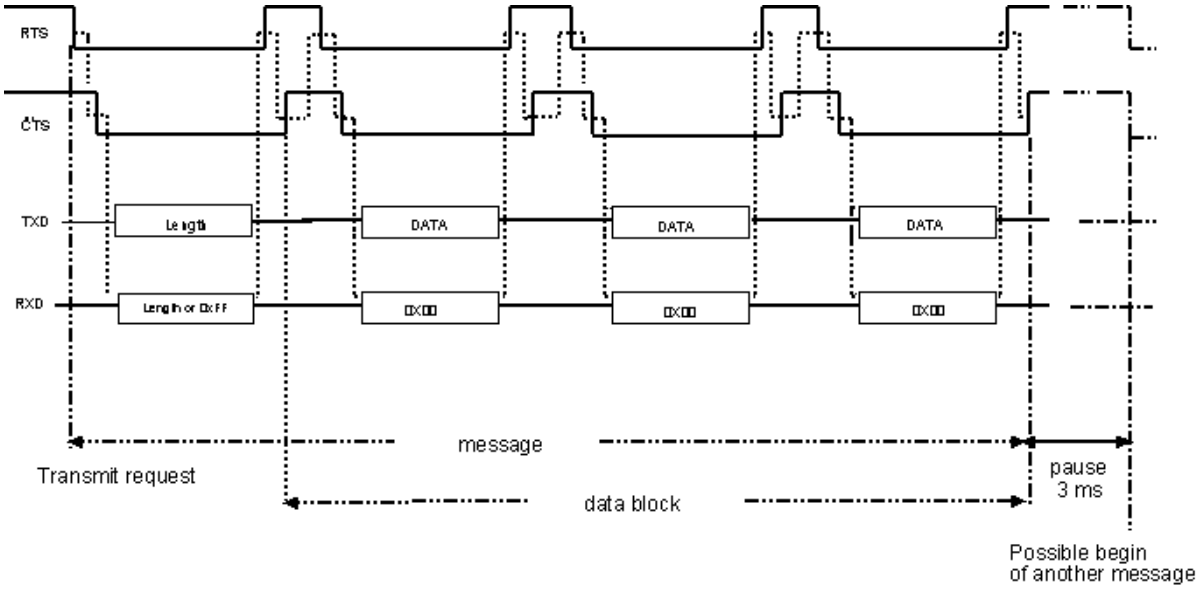


Figure 14 - Protocol if the BAU controller is transmitter

6.3.3 Definition of the synchronous signal at PEI-types 12 and 14

6.3.3.1 Signals and data formats for synchronous PEI-communication

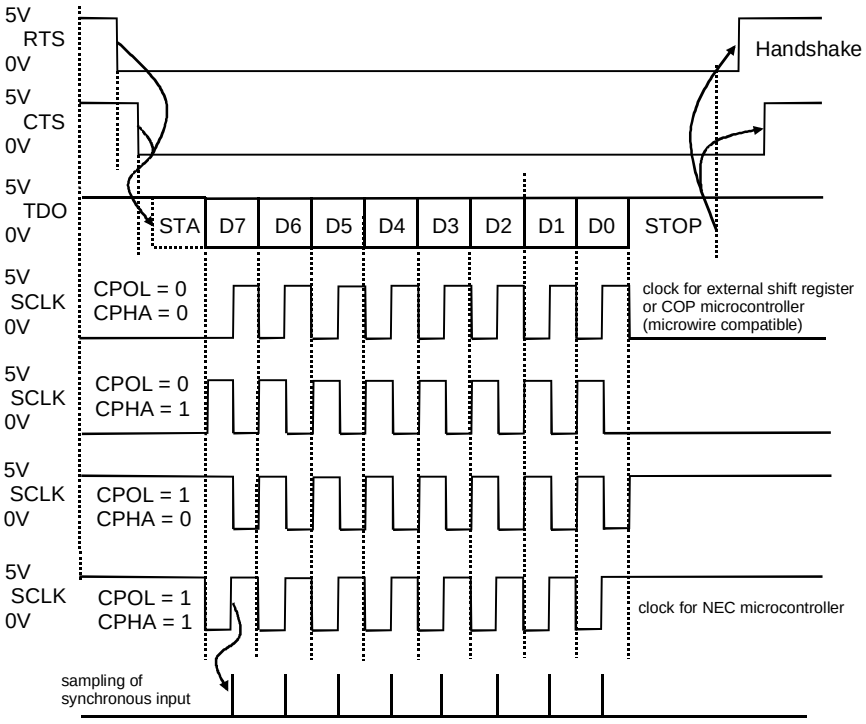


Figure 15 - Signals and data formats during synchronous PEI-communication

For the specific location of the system variables CPOL and CPHA and how to set the baudrate, please refer to [03].

6.3.3.2 The synchronous protocol

6.3.3.2.1 Features

- No gaps.
- Data transmission in both directions in parallel.
- Relation to the hardware handshake: to be explained.
- The hardware handshake protocol at the PEI-types shall be the same as the hardware handshake protocol of the asynchronous protocol. See there for more information.

6.3.3.2.2 PEI recognition and default settings

The serial port shall configure TDO as serial output; SCLK shall be configured as serial clock output. The clock phase and data format shall be configured according to CPHA and CPOL configuration in EEPROM. The baudrate to be used shall be located in the non-volatile memory at label "SyncRate".

6.3.4 Definition of the asynchronous signal at PEI-type 16

6.3.4.1 Signals and data formats for asynchronous PEI-communication

9600 bps; 8 data bits, no parity bit, one stop bit.

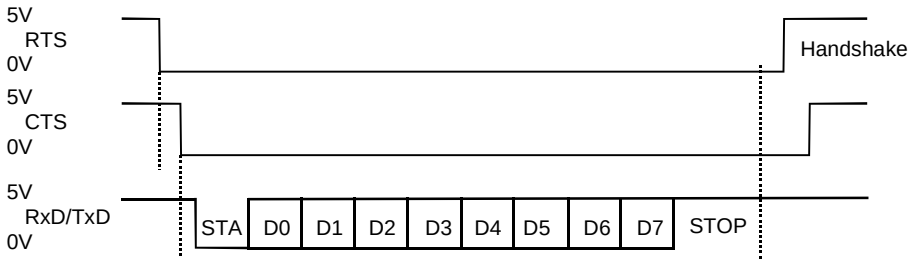


Figure 16 - Signals and data formats for asynchronous PEI-communication

6.3.5 Data transmission through the PEI

6.3.5.1 Octet transmission BAU is sender

Time out: BAU switches from RTS-False to RTS-True

The BAU shall detect a time out after approximately 130 ms. The time-out shall cover the entire data exchange (see Figure 14, time message). As can be gathered from the figure, the CTS-edge shall not be limited in time, as long as the entire data exchange is carried out during this time-out.

HOST switches from CTS-True to CTS-False

The succession of pulses is laid down in Figure 14. Except for the time out the maximum time shall not be limited. The RTS shall only switch to False if both the BAU octet and the BAU Acknowledge octet have been transmitted. The sequence of both octets shall be of no importance.

6.3.5.2 Octet transmission Host is Sender

HOST switches from CTS-False to CTS-True

The only limitation as regards time shall be the time-out. After the last exchange of data both handshake signals shall remain deactivated for at least 3 ms (see Figure 13). If the HOST is transmitting, only after termination of the stop-bit-time it is allowed to switch to CTS. As long as no transmission is carried out, there shall be no limitation as regards time.

BAU switches from RTS-True to RTS-False

When at the latest does the BAU change level?

In-between a message the BAU shall react within 0 ms to 3 ms.

6.3.5.3 Block transmission

Time between two octets

After the host has transmitted one octet, the BAU shall only switch his CTS-line to if when the RX-register is full, the stop bits shall have been detected and the transmitted octet shall have been entirely sent.

Delays

The 3 ms delay between two block transmissions shall always be foreseen. Other additional delays depend from one application to another. In case of EEPROM-programming one has to take into account a delay 20 ms per octet, this shall however only be relevant if the value of a distinct octet really changes. The number of octets that can be transmitted by means of a block is limited by the internal buffer of the BAU.

6.4 The default Protocol at PEI-type 10: FT1.2

6.4.1 Introduction

In order to have a reliable data transmission, a transmission protocol based on the international standard IEC 870-5-1 and 870-5-2 (DIN 19244) is defined for the BAU.

The balanced transmission procedure shall be used: that means each station may act simultaneously as primary station (initiating a message transfer) and secondary station (receiving a message). It shall be restricted to point-to-point in the BAU (no address field) and both stations shall have equal access rights; this is, there shall be no master/slave relation assigned to the station (Master/Master).

Only the transmission frame format FT1.2 is supported.

6.4.2 Physical Interface

The BAU and a station shall be connected via a 3-wire connection.

RxD: Received data

TxD: Transmit data

0 V: Signal 0 V

The data transmission shall be performed with 8 data bits and 1 stop bit with even parity (line idle shall be "1").

The transmission rate shall be selectable between the following values: 1 200, 2 400, 4 800, 9 600 and 19 200 baud. (The default value shall be 19 200.)

6.4.3 Transmission Frame Format

6.4.3.1 Overview

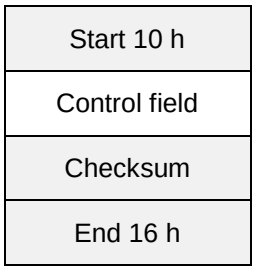
The FT 1.2 protocol shall include

1. frames with fixed length, and
2. frames with variable length and
3. single character frames

6.4.3.2 Frame with fixed length

6.4.3.2.1 Frame format

Frames with fixed length shall consist of a start character, one Control field, a frame checksum character and an end character.



6.4.3.2.2 Transmission rules

R1 Line idle shall be binary 1.

R2 Each character shall have one start bit (binary 0), 8 information bits, even parity and one stop bit (binary 1).

R3 Only restricted line idle intervals (see LINE_IDLE_TIMEOUT in clause 6.4.8 “Parameter descriptions” below) shall be admitted between characters of a frame.

R4 Upon detecting an error according to rule R6, a minimum interval (see LINE_IDLE_TIMEOUT in clause 6.4.8 “Parameter descriptions” below) shall be required between frames.

R5 The sequence of user data characters shall be terminated by an 8 bit checksum. The checksum shall be the arithmetic sum disregarding overflows (sum modulo 256) over all user data octets.

In frames with fixed length the checksum shall be equal to the Control field.

R6 The receiver shall check:

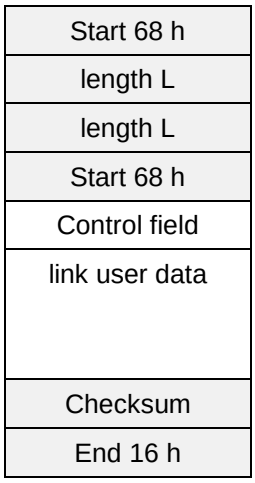
- per character:
  - the start bit, the even parity and the stop bit;
- per frame:
  - the specified start character, and
  - the frame checksum, and
  - the end character, and
  - upon detecting an error, the line idle interval shall be as specified by R4.

The frame shall be rejected if one of these checks fails; otherwise it shall be released to the user.

6.4.3.3 Frame with variable length

6.4.3.3.1 Frame format

Frames with user data shall consist of a start character, two equal characters that shall specify the number L of user data octets, a second start character, the user data, a frame checksum character and an end character.



Length shall specify the number of user data octets including the control field (range from 2 to 24).

6.4.3.3.2 Transmission rules

R1, R2, R3, R4, and R5: see transmission rules for frames with fixed length.

R6 The receiver shall check:

- per character:
  - the start bit, the even parity and the stop bit;
- per frame:
  - the specified start character at the beginning and at the end of the frame header
  - the identity of the two length specifications L
  - that the number of received characters is equal to L + 6
  - the frame checksum
  - the end character
  - upon detecting an error, the line idle interval specified by R4

The frame shall be rejected if one of these checks fails; otherwise it shall be released to the user.

6.4.3.4 Frame format of single character

One single character is specified:

The single character shall be used for a positive acknowledgment.

6.4.3.4.1 Transmission rules

R1, R2, R3, R4 see transmission rules for frames with fixed length.

R5 - The receiver shall check:

- per character:
  - the start bit, the even parity and the stop bit;
- per frame:
  - upon detecting an error, the line idle interval shall be as specified by R4.

The frame shall be rejected if one of these checks fails; otherwise it shall be released to the user.

6.4.4 Control Field

6.4.4.1 Contents and goals

The Control Field shall contain information that shall characterize the direction of the message, the type of the service provided and shall support control functions for suppressing losses or duplications of messages.

6.4.4.2 Control Field from primary station

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DIR	PRM = 1	FCB	FCV	Function code			

DIR: physical transmission direction 1 = BAU to external device

0 = external device to BAU

PRM: primary message 1 = message from primary (initiating) station, i.e. request

FCB: frame count bit: 0, 1 = alternating bit for successive SEND/CONFIRM

The frame count bit shall be used for suppressing losses and duplications of information transfers. The primary station shall alternate the FCB bit for each new SEND/CONFIRM transmission service. If an expected reply is timed out (missing) or garbled, then the same SEND/CONFIRM service shall be repeated with the same frame count bit.

FCV: frame count bit valid: the alternating function of bit FCB is valid

This bit shall always be set when communicating with the Service SEND_UDAT.

SEND/NO REPLY services and other transmission services shall not be used.

Function code

Function code	Service name	Service type	Service function	FCV	Frame format	format positive Confirm / response	format negative Confirm
00h	SEND_RESET	SEND/ CONFIRM expected	Reset of remote link	0	fixed length	single octet ACK	nothing / fixed length NACK
03h	SEND_UDAT	SEND/ CONFIRM expected	User data	1	variable length	single octet ACK	nothing / fixed length NACK
09h	REQ_STATUS	REQUEST/ RESPOND expected	Request status of link	0	fixed length	fixed length status respond	nothing / fixed length NACK
other	-	(not used)	Reserved	-	-		

6.4.4.3 Control Field from secondary station

Used only as the response to the REQ_STATUS Service.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DIR	PRM=0	RES	DFC	Function code			

DIR: physical transmission direction 1 = BAU to external station

0 = external station to BAU

PRM: primary message 0 = message from secondary (responding) station

RES: reserved (always 0)

DFC: data flow control: further messages may cause data overflow
(not used)

The secondary station may indicate with DFC bit = 1 to the primary station that further messages may cause a buffer overflow.

Usage by the BAU:

The BAU as primary station sends its frame regardless of the value of DFC in the last received CONFIRM frame from the secondary station. As secondary station the BAU does not set the DFC bit.

Function code:

Function code	Service name	Service type	Service function	Frame format
00h	CONFIRM_ACK	CONFIRM (not used)	positive acknowledgment	fixed length
01h	CONFIRM_NACK	CONFIRM	message not accepted (overload) not send from the BAU	fixed length
0Bh	RESPOND_STATUS	RESPOND	Status of link	fixed length
other	-	(not used)	Reserved	-

6.4.5 Transmission procedures

6.4.5.1 General requirements

Simultaneous data transmission in both directions (BAU ←→external station) shall be supported. However the primary station shall accept a new message transfer only if a previous message transfer is terminated either successfully or with an error indication (time-out).

Transmission errors shall only be detected by the receiving station. A secondary station receiving a disturbed SEND or REQUEST frame shall not reply. This shall be detected by the primary station timing out, because the expected CONFIRM or RESPOND frame is not received. A primary station receiving a disturbed CONFIRM or RESPOND frame shall detect the error and transmits the REQUEST frame once again.

6.4.5.2 Send/Confirm service

The transmission procedure for this service shall begin at first when the transmission procedure of a previous service is terminated.

When the request frame is received correctly by the secondary station, a positive CONFIRM (single character ACK) shall be transmitted to the primary station.

If the second station is unable to accept the message, e.g. due to an overload situation (unavailable buffer memory), a negative CONFIRM frame (NACK, message not accepted) or nothing (the same meaning) shall be sent (no NACK will be sent by the BAU).

If the primary station does not receive the CONFIRM frame before the exchange time-out (EXCHANGE_TIMEOUT), the message shall be repeated (see following clause).

The service shall be terminated if a CONFIRM frame is received or after the maximum number of repetitions.

6.4.5.3 Protection against message loss or message duplication

In the primary station the frame count bit (FCB) shall be alternated with each new SEND/CONFIRM service. If the CONFIRM frame is disturbed or timed out then the SEND frame shall be repeated with unchanged bit FCB. The maximum number of repetitions shall be defined in the parameter REPEAT_LIMIT (see table in clause 6.4.8).

6.4.5.4 Request/respond service

The transmission procedure for this service shall begin at first when the transmission procedure of a previous service is terminated.

On receiving a "Status of link" REQUEST frame, the secondary station shall send a RESPOND frame with the requested status of link.

6.4.6 Protocol Initialisation

After a reset, the station shall send a frame "Reset of remote link". On receiving the "Reset of remote link" frame, which shall have a FCB equal to zero, the secondary station shall delete messages in its buffer and shall be set to expect the next frame primary to secondary with FCV = valid (FCV=1) to have the opposite setting of FCB, i.e. FCB equal to one. If the BAU sends a RESET the BAU shall expect the next frame with FCV = valid (FCV=1) to have the FCB equal to one.

6.4.7 Examples of data frame transmission

6.4.7.1 Undisturbed send/confirm

The transmission of SEND/CONFIRM data frames may be initiated independently from both stations. However, the receipt of the associated CONFIRM frame is the condition for initiating a new transmission.

Station A	Service		Service	Station B
Send user data frame	SEND_UDAT	→	← SEND_UDAT	Send user data frame
Positive acknowledgment single character	ACK	→	← ACK	Positive acknowledgment single character
Send user data frame	SEND_UDAT	→		
			← ACK	Positive acknowledgment single character
			← SEND_UDAT	Send user data frame
Positive acknowledgment single character	ACK	→		

6.4.7.2 Disturbed Send/Confirm

If the SEND/CONFIRM data frame is disturbed and thus no CONFIRM frame is received within the time out, the SEND/CONFIRM expected data frame is repeated with the unaltered frame count bit.

Station A	Service	Service	Station B
Send user data frame	SEND_UDAT →	← SEND_UDAT	send user data frame e.g. with frame count bit = 1; SEND_UDAT is disturbed .
Error detected in SEND_UDAT of station B: wait a minimum interval (see line idle time-out) before taking in account the next data frame		← ACK	Positive acknowledgment single character
Send user data frame	SEND_UDAT →		
		← ACK	Positive acknowledgment single character
		← SEND_UDAT	→ Exchange time out of disturbed SEND_UDAT Repeated send user data frame with frame count bit = 1
Positive acknowledgment single character	ACK →		

NOTE A line idle time-out is always interpreted as end-of-frame.

6.4.7.3 Disturbed Send/Confirm and ignored Confirm

If the SEND/CONFIRM data frame is disturbed, the receiving station waits until it has detected a specified interval of line idle before accepting another frame. If a CONFIRM frame occurs during this interval, it will be ignored by the receiving station. In this case, the SEND/CONFIRM expected data frames of each station are repeated with the unaltered frame count bit.

Station A	Service	Service	Station B
Send user data frame e.g. with frame count bit = 0	SEND_UDAT	← SEND_UDAT	send user data frame e.g. with frame count bit = 1; SEND_UDAT is disturbed ...
Error detected in SEND_ UDAT of station B: wait a minimum interval (see line idle time-out) before taking in account the next data frame ⇒ ACK ignored		← ACK	Positive acknowledgment single character
Exchange time-out: Repeated send user data frame with frame count bit = 0	SEND_UDAT	← SEND_UDAT	... exchange time-out: Repeated send user data frame with frame count bit = 1
Positive acknowledgment single character	ACK	← ACK	Positive acknowledgment single character discharge the received Data

6.4.7.4 Disturbed Confirm

If the CONFIRM data frame is disturbed, the SEND/CONFIRM data frame is repeated with the unaltered frame count bit after the time out.

Station A	Service	Service	Station B
Send user data frame e.g. with frame count bit = 0	SEND_UDAT	← SEND_UDAT	Send user data frame e.g. with frame count bit = 1
Disturbed or missing ACK	ACK	← ACK	Positive acknowledgment
Send user data frame with frame count bit = 1	SEND_UDAT		
		← ACK	Positive acknowledgment
discharge the received data		← SEND_UDAT	Exchange time-out: Repeated send user data frame with frame count bit = 1
Positive acknowledgment	ACK		

6.4.8 Parameter descriptions

Name	Function	Default value
EXCHANGE_TIMEOUT	Time-out for end of exchange in case of SEND/CONFIRM or REQUEST/RESPOND	ca. 510 bits
REPEAT_LIMIT	Repeat limit the retransmissions due to transmission errors	3
LINE_IDLE_TIMEOUT	maximum time between two characters, minimum line idle time before an error is detected	ca.33 bits