EIBA Handbook Series

Release 3.0

Volume 3:
System Specifications

Part 6: Application Interfaces

Chapter 2: Physical External Interface

25.03.1999

Table of Contents

1. Overview

The Physical External Interface (PEI) is an interface situated in an EIB device between the bus access unit and the application module. The bus access unit contains all the EIB 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 EIB 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 EIB Tool Software, ETS, or an EIB bus monitor software.

The PEI is the EIB 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 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 20:	allows a manufacturer to download initial settings to the BCU. The behavior is implementation specific. Refer to the Chapter 3/4/4 for closer descriptions.

2. PEI types 3, 5, 7, 9, 11, 13, 15, 18:

   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: 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 analog values too. The application module's binary values are to be read and written by the internal EIB user application.

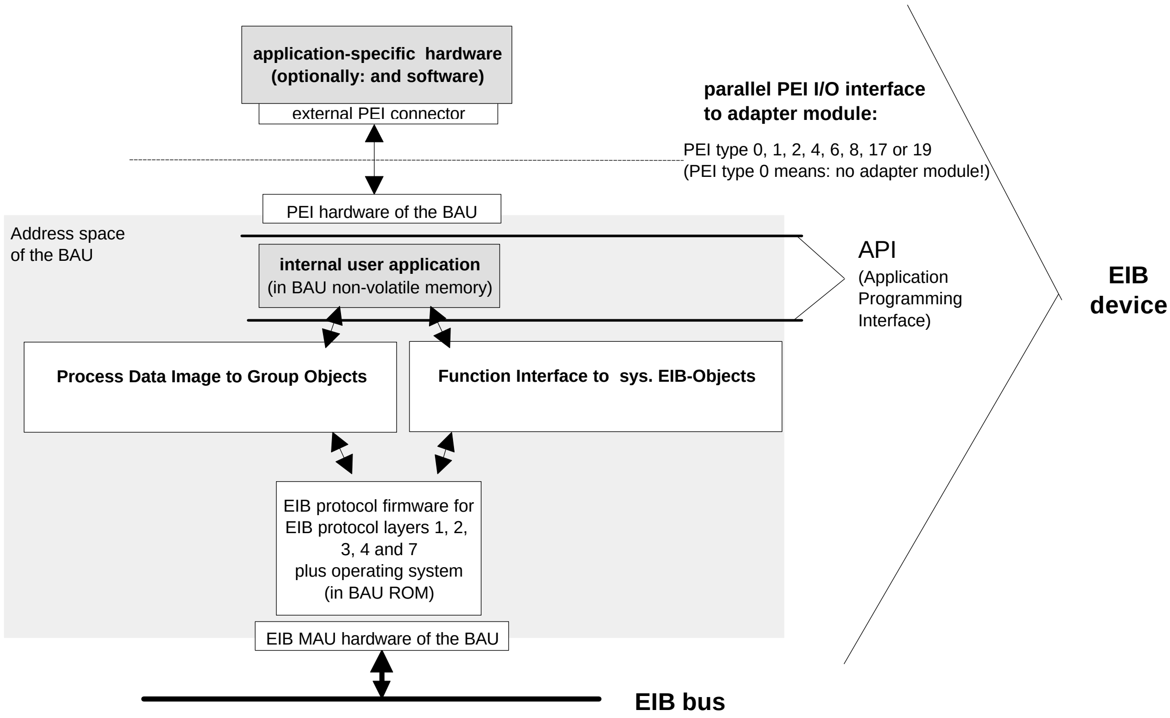

Fig. 3/6/2-1: A typical one-processor EIB Device with Parallel PEI I/O Communication

4. PEI types 10, 12, 14 and 16: 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 Chapter 3/6/3 "External Message Interface".

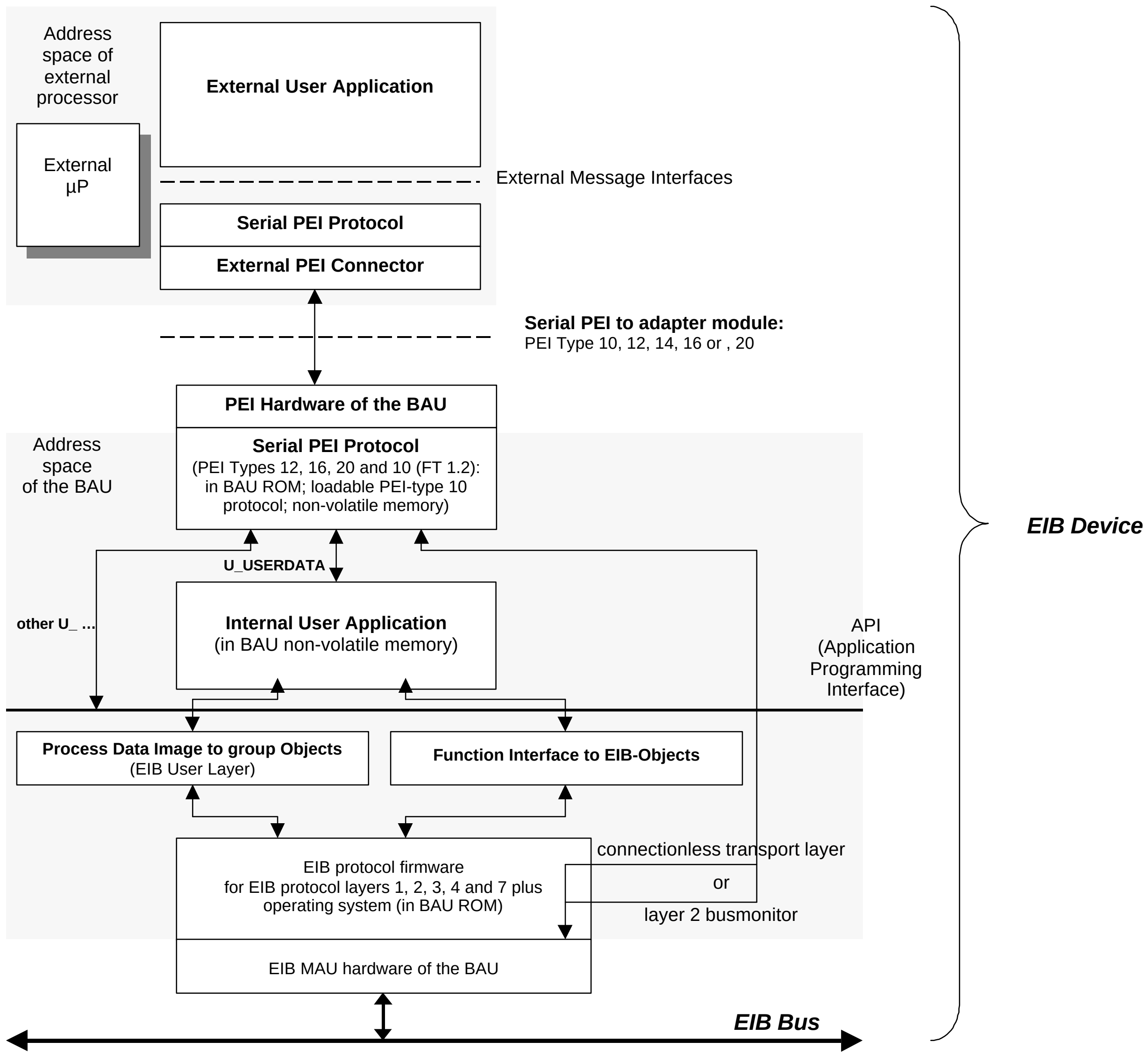

Fig. 3/6/2-2: A typical EIB Device communicating via Serial PEI, PEI Type 14

Each of the PEI types 10, 12, 14 and 16 stands for a different EIB 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.

EIB 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 EIB 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 is 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 is a system variable of the Bus Access Unit. It must be set by the network management client during download of the internal user application into the Bus Access Unit. EE_PEI_Type must be set to the hexadecimal value of the PEI type required by the internal user application. The bus access module's operating system, which is a part of the bus access module'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) has to be measured cyclically. The cycle time is implementation specific.
- The measured value must be transformed to a PEI type value.
- The PEI type value must 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 will set its serial interface according to the detected hardware type 10, 12, 14 or 16 (see paragraph 6).

   If in this case hardware and software type do however fit, there will 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 will 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

For the constructional specifications of the Physical External Interface, please refer to Part 9/1 "Cables and Connectors".

Fig. 3/6/2-3 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.

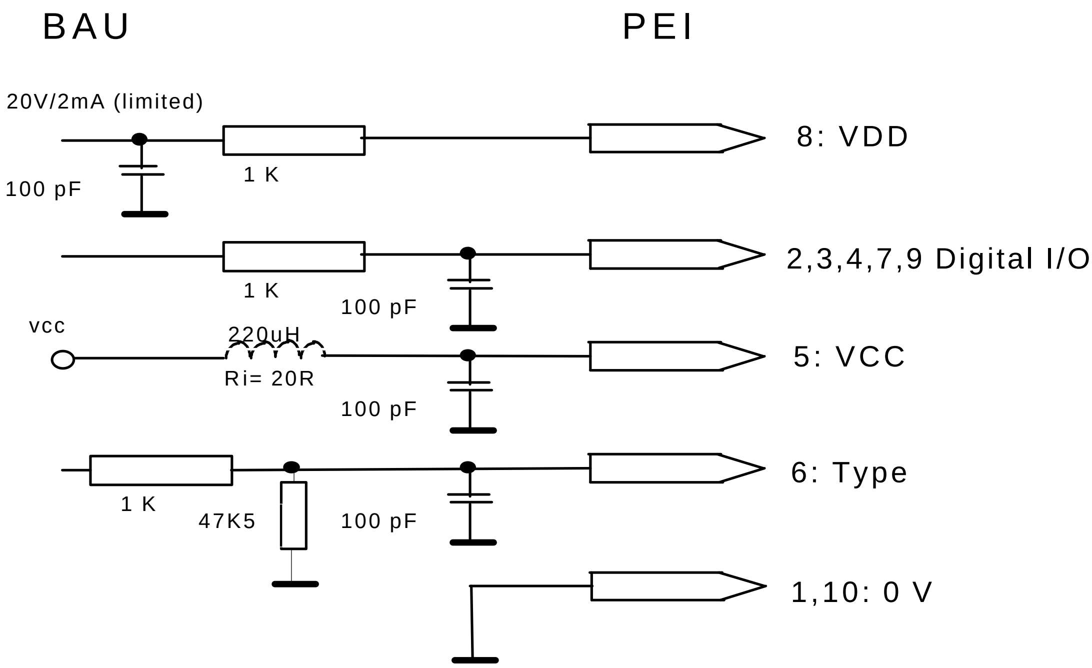

Fig. 3/6/2-3: Example of the Wiring Diagram of the PEI-Connector

According to Fig. 3/6/2-6 and Fig. 3/6/2-7 the PEI has 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 sub-paragraphs 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 is at pins 1 and 10. The internal resistance of both lines is Ri=10 kΩ.

The output of the bus access unit's power supply lines must be:

+ pin 8:	+24 -4 / +6	[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 is 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) must be in the following way:

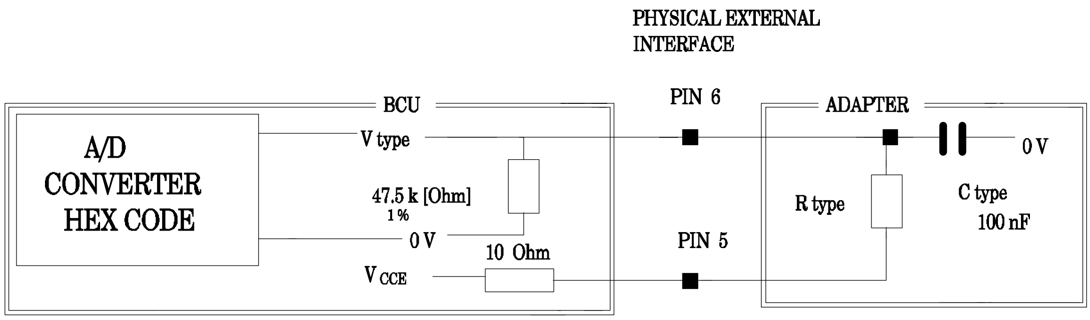

Fig. 3/6/2-4: PEI Type Line's Wiring Diagram

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	

Fig. 3/6/2-5: 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. Non-used output signals shall be not connected; non-used 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 are inputs to the bus access unit, TxD/TDO, RTS and SCLK are outputs. All signal lines except for the SCLK line are relevant for both synchronous and asynchronous protocols; the output SCLK is for synchronous protocols only. For more details see the data sheet of the bus access unit's microcontroller.

Non-used output signals shall be not connected; non-used input signals shall be connected to TTL high level.

4. PEI Logical Specification

The PEI standard interface between the EIB application module and the EIB 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 is male and has 10 pins. If the EIB device uses the parallel PEI I/O interface then the logical meaning of the 10 pins is according to Fig. 3/6/2-6, otherwise according to Fig. 3/6/2-7.

Note: In case of parallel PEI I/O communication the application modules are called PEI adapters.

In case of an EIB device mounted at the (wall) surface ("surface-mounted EIB device") the application module's connector is also male but has 12 pins. If the EIB device uses the parallel PEI I/O interface then the logical meaning of the 12 pins is according to Fig. 3/6/2-6, otherwise according to Fig. 3/6/2-7.

In case of an EIB device mounted at the DIN rail ("DIN rail-mounted EIB device") the application module's connector is 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 EIB device uses the parallel PEI I/O interface then the logical meaning of the ten lines is according to Fig. 3/6/2-6, otherwise according to Fig. 3/6/2-7.

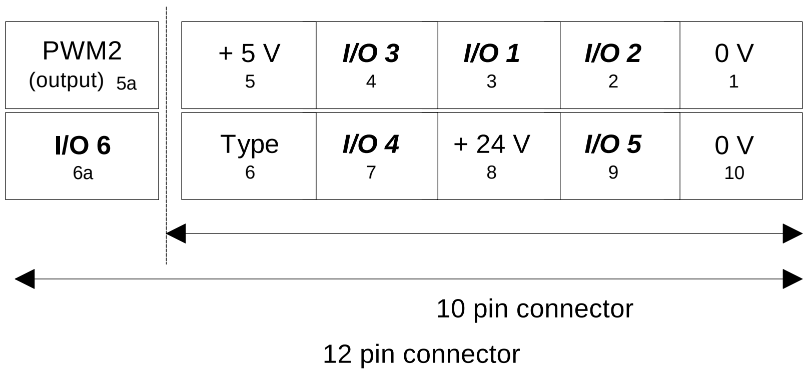

Fig. 3/6/2-6: Logical Specification of the Parallel PEI I/O Lines

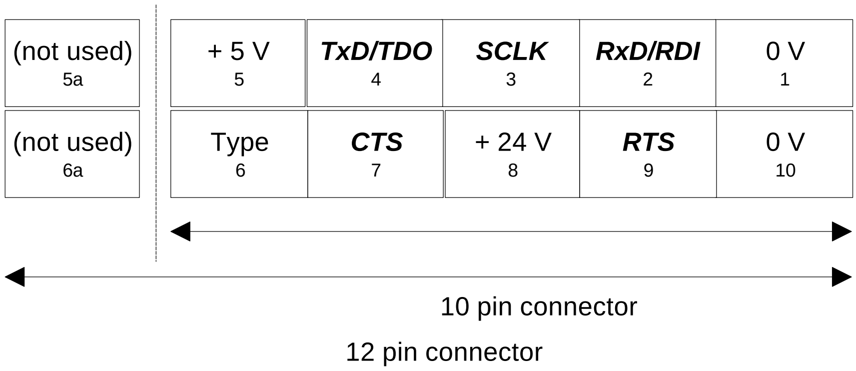

Fig. 3/6/2-7: Logical Specification of the Serial PEI Lines

4.1 Summary: PEI Type-dependent logical Properties of the PEI Connector Lines

The electrical and logical properties of the PEI connector lines depend on the PEI type given by the application module. For the various applications, different functional types of physical external interfaces are 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) are the same for all PEI types.

Fig. 3/6/2-8 shows the logical specification of the other PEI lines, dependent on the PEI type. The column headers give the logical names in relation to Fig. 3/6/2-6 and Fig. 3/6/2-7, distinguished in parallel and serial line usage.

PEI-Type	Functional description	PEI pin 2 I/O 2 RxD	PEI pin 3 I/O 1 SCLK (syn)	PEI pin 4 I/O 3 TxD	PEI pin 5a PWM2 -	PEI pin 6a I/O6 -	PEI pin 7 I/O 4 CTS	PEI pin 9 I/O 5 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

Fig. 3/6/2-8: PEI Type-dependent Specification of PEI Lines

5. Parallel PEI 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 4.1 gives an overview which PEI lines are relevant for each of the PEI types.

5.1 Meaning of PEI Type 0

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

5.2 Meaning of PEI Type 1

The PEI type 1 is reserved for applications that run without an application program.

5.3 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 paragraph 4.1 and the other explanations in Part 9/1 "Cables and Connectors".

5.3.1 Programmable I/O Configuration with PEI Type 17

This system variable EE_PortCDDR17 contains the direction bit register contents for port C. The operating system will 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 part 9/4 "BCU's and BIM's".

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 define serial PEI communication. See the PEI type table in paragraph 4.1 for information about the meaning of the PEI pins with respect to the serial PEI protocol.

At PEI type 14 there runs a synchronous protocol which serves to transfer data blocks of information from the internal user application to the external user application and vice versa. The data block format is explained in clause 6.2, the synchronous protocol in clause 6.3.1.2.

In distinction to the serial PEI type 14 protocol the PEI type 10, 12, 16 and 20 protocols serve to transfer messages from a specific EIB communication instance to the external user application and vice versa. The messages depend on the EIB communication instance which communicates to the external user application. See Chapter 3/6/3 "External Message Interface" for a description of the data format and the contents of the exchanged messages.

At PEI type 12 there runs the same synchronous protocol as for PEI type 14; therefore see also paragraph 6.3.1.2 for a detailed communication protocol explanation.

At PEI types 16 and 20 there runs an asynchronous protocol which is explained in the following.

At PEI type 10 there runs either the FT1.2 protocol as the default (asynchronous) protocol, or a synchronous or asynchronous protocol whose BAU protocol counterpart can be customer-defined. During device programming time the BAU protocol counterpart must 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 following figure shows the data block format. A data block of length n is transmitted transparent to the user.

block data

Fig. 3/6/2-9: Data Block Format for synchronous PEI Type 14 Communication

6.2.1.1 Block Data

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

6.2.2 Protocol Description

The protocol is as described in paragraph 6.3.2, concerning communication request and data exchange, but there is no exchange of the length byte.

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 data block of length n is composed of 1 octet for the length and n-1 octets for the message data.

length octet	message code	User data

Fig. 3/6/2-10: Message Format for Synchronous PEI Type 12 Communication

6.3.1.1 Length Octet

The length octet contains the number of subsequent block data octets. Fig. 3/6/2-11 shows the encoding of the length octet. The most significant bit is the even parity bit for the whole octet. The second-most and third-most significant bits build the length octet tag, which is always the bit sequence '01'. The other five bits allow to encode the number of octets following the length octet, i.e. the allowed value range for the number of following octets is 0..31.

P	0	1	X	X	X	X	X

Fig. 3/6/2-11: Encoding of the Length Octet

6.3.1.2 Message Code and Userdata

For definition of messagecodes and userdata see Chapter 3/6/3 "External Message Interface".

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) serve to transfer messages between the external user application and the BCU's communication stack. The message exchange consists of 4 phases:

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

6.3.2.1 Communication Request

The handling is the same, whatever microcontroller wants to communicate. A hardware handshake takes place on each byte transfer. It is a protocol of Request/Answer on the lines RTS (request to send) and CTS (clear to send). The communication initiator resets its RTS line and polls its CTS line, see Fig. 3/6/2-14. If CTS = 0 then the handshake is okay which is interpreted as a positive communication request.

6.3.2.2 Software Handshake

The first data exchange is a bi-directional transmission of the length byte. When a controller has nothing to send, it puts 0xFF as the length byte, otherwise it puts 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 is then 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.

The software handshake takes place on the first byte 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 sends the data octets. The other protocol instance responds in parallel by octets of value 0x00h.

6.3.2.4 Pause

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

6.3.2.5 Error Handling

In case of errors, protocol errors or time-outs the BAU resets the serial port. Then the BAU sets RTS high and polls CTS until CTS is also high. If high the BAU controller waits 10 ms before it requests a new data transfer. Otherwise it considers the request of the application controller. The time-out used for a data block transfer is about 130 ms.

6.3.2.6 Initialization

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

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

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

6.3.2.7 Protocol and Handshake Definition

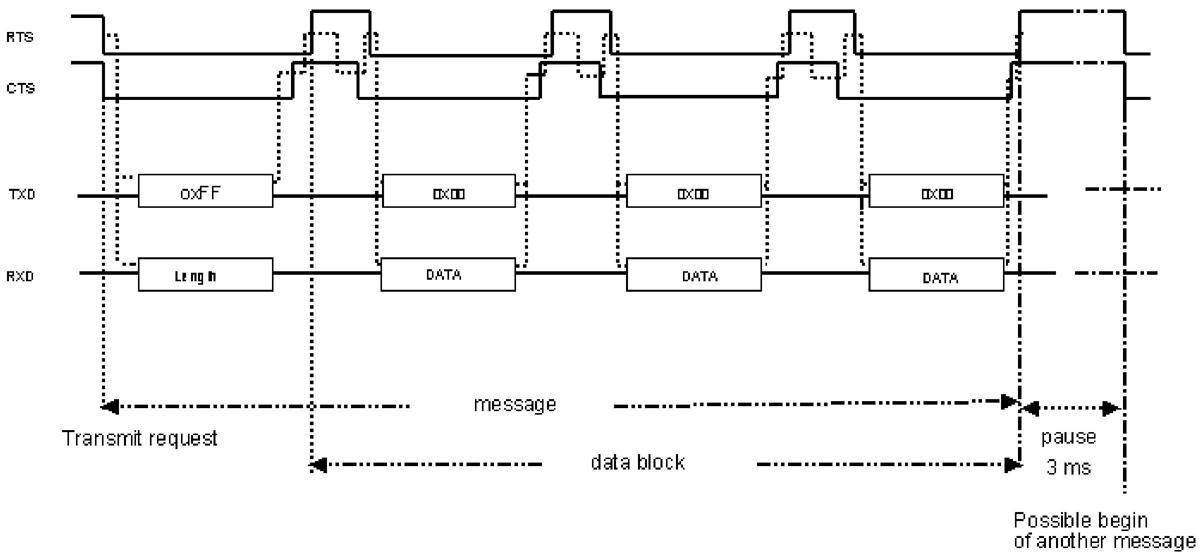

Fig. 3/6/2-12: Protocol, when the BAU Controller is Receiver

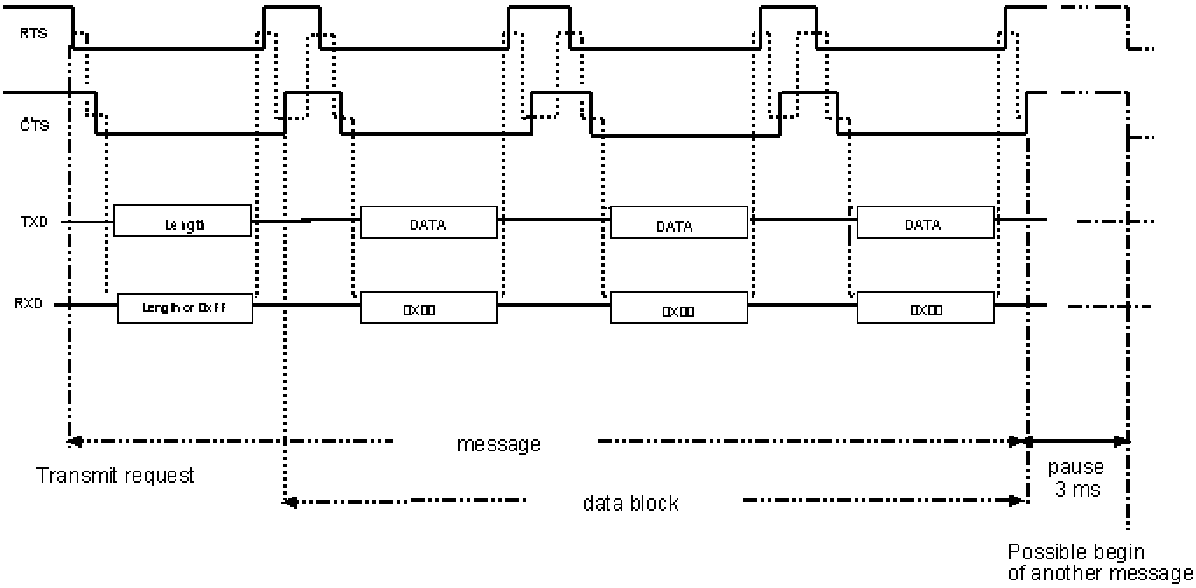

Fig. 3/6/2-13: Protocol, when the BAU Controller is Transmitter

6.3.3 Synchronous Signal Definition at PEI Types 12 and 14

6.3.3.1 Signals and Data Formats for Synchronous PEI Communication

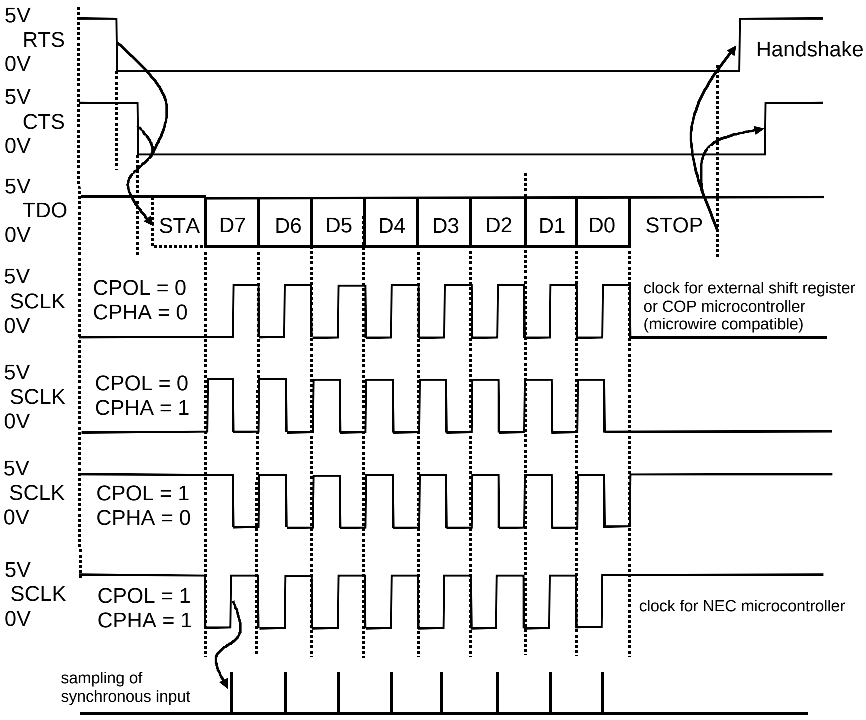

Fig. 3/6/2-14: Signals and Data Formats during Synchronous PEI Communication

For the specific location of the system variables CPOL and CPHA and how to set the baudrate, please refer to part 9/4 "BCU's and BIM's".

6.3.3.2 The Synchronous Protocol

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 is the same as the hardware handshake protocol of the asynchronous protocol. See there for more information.

6.3.3.2.1 PEI Recognition and default Settings

The serial port configures TDO as serial output, SCLK as serial clock output. The clock phase and data format are configured according to CPHA and CPOL configuration in EEPROM. The baud rate to be used is located in the non-volatile memory at label "SyncRate".

6.3.4 Definition of the Asynchronous Signal at PEI types 12 and 14

6.3.4.1 Signals and data formats for asynchronous PEI Communication

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

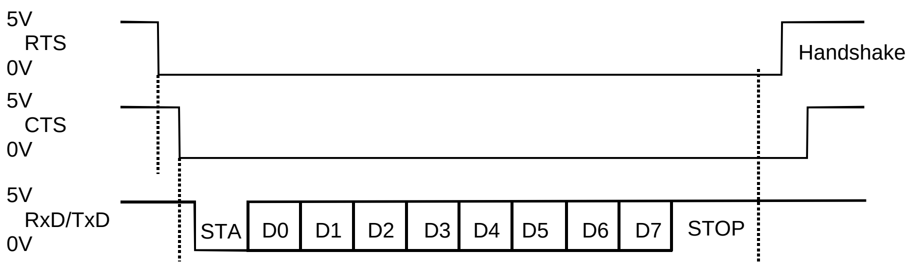

Fig. 3/6/2-15: Signals and data formats for asynchronous PEI communication

6.3.5 Data Transmission through the PEI

6.3.5.1 Byte Transmission BAU is Sender

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

The BAU detects a time out after approximately 130 ms (page 71). The time-out covers the entire data exchange (see Fig. 3/6/2-13, time message). As can be gathered from the figure, the CTS-edge is not 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 Fig. 3/6/2-13. Except for the time out the maximum time is not limited. The RTS only switches to False when both the BAU Byte and the BAU Acknowledge Byte have been transmitted. The sequence of both bytes is of no importance.

6.3.5.2 Byte Transmission Host is Sender

HOST switches from CTS-False to CTS-True

The only limitation as regards time is the time-out. After the last exchange of data both handshake signals must remain deactivated for at least 3 ms (see Fig. 3/6/2-12). When the HOST is transmitting, only after termination of the stop-bit-time is he allowed to switch to CTS. As long as no transmission is carried out, there is 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 reacts within 0 to 3 ms.

6.3.5.3 Block Transmission

Time between two bytes

After the host has transmitted one byte, the BAU can only switch his CTS-line to high when the RX-register is full, the stop bits have been detected and the transmitted byte has been entirely send.

Delays

The 3 ms delay between two block transmissions is always necessary. Other additional delays depend from one application to another. In case of EEPROM-programming one has to take a 20 ms/Byte delay into account, which is however only relevant when the value of a distinct byte really changes. The number of bytes which 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 is used: that means each station may act simultaneously as primary station (initiating a message transfer) and secondary station (receiving a message). It is restricted to point-to-point in the BAU (no address field) and both stations have equal access rights, i.e. there is 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 are connected via a 3-wire connection:

RxD: Received data

TxD: Transmit data

0 V: Signal 0 V

Data transmission is performed with 8 data bits and 1 stop bit with even parity (line idle is "1").

The transmission rate can be selected between the following values:

1200, 2400, 4800, 9600 and 19200 baud (default value is 19200).

6.4.3 Transmission Frame Format

The format FT1.2 includes frames with fixed length, frames with variable length and single character.

6.4.3.1 Frame with fixed Length

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

Start 10 h
Control field
Checksum
End 16 h

6.4.3.1.1 Transmission Rules

R1 Line idle is binary 1.

R2 Each character has 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 6.4.8) are admitted between characters of a frame.

R4 Upon detecting an error according to rule R6, a minimum interval (see LINE_IDLE_TIMEOUT in 6.4.8) is required between frames.

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

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

R6 The receiver checks:

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

The frame is rejected if one of these checks fails, otherwise it is released to the user.

6.4.3.2 Frame with variable Length

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

Start 68 h
length L
length L
Start 68 h
Control field
link user data
Checksum
End 16 h

Length specifies the number of user data byte including the control field (range from 2 to 23).

6.4.3.2.1 Transmission Rules

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

R6 The receiver checks:

- 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 is rejected if one of these checks fails, otherwise it is released to the user.

6.4.3.3 Frame format of single Character

One single character is specified:

ACK E5 h

The single character is used for a positive acknowledgment.

6.4.3.3.1 Transmission Rules

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

R5 - The receiver checks:

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

The frame is rejected if one of these checks fails, otherwise it is released to the user.

6.4.4 Control Field

The control field contains information that characterizes the direction of the message, the type of the service provided and supports control functions for suppressing losses or duplications of messages.

6.4.4.1 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 is used for suppressing losses and duplications of information transfers. The primary station alternates 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 is repeated with the same frame count bit.

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

This bit is always set when communicating with the Service SEND_UDAT.

SEND/NO REPLY services and other transmission services are not used.

Function code:

Function code	Service name	Service type	Service function	FCV	Frame format	format positive Confirm / response	format negative Confirm
0	SEND_RESET	SEND/ CONFIRM expected	Reset of remote link	0	fixed length	single byte ACK	nothing / fixed length NACK
03H	SEND_UDAT	SEND/ CONFIRM expected	User data	1	variable length	single byte 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
others	-	(not used)	Reserved	-	-		

6.4.4.2 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
0	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
others	-	(not used)	Reserved	-

6.4.5 Transmission Procedures

Simultaneous data transmission in both directions (BAU ⟷ external station) is supported. However the primary station accepts a new message transfer only when a previous message transfer is terminated either successfully or with an error indication (time-out).

Transmission errors are only detected by receiving station. A secondary station receiving a disturbed SEND or REQUEST frame does not reply. This is 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 detects the error and transmits the REQUEST frame once again.

6.4.5.1 Send/Confirm Service

The transmission procedure for this service can begin 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) is 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) can 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 is repeated (see following chapter).

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

6.4.5.1.1 Protection against Message Loss or Message Duplication

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

6.4.5.2 Request/Respond Service

The transmission procedure for this service can begin when the transmission procedure of a previous service is terminated.

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

6.4.6 Protocol Initialization

After a reset, the station sends a frame "Reset of remote link". On receiving the "Reset of remote link" frame, which has a FCB equal to zero, the secondary station will delete messages in its buffer and will 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. When the BAU sends a RESET the BAU expects 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 .

Station A	Service	Service	Station B
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: an 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

Station A	Service	Service	Station B
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 Parameters Description

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

Page left blank intentionally.