Bio-Rad Laboratories IMMUNOHEMATOLOGY

IH-Com LIS Connection Manual

V1.0-02/2016

The Complete Solution for Safe Transfusion

BIO-RAD

IDENTIFICATION

Document

IH-Com LIS Connection Manual Version: 1.0 - 02/2016
IH-Com Software Version: 5.0.8 and 5.0.9

REF H009269

IVD CE

Version	Date	IH-Com Version	Comments
1.0	30.03.10	3.0	First version
1.1	11.04.13	4.0	Minor updates related to **IH**-Com 4 plus graphic specifications
1.2	30.06.15	5.0	Separate Introduction "***3.1*** Introduction" on page 17 Update "***5*** Transfer protocols" on page 33 and "***6*** Configuration of the physical connection" on page 53 Addition of **IH**-Com Simple Protocol Addition of Socket Configuration Corrections in "***1.3.3*** Software" on page 10

Technical Information Contacts

Tel. +41 (0)26 67 45 111

Fax +41 (0)26 67 45 145

Manufacturer

DiaMed GmbH
Pra Rond 23
1785 Cressier FR
Switzerland

Copyright and Disclaimer

TABLE OF CONTENTS

Table of Contents

PAGE INTENTIONALLY LEFT BLANK

1 Generalities

Chapter Overview

This chapter contains basic information on the software and the document structure.

1.1 Appropriate Use

IH-Com is a software application to be used exclusively with an *in vitro diagnostic* instrument.

IH-Com is a software solution used for patient data, QC, sample and instrument management.

IH-Com may only be operated by trained personnel and is not intended for use in a direct patient environment.

The instructions contained in the LIS Connection Manual must be followed with particular attention to the safety instructions

1.2 Warranty limitation

Bio-Rad Laboratories recommends a backup of the computer before installation of **IH**-Com.

Bio-Rad denies any responsibility in case of:

- wrong use of the software;
- unauthorized modification (willingly or unwillingly);
- not complying with the instructions contained in the provided manuals;
- not complying with the safety instructions contained in the manuals;
- damages, in particular any data loss or financial loss which could possibly be attached to the use of the software.

When **IH**-Com is connected to an LIS, the user takes full responsibility for an error free transmission of orders and/or results (hardware, software, firmware, etc)..

Any warranty will, be deemed void if fault is found to have been caused by maltreatment, misuse, unauthorized maintenance of service or negligence of regular maintenance and service, accidental damage, incorrect storage or use of the products for operations outside their specified limitations, outside their specifications, contrary to the instructions given in this manual or with other than the manufacturer's original tips.

1.3 Glossary

The following terms, among others, are used in this manual.

1.3.1 Persons

Manufacturer

The manufacturer of **IH**-Com is:

DiaMed GmbH
Pra Rond 23
1785 Cressier FR
Switzerland.

Operator

The operator is the owner of **IH**-Com, when using it as the owner, or if transferring it to a thirdparty.

Personnel

Any person who operates **IH**-Com. These individuals should be qualified in accordance with the manufacturer's requirements.

Technical Personnel

Technical personnel are trained individuals who are allowed to perform specific tasks **IH**-Com.

Personnel qualifications

Certain personnel qualifications are required in order to perform the activities related to **IH**-Com.

The personnel qualifications define the minimum requirements which must be met by the authorized personnel.

Serious injury

A “serious injury” is an injury or illness that [21 CFR 803.3]:

- is life threatening;
- results in permanent impairment of a body function or permanent damage to a body structure or;
- necessitates medical or surgical intervention to preclude permanent impairment of a body function or permanent damage to a body structure.

Permanent means irreversible impairment or damage to a body structure or function, excluding trivial impairment or damage [21 CFR 803.3].

1.3.2 Product

Software

IH-Com distributed by the manufacturer.

Bio-Rad authorized gel card

Product based on column agglutination technique including a microtube system consumable where the antigen-antibody binding is fixed in Gel-Matrix.

> Only gel cards manufactured by Bio-Rad may be used.

Microplate

A microplate is a standard product with 96 wells, in which tests are performed. The reaction substance is either coated in the well of the Microplate or pipetted into the well of the microplate in order to get a antigen-antibody binding

Sample

Content of any sample tube.

Bio-Rad authorized reagent

A substance or compound that is added to a system in order to achieve a chemical reaction. Such a reaction is used to confirm the presence of another substance.

> Only reagents authorized by Bio-Rad may be used.

1.3.3 Software

Asynchronous Process

An asynchronous process in the data transfer is distinguished by the two remote stations (sender and receiver) being temporal independent of each other in starting to send data. The end of the sending process will not be awaited for further processing.

Division with remainder

A division with remainder is similar to normal division with the difference that the former returns two results, on the one hand the integer quotient of the division and also the remainder which is left over from this operation (the modulus).

The normal division of, for example, the dividend 16 by the divisor 10 produces as result the number 1.6 -> 16:10=1.6

On the other hand, the division with remainder produces the values 1 with remainder 6 -> 16 DIV 10 = 1 // 16 mod 10 = 6.

Frames

When data are broken down into several parts and sent one after the other, we are talking about data packages. If these data packages are supplemented with header data, control characters or similar in order to coordinate the data transfer, the total is called a frame.

In comparison, the term data packet can refer both to a subset of the user data as well as to a complete frame.

Handshake (process)

A handshake in the data transfer means that the user data are not simply sent via a data line/ connection but that sender and receiver communicate with each other using control characters in a question/answer process.

For example, the sender can ask the receiver whether it is ready to receive data by sending a control character. The sender then waits for a response from the receiver (it now becomes physically the receiver) which can communicate its readiness for the data receipt or even communicate that no data can currently be received.

An error correction process is usually also part of such a process. For example, if the receiver establishes that the received data show an error (e.g. using checksums), it can inform the sender about this and arrange for the missing data to be sent again.

Hexadecimal Number

In normal daily use, we use the decimal system with the digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. This is called a base 10 notational system.

There are many (infinite) possible number systems in mathematics. One of these is the hexadecimal system (from the Greek word **Hexa** = 6 and the Latin word **Decem** = 10, i.e.16) which is a base 16 number system. Here, there are not only the 10 well-known digits but also another 6 represented by the letters A to F. After the number 9, counting continues with A, B, C, D, E, F, and only then the 10 (which corresponds to the decimal value 16).

The hexadecimal system is particularly important in the world of computers. Because 4 bits can be covered with one digit; an 8-bit number can be conveniently represented by two hexadecimal digits which significantly simplifies the readability of binary information.

Example:

- Decimal 12 = binary 1100 = hexadecimal C
- Decimal 244 = binary 11110100 = hexadecimal F4

A comprehensive explanation of the hexadecimal system can also be found on the World Wide Web at http://de.wikipedia.org/wiki/Hexadezimal.

Synchronous Process

In the case of a synchronous process in the data transfer, two participants of a data transfer agree which side transfers its data. Only one side can transfer its user data at any time while the other side only transmits confirmations or other control information.

A synchronous process usually uses a handshaking process.

1.4 Typographical conventions

The following styles are used in this manual.

Description

This style, used in conjunction with illustration numbers, is preceded with the corresponding numbers:

Example:

1	First element.
2	Second element.
3	Etc...

Command

Any software command, button, function key, window, icon, option, tab, checkbox, selection box, article, menu, tool bar, field and section used in this document is represented by a bold italic font.

Example:

The ***Exit*** command allows to quit the software.Procedure

Each procedure step to be carried out step-by-step by the user is preceded by a letter.

Procedure

Example:

A	Start the **IH**-Com Communicator by double clicking on the desktop shortcut.
B	Select the ***Program/Login user*** menu.
C	Click ***New User*** (1).

Procedure result

A procedure result is shown by the following symbol .

Example:

A	Click on the ***Parameters*** button. *The **Parameter** screen is displayed.*

Cross reference

This style helps the user to find complementary information linked to the current subject.

Example:

See chapter " Cross reference" on page 12..

List of items

This style is used in order to display a list of elements.

*Example***:**

- item 1;
- item 2;
- item 3.

1.4.1 Labeling

Depending on the importance of the warning and the associated risks, two warning styles are defined.

The safety aspects are used in accordance with the requirements contained in the following norms:

- ANSI Z535.4;
- ISO 3864, ISO 3864-1:2002 and ISO3864-2:2004.

1.4.1.1 Warning

Indicates:

- **an imminently hazardous situation which, if not avoided, will result in death or serious injury**
- **a potentially hazardous situation which, if not avoided, could result in death or serious injury**
- **a potentially hazardous situation which, if not avoided, may result in minor or moderate injury**
- **a risk of minor damage, but no possible injury**

1.4.1.2 Note

designates:

- a preferred procedure or a recommended usage
- a general or purely informative remark

2 Safety

Chapter contents

This chapter sets out safety instructions to ensure safe and trouble-free operation of the **IH**-Com and its associated instruments.

2.1 Introduction

2.1.1 Principles

⚠ **Before carrying out any operation whatsoever on the IH-Com, it is imperative to read this chapter and fully understand it.**

In case of any doubt, consult your Bio-Rad representative.

2.1.2 Importance of the safety instructions

⚠ **All the safety instructions in this LIS Connection Manual and in the software must be complied with in order to prevent accidents to persons, damage to equipment or pollution of the environment.**

In a similar manner, the legal bylaws and the recognised technical rules which apply in the country of use of the **IH**-Com must be adhered to.

2.1.3 Disregarding the safety rules

Disregarding the safety rules, as well as existing legal and technical regulations, may lead to accidents, property damages or environmental pollution.

⚠ **Disregarding the instructions for use given by the manufacturer may reduce the level of protection offered by the instrument.**

2.2 General safety instructions

⚠	**Repair and service operations must only be performed by a qualified service engineer appointed by the manufacturer.** **Take into account all the warnings and follow all instructions provided with the IH-Com and in the documentation.**
⚠	**The IH-Com may only be operated with the instruments and with the accessories approved by the manufacturer (example: external barcode reader).** **Only use virus free USB sticks on all of our IHD instruments and computers.**

3 BASICS OF SERIAL DATA TRANSFER

3.1 Introduction

In **IH**-Com there is a variety of combination possibilities for the interface with LIS systems regarding:

- data content (content and format of data to be transferred)
- transfer protocol (characterizing the transfer of data) and
- physical connection (connection type)

Following combinations are possible:

Connection Type				
	Serial Connection (Serial Cable)	TCP/IP (Server, Client) (Network Cable)	FTP client	File (UNC/DFS)
ASTM 1381	x	x	-	-
Maestro (emulation)	x	x	-	-
IH-Com Simple	-	x	-	-
None or FTP	-	-	x	x

Table 1: Possible combination of transfer protocol and connection type

Data Content			
	ASTM 1394	Maestro (emulation)	External LIS
ASTM 1381	x	x	x
Maestro (emulation)	x	x	x
IH-Com Simple	x	x	x
None or FTP	x	x	x

Table 2: Possible combination of transfer protocol and data content

Transfer protocols ASTM-1381 Standard, Maestro (emulation) and **IH**-Com Simple are described in this document.

For better understanding the basics of the serial interface are to be focussed on which demonstrate the necessity of a well aligned configuration as well as the comparison with TCP/IP regarding reliability, performance and error-proneness.

Additionally the configuration of such LIS interfaces in **IH**-Com is described.

A description of the transfer Protocol FTP (File Transfer Protocol) is not part of the document. Refer for details of the functionality of FTP to the LIS server software documentation.

Refer to the documents IH-Com HOST Connection Manual ASTM 1394 and IH-Com HOST Connection Manual Maestro Emulation for details of data content. Data content of the external LIS is described in the customer specific documentation.

3.2 History of the serial interface and transmission

Serial data transfer is one of the oldest methods of transferring electronic data. The basics of this method were already used with the advent of the computer age in the second half of the last century.

The first industrial specification which described serial data transmission (EIA-232, prepared by the Electronic Industries Alliance) was applicable as early as 1962. At that time, the transfer envisaged a transfer of data on two data lines (voltage + ground). The sender transmitted the data which it wanted to transfer bit by bit over the line and the recipient had to reassemble these data bits back to the original data (*Fig. 3-1*).

As very long telephone lines were mainly used as carrier medium at this time, the first specification envisaged a data rate of 100 bits per second. Up to 19,200 bits per second were already possible over short distances, for example between peripheral equipment such as a printer or a server/terminal client connection. In order to realize these speeds, the data had to be transferred very cleanly which is why connectors with 24 pins (instead of the usual 9 today) were still the standard at that time. On the one hand, there were still many control lines at that time (for example, for modems or their predecessors - the acoustic couplers) and on the other hand, these first specifications envisaged an own ground line per data and control line.

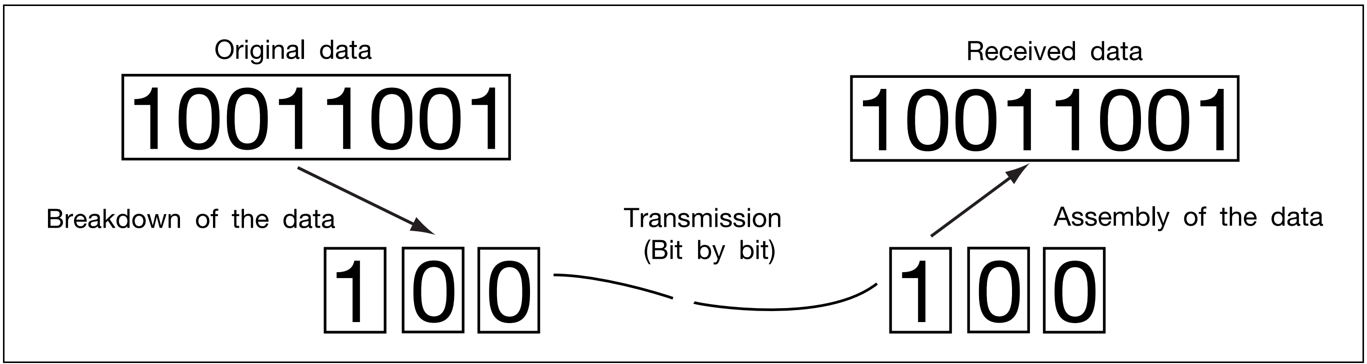

Fig. 3-1: Serial transmission

In the course of the years, the specifications have been adapted to the new technical possibilities, mainly the better signal quality of the lines and the modern chips which produce the signals. The most current version of the standard is the 1997 standard ANSI/EIA/TIA-232-F-1997.

> In addition to the simpler 9-pin plug connections, this also envisages data transfer rates of up to 500,000 bits per second.

3.3 Technical realization of the serial transfer

The serial transfer basically functioned similarly to the exchange of information earlier in shipping. When two ships met and wanted to exchange information, they used specific arm positions which were shown by a sailor or officer using two flags.

The ship now had a sender which could "announce" data.

Accordingly, the other ship was the receiver (using an observer of the displayed flag position) which could "read" these data. The individual pieces of information were then reassembled by the receiver back to the message which the sending ship wanted to transmit.

In principle, the serial data transfer in computer systems functioned exactly the same. However, signal levels instead of flags are used here. In the case of the serial port, different voltages are used for this.

The industry standard ANSI/EIA/TIA-232-F-1997 envisages a voltage between -15 and -3 Volts to signal a bit with the value 1 while a voltage of +3 to +15 Volts signals a bit with the value 0. Voltages between -3 and +3 Volts are not defined; they announce a signal change between 0 and 1.

3.3.1 Serial signal voltages

Applied signal voltage	Is interpreted as...
-15 Volt … -3 Volt	1
-3 Volt … +3 Volt	-
+3 Volt … +15 Volt	0

While the specifications were previously always complied with that +15 an -15 Volts were really used, many serial interfaces today use the signal voltage of +/-5 Volt used as standard in the PC. This particularly concerns interfaces directly on the main board.
This is in fact within the specification, however mainly older peripheral equipment and also computer interfaces do not detect this low voltage correctly and only report errors or do not even establish the connection.
If a serial connection is not established, it is always recommended to check whether either of the two communication points has an older interface. In this case, it is advisable to replace them or, if this is not possible, to use an expansion card for the other point (the PC) which generates the required 15 Volts from the available 5 or 3.3 Volts using special electronic components.

3 Basics of serial data transfer

Two data lines are used which cross over to connect the receiver unit of one side with the transmitter unit of the other side.

The data are now announced or read within specified time slices both on the sender side as well as on the receiver side (see figure below).

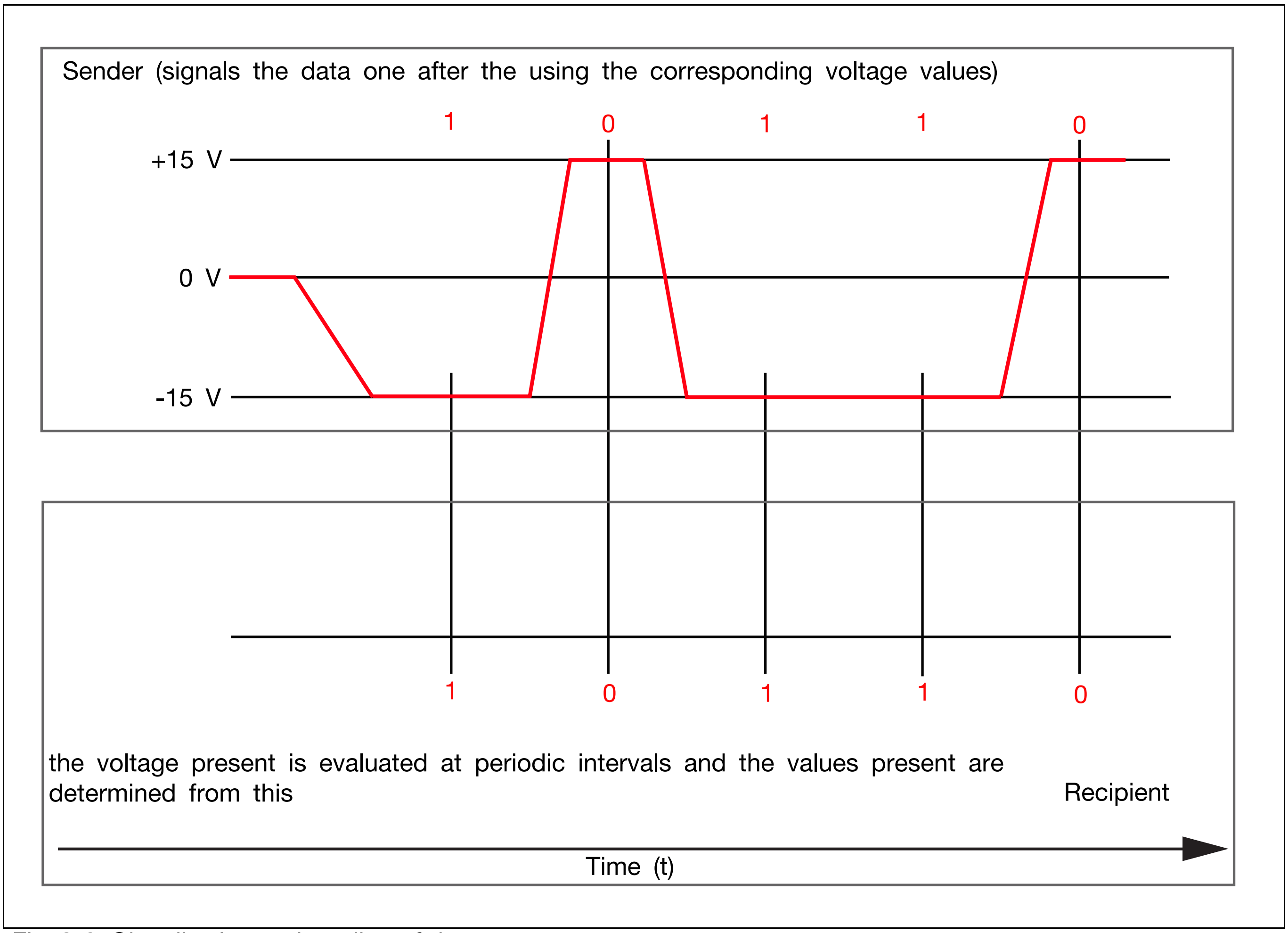

Fig. 3-2: Signalization and reading of data

The length of these time slices is different depending on the specified transfer rate (the baud rate). Or to put it another way; the higher the data rate, the shorter is the time in which the bit to be transferred is announced (and the reading time slice must be correspondingly small).

3.3.2 Baud rates and bit duration

Here are some examples of how long a bit is signaled depending on the specified baud rate:

Baud rate	Signal duration
50	20 ms
300	3.3 ms
1,200	0.833 ms
2,400	0.417 ms
4,800	0.208 ms
9,600	0.104 ms
19,200	0.052 ms
57,600	0.017 ms
115,200	0.00868 ms

3.3.3 Receiver with doubled baud rate

As no synchronization between receiver and sender is performed (both time slices run with the same system clock rate and are therefore slightly offset), it is very important that both sides use the same settings of the serial port.

For example, if the receiver is set to double the baud rate, it has no chance to read the data correctly:

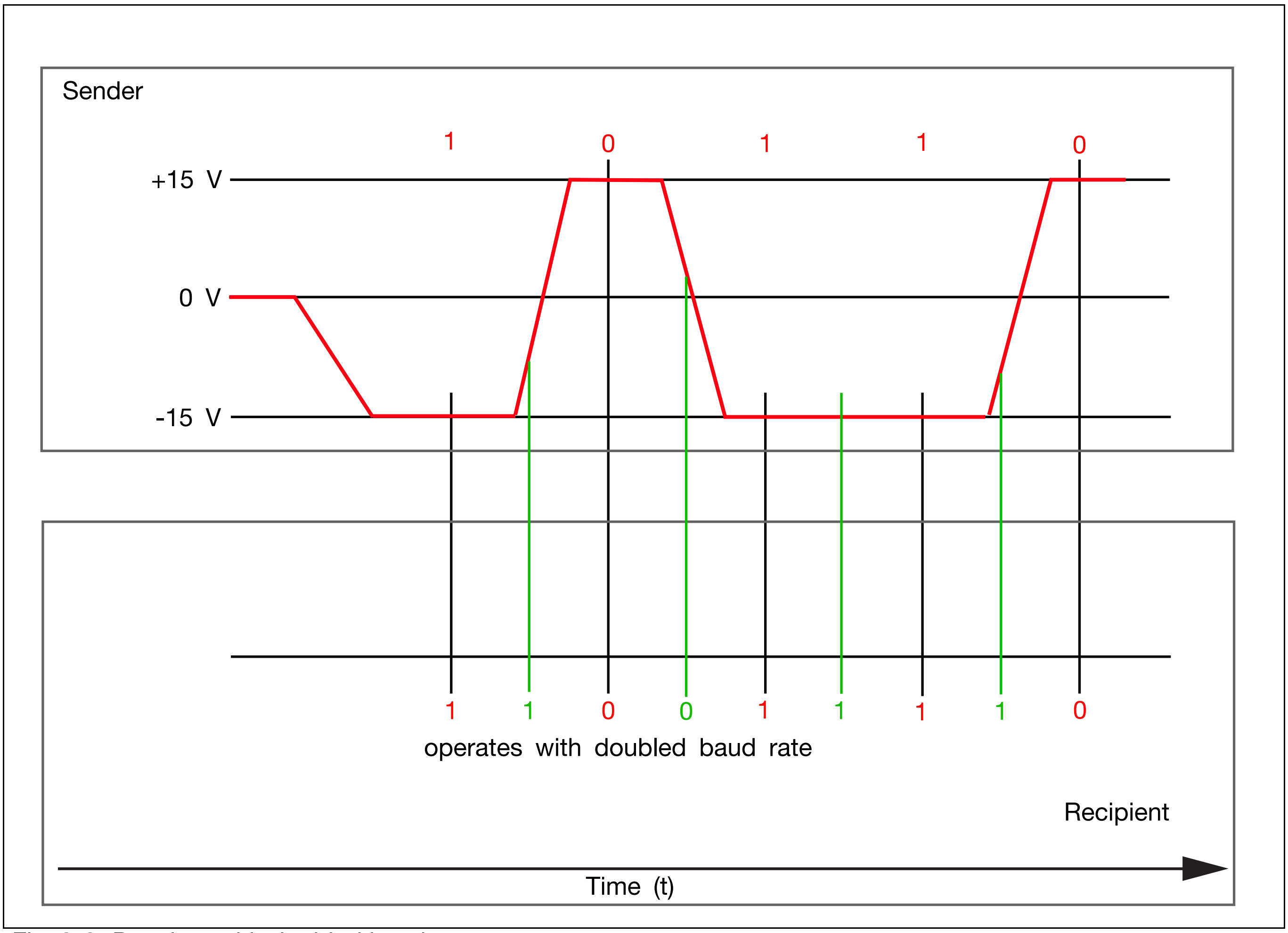

Fig. 3-3: Receiver with doubled baud rate

It is assumed in the example that the reading is performed during the signal change, while the voltage is still smaller than -3 or larger than +3 Volts.

Here, the receiver would read the bit sequence 110011110 instead of the transmitted bit sequence 10110. The data would be completely falsified. As long as the not defined range is not encountered (provided it is correctly recognized at all), the receiver does not detect any errors and must assume that the received data are correct.

In order to counteract such problems, error correction methods had to be developed. Some of these still exist today for the serial transmission; however this means that the serial port must be more extensively configured and that the same settings must absolutely be set on both the sender side and for the receiver.

3.4 Error correction methods of serial data transfer

There are some methods which should make the transfer via the serial interface more reliable.

The main parameters of the serial interface today are:

- Start bit
- Number of data bits
- Parity bit (optional)
- Stop bit (optional)
- Flow control

A "packet" of bits which should be transferred via the serial interface looks like the following today:

Start bit	Defined number of data bits (usually 8)	Parity bit	Stop bit

3.4.1 The Start Bit

The start bit is standard today. A single bit (without carrying user data) is sent at the start of a new transfer to initialize the transfer. The sender signals the start bit and then starts its transmission interval. The receiver can also start its read interval and otherwise ignores the start bit.

3.4.2 Number of Data Bits

The data bits (with the actual user data) are signaled after the start bit. A fixed number of bits which should be transmitted at once in one data packet must also be specified for the serial ports. Both the receiver as well as the sender must be configured for the same number of data bits. The usual configuration defaults to 8 data bits.

In the example above of the doubled baud rate of the receiver, there would already have been an opportunity to notice an error using the number of data bits as too many bits will be read.

3.4.3 Parity bit

The parity bit is a simple mechanism to recognise an incorrectly transferred bit.

The two most frequent parities are "even" and "odd".

This means the number of bits with the value 1 in the user data. Depending on their number, the parity bit is set to 1 to make the number even or odd.

A few examples for this:

User data	Required parity	Number of bits with value = 1	Parity bit value
11001001	Even	4	0
11001001	Odd	4	1
11001011	Even	5	1
11001011	Odd	5	0

If a bit is recognized incorrectly, this can be detected using the parity bit.

Here are also a few examples for this (all with even parity, i.e. even number bit 1):

Transmitted	Read	Number of bits with value = 1	Parity	Error detected
11010101	11010101	5	1	No, OK
11010101	11010111	6	1	Yes, OK
11010101	11010001	4	1	Yes, OK
11010101	11000001	3	1	No, ERROR

The last example already shows that the parity is not a universal remedy for transmission errors. If two bits have been changed, the parity is correct again. In this case, the receiver would not detect the transmission error and would thus recognise the falsified user data as correct.

3.4.4 Stop bit

If a stop bit is used, this terminates a data packet. Thereby, the stop bit can have the length of 1, 1.5 or two. One, one and a half and two stop bits are indeed used commonly; however, this does not mean the number but the duration of how long the stop bit is signaled. The receiving side can thus check whether it has correctly recognized (in the number) all bits previously.

3.4.5 Flow control

Flow control was also implemented quite early. Some of the control lines of the serial interface are used for this and in parallel with the two data lines used. For example, it can be signaled "a new bit starts here" by switching the signal line from a High level (+15 V) to a Low level (-15 V) or vice versa. The opposite side can thus better recognise the data.

Alternatively, one of the control lines can also be used as "a bit is present" signal. As long as the signal is present, the receiver can read the bit; if the signal is switched off, the opposite side then knows that a new bit is starting soon.

The variants which exist for flow control are very different. Most also function with the 9-pin connector system which is used today. However, there are also methods which originate from the times of the 24-pin connector system and use control lines which no longer exist today.

It is important that both sender and receiver use the same or no flow control. It is usual today not to use any flow control.

3.4.6 Serial Interface Summary

The serial interface is a very old connection system between two computers or between one computer and a peripheral device.

It is a point-to-point connection which can only connect one transmit/receive unit with another transmit/receive unit.

The data rates are generally quite low, error detection is indeed available, however quite unreliable. A check of the user data must generally be implemented in the software as the interface itself is too imprecise.

Due to the long history of the interface, compatibility problems can occur again and again.

The length of the line between the two devices to be connected is dependent on the data rate which should be used for the transfer.

The following values can be used as guidelines for this:

Target data rate	Maximum line length
2,400 Baud	900 m
4,800 Baud	300 m
9,600 Baud	152 m
19,200 Baud	15 m
57,600 Baud	5 m
115,200 Baud	< 2 m

PAGE INTENTIONALLY LEFT BLANK

4 DATA TRANSFER VIA A NETWORK CONNECTION

Chapter Overview

The requirement to exchange the data not only using (partially unreliable) data carriers but to connect the computers directly – to network them – already emerged very early after the advent of the computer age.

Research for this, initially supported by the military, started as early as 1962. ARPANET which networked some important USA universities and the Pentagon was created in 1969.

The network grew over time and research institutions from other countries were connected to the existing network.

As different physical data transfer methods (e.g. Ethernet, Token Bus, Token Ring, FDDI) were used in different regions of the world, the necessity was recognized to develop a protocol which used and abstracted these transfer methods in order to create a standardized transport layer. This was the birth of the Internet Protocol (IP).

Another protocol was also developed for further standardization which used the Internet Protocol, the most well-known today TCP (Transmission Control Protocol). The entire Internet and very many local networks today are based on this TCP/IP combination.

The network communication today is structured as follows:

Transmission	Network communication			
Applications	FTP, DNS, HTTP, SMTP, NTP, ...			
Data Layer	TCP		UDP	
Abstraction Layer	Internet Protocol (IP)			
Physical transfer	Ethernet	Token Bus	Token Ring	FDDI

The transmission model is thus more complex than via the serial interface which only contains the physical transmission above and directly above that the applications.

4.1 Physical transfer using Ethernet as an example

The data are also transferred in the form of electrical signals for the network transmission. And just like the serial interface, these data must be transmitted bit by bit. And voltages are also used again which signal a bit and (in this case several). The receivers determine the data pattern from this which should be transferred. However, there are some differences.

4.1.1 History of Ethernet

As the representative for the numerous physical data transfer methods in the network, Ethernet should be examined more closely here. Ethernet is the most widely used method today, mainly in Europe, Asia and the American double continent.

The first version of Ethernet was standardized in the year 1976. This first version was designed for a data transfer rate of 3 MBit/s. It was therefore already significantly faster than, e.g. the serial interface at that time.

The specifications were revised as early as 1979/1980 and envisaged data rates between 1 and 20 MBit/s. At that time, the IEEE (Institute of Electrical and Electronics Engineers) standard was internationally anchored in the standard 802.xxx. Since that time, Ethernet has been supported by more and more (also large and well-known) manufacturers and an increasing number of products according to this standard appeared. Various sub-groups of the standard were formed as Ethernet was and is capable of running using various media (initially also telephone lines as Cat. 3, coaxial cable, twisted pair cable to glass fibers and using wireless).

In 1995 Ethernet was standardized with 100 MBit/s, and later for 1,000 MBit/s (Gigabit Lan) and 10,000 MBit/s.

4.1.2 Addressing in Ethernet

In order to participate in an Ethernet-based network, each Ethernet device (e.g. a network card) requires a unique address. A 48 bit long code is used for this which is unique in the world. This bit pattern used for the identification of the Ethernet device is called the MAC address. The 48 bits for the MAC address are usually stated as hexadecimal numbers. For example, a MAC address can be stated as: 08-00-20-ae-fd-7e; alternatively the format 08:00:20:ae:fd:7e is also possible and valid.

For better classification, the first 24 bits are used as manufacturer identifier. In this way, large manufacturers can allocate continuous address ranges for their devices such as, e.g. network cards.

Reserved MAC Range	Manufacturer
00-50-8B-xx-xx-xx	Compaq
00-07-E9-xx-xx-xx	Intel
00-60-2F-xx-xx-xx	Cisco
00-15-F2-xx-xx-xx	Asus

The address FF-FF-FF-FF-FF-FF has a special role. This is designated as broadcast address. If data are sent to this address, the data are received and evaluated by all reachable receivers.

4.1.3 Ethernet data packets and data transfer

The data themselves are now transmitted in the form of data packets. These data packets have a fixed structure. First, they contain the 48 bits of the MAC address to which the packet should be sent. This is followed by the 48 bits of the MAC address of the sender. Then a type field consisting of 16 bits is sent, followed by the actual user data and a 32 bits long checksum.

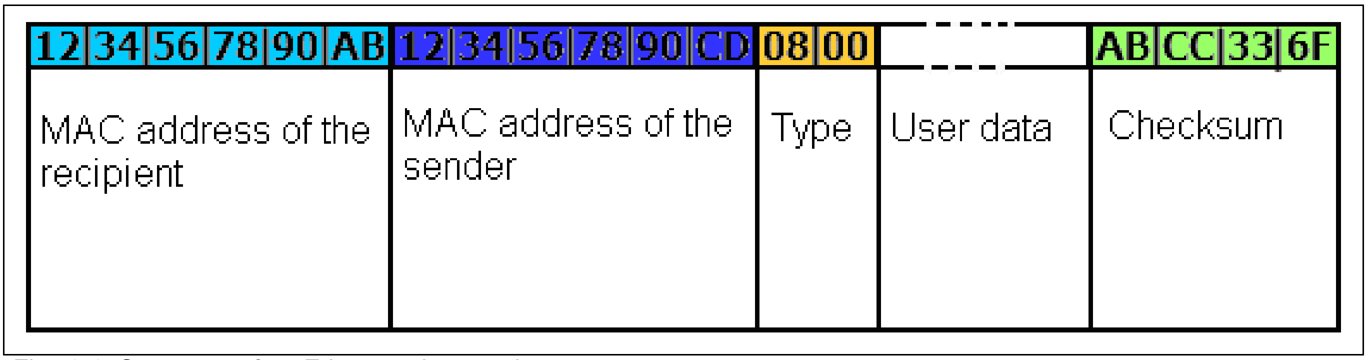

Fig. 4-1: Structure of an Ethernet data packet

In contrast to the serial interface which is a point-to-point connection (there is only one sender for exactly one receiver), the network is a point-to-multiple points connection. This means that there is one sender and any number of receivers. Because several computers participate in the network (in the example of networking computers using network cards).

If one computer now wants to send data to another computer, it first checks whether data are currently being transferred on the data line(s). If no data are currently being transferred, the computer simply starts sending; the individual bits of the data packet are signaled one after the other. All connected receivers compile the data packet from this and check whether their own MAC address is the recipient address (or whether the data packet has been sent to the broadcast address). If the data packet is not intended for one of the receivers, it discards it.

If two computers now want to send data at the same time and both detect that no data are currently being transferred, this can result in both computers sending simultaneously. As they share the same (physical) line, the signals overlap and result in a data collision. The data can no longer be kept apart from each other. The receivers recognise this and send a collision signal. All senders then stop transmitting and go into a wait cycle which has a random length so that the two computers do not attempt to send again at the same time.

This method is called CSMA/CD (Carrier Sense Multiple Access/Collision Detection).

In principle, this method can be imagined as a room in which there are several persons. These would like to communicate something to someone else. If there is currently silence, one starts to speak. If two persons start speaking simultaneously, this is recognized and both stop speaking and wait to see whether someone else starts talking.

4.1.4 Ethernet in comparison with the serial interface

Ethernet is to a certain extent related to the serial transfer using the serial port. The data are transferred one bit after the other for both.

The main difference is that two devices are connected directly using the serial interface while any number of devices are connected to the same line in the network. Nevertheless, significantly higher data transfer rates are possible using a network connection. In contrast to the serial interface, Ethernet uses a checksum for the user data. Bit errors can be more reliably established with this. In this way, Ethernet has a significantly lower susceptibility to errors.

4.2 The Internet Protocol IP

In order to exchange data in an Ethernet network, all nodes must be connected to the same line. However, the more that do this the higher the probability that collisions occur. The performance is also reduced more strongly.

These problems can indeed be alleviated by the use of switches instead of hubs as connection element as a switch notes which Ethernet address can be found where in the network and only forwards a data packet to where it is also needed. However, this technique was not sufficient for larger networks. On the one hand, there are still methods other than Ethernet and on the other hand, Ethernet packets can be processed differently depending on the operating system.

It was necessary to put another protocol in between here which took over the management of the data packets and their addressing. The answer to this was the Internet Protocol IP.

The Internet Protocol resulted in a new type of addressing – the IP addresses. These consist of four groups of 8 bits each, or expressed more understandably of four decimal numbers between 0 and 255 (for IP Version 4).

The MAC address of a computer no longer had to be known with this but the IP assigned to it which the Internet Protocol binds internally to the available MAC address.

Data are also transferred in data packets using the Internet Protocol. These have a significantly more complex structure in comparison with Ethernet. They consist of a packet header which consists of version information, some signal bits, length information, the sending and destination address and a header checksum. This is followed by the data which also have checksums. Data errors can be reliably established using the length information from the header and the checksums.

Another major benefit is that data packets can be "routed" using IP. If an IP is addressed which is not located locally in the network but in a different network which can be reached using an external connection (Internet), the data are routed internally from switch to switch (which are each addressed via Ethernet and their MAC address) until the data packet arrives in the destination network and can be transferred to the IP destination.

Details about the functionality of the Internet Protocol, the structure of the packets and the checksums can be found in RFC 791 at http://tools.ietf.org/html/rfc791.

4.3 The TCP Transmission Protocol

Using the Internet Protocol, the data can now already be exchanged over a large distance and also beyond MAC limits. There is also good error detection. What is still missing here is a specification how the data transfer itself should function in order to transport user data from A to B.

The Transmission Control Protocol TCP is used for this.

TCP controls the data transmission. It establishes a connection to the receiver and transfers the data. Data packets, the TCP packets, are used again for this. In addition to checksum numbers, these also contain packet numbers, acknowledgement numbers and information whether further packets still follow for the current data transfer and at which point of the transfer the current data packet is.

However, the most important difference is that TCP functions using a handshaking process[1]. Information is sent which the receiver must acknowledge. If the receiver has established that the data are not OK (i.e. one of the checksums is not correct or the length information is different, …), it requests the individual packet again. TCP makes a synchronous process out of the previous asynchronous process (Ethernet (= serial) -> Internet Protocol)[2].

This increases the reliability; erroneous modification of the user data is practically ruled out as too many security systems check the data.

1. *See glossary: “ Handshake (process)” on page 10*
2. *See glossary: “ Asynchronous Process” on page 10 and “ Synchronous Process” on page 11*

4.4 Network Transmission Summary

The network transmission is also serial at the physical level. Data packets are transferred bit by bit.

There are various physical transfer methods; the most widely used is Ethernet. It is based on hardcoded addresses which are 48 bits long and are called MAC addresses.

The transfer itself is performed in the CSMA/CD process whereby all stations in the network initially capture the data and then check whether the data are intended for the respective station in order to process or discard the data afterwards.

The data are checked using a checksum.

The higher layer Internet Protocol IP uses the lower layer protocol (e.g. Ethernet) to transport the data based on a destination IP. In doing so, data can be transported out of a local network via intermediate stations into other networks.

There are both a length check as well as checksums in order to control the data.

The protocol TCP is used as top layer. This makes a synchronous transfer from the asynchronous transfer of the lower layers (which TCP itself uses for the transfer) by using a handshaking process[1] in which not only the data themselves are exchanged but also status information about these data and their integrity. If data are detected as transferred with errors, TCP manages the data transmission again so that the user data are transferred reliably.

4.4.1 Comparison of serial transmission with network transmission

The network transmission and the serial transfer via the serial port are similar at the physical level. However, one is a point-to-point connection while the other is a point-to-multiple point connection.

However, the greatest difference is that a series of protocols exist for the network which manage the transfer and perform efficient error management. If a software package uses a network connection based on TCP/IP, the application only needs to transfer the user data to the protocol which performs the rest itself.

However, the most important difference is that a (more reliable) synchronous transfer is made from the asynchronous transfer by using TCP.

Description	Transfer using Serial port	Transfer using TCP/IP
Connection type	Point-to-point	Point-to-point or Point-to-multiple points
Speed	max. 0.5 MBit/s	usual 100 to > 1,000 MBit/s
Max. distance	approx. 900 meters	Global
Error management	Rudimentary, limited bit error detection	Extensive using various checksums at various levels
Synchronicity	Asynchronous	Synchronous
Security against interception	Very high (direct connection)	Very low without the use of encryption methods

1. *See glossary: " Handshake (process)" on page 10*

5 TRANSFER PROTOCOLS

Chapter Overview

The transfer protocols ASTM 1381, Maestro (emulation) and **IH**-Com Simple are so-called high level protocols. They define the method of data transfer (hand shake) via optional physical media and possibilities of error detection during transfer. Insofar procedures resemble e.g. Ethernet because they concern only the transfer irrespective which data is transferred. Both text data and image data may be transferred. Both would not make a difference for a low level protocol.

5.1 Necessity of transmission protocols

The transfer of data via serial interface is due to missing error detecting possibilities prone to errors. Therefor the development of appropriate transfer protocols was necessary.

From Version 3.0, **IH**-Com also supports two of these transmission protocols in addition to the serial interface: the ASTM 1381 international standard protocol of medical technology and the Maestro transmission protocol developed by DiaMed.

Connections via TCP/IP are safer than connections via serial interface, as TCP ensures the file transmission with error detection and acknowledgement for receiving of single data packages. For transfer of large amounts of data it is nevertheless useful to know if the remote station received the data completely. This security is only possible with an additional protocol. ASTM 1381 is convenient but too extensive. For this reason a simple option - Simple Protocol - was developed. Usage of ASTM 1381 via TCP/IP takes place for older serial connections for compatibility reasons.

5.2 The ASTM 1381 Protocol

The ASTM 1381 protocol belongs to a series of protocols which are developed and published by the American Society for Testing and Materials (renamed in 2001 to ASTM International)[1].

ASTM 1381 like ASTM 1394 handles the communication between medical devices and their host systems as well as the communication between medical analysis and diagnostic support software products and the host systems.

The current version of ASTM 1381 has been in place since 1995. (ASTM E 1381 – 95).

As most medical devices in the past were connected to corresponding evaluation computers or LIS systems using a serial cable, it was necessary to guarantee the reliability and correctness of the transferred data. As the serial interface was too unreliable for this and no error detection available, a method was created using the ASTM 1381 protocol which[2] on the one hand converts the communication via the serial interface into a synchronous process and also implements error detection and correction using a handshake process.

The method assumes that sender and receiver exchange their data alternately.

5.2.1 ASTM 1381 Data Transfer Process

If one of the two stations has data to send, it is first checked whether a data transfer is currently taking place. If this is the case, the station waits until the current transfer has been completed.

If the line is free, a control character[3], a so-called Enquire, is first set at the receiver, that data is to be sent. If the remote station signals readiness data can be received.

The receiver can respond to this positively (according to the standard with the control character ACK) or negatively (NAK).

If the response was negative, an ENQ (Enquire) is transmitted again after a defined time (10 seconds according to the standard). This is repeated until a positive response is received or a definable number of attempts has been reached; the latter is considered a transmission error and the attempt to transfer the data is aborted.

If the enquiry has been responded to positively (i.e. the receiver is ready to accept the data), the user data are then broken down into individual (ASTM 1381) packets and transferred packet by packet.

Such a transmission packet has the following structure:

Start control character (standard: STX)	Packet number (single digit)	User data (max. 247 bytes)	Control characters for end of block or text (ETB / ETX)	Checksum (two-digit)	Line break (two-digit)

A data packet consists of a stat control character followed by a packet number which starts with 0 and after the packet number 9 starts counting again from 0. Then the actual user data which are permitted to be in a packet up to 247 bytes long.

1. *ASTM International can be found on the World Wide Web at www.astm.org*
2. *See glossary: “ Handshake (process)” on page 10*
3. *Control characters are special characters in the computer's character set which are not assigned visually to a letter, a digit, a special character or a graphic but have a special function such as, e.g. the control character 9 which corresponds to the Tab key, or the control character 13 which corresponds to the Enter key. The most important control characters can be found in the positions 0 to 29 in the character set. A list of these control characters and their input form in IH-Com can be found in the Appendix. See “7.2 Control characters used by IH-Com” on page 66*

If the current packet does not yet contain all data and there are other packets to send, the control character for the end of a text block is sent next whereby the receiver knows that other packets will follow.

On the other hand, if the packet does contain the last of the user data to be transmitted, the control character for the end of text is sent which signals to the receiver that all packets for the current user data have been transferred.

The receiver can then compile the actual user data from all packets including the last one.

In doing so, each packet is acknowledged after the receipt. Either with an ACK if everything is OK with the packet or with a NAK if there is a reason for the receiver to assume that the data in the packet are not OK (e.g. if the packet numbers are not sequential or the checksum does not match or one of the control characters is wrong).

If the last packet has been transferred and positively acknowledged by the receiver (ACK), an EOT is transmitted by the sender which puts the sender and receiver into a neutral state again.

The following diagrams should explain the communication from the perspective of the sender and of the receiver of the data.

5.2.1.1 ASTM 1381 sending data diagram

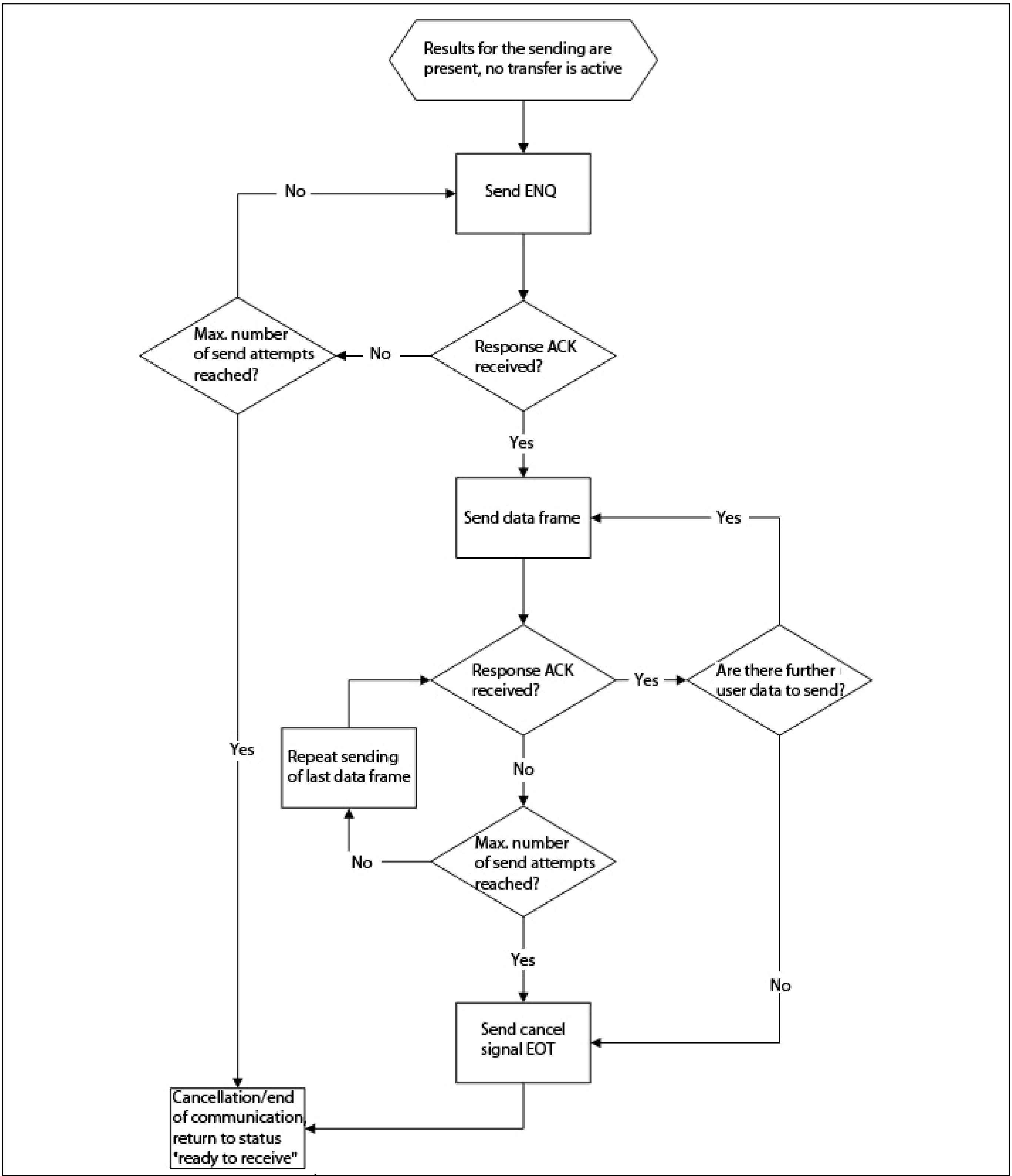

Fig. 5-1: ASTM 1381 sending data[1]

1. *A frame or data frame corresponds to a data packet as described above*

5.2.1.2 ASTM 1381 receiving data diagram

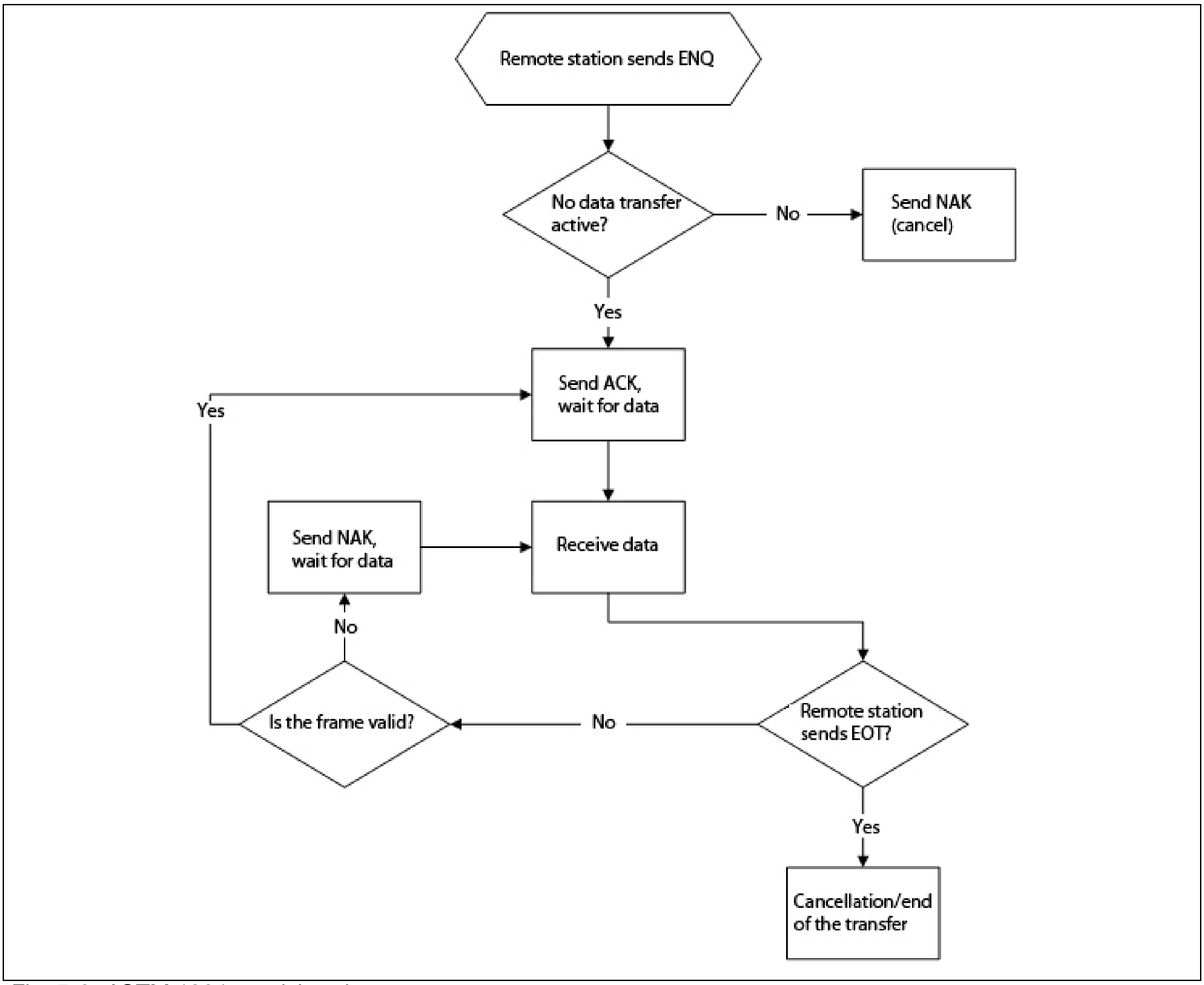

Fig. 5-2: ASTM 1381 receiving data

In order to ensure the correctness of data, these are secured by a checksum with respect to the number of bytes (calculated according to the ASTM 1381) and the correct control character is checked.

5.2.2 Activating ASTM 1381 in IH-Com

Beside the requested data content the transfer protocol **ASTM 1381** must be selected under ***Protocol*** (1). Additionally the physical connection according export and import path must be selected (3). Impossible combinations can not be selected. In that case an exclamation mark is indicated beside the entry and the OK button is inactive. If the exclamation mark is touched with the cursor, a tooltip with detailed information will be displayed.

For proper transfer of all language specific characters the appropriate ***Transfer Code Page*** must be selected under ***Code page*** (2). It will be initially selected automatically in **IH**-Com according to the system language. It is imperative that the system language corresponds with the remote station, the LIS.

In order to activate the transfer protocol ASTM 1381 the dialogue for the host interface must be changed.

Refer to the **IH**-*Com Service Manual.*

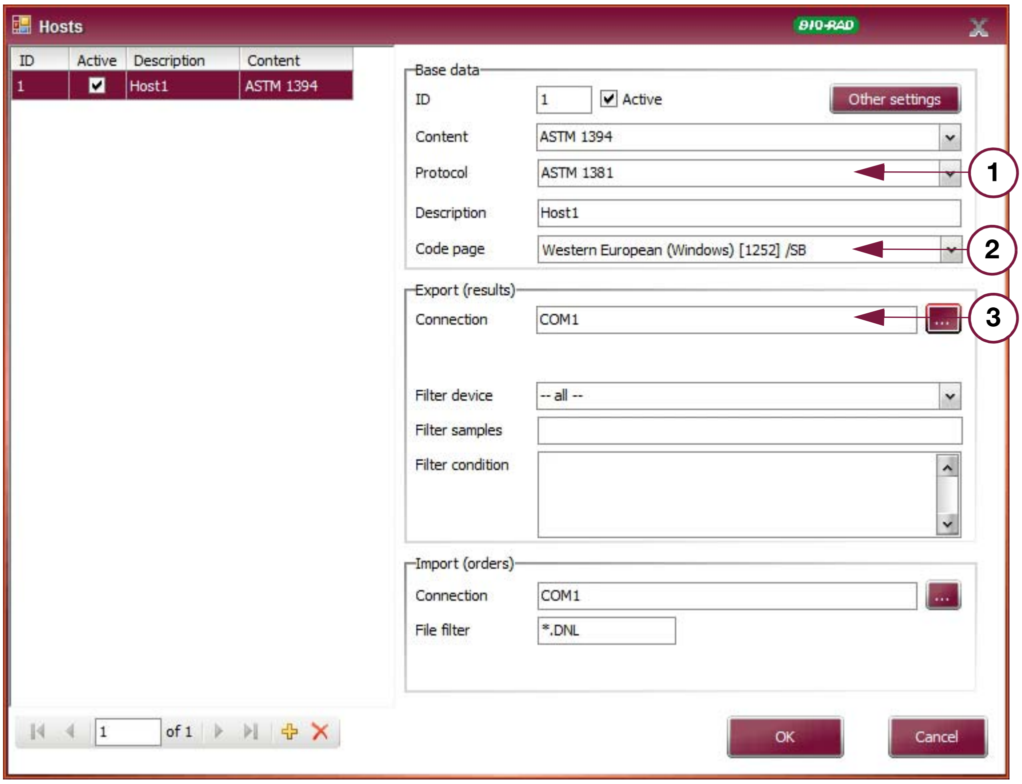

Fig. 5-3 *Activating ASTM 1381 in* **IH**-*Com*

1	Protocol
2	Code page
3	Connection

In order to activate the transfer protocol ASTM 1381, confirm the selected settings and restart the **IH**-Com Communicator.

5.2.3 Configuration of ASTM 1381 in IH-Com

The defined standard values for using ASTM 1381 protocol in **IH**-Com correspond to the standard ASTM E 1381-95. Therefore they may not be changed individually. The function beyond the standard definitions is not ensured.

Overview of the used control characters (according the defined standard):

Definition	Control character / Value
Control character for positive confirmation after an Enquiry or a proper received data package.	<ACK>
Control character, which requests the remote station, if it is ready for receiving data to be sent.	<ENQ>
In case of errors this control character is the sign for transfer termination. After sending the last data package and its positive confirmation it is the sign for end of communication.	<EOT>
Control character of a single package, which labels this package as intermediate package that is followed by other data packages.	<ETB>
Control character of a single package, which labels the package as end package not followed by other packages.	<ETX>
Control character sent as a negative response to an Enquire or an error message for a false data package.	<NAK>
Control character designating the beginning of a data package.	<STX>
Interval in seconds between two ENQ-requests, in case of not confirmed with ACK (ready for receiving data).	10
Number of sending trials for the ENQ. If this number is reached, an error will be indicated for not sending the data.	5
Number of trials, how often a package is resent after a negative response. If this number is reached an EOT is sent and the communication stopped.	5

In order to check the data transfer during installation or if problems occur, an additional log can be activated in **IH**-Com.

A Select the option ***Log LIS debug messages*** via menu ***Configuration / Common*** from tab ***Database / Backup*** in **IH**-Com Client.

B After restarting **IH**-Com Communicator a file **Raw_*.log** is created daily in directory ***C:\ProgramData\IH-Com\Log*** of the **IH**-Com Communicator.

Data transfer inclusive control characters is recorded in this file.

Example:

```
2015-05-22 15:52:25.3787   rec     <ENQ>
2015-05-22 15:52:27.6250   snd     <ACK>
2015-05-22 15:52:27.8278   rec

<STX>1H|\^&|||BECOMLAB||||||||1|20150521224144<CR><ETX>DC<CR><LF>

2015-05-22 15:52:28.0462   snd     <ACK>
2015-05-22 15:52:28.3114   rec

<STX>2P|1||21155153||Patient^Kathye||19320728|F|||||||||||||||||||||||||<CR><ETX>5D<
CR><LF>

2015-05-22 15:52:28.3425   snd     <ACK>
2015-05-22 15:52:28.5921   rec

<STX>3O|1||21155153^^^\^^^|^^^3|R|20150521181216|20150520214520||||N||||||||||<CR><E
TX>83<CR><LF>

2015-05-22 15:52:28.5921   snd     <ACK>
2015-05-22 15:52:28.7793   rec     <STX>4L|1|N<ETX>FA<CR><LF>
2015-05-22 15:52:28.7793   snd     <ACK>
2015-05-22 15:52:28.9821   rec     <EOT>
```

Fig. 5-4 *Log File*

After solving the problem it is recommended to off this log and delete the log files manually from the directory.

5.3 The Maestro Transfer Protocol Emulation

The Maestro Transfer Protocol also uses control characters to mark the packet beginning and the packet end. It also has a checksum process whereby errors in the user data can also be detected with this protocol.

However, in contrast to ASTM1381, the Maestro Transfer Protocol dispenses with the breakdown of the data into separate packets. The negotiation of the data connection is also dispensed with. In comparison with ASTM1381, the Maestro Transfer Protocol is thus extremely simplified.

5.3.1 Data transfer process using Maestro Transfer Protocol

The Maestro Transfer Protocol does not initiate the communication with an Enquire but takes all of the user data to be sent and provides these with control characters at the start and at the end (frame start character and frame end character) and a checksum before the end control character which always has a length of 2 digits.

It is then checked whether a data transfer is currently taking place. If this is the case, there is a wait, otherwise the packet is sent in one piece.

The receiver then checks the packet and sends a positive or negative receipt confirmation.

If a negative receipt confirmation is transmitted as response, the sender decides whether to send this data packet again. There is also no "end of transmission" control character.

The Maestro Transfer Protocol thus only knows three packet types:

- Packet with the user data:

STX	<User data>	Checksum	ETX

- Positive receipt confirmation:

STX	OK	Checksum	ETX

- Negative receipt confirmation:

STX	ER	Checksum	ETX

Therefore, a packet with the user data is always sent which is acknowledged with a positive or a negative receipt confirmation.

The confirmation are common payload packages containing either the text **OK** or **ER**.

The checksum is also determined in a simplified way. The decimal values of the individual user data characters according to the ASCII table[1] are summed for this. The resulting number is converted to a hexadecimal number[2] and the last two digits (or if the number only has one digit, a leading zero is added) are used as checksum. <CR> and <LF> are included in the calculation of the check sum.

Example for a data package:

`<STX>OK<CR><LF><ETX>: O (79)  K (75) <CR> (13) <LF> (10)`

79+75+13+10 = 177

177 is hexadecimal B1. The check sum is B1.

The diagrams for the sending and receiving of data with the Maestro Transfer Protocol are shown below.

1. *The ASCII table and the associated decimal values can be found in the Appendix, chapter "7.1 ASCII Table" on page 57*
2. *See glossary: " Hexadecimal Number" on page 10*

5.3.1.1 Maestro Transfer Protocol - Send data diagram

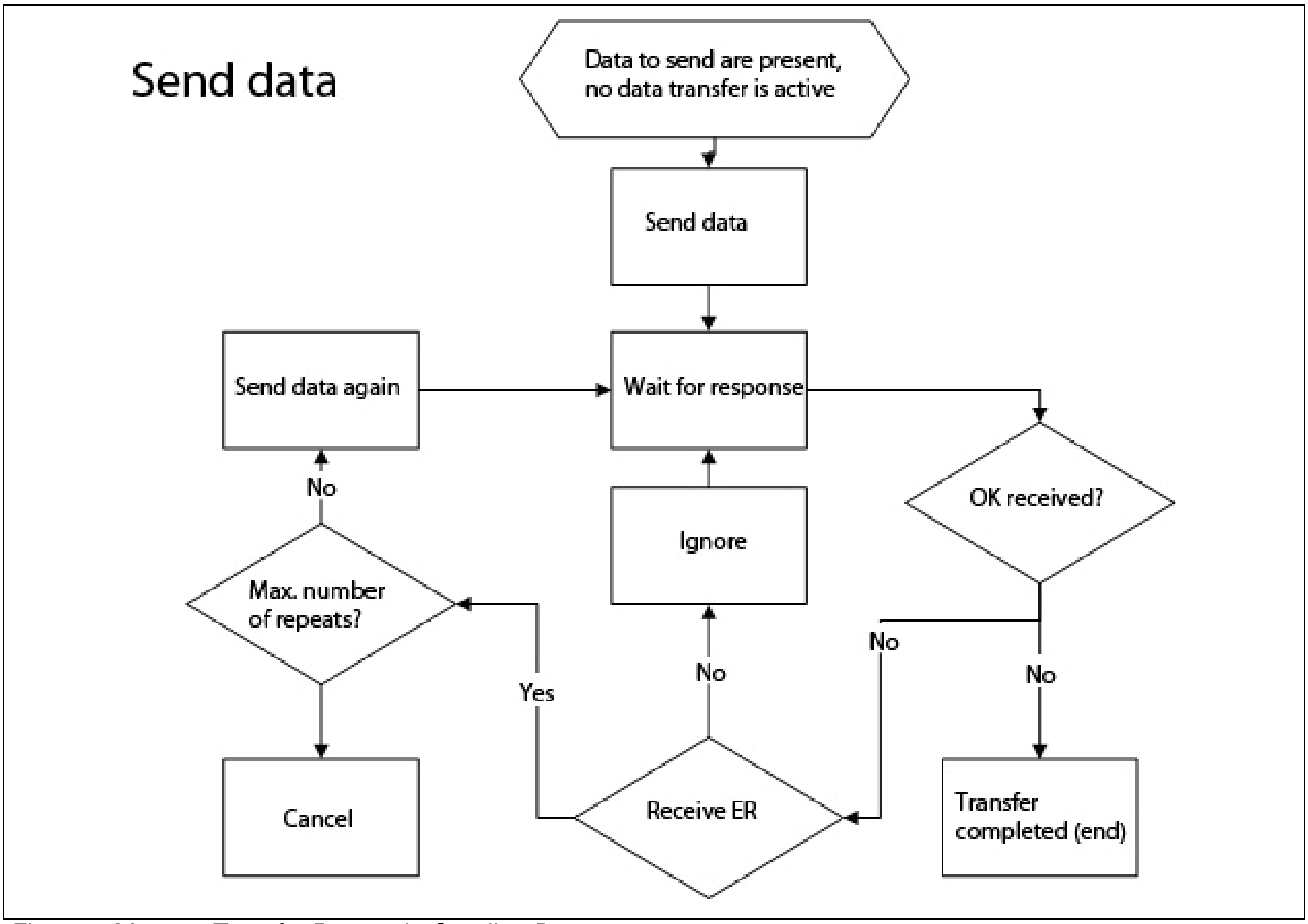

Fig. 5-5: Maestro Transfer Protocol - Sending Data

5.3.1.2 Maestro Transfer Protocol - Receiving Data diagram

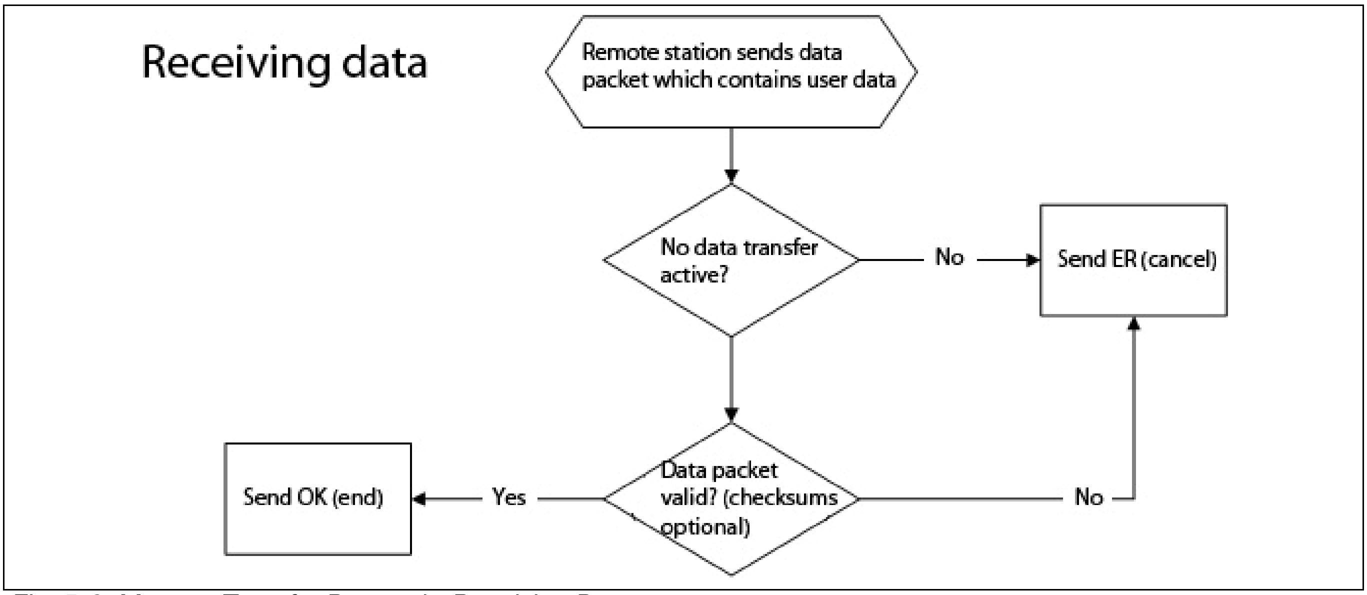

Fig. 5-6: Maestro Transfer Protocol - Receiving Data

5.3.2 Activating Maestro Transfer Protocol Emulation in IH-Com

In order to activate the ***Maestro Transfer Protocol*** it must be switched to the dialogue of host interfaces.

Refer to the ***IH****-Com Service Manual.*

There additionally to the requested data content (***Content***(1)) the transfer protocol ***Maestro (emulation)*** (2) must be selected under ***Protocol***. As well the physical connection according to export and import path must be selected (4).

For proper transfer of all language specific characters the appropriate ***Transfer Code Page*** must be selected under ***Code page*** (3). In **IH**-Com it is initially selected automatically according the system language. It is imperative that it corresponds with the remote station, the LIS. Maestro Transfer Protocol supports only ***Single Byte Code Pages, Unicode (UTF-32)*** and ***Japanese (Shift-JIS).***

Impossible combinations can not be selected. In that case an exclamation mark is indicated beside the entry and the **OK** button is inactive. Not recommended combinations are also indicated with an exclamation mark beside the entry. But it is possible to set such combinations. In that case the **OK** button is active. If the exclamation mark is touched with the cursor, a tooltip with detailed information will be displayed.

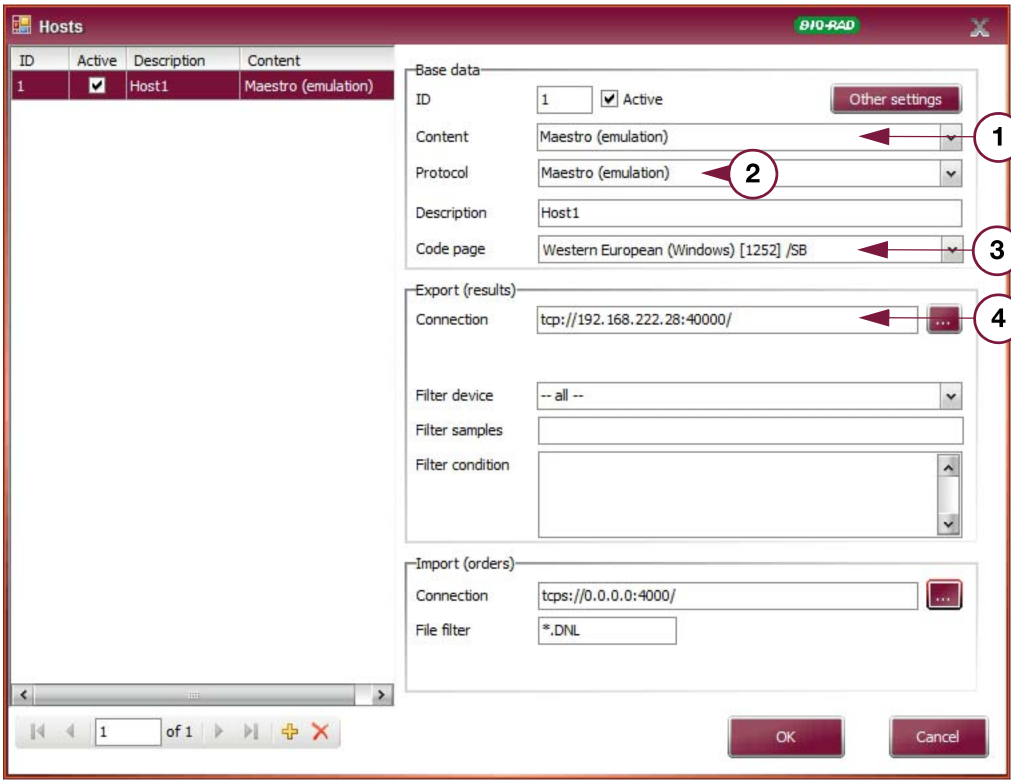

Fig. 5-7: Activating Maestro Transfer Protocol in ***IH****-Com*

1	Content
2	Protocol
3	Code page
4	Connection

In order to activate the ***Maestro Transfer Protocol***, confirm the selected settings and restart the **IH**-Com Communicator.

5.3.3 Configuration of the Maestro Transfer Protocol Emulation in IH-Com

The most important parameters for the use of the Maestro Transfer Protocol can be individually configured in the system options. System options can be found in the dialogue of host connections via button ***Other settings***.

The defaults correspond to the information specified in the IH-Com Host Connection Manual. If these defaults are changed, the user moves outside the standard which should only happen if it is really necessary. Operation outside the standard cannot be guaranteed.

All settings for the Maestro Transfer Protocol can be found in the ***System options*** in group ***Protocol***.

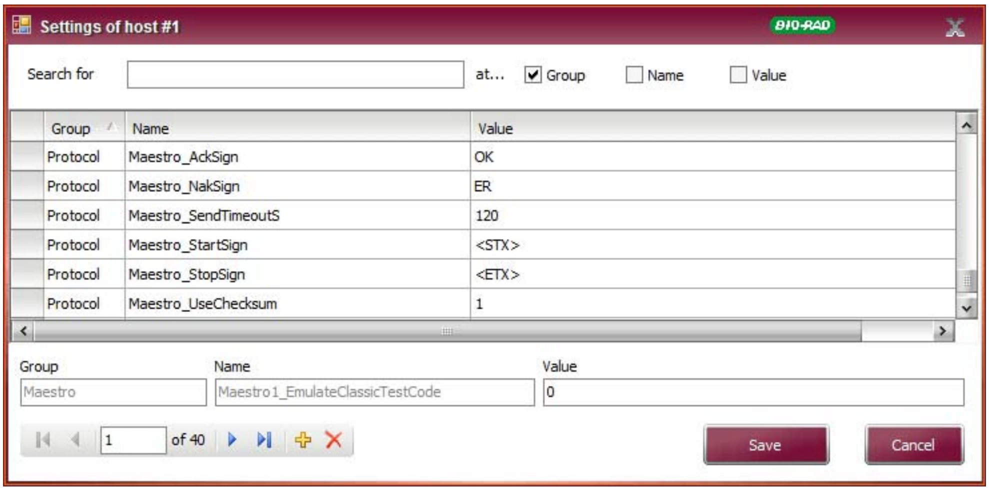

Fig. 5-8: System options for Maestro Transfer Protocol in ***IH****-Com*

System options Maestro Transfer Protocol

Following settings can be done:

Option	Definition	Standard value
Maestro_AckSign	Text which informs that the last data package was proper	OK
Maestro_NakSign	Text which informs that the last data package contains an error and is discarded.	ER
Maestro_SendTimeoutS	Amount in seconds **IH**-Com tries to sent data. If no receiving confirmation takes place during this time, **IH**-Com stops sending trials with ***Timeout***.	120
Maestro_StartSign	Control character labelling beginning of a data package.	<STX>
Maestro_StopSign	Control character labelling ending of a data package.	<ETX>
Maestro_UseChecksum	If the value is 1 check sums of received data packages are checked. If non corresponding the package will be reported as error. If the value is 0 check sums are ignored.	1

Signs for OK and ER are common text and not control characters.

It is technically possible to enter control characters of ASCII codes under 32 via keyboard but these settings are not visible in the system options because these control characters are not presentable.

Therefore IH-Com supports the entry of control characters via so-called tags. They are representatives of a control character being transformed internally to the not presentable control character.

Text <STX> is transformed internally to the control character with ASCII value 2.

Find a table of control character in chapter “7 Appendix” on page 57.

5.4 IH-Com Simple Protocol

5.4.1 Overview

The **IH**-Com Simple Protocol is a simplified subset of ASTM E1381 protocol definition.

The data link layer has procedures for link connection and release, delimiting and synchronism, sequential control, error detection, and error recovery. It uses a character-oriented protocol to send messages between directly connected systems. Some restrictions are placed on the characters which can appear in the message content.

The data link mode of operation is one-way transfer of information with alternate supervision. Information flows in one direction at a time. Replies occur after information is sent, never at the same time. It is a simplex stop-and-wait protocol.

Protocol is controlled by transmission control characters. The three states of this protocol are idle, sending and receiving.

5.4.2 Idle state

A system which is not receiving and does not have information to send is in ***Idle*** state. It acts as a receiver, waiting for the other system.

5.4.3 Sending state

5.4.3.1 Start sending

A system which has information to send and is in ***Idle*** state (sender) starts transfer of data encapsulated with start code and stop code (message).

The message structure is illustrated as follows:

`<STX>DATA<ETX>`

Where:

<STX> = Start code (start of message)

`DATA` = Message

<ETX> = Stop code (end of message)

This system is in ***Sending*** state.

5.4.3.2 Response

After completion of transmission, sender is awaiting <ACK> or <NAK> as a response. All other characters will be ignored.

A reply of <ACK> signifies the last message was received successfully and the receiver is going to ***Idle*** state. This will bring sender back to “Idle” state too.

A reply of <NAK> signifies the last message was not successfully received and the receiver is prepared to receive the frame again. Sender will do up to 4 further retries to send previous message. After receiving <ACK> or after last unsuccessful retry sender and receiver reach “Idle” state.

5.4.3.3 Timeout

If there is no response within 30 seconds a timeout occurs and sender goes back to ***Idle*** mode. Sender should wait at minimum 15 seconds before doing next attempt.

5.4.3.4 Checksum (optional)

Optionally it is possible to add a checksum to message as follows:

`<STX>DATA[C1][C2]<ETX>`

Where:

<STX> = Start code (start of message)

DATA = Message

[C1] = most significant character of checksum (0 to 9 and A to F)

[C2] = least significant character of checksum (0 to 9 and A to F)

<ETX> = Stop code (end of message)

This checksum permits the receiver to detect a defective message.

Enabling the checksum feature should be same at both sites.

The checksum is initialized to zero with the start code (<STX>) character. Each character in the message text is added to the checksum (modulo 256). The computation for the check-sum does not include the start code (<STX>), the checksum characters or the trailing stop code (<ETX>).

The checksum is an integer represented by eight bits, it can be considered as two groups of four bits. The groups of four bits are converted to the ASCII characters of the hexadecimal representation. The two ASCII characters are transmitted as the checksum, with the most significant character first.

For example, a checksum of 122 can be represented as **01111010** in binary or **7A** in hexadecimal. The checksum is transmitted as the ASCII character **7** followed by the character **A**.

In spite of ability of using multi-byte character sets it is required to compute checksum before coding and after decoding data of transmission stream.

5.4.4 Receiving state

5.4.4.1 Receiving

A system which is in ***Idle*** state and receives <STX> (receiver) starts ***Receiving*** state.

5.4.4.2 Response

After completion of transmission (receiving <ETX>), receiver will reply <ACK> or <NAK> as a response. This will bring receiver back to ***Idle*** state.

All characters before <STX> will be ignored.

A reply of <ACK> signifies the last message was received and processed successfully.

A reply of <NAK> signifies the last message was not successfully received or processed.

5.4.4.3 Checksum

Optionally it is possible to await a checksum at message.

*See chapter "**5.4.3.4** Checksum (optional)" on page 48.*

Enabling the checksum feature should be same at both sites.

5.4.5 Restrictions

5.4.5.1 Restricted message characters

The data link protocol is designed for sending character based message text. Restrictions are placed on which characters may appear in the message text. The restrictions make it simpler for senders and receivers to recognize replies and frame delimiters.

None of the transmission control characters may appear in message text. The restricted characters are: start code (<STX>), stop code (<ETX>), positive acknowledge (<ACK>) and negative acknowledge (<NAK>).

5.4.5.2 Message coding

Compared to ASTM E 1381 this protocol is not restricted to seven-bit ASCII characters.

Using of any single-byte code page is possible if both sites use same one.

Only following multi-byte code pages is supported: UTF-32.

5.4.6 Activating Simple Protocol in IH-Com

In order to activate the ***Simple Protocol*** it must be switched to the dialogue of host interfaces.

Refer to the **IH**-*Com Service Manual.*

There additionally to the requested data content (***Content***(1)) the transfer protocol ***Simple (TCP/IP only)*** (2) must be selected under ***Protocol***. As well the physical connection according to export and import path must be selected (4). For ***Simple (TCP/IP only)*** only TCP/IP is available.

For proper transfer of all language specific characters the appropriate ***Transfer Code Page*** must be selected under ***Code page*** (3). In **IH**-Com it is initially selected automatically according the system language. It is imperative that it corresponds with the remote station, the LIS.

Impossible combinations can not be selected. In that case an exclamation mark is indicated beside the entry and the OK button is inactive. Not recommended combinations are also indicated with an exclamation mark beside the entry. But it is possible to set such combinations. In that case the **OK** button is active. If the exclamation mark is touched with the cursor, a tooltip with detailed information will be displayed.

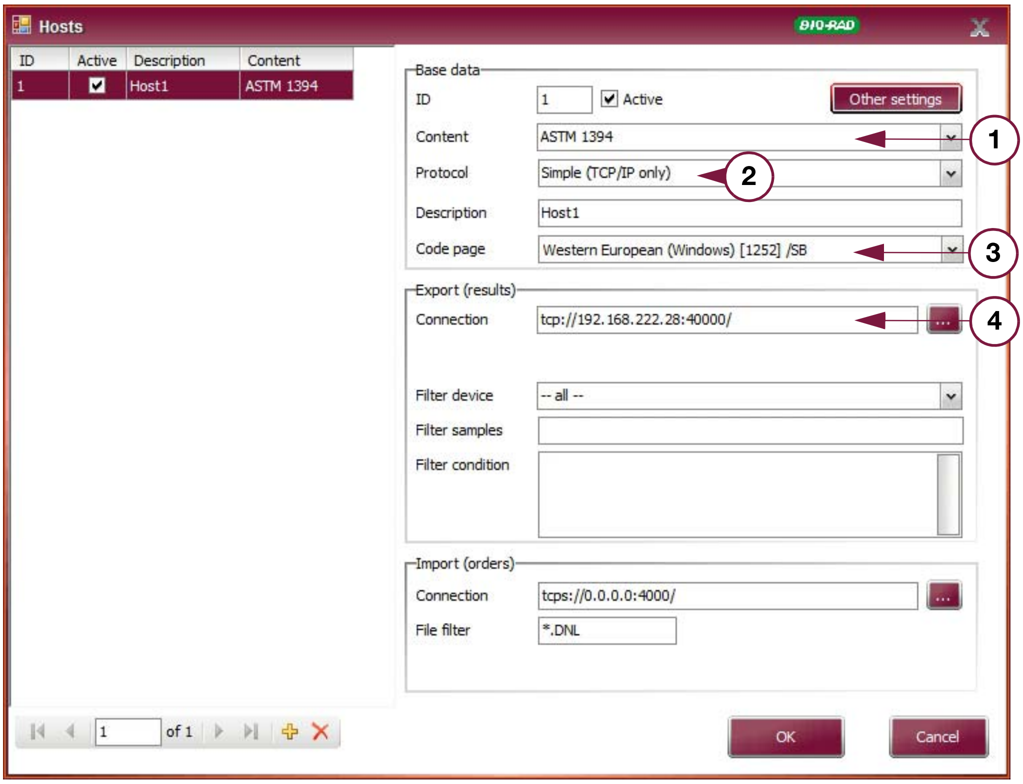

Fig. 5-9: Activating Simple Protocol in **IH**-*Com*

1	Content
2	Protocol
3	Code page
4	Connection

In order to activate ***Simple Protocol***, confirm the selected settings and restart the **IH**-Com Communicator.

5.4.7 Configuration of Simple Protocol in IH-Com

In the system options all important values for using ***Simple Protocol*** can be set individually. These system options can be found in the dialogue for host interfaces via ***Other settings*** button.

All setting for ***Simple Protocol*** are defined in the system options of group ***Protocol***.

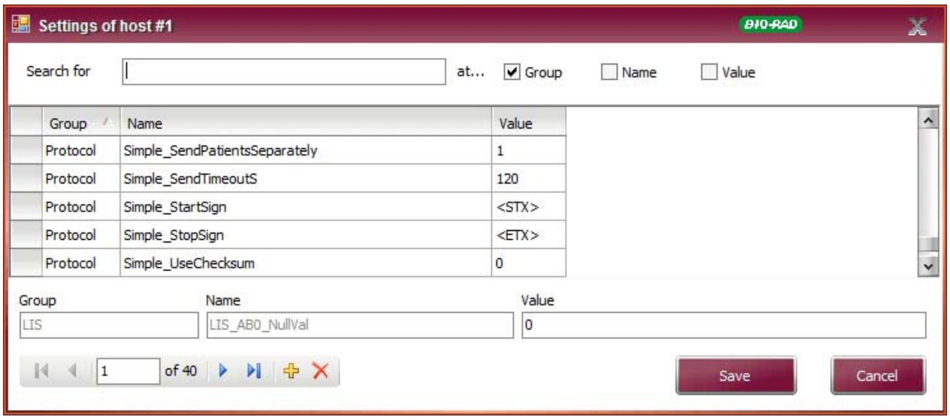

Fig. 5-10 *System options for Simple Protocol in* **IH**-*Com*

Following settings can be done:

Option	Definition	Standard value
Simple_SendPatientsSeparatly	If this value is 1, each sample or each patient is transferred as a separate data package. If this value is 0, all samples to be sent are transferred in one data package.	1
Simple_SendTimeoutS	Number of seconds **IH**-Com tries to send data. If no receiving confirmation from the remote station is sent during this time, **IH**-Com stops sending trials with ***Timeout***	120
Simple_StartSign	Control character designating beginning of a data package.	<STX>
Simple_StopSign	Control character designating ending of a data package.	<ETX>
Simple_UseChecksum	If the value is 1 check sums of received data packages are checked. If non corresponding the package will be reported as error. If the value is 0 check sums are ignored.	0

Table: System Options Simple Protocol and default values

It is technically possible to enter control characters of ASCII codes under 32 via keyboard but these settings are not visible in the system options because these control characters are not presentable.

Therefore IH-Com supports the entry of control characters via so-called tags. They are representatives of a control character being transformed internally to the not presentable control character.

Text <STX> is transformed internally to the control character with ASCII value 2.

Find a table of control character in chapter "7 Appendix" on page 57.

6 CONFIGURATION OF THE PHYSICAL CONNECTION

6.1 Configuration of the serial port

In system options the most important values for using the physical connection via serial interface can be adjusted individually. Navigate via button ***Other settings*** in dialogue host interfaces to the system options.

It is important that the configuration in **IH**-Com is identical to that used by the receiver.

From the hardware aspect, the serial ports are in fact already configured in Windows using the Control Panel. These settings are overwritten when opening the ports with those from IH-Com.

It must also be noted that the settings apply to all serial ports. If more than one serial port on a computer is used, the settings apply to all connections.

All settings for the serial port are in the group ***Serial***.

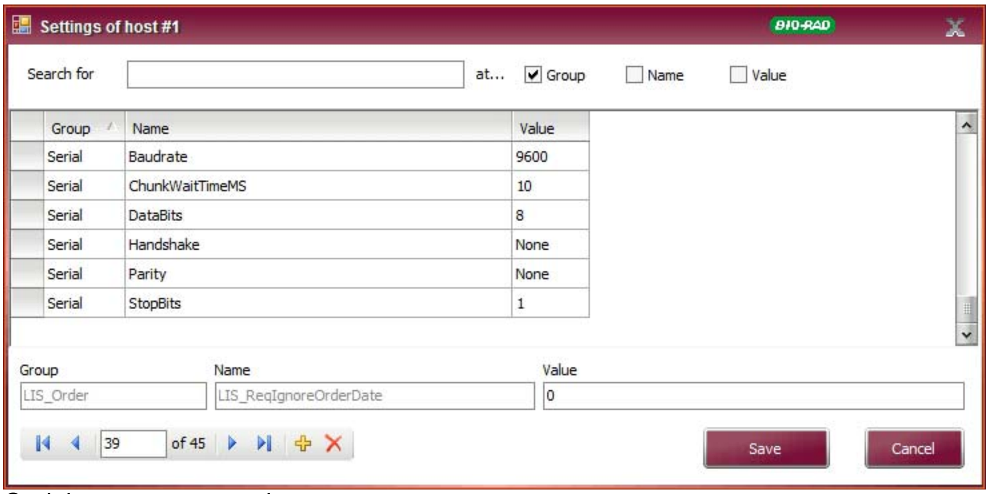

Fig. 6-1: Serial port system options

6.2 System options table for the serial port

The following settings can be done:

Option	Definition	Standard setting	Possible values
Baudrate	Data rate for communication *For detailed information refer to chapter “**3.3** Technical realization of the serial transfer” on page 19*	9600	300 1200 2400 4800 9600 14400 19200 28800 38400 57600 76800 115200 230400 250000
ChunkWaitTimeTS	Waiting time in seconds in between two single data blocks, if no protocol is used	10	each integer > 0
DataBits	Number of data bits per package	8	each integer > 0
Handshake	Handshake of serial connection (only ***None*** is still supported)	None	None
Parity	Parity check *For detailed information refer to chapter “**3.4** Error correction methods of serial data transfer” on page 23*	None	None Even Odd Mark Space
StopBits	Number of stop bits	1	0 1 1.5 2

The defaults of the serial port correspond exactly to the specifications of the ASTM1381 standard.

The entries of the ***Baud Rate, Parity*** and ***Stop Bits*** must be made in exactly the text form as shown in the above table.

6.3 Configuration of the connection via TCP/IP

The physical connection via TCP/IP can be adjusted individually. Only one system option is available. Navigate via button ***Other settings*** in dialogue host interfaces in group TCP.

Following setting can be done:

Option	Definition	Standard value
AutoReConnect	If the value is 1, **IH**-Com tries automatically after a time out to connect to the remote station. If the value is 0, a restart must take place after a time out.	1

PAGE INTENTIONALLY LEFT BLANK

7 APPENDIX

Chapter Overview

This chapter contains additional and helpful information for the LIS Connection Manual.

7.1 ASCII Table

The ASCII table is a character set table. This means it shows which internal values (decimal numbers) are assigned to the respective characters.

The original ASCII character set only included 128 characters of which the first 32 characters are control characters.

See section "7.2 Control characters used by IH-Com" on page 66

The extended ASCII character set includes 255 characters. This is shown below for the sake of completeness.

Decimal value	ASCII character	Display	Comment/description
1	NUL	-	Control character
2	SOH	-	Control character
3	STX	-	Control character
4	ETX	-	Control character
5	ENQ	-	Control character
6	ACK	-	Control character
7	BEL	-	Control character
8	BS	-	Control character
9	TAB	-	Control character
10	LF	-	Control character
11	VT	-	Control character
12	FF	-	Control character
13	CR	-	Control character
14	SO	-	Control character
15	SI	-	Control character
16	DLE	-	Control character
17	DC1	-	Control character
18	DC2	-	Control character
19	DC3	-	Control character
20	DC4	-	Control character
21	NAK	-	Control character
22	SYN	-	Control character

Decimal value	ASCII character	Display	Comment/description
23	ETB	-	Control character
24	CAN	-	Control character
25	EM	-	Control character
26	SUB	-	Control character
27	ESC	-	Control character
28	FS	-	Control character
29	GS	-	Control character
30	RS	-	Control character
31	US	-	Control character
32			Space
33	!	!	Punctuation marks and special characters
34	"	"	Punctuation marks and special characters
35	#	#	Punctuation marks and special characters
36	$	$	Punctuation marks and special characters
37	%	%	Punctuation marks and special characters
38	&	&	Punctuation marks and special characters
39	'	'	Punctuation marks and special characters
40	(	(	Punctuation marks and special characters
41	)	)	Punctuation marks and special characters
42	*	*	Punctuation marks and special characters
43	+	+	Punctuation marks and special characters
44	,	,	Punctuation marks and special characters
45	-	-	Punctuation marks and special characters
46	.	.	Punctuation marks and special characters
47	/	/	Punctuation marks and special characters
48	0	0	Numbers
49	1	1	Numbers
50	2	2	Numbers
51	3	3	Numbers
52	4	4	Numbers
53	5	5	Numbers
54	6	6	Numbers
55	7	7	Numbers

Decimal value	ASCII character	Display	Comment/description
56	8	8	Numbers
57	9	9	Numbers
58	:	:	Punctuation marks and special characters
59	;	;	Punctuation marks and special characters
60	<	<	Punctuation marks and special characters
61	=	=	Punctuation marks and special characters
62	>	>	Punctuation marks and special characters
63	?	?	Punctuation marks and special characters
64	@	@	Punctuation marks and special characters
65	A	A	Uppercase letters
66	B	B	Uppercase letters
67	C	C	Uppercase letters
68	D	D	Uppercase letters
69	E	E	Uppercase letters
70	F	F	Uppercase letters
71	G	G	Uppercase letters
72	H	H	Uppercase letters
73	I	I	Uppercase letters
74	J	J	Uppercase letters
75	K	K	Uppercase letters
76	L	L	Uppercase letters
77	M	M	Uppercase letters
78	N	N	Uppercase letters
79	O	O	Uppercase letters
80	P	P	Uppercase letters
81	Q	Q	Uppercase letters
82	R	R	Uppercase letters
83	S	S	Uppercase letters
84	T	T	Uppercase letters
85	U	U	Uppercase letters
86	V	V	Uppercase letters
87	W	W	Uppercase letters
88	X	X	Uppercase letters

Decimal value	ASCII character	Display	Comment/description
89	Y	Y	Uppercase letters
90	Z	Z	Uppercase letters
91	[	[	Punctuation marks and special characters
92	\	\	Punctuation marks and special characters
93	]	]	Punctuation marks and special characters
94	^	^	Punctuation marks and special characters
95	_	_	Punctuation marks and special characters
96	`	`	Punctuation marks and special characters
97	a	a	Lowercase letters
98	b	b	Lowercase letters
99	c	c	Lowercase letters
100	d	d	Lowercase letters
101	e	e	Lowercase letters
102	f	f	Lowercase letters
103	g	g	Lowercase letters
104	h	h	Lowercase letters
105	i	i	Lowercase letters
106	j	j	Lowercase letters
107	k	k	Lowercase letters
108	l	l	Lowercase letters
109	m	m	Lowercase letters
110	n	n	Lowercase letters
111	o	o	Lowercase letters
112	p	p	Lowercase letters
113	q	q	Lowercase letters
114	r	r	Lowercase letters
115	s	s	Lowercase letters
116	t	t	Lowercase letters
117	u	u	Lowercase letters
118	v	v	Lowercase letters
119	w	w	Lowercase letters
120	x	x	Lowercase letters
121	y	y	Lowercase letters

Decimal value	ASCII character	Display	Comment/description
122	z	z	Lowercase letters
123	{	{	Punctuation marks and special characters
124	|	|	Punctuation marks and special characters
125	}	}	Punctuation marks and special characters
126	~	~	Tilde
127	DEL	□	Delete character
128	€	€	Start of extended characters
129	□	□	Extended characters
130	‚	‚	Extended characters
131	ƒ	ƒ	Extended characters
132	„	„	Extended characters
133	…	…	Extended characters
134	†	†	Extended characters
135	‡	‡	Extended characters
136	ˆ	ˆ	Extended characters
137	‰	‰	Extended characters
138	Š	Š	Extended characters
139	‹	‹	Extended characters
140	Œ	Œ	Extended characters
141	□	□	Extended characters
142	Ž	Ž	Extended characters
143	□	□	Extended characters
144	□	□	Extended characters
145	‘	‘	Extended characters
146	’	’	Extended characters
147	“	“	Extended characters
148	”	”	Extended characters
149	•	•	Extended characters
150	–	–	Extended characters
151	—	—	Extended characters
152	˜	˜	Extended characters
153	™	™	Extended characters
154	š	š	Extended characters

Decimal value	ASCII character	Display	Comment/description
155	›	›	Extended characters
156	oe	oe	Extended characters
157	□	□	Extended characters
158	ž	ž	Extended characters
159	Ÿ	Ÿ	Extended characters
160			Extended characters
161	¡	¡	Extended characters
162	¢	¢	Extended characters
163	£	£	Extended characters
164	¤	¤	Extended characters
165	¥	¥	Extended characters
166	¦	¦	Extended characters
167	§	§	Extended characters
168	¨	¨	Extended characters
169	©	©	Extended characters
170	ª	ª	Extended characters
171	«	«	Extended characters
172	¬	¬	Extended characters
173			Extended characters
174	®	®	Extended characters
175	¯	¯	Extended characters
176	°	°	Extended characters
177	±	±	Extended characters
178	²	²	Extended characters
179	³	³	Extended characters
180	´	´	Extended characters
181	ì	ì	Extended characters
182	¶	¶	Extended characters
183	·	·	Extended characters
184	¸	¸	Extended characters
185	¹	¹	Extended characters
186	º	º	Extended characters
187	»	»	Extended characters

Decimal value	ASCII character	Display	Comment/description
188	¼	¼	Extended characters
189	½	½	Extended characters
190	¾	¾	Extended characters
191	¿	¿	Extended characters
192	À	À	Extended characters
193	Á	Á	Extended characters
194	Â	Â	Extended characters
195	Ã	Ã	Extended characters
196	Ä	Ä	Extended characters
197	Å	Å	Extended characters
198	Æ	Æ	Extended characters
199	Ç	Ç	Extended characters
200	È	È	Extended characters
201	É	É	Extended characters
202	Ê	Ê	Extended characters
203	Ë	Ë	Extended characters
204	Ì	Ì	Extended characters
205	Í	Í	Extended characters
206	Î	Î	Extended characters
207	Ï	Ï	Extended characters
208	Ð	Ð	Extended characters
209	Ñ	Ñ	Extended characters
210	Ò	Ò	Extended characters
211	Ó	Ó	Extended characters
212	Ô	Ô	Extended characters
213	Õ	Õ	Extended characters
214	Ö	Ö	Extended characters
215	×	×	Extended characters
216	Ø	Ø	Extended characters
217	Ù	Ù	Extended characters
218	Ú	Ú	Extended characters
219	Û	Û	Extended characters
220	Ü	Ü	Extended characters

Decimal value	ASCII character	Display	Comment/description
221	Ý	Ý	Extended characters
222	Þ	Þ	Extended characters
223	ß	ß	Extended characters
224	à	à	Extended characters
225	á	á	Extended characters
226	â	â	Extended characters
227	ã	ã	Extended characters
228	ä	ä	Extended characters
229	å	å	Extended characters
230	æ	æ	Extended characters
231	ç	ç	Extended characters
232	è	è	Extended characters
233	é	é	Extended characters
234	ê	ê	Extended characters
235	ë	ë	Extended characters
236	ì	ì	Extended characters
237	í	í	Extended characters
238	î	î	Extended characters
239	ï	ï	Extended characters
240	ð	ð	Extended characters
241	ñ	ñ	Extended characters
242	ò	ò	Extended characters
243	ó	ó	Extended characters
244	ô	ô	Extended characters
245	õ	õ	Extended characters
246	ö	ö	Extended characters
247	÷	÷	Extended characters
248	ø	ø	Extended characters
249	ù	ù	Extended characters
250	ú	ú	Extended characters
251	û	û	Extended characters
252	ü	ü	Extended characters
253	ý	ý	Extended characters

Decimal value	ASCII character	Display	Comment/description
254	þ	þ	Extended characters
255	ÿ	ÿ	Extended characters

7.2 Control characters used by IH-Com

Control character	IH-Com Tag	ASCII Value	Designation	Description
NUL	<NUL>	0	Null	Fill character without information content.
SOH	<SOH>	1	Start of header	Marks the start of a byte sequence which represents the destination of the message or change in machine-readable form.
STX	<STX>	2	Start of Text	Marks the start of the actual message and thus the end of the "header"
ETX	<ETX>	3	End of Text	Marks the end of the message to be transmitted. Used as "cancel" character for terminal input.
EOT	<EOT>	4	End of Transmission	Marks the end of the complete transmission which can consist of several messages including headers. Used as "session end/logout" for some command interpreters.
ENQ	<ENQ>	5	Enquiry	In a bidirectional communication direction as enquiry whether the remote station can respond. Also paraphrased as "WRU" [who are you]. Usually named as "Wer Da?" (who is there?) on German teleprinters.
ACK	<ACK>	6	Acknowledge	Control character which expresses the positive response to a previous enquiry.
BEL	<BEL>	7	Bell	Generates a sound signal (bell or beep) at the receiving terminal. Used as alarm character or to draw attention to error situations.
BS	<BS>	8	Backspace	Moves the print head/cursor one position back. (And deletes the character printed there).
TAB	<TAB>	9	Horizontal Tab	Moves the print head/cursor to the next predefined position (tab stop) in the current line.
LF	<LF>	10	Line Feed	Moves the print head/cursor to the next line. If agreed between sender and receiver, it means "New Line" whereby the first print position of the next line is moved to.
VT	<VT>	11	Vertical Tab	Moves the print head/cursor to the next predefined line.
FF	<FF>	12	Form Feed	Moves the print head/cursor to the first print position on the next page.
CR	<CR>	13	Carriage Return	Moves the print head/cursor back to the first print position of the current line.
SO	<SO>	14	Shift Out	Exit the current presentation mode until "Shift In" occurs. This is used for temporary changeover to different character sets, fonts and similar.

Control character	IH-Com Tag	ASCII Value	Designation	Description
SI	<SI>	15	Shift In	Return from the mode initiated by "Shift Out".
DLE	<DLE>	16	Data Link Escape	Control character which announces that a specified number of following characters have a different meaning, e.g. control information.
DC1	<DC1>	17	Device Control 1	Device-specific control character, for example to activate and deactivate specific device functions (e.g. font for printers).
DC2	<DC2>	18	Device Control 2	*Control character " DC1" p. 67*
DC3	<DC3>	19	Device Control 3	*Control character " DC1" p. 67*
DC4	<DC4>	20	Device Control 4	*Control character " DC1" p. 67*
NAK	<NAK>	21	Negative Acknowledge	Control character which expresses the negative response to a previous enquiry.
SYN	<SYN>	22	Synchronous Idle	Control character which also enables the synchronization in the absence of signals to be transmitted for synchronous data transfers.
ETB	<ETB>	23	End of Transmission Block	Control character which indicates the end of a block of transferred data blocks if this end of block cannot be recognized from the data themselves.
CAN	<CAN>	24	Cancel	Control character which indicates that the data just transferred are/were defective and must be discarded.
EM	<EM>	25	End of Medium	Control character which indicates the (physical or logical) end of the storage medium.
SUB	<SUB>	26	Substitute	Used to replace a character which is invalid or incorrect.
ESC	<ESC>	27	Escape	Control character which should indicate the extension of the character set. It is itself the start of a sequence of directly following characters which have a particular meaning.
FS	<FS>	28	File Separator	Separator character which logically subdivides the data blocks. The precise meaning of the logical units "File", "Group", "Record", "Unit" is not defined; however the order should be from "File" as the highest level division unit to "Unit" as the lowest level division unit.

Control character	IH-Com Tag	ASCII Value	Designation	Description
GS	<GS>	29	Group Separator	*Control character “ FS” p. 67*
RS	<RS>	30	Record Separator	*Control character “ FS” p. 67*
US	<US>	31	Unit Separator	*Control character “ FS” p. 67*

7.3 CE compliance

The **IH**-Com software has been designed to meet all current requirements for safety at work and operating. It complies with all relevant European Community directives and associated harmonized standards.

7.3.1 European directives

Reference	Designation
IVD 98/79/EC	In Vitro Diagnostic Medical Devices Directive
EN ISO 13485:2012	Medical devices - Quality management systems - requirements for regulatory purposes
EN ISO 14971:2012	Medical devices - Application of risk management to medical devices
EN 62304:2006	Medical device software - Software life-cycle processes
EN 62366:2008	Medical devices - Application of usability engineering to medical devices

7.4 Device disposal

7.4.1 General warnings

In order to protect the persons and the environment, any instrument and its accessories must be disposed of in an appropriate way. Its is mandatory to strictly apply laws and local bylaws relative to an appropriate disposal procedure.

7.5 List of documents

Service and user manual is available on www.diamed.ch

section «distributors - secret area - manuals download»

PAGE INTENTIONALLY LEFT BLANK

INDEX

PAGE INTENTIONALLY LEFT BLANK

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

....

Bio-Rad
Laboratories

Clinical
Diagnostics Group

DiaMed
Pra Rond 23, 1785 Cressier
Switzerland
www.bio-rad.com/immunohematology