Gateway Manual

Protocol Conversion

EtherNet/IP Adapter (Slave) to ASCII

on the netTAP

NT 40-EN-RS

NT 40-RS-EN

Edition: 1
Language: English

Hilscher Gesellschaft für Systemautomation mbH
Rheinstraße 15
D-65795 Hattersheim
Germany

Web: www.hilscher.com

List of Revisions

Index	Date	Version	Chapter	Revision
1	09 Aug 2005	V1.011	all	created

Although this protocol implementation has been developed with great care and intensively tested, Hilscher Gesellschaft für Systemautomation mbH cannot guarantee the suitability of this protocol implementation for any purpose not confirmed by us in writing.

Guarantee claims shall be limited to the right to require rectification. Liability for any damages which may have arisen from the use of this protocol implementation or its documentation shall be limited to cases of intent.

We reserve the right to modify our products and their specifications at any time in as far as this contribute to technical progress. The version of the manual supplied with the protocol implementation applies.

Table of Contents

1 Introduction

1.1 General

The netTAP NT 40-RS-EN and NT 40-EN-RS is a gateway system between EtherNet/IP and ASCII (RS-232, RS-422 or RS-485). It has a serial interface and an Ethernet interface. Two independent tasks read and write a block of shared memory. One of the tasks takes care of the EtherNet/IP communication; the other task provides ASCII connectivity.

1.2 About this Manual

In this manual the connection between devices with EtherNet/IP and ASCII protocol based on the gateway netTAP is described. The protocol conversion described here can be used with the following devices:

- NT 40-EN-RS2\CCE
- NT 40-EN-RS2\D9F
- NT 40-EN-RS2\D9M
- NT 40-EN-RS4\CCE
- NT 40-EN-RS4\D9F
- NT 40-EN-RS4\D9M
- NT 40-EN-RSI2\CCE
- NT 40-EN-RSI2\D9F
- NT 40-EN-RSI2\D9M
- NT 40-EN-RSI4\CCE
- NT 40-EN-RSI4\D9F
- NT 40-EN-RSI4\D9M
- NT 40-RS-EN

The ASCII protocol can be configured as Master or Slave. The EtherNet/IP Protocol works as Adapter (Slave) only.

Data received from the ASCII communication partner is stored in a shared memory and transferred to an EtherNet/IP Scanner. If the netTAP receives data from an EtherNet/IP scanner, this data can be transmitted to the ASCII communication partner.

The EtherNet/IP protocol is described in a separate document: Protocol manual *EtherNet/IP Slave (Adapter)*. This document is named eis_pre.pdf.

The netTAP gateway is configured with SYCON.net. This tool is described in a separate manual. This document is named ComProDTM_en.pdf.

The following figure shows how to connect the netTAP.

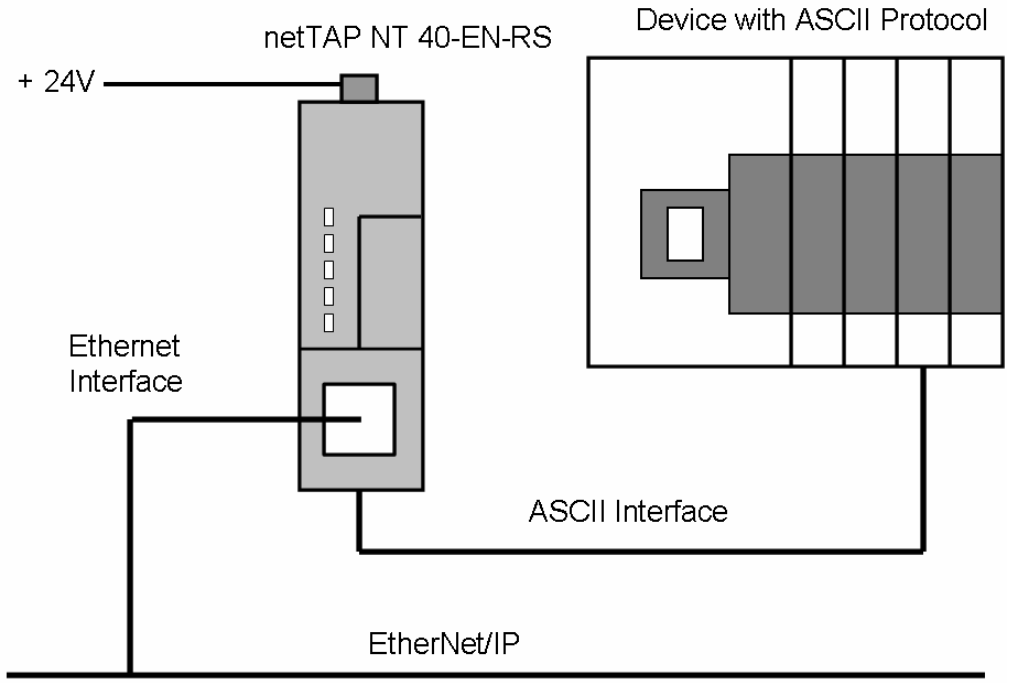

Figure 1: Connection of the devices at the netTAP 40-EN-RS

Values followed by 'h' are in hexadecimal notation such as 1Eh = 30. Values without following letters are decimal.

All IP address in this document have host byte order.

2 Startup Guideline

2.1 General

Before you can use your netTAP, consider the following sequence:

- A firmware has to be downloaded to the netTAP; it might be possible that a firmware is already loaded.
- A configuration has to be created and downloaded to the netTAP.
- Connect the netTAP to the Ethernet network and an EtherNet/IP Scanner (Master).
- The ASCII communication partner needs to be connected and configured properly. Make sure that the physical interface and cable are correct. Please consult the device user's manual.

3 Communication Mechanism

3.1 Data Management - Shared Memory

3.1.1 Shared Memory in the Gateway

Data exchange is carried out via the shared memory in the netTAP (called image memory). It consists of 240 bytes input and 240 bytes output. The amount of bytes exchanged over EtherNet/IP and the ASCII protocol are independent from each other. The data offset in the shared memory always starts at 0 (zero) in the shared memory.

The netTAP supports different communication mechanisms between EtherNet/IP Scanner and the ASCII device.

The ASCII protocol implementation on the netTAP supports unidirectional and bi-directional data transfer and Master and Slave operation.

The netTAP offers four options regarding its communication mechanism. It depends on application requirements and the operating mode. These modes are described in section *Operating Modes* on page *11*.

The terms "input" and "output" are always seen from the viewpoint of the EtherNet/IP Scanner.

- **Input Data**

  Data that are received from the EtherNet/IP Scanner.

- **Output Data**

  Data that are sent from the EtherNet/IP Scanner.

3.1.2 Communication Principles

The following cases may be considered:

- **Option 1:** Data transmission is **unidirectional** and is carried out from the EtherNet/IP Scanner to the ASCII communication partner. Thus, the EtherNet/IP Scanner initiates data transmission with the netTAP (1). Now the netTAP sends a telegram to the ASCII communication partner (2). Therefore the netTAP is configured as "**Master**" in the ASCII table (see section *ASCII Table* on page *19* for details).

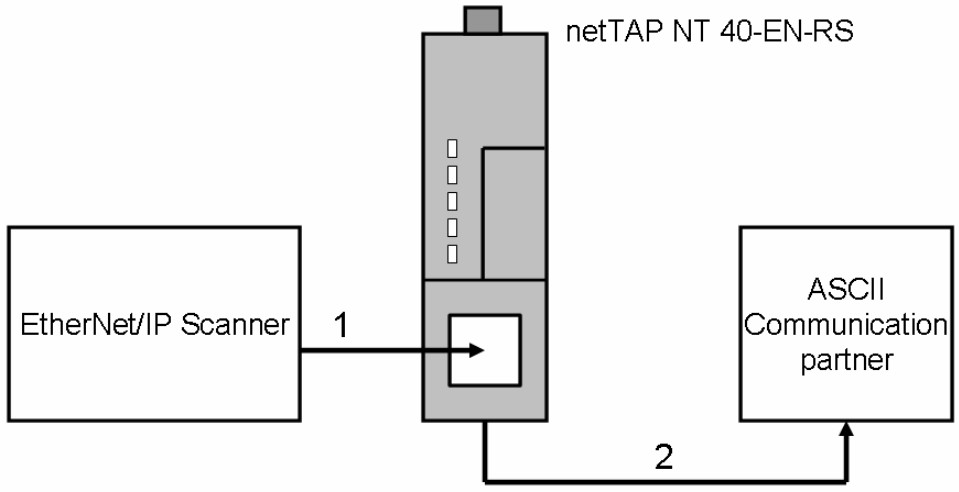

Figure 2: Unidirectional Data Transmission EtherNet/IP → ASCII

- **Option 2:** Data transmission is **unidirectional** and is carried out from the ASCII communication partner to the EtherNet/IP Scanner. The ASCII communication partner initiates the data transmission with a telegram to the netTAP (1). Then the netTAP transfers the new data to the image memory and from there to the EtherNet/IP Scanner (2). Therefore the netTAP is configured as "**Slave**" in the ASCII table (see section *ASCII Table* on page *19* for details).

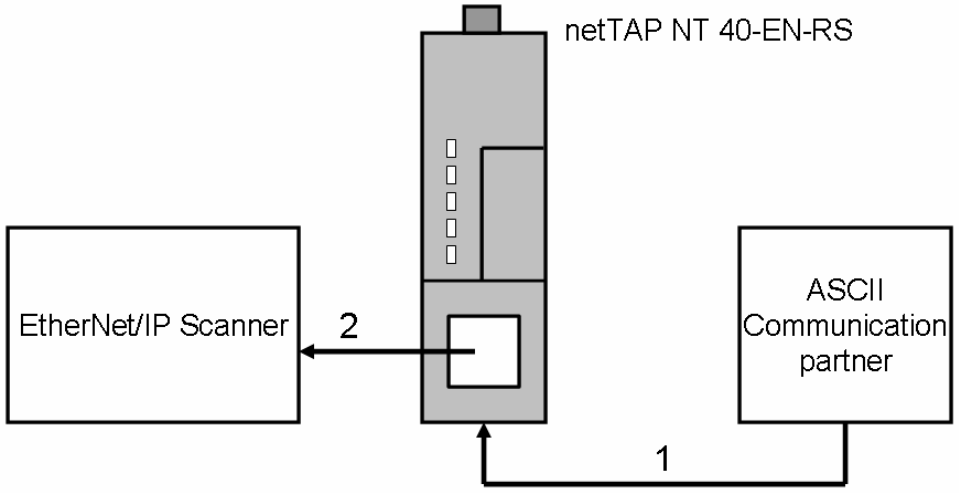

Figure 3: Unidirectional Data Transmission ASCII → EtherNet/IP

- **Option 3:** Data transmission is **bidirectional**. The EtherNet/IP Scanner initiates the data transmission with the netTAP (1). The netTAP then sends a telegram to the ASCII communication partner (2). Therefore the netTAP is configured as "**Master**" in the ASCII table (see section *ASCII Table* on page *19* for details). The ASCII communication partner then sends a reply telegram back to the netTAP (3). Using handshake bits (see section for details), the netTAP then transfers the new data to the EtherNet/IP Scanner via the shared memory (4).

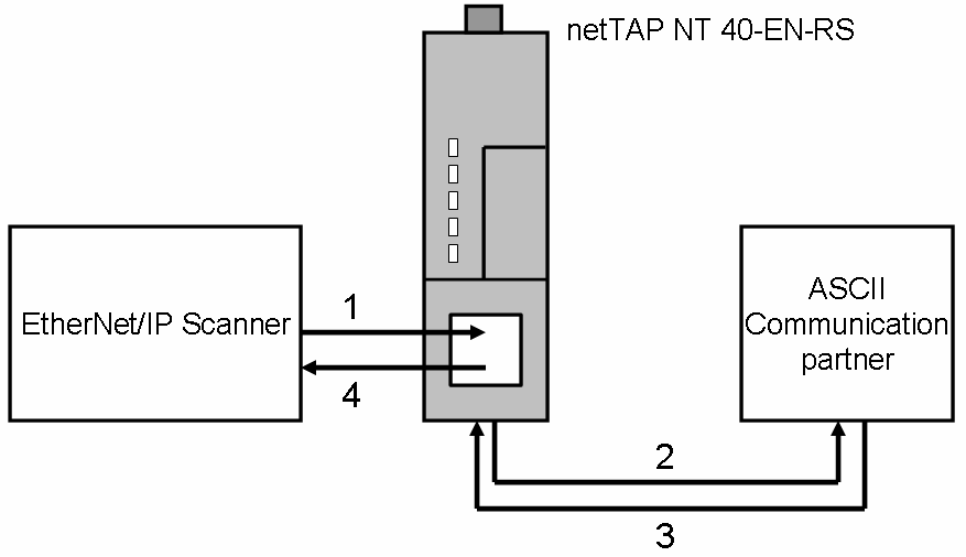

Figure 4: Bidirectional Data Tansmission EtherNet/IP → ASCII → EtherNet/IP

- **Option 4:** Data transmission is **bidirectional**. The ASCII communication partner initiates the data transmission with a telegram to the netTAP (1). Therefore the netTAP is configured as "Slave" in the ASCII table (see section *ASCII Table* on page *19* for details). The netTAP then transfers the new data to the EtherNet/IP Scanner via the shared memory (2). The EtherNet/IP Scanner sends a reply back to the netTAP (3). The netTAP then sends a telegram back to the ASCII communication partner (4).

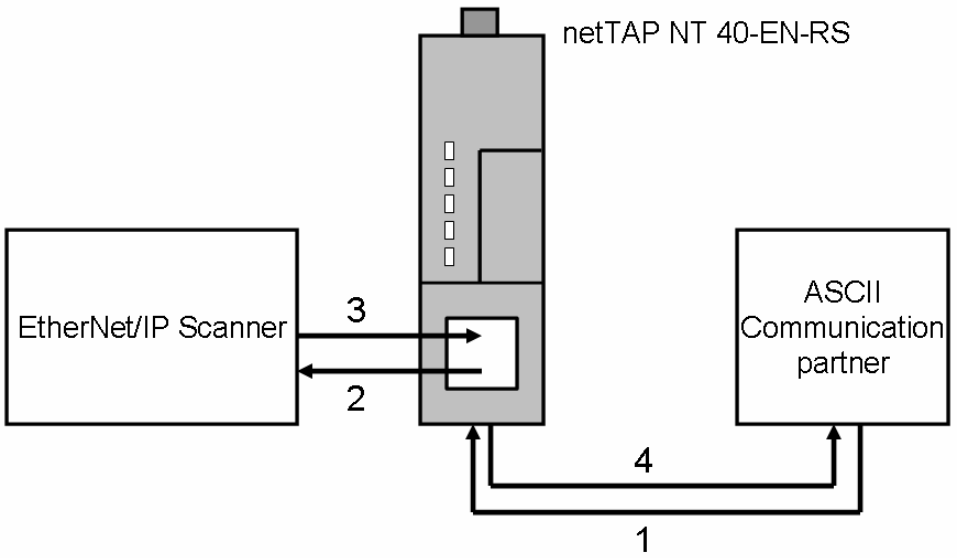

Figure 5: Bidirectional Data Transmission ASCII → EtherNet/IP → ASCII

3.1.3 Operating Modes

With respect to the ASCII device connected to the netTAP, one of four operating modes can be chosen. These operating mode have to be configured additionally to the ASCII settings. Configuration and meaning of the individual parameters are described in section *Settings - Parameterizing - Configuration.* In all modes input data is set to 0 (zero) after power-up or reset!

3.1.3.1 Operating Mode ‘No Send to Serial‘

In operation mode 'no send to serial' the netTAP allows data transmission only from the ASCII device to the EtherNet/IP Scanner. Therefore it is considered **unidirectional**; the netTAP is configured as **Slave** in the ASCII table.

As soon as the netTAP receives an ASCII telegram, the new data is passed to the process image. Max 240 bytes can be received per ASCII telegram. Previously received data in the process data image gets overwritten.

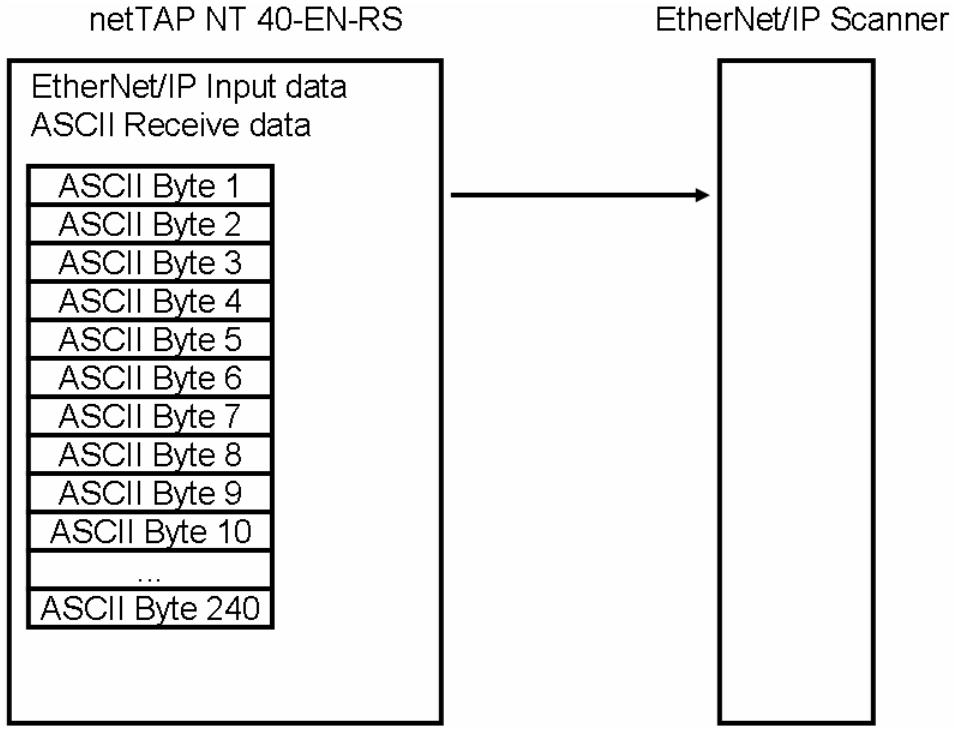

Figure 6: Operating Mode ‘No Send to Serial‘

3.1.3.2 Operating Mode ‘Cyclic‘

In operating mode 'Cyclic' the netTAP allows data transmission from the ASCII communication partner to the EtherNet/IP Scanner and from the EtherNet/IP Scanner to the ASCII communication partner. Therefore it is considered **bidirectional**; the netTAP is configured as **Slave** in the ASCII table.

As soon as the netTAP receives an ASCII telegram, the new data is transferred into the process image. 240 bytes can be received per ASCII telegram. Previously received data in the process data image gets overwritten.

Data received from the EtherNet/IP Scanner is transmitted to the ASCII communication partner in a cyclic manner. The time interval and the number of bytes is configurable (see section *BRIDGE Table* on page *22*). Data transmission always starts at offset 0 in the shared memory.

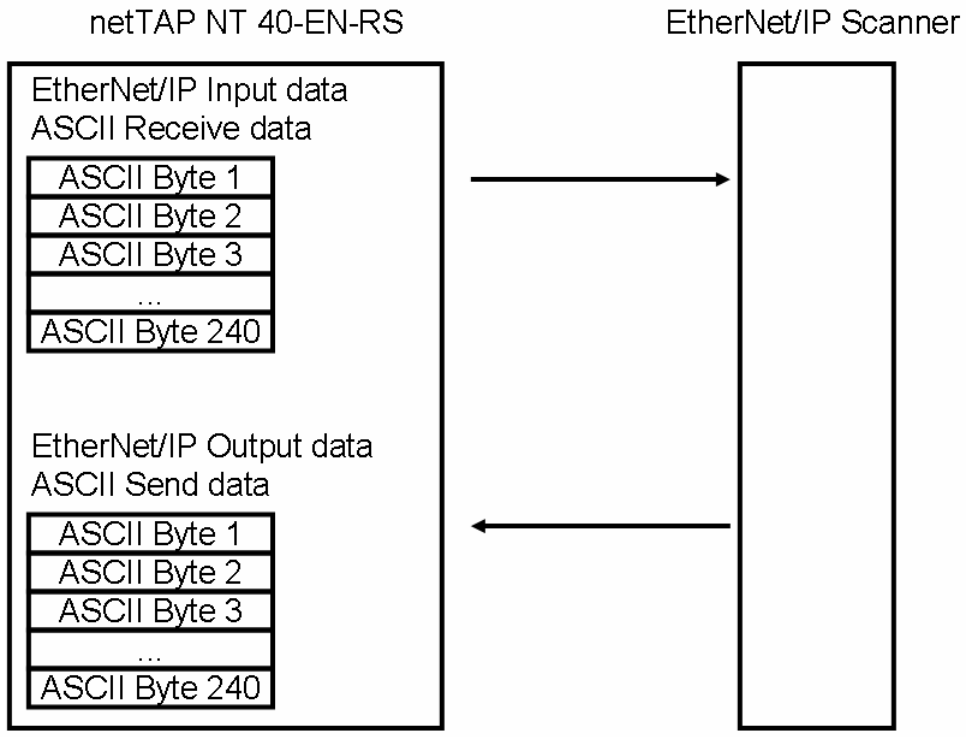

Figure 7: Operating Mode ‘Cyclic‘

3.1.3.3 Operating Mode ‘Change‘

In operating mode 'Change' the netTAP allows data transmission from the ASCII communication partner to the EtherNet/IP Scanner and from the EtherNet/IP Scanner to the ASCII communication partner. Therefore it is considered **bidirectional**; the netTAP is configured as **Slave** in the ASCII table.

As soon as the netTAP receives an ASCII telegram the new data is transferred to the shared memory and is transferred to the EtherNet/IP Scanner. The netTAP compares data received from the EtherNet/IP Scanner against data stored in the process data image. If they differ, an ASCII telegram with the new data is being created and sent to the ASCII device.

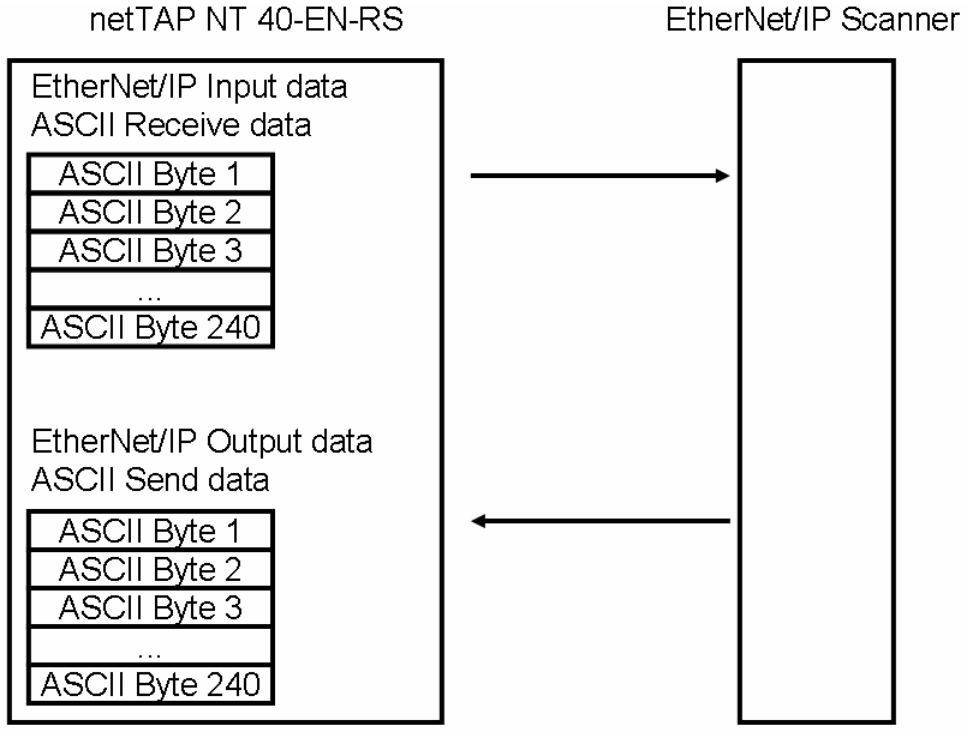

Figure 8: Operating Mode ‘Change‘

The range of data compared can be configured in the Bridge table (see section *Parameters for Operating Mode 'Change'* on page *23* for details). Below is an example of the relationship between the 'Serial length', 'Change length' and 'Change offset'.

Example:

Parameter	Configured Values
Serial length	20
Change length	10
Change offset	8

Table 1: Example Operating Mode 'Change'

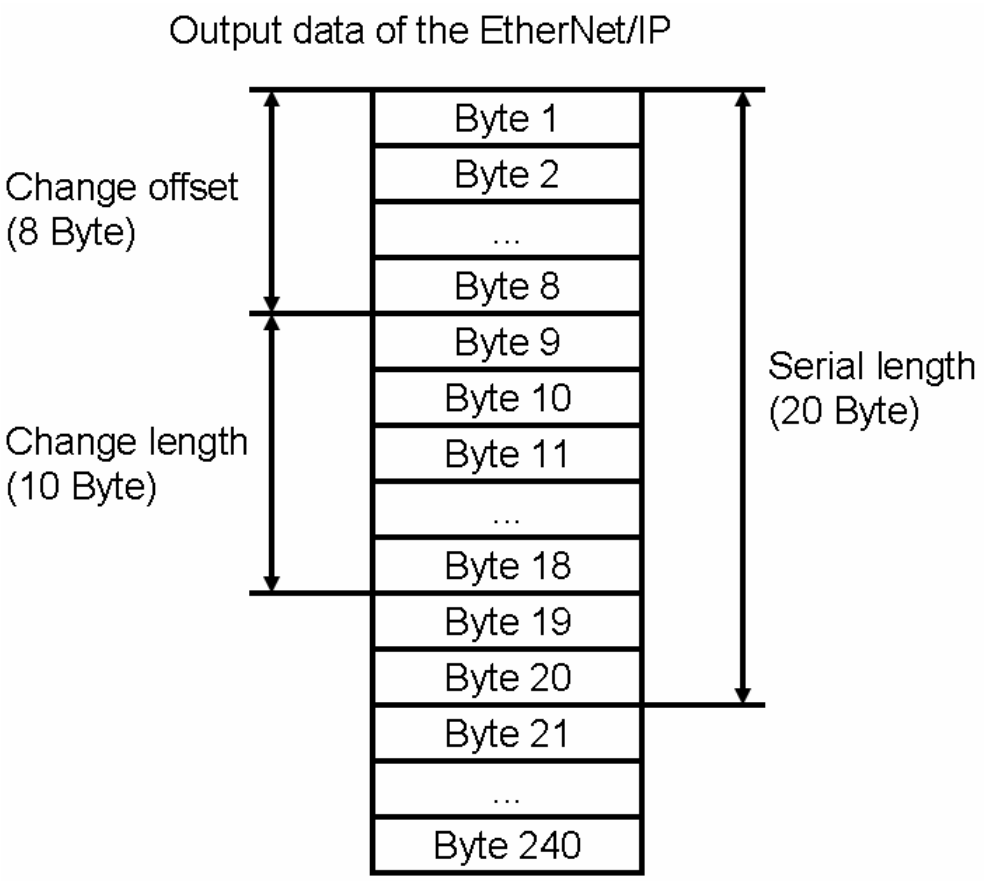

Figure 9: Connection between 'Serial length', 'Change length' and 'Change offset'

If data received from the EtherNet/IP Scanner differs from the previously received data in bytes 9 to 18, the netTAP creates an ASCII telegram containing bytes 1 to 20 and sends it to the ASCII communication partner. Changes in bytes 1 to 8 and bytes 19 and 20 have no effect. Data in bytes 21 to 240 are not sent to the ASCII communication partner.

3.1.3.4 Operating Mode ‘Counter‘

In operating mode 'Counter' the netTAP allows data transmission from the ASCII communication partner to the EtherNet/IP Scanner, and from the EtherNet/IP Scanner to the ASCII communication partner. Therefore it is considered **bidirectional**; the netTAP is configured as **Slave** in the ASCII table.

The transmission of new data to the ASCII communication partner is triggered when the value of the sequence counter has changed. This counter is located in an additional data block that precedes the actual data field in the process data image send by the EtherNet/IP Scanner. This additional data block consumes 12 bytes. The max number of user data bytes is still 240.

The process data image received from the EtherNet/IP Scanner (master output data) has the following structure:

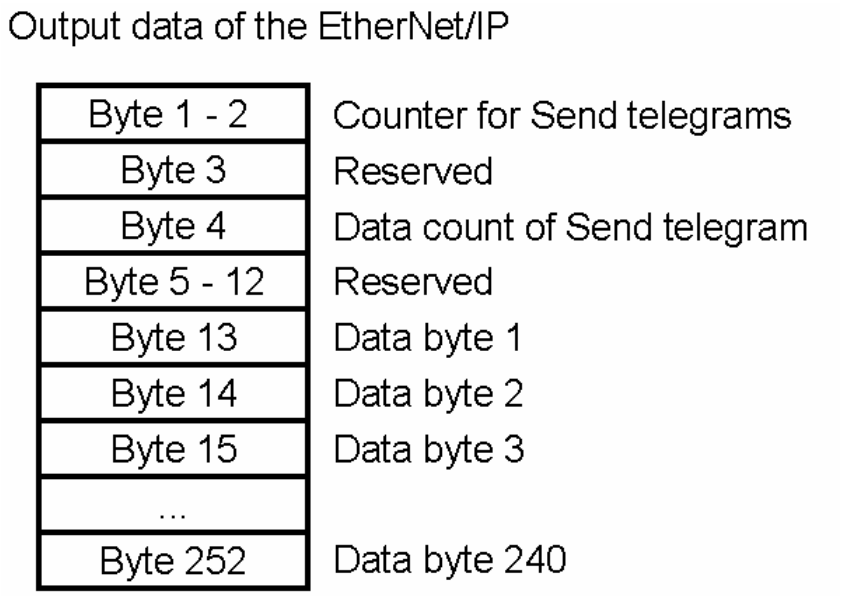

Figure 10: EtherNet/IP Output Data in Operating Mode ‘Counter‘

Byte	Meaning
1 – 2	Sequence counter; a change causes a telegram to the ASCII partner; the value can be selected freely.(Motorola format) Overflow causes 65535 => 0
3	Reserved
4	Number of data bytes of the next telegram.
5 - 12	Reserved
12 - 252	User data.

Table 2: Meaning of EtherNet/IP Output Data

The netTAP sends a data block of 12 additional bytes to the EtherNet/IP Scanner. This data block contains a sequence counter, which is incremented each time a new ASCII telegram has been received. This data block precedes the actual data field in the process data image. The application running on the EtherNet/IP Scanner can detect new data telegrams as well as lost telegram. The max number of user data bytes is still 240.

The process data image sent by the netTAP to the EtherNet/IP Scanner (master input data) has the following structure:

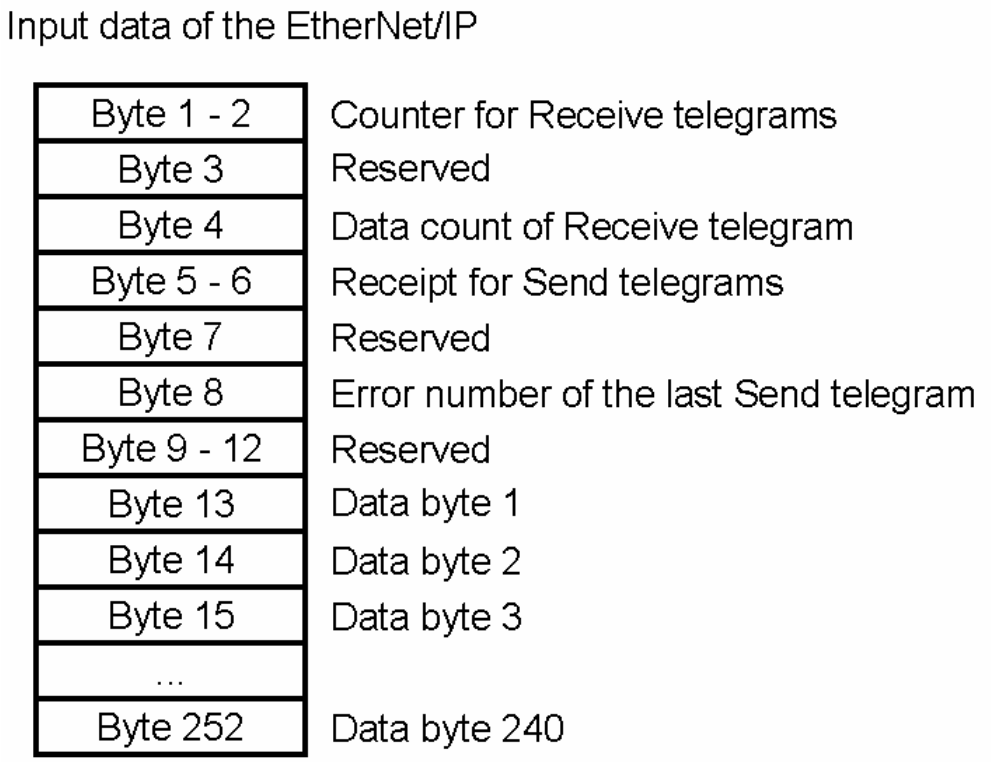

Figure 11: EtherNet/IP Input Data in Operating Mode 'Counter'

Byte	Meaning
1 - 2	Sequence counter; the value changes each time a new ASCII telegram has been received (Motorola format) Overflow causes 65535 => 0.
3	Reserved
4	Number of bytes in telegram
5 - 6	In these bytes the number of the last send job (byte 1 and 2 of the EtherNet/IP output data) to the ASCII communication partner is entered after the transmission. After this receipt is received the next job from the EtherNet/IP Scanner is started.
7	Reserved
8	In this byte the value 0 is entered after a successful transmission to the ASCII communication partner. If an error has occurred during the transmission the error number is entered here. The error numbers are described in section *Error Messages*.
9 – 12	Reserved
13 - 252	User data of the last telegram from the ASCII communication partner.

Table 3: Meaning of EtherNet/IP Input Data

Thereby it has to be noted, that the user data starts with byte 13 of the EtherNet/IP in contrary to the other operating modes.

3.2 Task States in the EtherNet/IP Input Data

The netTAP can send additional diagnostic information to the EtherNet/IP Scanner. Sometimes it is useful to have the task status information available to detect communication issues or to trouble-shoot network problems during system set-up.

Two different structures are available. The diagnostic data is mapped independently into the input process data of the EtherNet/IP Scanner. See *Table 5: Task State SERIAL_STS* and *Table 6: Task State BRIDGE_STS* for details.

Task states and data types have the following structure.

Data type	Description	Range of Value	Length in Bytes
INT8	Integer 8 Bit	-128 … 127	1
INT16	Integer 16 Bit	-32.768 … 32.767	2
UINT8	Unsigned Integer 8 Bit	0 … 255	1
UINT16	Unsigned Integer 16 Bit	0 … 65.535	2

Table 4: Data Types of Task States

3.2.1 ASCII Task Diagnostic

State	Data type	Meaning
Send telegrams	UINT16	Number of ASCII telegrams sent without error. Overflow causes 65535 => 0
Receive telegrams	UINT16	Number ASCII telegrams received without error Overflow causes 65535 => 0
Error bits [HEX]	UINT16	Error class; meaning of the bits is described in section *Extended Task State - ASCII Task* on page *30.*
Last error	UINT8	Last error code detected Error codes are described in section *Error Codes - ASCII Task* on page *34*.

Table 5: Task State SERIAL_STS

3.2.2 Bridge Task Diagnostic

State	Data type	Meaning
Send Count	UINT16	Number of telegrams sent to ASCII task without error Overflow causes 65.535 => 0.
Receive Count	UINT16	Number of telegrams received from ASCII task Overflow causes 65.535 => 0
Error Count	UINT16	Number of errors deteced Overflow causes 65535 => 0
Last Error	UINT8	Last error Code detected Error numbers are described in section *Extended Task State - Bridge EIPASC* on page *31*.

Table 6: Task State BRIDGE_STS

4 Settings - Parameterizing - Configuration

4.1 Network Parameter

Network parameters for both protocols of the netTAP have to be configured:

- **ASCII**

  Network parameters for the ASCII protocol (baud rate, parity, …) are entered in the *ASCII* table.

- **EtherNet/IP**

  Number of input and output bytes are entered in *IO_LENGTH* table.

- **Ethernet**

  IP address, net mask and gateway address are entered in the *IP_SETUP* table.

4.2 ASCII Table

In the table *ASCII* the parameters of the ASCII protocol are configured. The default values are characterized by an underline:

Parameter	Meaning	Range of Value
Interface and RTS	With this parameter the serial physics of the netTAP as well as the RTS control are configured. If the RTS control is switched on, then the control lines RTS and CTS are served by an RS232C interface or by an RS485/RS422 interface that switches through the data drivers only while sending. This does not alter the protocol sequence. Thereby the protocol expiration does not change. With a RS485 interface the RTS control must always be switched on.	RS232 RTS OFF RS232 RTS ON RS422 RTS OFF RS422 RTS ON RS485 RTS ON
Baud rate	Specifies the transmission rate.	50 Baud 100 Baud 110 Baud 150 Baud 200 Baud 300 Baud 600 Baud 1200 Baud 2400 Baud 4800 Baud 9600 Baud 19200 Baud 38400 Baud
Data bits	Defines the quantity of data bits.	7, 8
Stop bits	Specifies the number of stop bits. The netTAP supports only one stop bit, therefore this value can not be changed.	1
Parity	Defines the parity bit.	none even odd
Mode	Operating Mode of the ASCII protocol	Slave Master
End mode	Specifies the end mode of the telegram. In Slave mode none of the following end modes is allowed: 'End mode = 2' or 'End mode = 3' or 'End mode = 5'.	0 = only time control 1 = end identifier 2 = acknowledge telegram 3 = end identifier / acknowledge telegram 4 = fixed data count 5 = pass data count forward
Checksum mode	Defines the checksum mode	no binary 7 bit binary 8 bit BCC BCC in ASCII
Checksum area	Defines the telegram area over which the checksum is created	only user data with start identifier with end identifier complete telegram

Character filter	Defines the filter function for certain characters	no filter character duplication
Character [HEX]	Defines the character for the active filter in Hexadecimal.	0 … FFh
Telegram timeout	Timeout for receive of the complete telegram. Using 'Telegram timeout = 0' means that 'Start timeout' and 'Character delay time' is valid and used from the protocol.	1 … 1000 … 65535 0 = none timeout
Start timeout	Max waiting time for the telegram start in milliseconds. Only valid if 'Telegram timeout = 0'. Not in slave mode.	1 … 65535 0 = none timeout
Character delay time	Max time between two characters in a telegram in milliseconds. Only valid if 'Telegram timeout = 0'.	1 … 65535 0 = none timeout
Retries	Number of retries in error case	0 … 3 … 10
Length of telegram start	Length of the telegram start. If length is 0 then no start is send. If length is negative the text is interpreted as Hex characters 0..F In any other cases the ASCII Text is used.	-4 … 0 … 8
Telegram start	Telegram start identifier of 0..8 characters. This is used for each send telegram	8 ASCII characters or 4 hex characters 00h – FFh
Length of telegram end	Length of the telegram end. If length is 0 then no end is send. If length is negative the text is interpreted as Hex characters 0..F In any other cases the ASCII Text is used.	-4 … 0 … 8
Telegram end	Telegram end identifier of 0..8 characters. This is used after each send telegram	8 ASCII characters or 4 hex characters 0x00 - 0xff
Length of ACK-telegram	Length of the acknowledge telegram. If length is 0 then no end is send. If length is negative the text is interpreted as Hex characters 0..F In any other cases the ASCII Text is used.	-4 … 0 … 8
ACK-telegram	Acknowledge-Telegram identifier of 0..8 characters.	8 ASCII characters or 4 hex characters 00h – FFh
Length of NACK-telegram	Length of the negative acknowledge telegram. If length is 0 then no end is send. If length is negative the text is interpreted as Hex characters 0..F In any other cases the ASCII Text is used.	-4 … 0 … 8
NACK-telegram	Negative Acknowledge Telegram identifier of 0..8 characters.	8 ASCII characters or 4 hex characters 00h – FFh
Telegram length device	Length of the telegram of the coupling device, for specifying the end of the telegram Only in End mode 'fixed data count'	0 … 240
Telegram sequence time	Specifies the min time between two send telegrams	0 = no Telegram sequence time 1 … 65535

Table 7: ASCII Protocol Settings

4.3 IO_LENGTH Table

The *IO_LENGTH Table* contains all setting related to the EtherNet/IP protocol . Default values are underlined:

Parameter	Meaning	Range of Value
Inst01 Length (In)	Number of input bytes	0 … 64 … 504
Inst02 Length (Out)	Number of output bytes	0 … 64 … 504

Table 8: EtherNet/IP Protocol Settings

4.4 BRIDGE Table

The *Bridge Table* contains all setting in regards to the operating mode of the netTAP. The different modes are described in section *Operating Modes* . Default values are underlined.

Parameter	Range of Value
Function mode	Set operating mode: No send to serial Cyclic Change Counter
Serial length	0 … 40 … 240
Setup time x10ms	0 … 20 … 60000
Change length	0 … 20 … 240
Change offset	0 … 239

Table 9: Bridge EIPASC

4.4.1 Parameters for Operating Mode ‘No Send to Serial‘

Parameter	Meaning
Serial length	no meaning
Setup time x10ms	no meaning
Change length	no meaning
Change offset	no meaning

Table 10: Meaning of the Parameters in Operating Mode ‘No send to serial‘

4.4.2 Meaning of the Parameters in the Operating Mode ‘Cyclic‘

Parameter	Meaning
Serial Length	Number of bytes, which are sent cyclically to the ASCII communication partner. This value must not be zero for this operation mode!
Setup time x10ms	Time slice in steps of 10 milliseconds, in which the data are sent cyclically to the ASCII communication partner. This value must not be zero for this operation mode!
Change length	no meaning
Change offset	no meaning

Table 11: Meaning of the Parameters in Operating Mode ‘Cyclic‘

4.4.3 Parameters for Operating Mode ‘Change‘

Parameter	Meaning
Serial length	Number of bytes, which are send to the ASCII communication partner when changes take place. This value must not be zero for this operation mode!
Setup time x10ms	no meaning
Change length	Number of bytes, which are compared: value must not be zero for this operation mode!
Change offset	Byte offset, from which the data is compared.

Table 12: Meaning of the Parameters in Operating Mode ‘Change‘

With the configuration of the operating mode 'Change' it has to be noted that the data area, which is to be supervised, is completely within the transmit data of the ASCII telegram. If the range is outside of transmitting data the task reports an initialization error.

4.4.4 Parameters for Operating Mode ‘Counter‘

Parameter	Meaning
Serial length	no meaning
Setup time x10ms	no meaning
Change length	no meaning
Change offset	no meaning

Table 13: Meaning of the Parameters in Operating Mode ‘Counter‘

4.5 SERIAL_STS Table

Diagnostic information maintained by the ASCII task can be mapped into the input data image of the EtherNet/IP Scanner. The table SERIAL_STS defines whether or not this information is mapped. The task state structure is described in section *ASCII Task Diagnostic* on page *17*. Default values are underlined.

Parameter	Meaning	Range of Value
Serial state	The ASCII task state is not mapped into the input data.	off
	The ASCII task state is mapped into the input data.	on
Start offset (byte)	Byte offset to set the start offset for the ASCII task state in the input area.	0 … 240 … 248

Table 14: SERIAL_STS Table

4.6 BRIDGE_STS Table

Diagnostic information maintained by the bridge task EIPASC can be mapped into the input data image of the EtherNet/IP Scanner. The table *BRIDGE_STS* defines whether or not this information is mapped. The task state structure is described in section *Bridge Task Diagnostic* on page *17.* Default values are underlined.

Parameter	Meaning	Range of Value
Bridge state	The EIPASC task state is not mapped into the input data.	Off
	The EIPASC task state is mapped into the input data.	on
Start offset (byte)	Byte offset to set the start offset for the EIPASC task state in the input area.	0 … 248

Table 15: BRIDGE_STS Table

4.7 IP_SETUP Table

Settings in regards to IP parameters are set in this table. Default values are shown underlined:

Parameter	Meaning	Range of Value
Flags 0	See figure *Parameter Flag 0 Definition*	0 … 7 … 255
Flags 1	See figure *Parameter Flag 1*	0 … 2 … 255
IP address Byte 0	First (lowest) byte of the own IP address	0 … 255
IP address Byte 1	Second byte of the own IP address	0 … 255
IP address Byte 2	Third byte of the own IP address	0 … 255
IP address Byte 3	Fourth (highest) byte of the own IP address	0 … 255
Net mask Byte 0	First (lowest) byte of the net mask	0 … 255
Net mask Byte 1	Second byte of the net mask	0 … 255
Net mask Byte 2	Third byte of the net mask	0 … 255
Net mask Byte 3	Fourth (highest) byte of the net mask	0 … 255
Gateway Byte 0	First (lowest) byte of the IP address of the Gateway	0 … 255
Gateway Byte 1	Second byte of the IP address of the Gateway	0 … 255
Gateway Byte 2	Third byte of the IP address of the Gateway	0 … 255
Gateway Byte 3	Fourth (highest) byte of the IP address of the Gateway	0 … 255

Table 16: IP_SETUP Parameter List

The parameter flag 0 is a bit-coded byte with the following meaning:

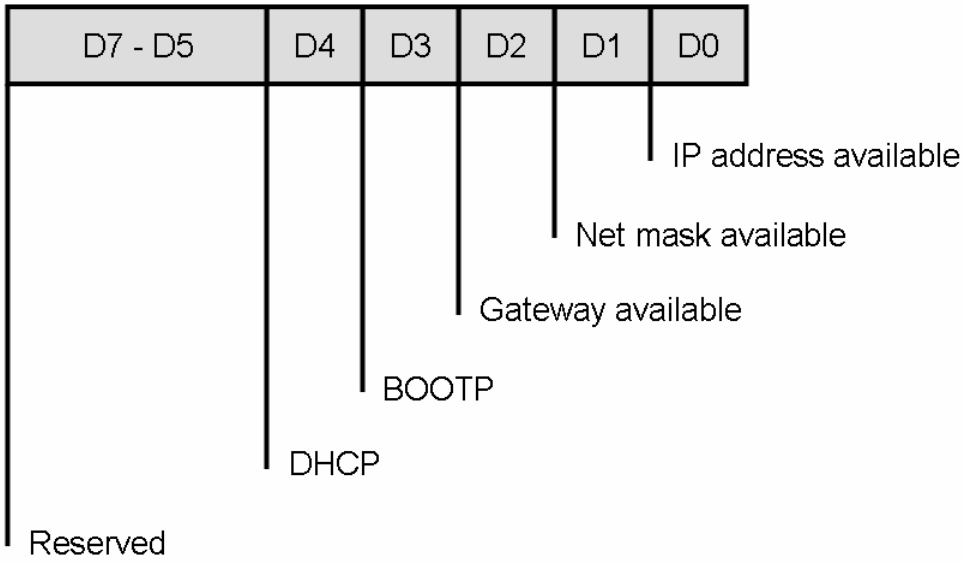

Figure 12: Parameter Flag 0 Definition

Parameter	Meaning
IP address available	If this bit is set, the contents of the parameters IP address byte 0 to IP address byte 3 is evaluated.
Net mask available	If this bit is set, the contents of the parameters net mask byte 0 to IP address byte 3 is evaluated.
Gateway available	If this bit is set, the content of the parameters Gateway byte 0 to IP address byte 3 is evaluated.
Enable BOOTP	The device contains the IP parameter of a BOOTP Server.
Enable DHCP	The device contains the IP parameter of a DHCP Server.

Table 17: Parameter Flag 0 Definition

The parameter flag 1 is a bit-coded byte with the following meaning:

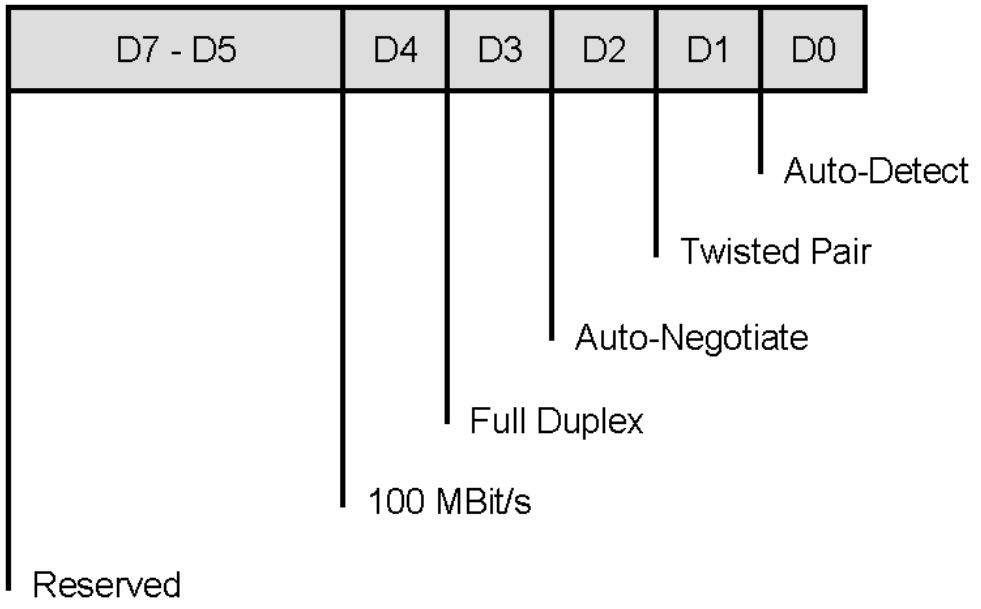

Figure 13: Parameter Flag 1 Definition

Parameter	Meaning
Auto-Detect	If this bit is set, an automatic detection of the Ethernet interface takes place.
Twisted Pair	If this bit is set, twisted pair is selected as Ethernet interface, otherwise AUI.
Auto-Negotiation	If this bit is set, the device negotiates automatically the duplex mode and the data transmission rate with the hub or switch.
Full Duplex	If this bit is set, the device works in full-duplex mode, otherwise it works in half duplex mode.
100 MBit/s	If this bit is set, the device works with a transmission rate of 100 MBit/s, otherwise with 10 MBit/s.

Table 18: Parameter Flag 1 Definition

If more than one configuration method is chosen (e.g. DHCP and manually entered IP parameters), the netTAP tries to process all methods successively. As soon as one was successful, the netTAP starts with these parameters.

5 Diagnostic

5.1 LEDs

The LEDs on the netTAP shows the status of the gateway:

LED	Status	Meaning
RDY	On	netTAP NT 40-EN-RS / NT 40-RS-EN is ready
	Flashing cyclic (5 Hz)	Firmware download is in progress
	Flashing cyclic (1 Hz)	Device is in bootloader mode and is waiting for firmware download
	Flashing irregular (*)	Hardware or heavy runtime error detected
	Off	Device has no power supply or hardware defect
RUN	On	Communication is running
	Flashing cyclic (5 Hz)	No connection established
	Flashing irregular (*)	Configuration missing or faulty, device needs commissioning
	Off	No communication
ERR	On	EtherNet/IP error
	Off	No error
STA	-	LED without function
ACT	On	A connection to the Ethernet exsits
	Flashing	The device sends/receive Ethernet frames
	Off	The device has no connection to the Ethernet

Table 19: LEDs

(*) 3 times fast at 5 Hz, 8 times between 0.5 Hz and 1 Hz

5.2 Extended Task State

The configuration tool SYCON allows monitoring diagnostic information and helps trouble shooting network issues.

5.2.1 Extended Task State - ASCII Task

5.2.1.1 ASCII Protocol

The following task state contains the common state, statistic and error information.

State	Meaning
Task state	Current condition of the protocol procedure. 0 = Not initialized 1 = Idle mode 2 = Telegram following time runs 3 = Transmission mode 4 = Receive mode
Send telegrams	Number of telegrams sent without error.
Receive telegrams	Number of telegrams received without error.
Send errors	Number of faulty telegrams sent.
Receive errors	Number of faulty telegrams received.
Error Bits [HEX]	Detected error are converted to an error class, error classes are shown here. For details see section *Figure 14: Error Bit* on page *30*
Last Error	Last error detected

Table 20: Extended Task Status - ASCII Protocol

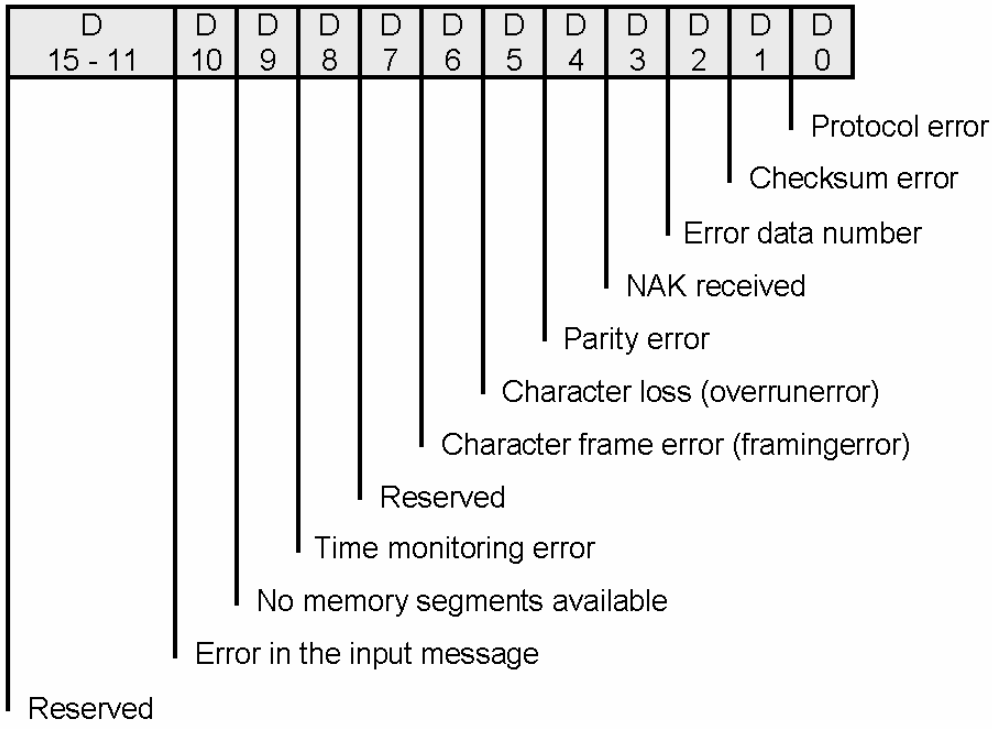

Figure 14: Error Bit Definition

The corresponding bit is set for each error. The bit is cleared only after reset or after power cycling the netTAP.

5.2.2 Extended Task State - Bridge EIPASC

5.2.2.1 EIPASC General

The following task state structure contains information about the common state, statistic and holds additional error information.

State	Meaning
Conversion	Actual conversion of the bridge task: EIP ←→ ASC
Version bridge	Version of the bridge task
Version function	Version of the function module
Data send count	Number of telegrams sent to the ASCII communication partner.
Data receive count	Number of telegrams received by the ASCII communication partner.
Last error	Number of the last detected error. The error numbers are described in section *Error Codes - EIPASC*.
Error count	Number of detected errors.

Table 21: Extended Task State Definition - EIPASC General

5.2.2.2 EIPASC Setup

The following task state structure contains information about the current configuration of the bridge task. Only values of parameters in use are shown, all unused parameters are set to 0.

State	Meaning
Function mode	Actual operating mode of the conversion No send to serial Cyclic Change Counter
Data length	Configured data length for the telegram to the ASCII communication partner in the modes 'Cyclic' and 'Change'
Setup time	Configured time interval for the telegram to the ASCII communication partner in the 'Cyclic' mode.
Change length	Configured data length for the supervision of data changes in the 'Change' mode.
Change offset	Configured offset for the supervision of data changes in the 'Change' mode.
Serial state	ASCII state in the input data of the EtherNet/IP Off On
Serial offset	Offset from which the ASCII state is faded in into the input data of the EtherNet/IP
Bridge state	Bridge state in the input data of the EtherNet/IP Off On
Bridge offset	Offset from which the bridge state is faded in into the input data of the EtherNet/IP

Table 22: Extended Task State Definition - EIPASC Setup

5.2.3 Extended Task State - TCP_UDP Task

5.2.3.1 TCP_UDP Task Information

State	Meaning
Task state	Task state of the Task: 0 = Task not initialized 1 = Task is running 2 = Task initialized 3 = Task reports an error by initialization
Error count	Number of errors appeared
Last error	Last error appeared (description see TCP/UDP protocol manual)

Table 23: Extended Task State - TCP_UDP Task Information

5.2.4 Extended Task State - IP Task

5.2.4.1 IP Task Information

State	Meaning
Task state	State of the Task: 1 = Task is running 2 = initialization is running 3 = initialization has failed
Error count	Counter for appeared errors
Last error	Last appeared error
IP address	IP-Address of the device
Net mask	Net mask of the device
Gateway	Gateway of the device

Table 24: Extended Task State - IP Task Information

5.2.4.2 IP Ethernet State

State	Meaning
MAC address (hex)	MAC address of the device
Interface	Actual known Ethernet interface
Speed	Transmission rate
Duplex mode	Shows the actual Duplex mode: Half-/Full duplex
Twisted pair link	State of the Twisted Pair connection

Table 25: Extended Task State - IP Ethernet State

6 Error Messages

6.1 General

The following tables show error messages of the individual protocols. These can be displayed with the diagnostic program SYCON.

6.2 Error Codes - ASCII Task

6.2.1 Initialization Errors

Error number	Meaning
10	Serial interface occupied. The serial interface has already been initialized by another task.
11	Sum of all baudrates to high The sum of all Baud rates on all initialized interfaces is too big.
12	Error 'Communication line' Parameterized interface at the device not available.
13	Error 'Baudrate' Invalid value for the initialization parameter 'Baudrate' initialization parameter.
14	Error 'Parity' Invalid value for the initialization parameter 'Parity'.
15	Error 'Databits' Invalid value for the initialization parameter 'Data bits'.
16	Error 'Stopbits' Invalid value for the initialization parameter 'Stop bits'.
17	Error 'RTS-Control' Invalid value for the initialization parameter 'RTS-Control'.
50	Error 'Mode' Invalid value for the initialization parameter 'Mode'.
51	Error 'End mode' Invalid value for the initialization parameter 'End mode'.
52	Error 'Checksum mode' Invalid value for the initialization parameter 'Checksum mode'.
53	Error 'Checksum area' Invalid value for the 'Checksum area'.
54	Error 'Character filter' Invalid value for the 'Character filter'.
55	Error 'Telegram timeout' Invalid value for the initialization parameter 'Telegram timeout'.
56	Error 'Start timeout' Invalid value for the initialization parameter 'Start timeout'.
57	Error 'Character delay time' Invalid value for the initialization parameter 'Character delay time'.
58	Error 'Retries' Invalid value for the initialization parameter 'Retries'.
60	Error 'Length of telegram start' Invalid value for the initialization parameter 'Length of telegram start'.
61	Error 'Length of telegram end' Invalid value for the initialization parameter 'Length of telegram end'.
62	Error 'Length of ACK-telegram' Invalid value for the initialization parameter 'Length of ACK-telegram'.
63	Error 'Length of NACK-telegram' Invalid value for the initialization parameter 'Length of NACK-telegram'.
64	Error 'Telegram length device' Invalid value for the initialization parameter 'Telegram length device'.

Table 26: Initialization Errors ASCII

6.2.2 Protocol Errors

Error Number	Error
100	Parity error The interface controller has detected a parity error.
101	Framing error The interface controller has detected a parity error.
102	Overrun error The interface controller has detected an “overrun” error..
103	To much/less data received More data has been received than can be accommodated in the receiving buffer or too little to make a proper telegram testing possible.
104	CRC-Error A CRC error has been determined in the received telegram.
105	Timeout telegram Time monitoring error. A time monitoring error has occurred as no answer has been received from the slave within the defined slave waiting time.
106	Protocol error The protocol procedure is incorrect.
108	NACK-telegram received The couple partner had answered with a NACK-telegram.
110	Error data start / data end The quantity of the received data is inconsistent with the configured length of the 'telegram start length' or 'telegram end length'.
111	Error checksum start / end The quantity of the received data is inconsistent with the configured checksum or checksum area.

Table 27: Protocol Errors ASCII

6.2.3 System Errors

Error Number	Error
210	Error by opening the data base The parameter database is not available.
212	Error by reading the data base The parameter database is inconsistent.
213	System error 'RcsPutStructure' Internal error.

Table 28: Internal System Errors ASCII

6.3 Error Codes - EIPASC

6.3.1 Initialization Errors

Error Number	Meaning
50	Serial task not found The Token of the ASCII protocol task could not be found.
51	Invalid Mode Invalid parameterizing of the tables SERIAL_STS or BRIDGE_STS detected.
52	Invalid setup time Invalid parameter for ‘Setup time x10ms‘.
53	Invalid serial length Invalid parameter for ‘Serial length‘.
54	Invalid change length Invalid parameter for ‘Change length‘.
55	Invalid change offset Invalid parameter for ‘Change offset‘.
56	Invalid change length and change offset Invalid parameter for ‘Serial length‘ in combination with ‘Change offset‘.
57	Invalid serial length in combination with change offset and change length Invalid parameter for ‘Serial length‘ in combination with ‘Change offset‘ and ‘Change length‘.
58	EtherNet/IP task not found The Token of the EtherNet/IP task could not be found

Table 29: Initialization Errors EIPASC

6.3.2 Runtime Errors

Error Number	Meaning
160	Invalid telegram header Invalid Telegram Header received
165	Invalid data count Invalid number of data received
176	Invalid acknowledge Invalid receipt received

Table 30: Runtime Errors EIPASC

6.3.3 System Errors

Error Number	Meaning
200	Task not initialized The task could not be initialized.
210	Error at opening the data base The parameter database is not available.
212	Error at reading the data base The parameter database is inconsistent.
213	System error 'RcsPutStructure' Internal error.
217	General system error Common system error.
218	Error 'Memory allocation' Error in the internal memory assignment.
219	Error ‘Dualport memory‘ Error with Dualport memory.

Table 31: Internal System Errors EIPASC

7 Examples

7.1 ASCII Protocol Configuration

When you configure the ASCII side of the netTAP, consider the following:

The ASCII protocol cannot resolve communication conflicts. If the netTAP and the ASCII communication partner send telegram at the same time, information may get lost (half-duplex communication). To help solving the conflict, you have the following options:

You can configure the netTAP to be a "Master". Configured as a Master, the netTAP sends the first telegram and initiates the transaction. Then the "Slave" devices replies with an ASCII telegram. This telegram may contain user data or is an acknowledgement to the previous Master telegram. The down side of this approach is that the transaction can only be initiated by the netTAP.

The other option is to configure the netTAP to be a "Slave". As a Slave, the netTAP can send and receive ASCII telegram at any time. Since the ASCII protocol implementation on the netTAP supports half-duplex communication only, you have to make sure on an application level that neither the netTAP nor the other device send an ASCI telegram at the same time. Only one ASCII device has the permission to send at any given time. Then the "send token" can be passed to the other device.

The following sections explain the communication mechanism. For better understanding of the please read chapter 1, 2 and 4 of manual *'ASCII protocol manual*' (asc_pre.pdf).

7.1.1 Example 1: Master/Slave with End Identifier

The ASCII communication partner is Master; the netTAP is Slave. Data exchange is bidirectional. The data transfer sequence is carried out as follow:

- The ASCII device (Master) starts the communication using the following telegram to the netTAP:

  <Send Data>END

  The text within the bracket represent the user data. "END" is the end identifier.

- The netTAP uses the END identifier to determine the end of the telegram. The netTAP strips off the end identifier and copies the content of <Send Data> into the input data image.

- With the next network cycle on the Ethernet side, the new data is transferred to the EtherNet/IP Scanner. The application running on the EtherNet/IP Scanner now creates a telegram with <Answer Data> and sends it to the netTAP.

- The netTAP sends <Answer Data> to the ASCII device followed by the end identifier END:

  <Answer Data>END

  For example, <Answer Data> can be "ACK" for a positive acknowledgement; it can be "NACK" for a negative acknowledgement.

The table below shows the settings for the ASCII protocol for this example:

Parameter	Value
Interface and RTS	RS232 RTS OFF
Baudrate	9600
Data bits	8
Stop bits	1
Parity	even
Mode	slave
End mode	end identifier
Checksum mode	none
Checksum area	only user data
Character filter	no filter
Character [HEX]	
Telegram timeout	1000
Start timeout	0
Character delay time	0
Retries	0
Length of telegram start	0
Telegram start	
Length of telegram end	4
Telegram end	END
Length of ACK-telegram	0
ACK-telegram	
Length of NACK-telegram	0
NACK-telegram	
Telegram length device	0
Telegram sequence time	0

Table 32: Parameter of the ASCII-Protocol: Example 1

7.1.2 Example 2: Master/Slave Using Telegram Timeout

The ASCII communication partner is Master; the netTAP is configured to be Slave. Data exchange is bidirectional. The communication has the following sequence:

- The ASCII device (Master) starts the communication by sending the following telegram to the netTAP:

  <Send Data>

- <Send Data> is copied into the input data image of the netTAP. The ASCII telegram is considered complete if no new character has been received after the configured telegram timeout of 50 ms

- With the next network cycle on the Ethernet side, the new data is transferred to the EtherNet/IP Scanner. The application running on the EtherNet/IP Scanner now creates a telegram with <Answer Data> and sends it to the netTAP.

- The netTAP sends <Answer Data> to the ASCII device:

  <Answer Data>

  For example, <Answer Data> can be "ACK" for a positive acknowledgement; it can be "NACK" for a negative acknowledgement.

The table below show the settings for the ASCII protocol for this example:

Parameter	Value
Interface and RTS	RS232 RTS OFF
Baudrate	9600
Data bits	8
Stop bits	1
Parity	even
Mode	slave
End mode	only time control
Checksum mode	none
Checksum area	only user data
Character filter	no filter
Character [HEX]	
Telegram timeout	50
Start timeout	0
Character delay time	0
Retries	0
Length of telegram start	0
Telegram start	
Length of telegram end	0
Telegram end	
Length of ACK-telegram	0
ACK-telegram	
Length of NACK-telegram	0
NACK-telegram	
Telegram length device	0
Telegram sequence time	0

Table 33: Parameter of the ASCII Protocol: Example 2

7.1.3 Example 3: Master/Slave with Character Timeout

The ASCII communication partner is Slave; the netTAP is configured to be a Master. The data exchange is bidirectional. The communication has the following sequence:

- The netTAP (Master) starts the communication by sending the following telegram to the ASCII communication partner.

  <Send Data>

- The ASCII communication partner (Slave) creates an reply and send <Answer data> to the netTAP. The ASCII device has to be send first character within the configured start timeout of 1000 ms. Otherwise the netTAP reports a timeout error to the EtherNet/IP Scanner (error bit D4 will be set).

- The netTAP copies the answer telegram <Answer Data> into the input data image. The telegram is considered complete if no new character has been received after 10 ms (character timeout).

- With the next network cycle on the Ethernet side, the new data is transferred to the EtherNet/IP Scanner. The application running on the EtherNet/IP Scanner now creates a telegram with <Answer Data> and sends it to the netTAP.

The table below show the settings for the ASCII protocol for this example:

Parameter	Value
Interface and RTS	RS232 RTS OFF
Baudrate	9600
Data bits	8
Stop bits	1
Parity	even
Mode	master
End mode	only time control
Checksum mode	none
Checksum area	only user data
Character filter	no filter
Character [HEX]	
Telegram timeout	1000
Start timeout	0
Character delay time	10
Retries	0
Length of telegram start	0
Telegram start	
Length of telegram end	0
Telegram end	
Length of ACK-telegram	0
ACK-telegram	
Length of NACK-telegram	0
NACK-telegram	
Telegram length device	0
Telegram sequence time	0

Table 34: Parameter of the ASCII Protocol: Example 3

7.1.4 Example 4: Master/Slave with Acknowledge Telegram

The Gateway is configured to be a Master: the ASCII communication partner is Slave. The data exchange is **unidirectional**. The communication has the following sequence:

- The netTAP (Master) starts the communication by sending the following telegram to the ASCII communication partner:

  <Send Data>

- The ASCII communication partner (Slave) has to answer with an acknowledge telegram within the configured telegram timeout of 1000 ms. The netTAP will wait and send 3 retry telegrams (parameter: Retries = 3). The netTAP reports an timeout error if all attempts fail (error bit D4 will be set).

  If the netTAP receives a valid telegram, the netTAP send an acknowledge telegram to the ASCII device (ACK = 06h). If, on the other hand, an error was detected in the answer telegram, the netTAP replies with an no-acknowledge telegram (NACK = 15h). in either case the netTAP acknowledges the handshake bit 'Send Order'.

The table below show the settings for the ASCII protocol for this example:

Parameter	Value
Interface and RTS	RS232 RTS OFF
Baudrate	9600
Data bits	8
Stop bits	1
Parity	even
Mode	master
End mode	acknowledge telegram
Checksum mode	none
Checksum area	only user data
Character filter	no filter
Character [HEX]	
Telegram timeout	1000
Start timeout	0
Character delay time	0
Retries	3
Length of telegram start	0
Telegram start	
Length of telegram end	0
Telegram end	
Length of ACK-telegram	-1
ACK-telegram	6
Length of NACK-telegram	-1
NACK-telegram	15
Telegram length device	0
Telegram sequence time	0

Table 35: Parameter of the ASCII Protocol: Example 4

7.1.5 Example 5: Slave/Slave

The netTAP is configured to be a Slave. Since the ASCII protocol implementation on the netTAP supports half-duplex communication only, you have to make sure on an application level that neither the netTAP nor the other device send an ASCI telegram at the same time. Only one ASCII device has the permission to send at any given time. Then the "send token" can be passed to the other device.

- For example, if the ASCII communication partner starts the communication and sends <Send Data>END, the netTAP recognizes 'END' as the end of the telegram. The <Send Data> are copied into the input data image and then transferred to the EtherNet/IP Scanner.
- If the EtherNet/IP Scanner starts the communication, the telegram <Send Data>END is send from the netTAP to the ASCII communication partner. The handshake bit 'Send Order' is automatically acknowledged by the converter.

The table below show the settings for the ASCII protocol for this example:

Parameter	Value
Interface and RTS	RS232 RTS OFF
Baudrate	9600
Data bits	8
Stop bits	1
Parity	even
Mode	slave
End mode	end identifier
Checksum mode	none
Checksum area	only user data
Character filter	no filter
Character [HEX]	
Telegram timeout	1000
Start timeout	0
Character delay time	0
Retries	3
Length of telegram start	0
Telegram start	
Length of telegram end	4
Telegram end	ENDE
Length of ACK-telegram	0
ACK-telegram	
Length of NACK-telegram	0
NACK-telegram	
Telegram length device	0
Telegram sequence time	0

Table 36: Parameter of the ASCII Protocol: Example 5

8 Technical Data

8.1 NT 40-EN-RS / NT 40-RS-EN with EtherNet/IP Adapter (Slave) to ASCII

EtherNet/IP Interface	Value
Product Name	NT 40-EN-RS, NT 40-RS-EN
Vendor ID	283
Ethernet Transmission rates	10 MBit/s or 100 MBit/s
Ethernet Duplex Mode	Half-Duplex or Full-Duplex
Connections	1
User data length	max. 252 Byte Input max. 252 Byte Output

Table 37: Technical Data EtherNet/IP Interface

ASCII Interface	Value
ASCII	Master or Slave
ASCII Duplex mode	Half-Duplex
ASCII Baud rate	50 Bit/s .. 38,4 kBit/s
ASCII Data bits	7 or 8
ASCII Stop bits	1
User data length	max. 240 Byte Send- and Receive data

Table 38: Technical Data ASCII Interface

Gateway Function	Value
Size of process data image for input data	240 bytes
Size of process data image for output data	240 bytes

Table 39: Technical Data Gateway function

9 Lists

9.1 List of Figures

9.2 List of Tables