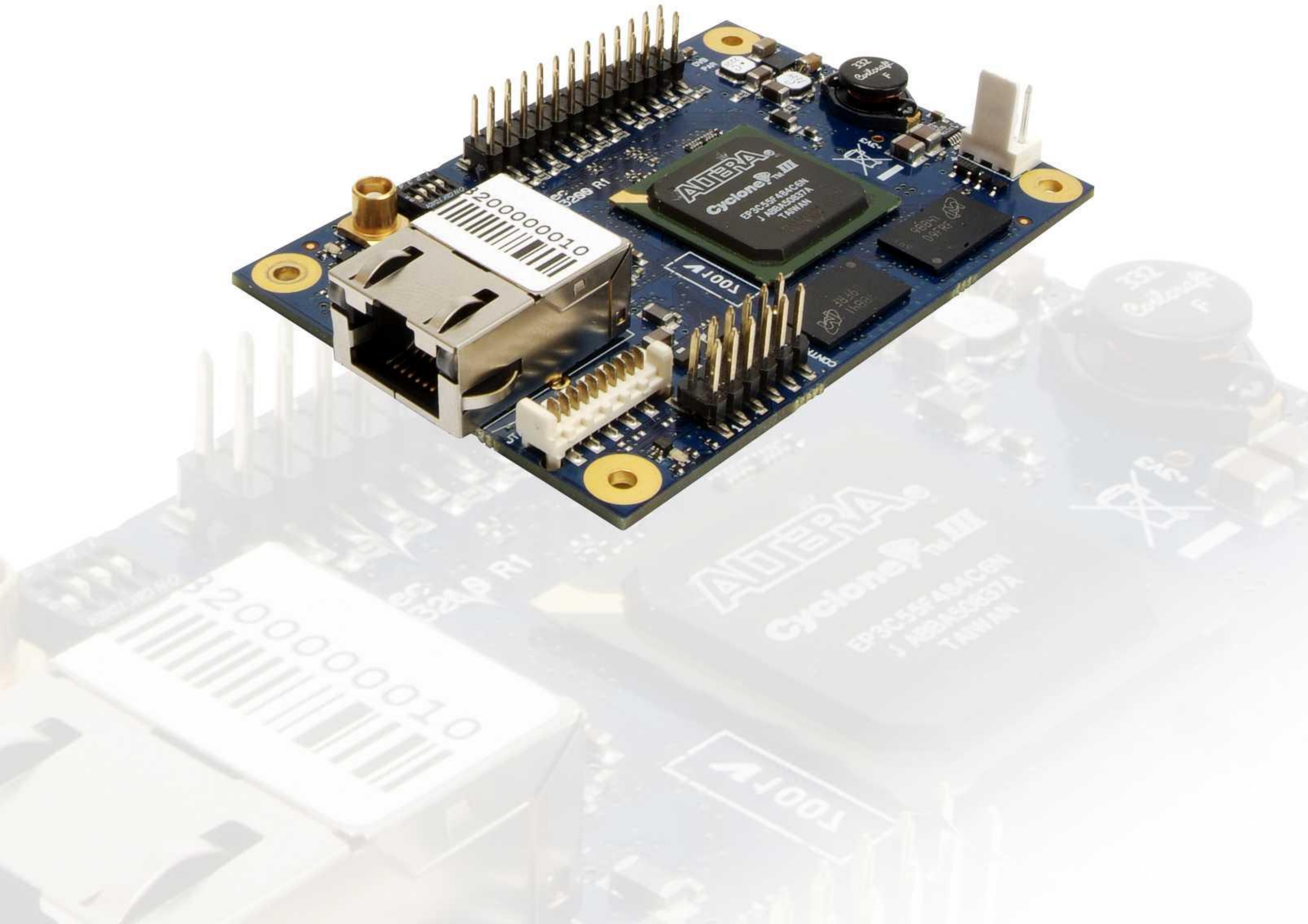
DTM-3200

OEM TS-over-IP Converter



USER **MANUAL**

April 2012

Table of Contents

Revision History

<table>
<tr><th>Version</th><th>Date</th><th>Changes</th></tr>
<tr><td>2.0</td><td>2012.03.28</td><td>Important notice for upgrading to v2.0:
Firmware update v2.0 contains both application and failsafe firmware. Any interruption of the upgrade process may result in an unusable DTM-3200 module. The first boot after this update will take longer than usual, this is normal behavior.

Fixed issues:
• DTM-3200 unicasts to all ports of a switching hub
• Category 0x81, setting 0x14 'PCR present' does not work
• Category 0x81, setting 0x15 'Rate change counter': value is incorrect
• Category 0x81, setting 0x19 ‘Delay factor’: value is incorrect
• Corruption of IP packet headers leading to occasional packet loss
• DHCP client is started twice
• The serial command protocol is case-sensitive, should be case insensitive
• After extensive communication and reboot the application firmware may get corrupted
• Dtm3200Util user interface locks up when repeatedly sending a setting to category 0x82, while no Ethernet cable has been attached

Fixed issues related to I2C control:
• Accessing other I2C devices while DTM-3200 is processing a command causes the DTM-3200 to become unresponsive
• DTM-3200 does not generate an I2C error response when an incorrect command has been issued
• DIP switches #3 and #4 have no effect when used to set I2C slave address
• An I2C master has to wait at least 10ms between issuing a command and trying to read the response

Changes affecting customer design:
• The I2C read behavior has changed: The DTM-3200 will always acknowledge its address when a read-cycle is initiated. When there is no data available to return to the master, zeroes will be returned. This also means that zeroes are appended to a message when the master reads more bytes than available
• The firmware update ASCII encoding (does not apply to I2C) has been altered and is incompatible with previous versions, including older failsafe firmware
• Enabling the DHCP client by writing an IP address with value 0.0.0.0 is no longer supported. Writing 0.0.0.0 will result in an error-response, since it is an incorrect fixed IP address. The user should use the DHCP-enable setting instead
• Every message sent by the DTM-3200 on the RS-XXX port now uses uppercase ASCII characters. Users checking for lowercase characters should change this to uppercase. e.g. "3A" instead of "3a", "R" instead of "r"

Other changes:
• Added 'VBR' mode, category 0x81, setting 0x16, data 0x03
• Changed behavior of the DHCP enable setting (category 0x03, setting 0x04)
• Added 'Volatile Storage' setting (category 0x02, setting 0x03), giving the user the option to choose between RAM or FLASH storage of settings
• Added checking for 'r' and 'R' in the read/write-field of a command message. Previously any character other than 'w' or 'W' would result in a read reply
• Added protocol checks for incorrect settings; these will result in an 'E' reply
• Removed automatic reboot at the end of the firmware update process
• Firmware update speed has been improved, compressed programming files are now supported and ASCII encoding has been altered</td></tr>
</table>

		• MPEG null packets (PID 8191) are now ignored when scanning for PCRs, to prevent incorrect detection of PCRs resulting in lock errors
1.0	2010.03.31	Initial release to the field

1. Introduction

1.1 General description

The DTM-3200 is a compact OEM module to convert ASI to or from a Transport Stream over IP (TS-over-IP or TSoIP). Next to the serial ASI interface, the unit supports a parallel interface. The direction of the conversion (TSoIP to ASI or ASI to TSoIP) can be configured programmatically through the control interface.

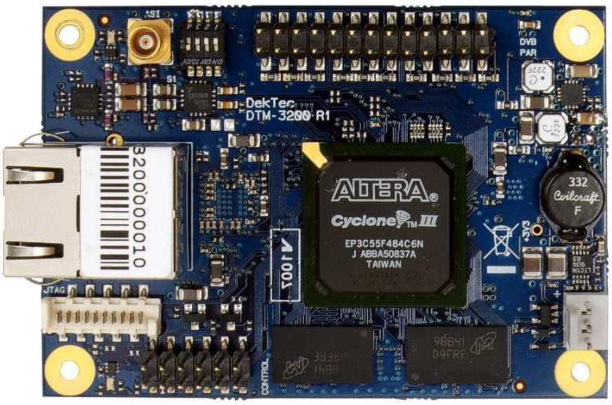


Figure 1. The PCB of the DTM-3200

A development kit (DTM-3200-DEVKIT; refer to Appendix C) is available for easy setup and experimentation with the DTM-3200.

1.2 TSoIP-to-ASI mode

When configured as TSoIP-to-ASI converter, the unit accepts unicast and multicast streams over its Gigabit-Ethernet port. Key features include de-encapsulation of UDP or RTP, IP jitter removal and error correction according to SMPTE 2022-1. The resulting stream is transmitted simultaneously on the ASI output connector and the parallel pin header.

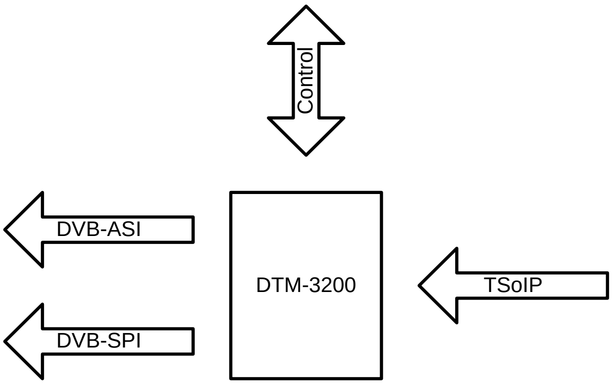


Figure 2. The DTM-3200 configured as IP-to-ASI converter

1.3 ASI-to-TSoIP mode

When configured as ASI-to-TSoIP converter, the DTM-3200 accepts ASI or parallel input. The ASI input is selected automatically when a signal is available; otherwise the parallel input is used. Key features include encapsulation of UDP or RTP, controlled scheduling of IP packets to prevent IP jitter (zero jitter playout), and adding forward error correction (FEC) according to SMPTE 2022-1.

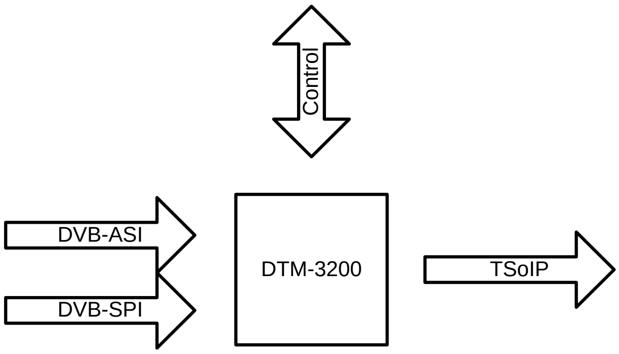


Figure 3. The DTM-3200 configured as ASI-to-IP converter

1.4 Control

The unit can be managed and controlled through one of the available control interfaces: I^2C or RS-232/485/422. Settings applied through a control interface are stored persistently on the unit and will be automatically be reloaded after a power cycle. It is not possible to configure the device via the Ethernet interface.

There are three ways to control the DTM-3200:

1. From a development PC using the serial RS-232 control interface. This way of controlling will mostly be used for experimental usage, or for pre-configuring the DTM-3200.
2. Using a controller on-board of the equipment that uses the DTM-3200 for I/O conversion. In this case I^2C is a plausible choice, but the other serial interfaces may also be used.
3. Stand-alone mode. The DTM-3200 is pre-configured and no dynamic control is applied.

Two Windows-based test tools are available:

1. ***Dtm3200Util*** – GUI tool to view status and control settings of the DTM-3200. The tool can also be used to upload firmware. Dtm3200Util is especially convenient for experimentation with the DTM-3200.
2. ***DtmUart*** – Command-line tool to send commands to the DTM-3200, and inspect the return messages. Multiple commands can be combined in a script to apply a group of settings in one go. ***DtmUart*** is useful for studying the low-level structure of command and return messages, to debug control handlers. It is also useful to apply a pre-defined group of setting values from a script.

1.5 Theory of operation

Essentially, the DTM-3200 consists of two subsystems:

- A Stream Processor, which converts the IP packets to a base-band Transport Stream and outputs it as ASI and parallel, or vice versa;
- A processor subsystem that handles all internal and external control (I^2C, RS232/485/422).

The DTM-3200 is operating either as IP-to-ASI converter, or as ASI-to-IP converter.

1.5.1 IP-to-ASI converter - Functional block diagram

Figure 4 shows the functional block diagram of the DTM-3200 when it is configured as IP-to-ASI converter.

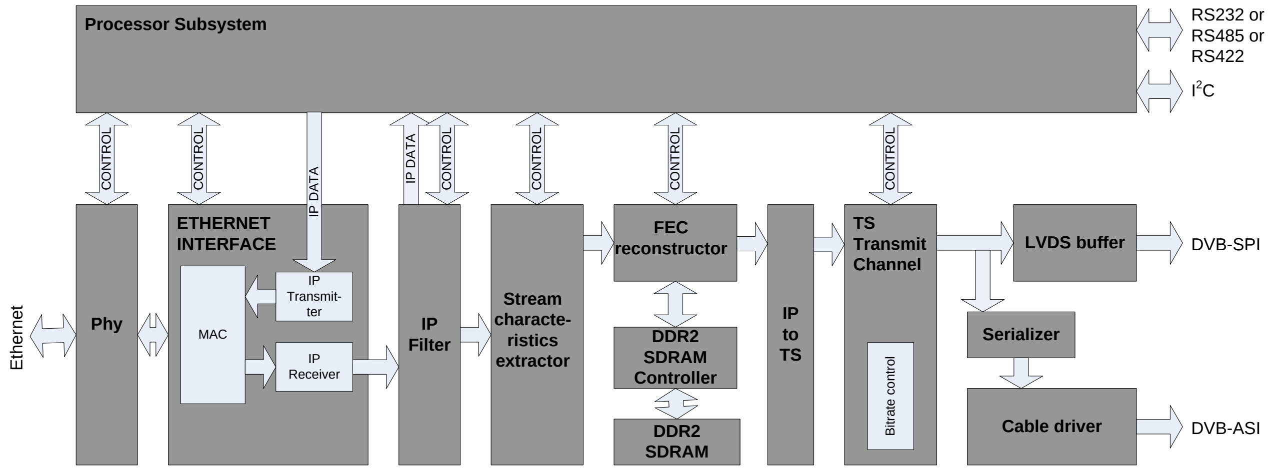


Figure 4: Functional block diagram of the DTM-3200 when configured as IP-to-ASI converter

The incoming Ethernet packets are received by the physical layer interface (PHY). The Ethernet interface checks the packets for corruption and correctness, and sends them to the IP filter that selects the desired stream. Other IP packets are sent to the processor subsystem in order to support low-level IP protocols like ARP and DHCP. From the IP Filter the transport stream data enters the stream characteristics extractor. This block will analyze the stream for specific characteristics like PCR information. With these characteristics, the DTM-3200 determines the bitrate for transmitting the stream at the output. The FEC Reconstructor uses the FEC streams (if available) to reconstruct missing packets (if any).

In the next step, the data is stored in SDRAM. Jitter on the IP input stream may cause late arrival of some IP packets. The memory is used as a buffer to 'de-jitter' the stream. The size of the de-jitter buffer can be set via the control interface. The IP stream is then converted to a Transport Stream (TS) with the correct bitrate. The resulting stream is transmitted as an ASI and as a parallel stream at the same time. The TS contents are not affected by the DTM-3200.

1.5.2 ASI-to-IP converter mode - Functional block diagram

Figure 5 shows the functional block diagram of the DTM-3200 when it is configured as ASI-to-IP converter.

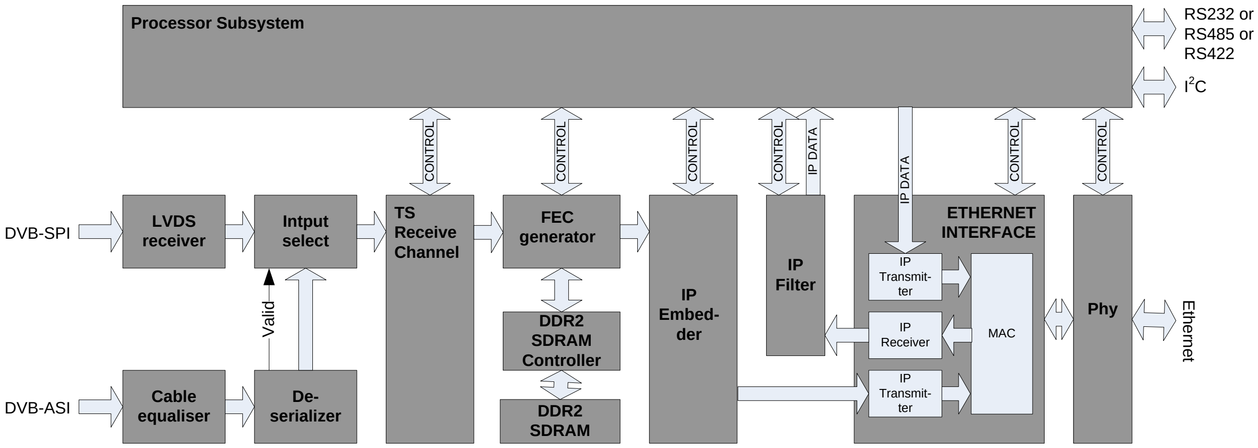


Figure 5: Functional block diagram of the DTM-3200 when configured as ASI-to-IP converter

The incoming stream is received either through the serial or the parallel input. Only one stream can be received at a time. If the Deserializer detects a valid input stream, the Input Selector selects the DVB-ASI input, otherwise the Input Selector selects the parallel input. The TS Receive Channel synchronizes to the stream and determines the packet size (188 or 204 bytes). If the DTM-3200 is configured to generate FEC packets, the FEC Generator will create row and column FEC data. The IP Embedder embeds the TS packets in IP packets. Finally, the PHY transmits the IP packets through the Ethernet interface.

1.6 List of abbreviations

ASI	Asynchronous Serial Interface. Shorthand for DVB-ASI.
auto-MDIX	Automatic medium-dependent interface crossover. Technique to automatically detect the type of network cable: straight-through or crossover.
DHCP	Dynamic Host Configuration Protocol. Network protocol to automatically assign an IP address to a network port from a server.
DVB	Digital Video Broadcasting
FEC	Forward Error Correction
IP	Internet Protocol
MAC	Media Access Controller
Mbps	Megabit per second
NA	Not Applicable
NC	Not Connected
PCR	Program Clock Reference
R/W	Read / Write
RO	Read Only

RTP	Real-time Transport Protocol
SPI	Synchronous Parallel Interface. Shorthand for DVB-SPI.
TSoIP	Transport Stream over IP
UDP	User Datagram Protocol
WO	Write Only

1.7 References

[1] SMPTE-2022-1, Forward Error Correction for Real-Time Video/Audio Transport Over IP Networks

[2] SMPTE-2022-2, Unidirectional Transport of Constant Bit Rate MPEG-2 Transport Streams on IP Networks

2. Getting Started

2.1 Introduction

This section provides a walkthrough for getting started with the DTM-3200. Two set-ups are described: one for receiving a Transport Stream over IP (TSoIP) and converting it to ASI, and one the other way around, receiving ASI and converting it to TSoIP.

The description below assumes that the DTM-3200 development kit (see Appendix B) is used to control the DTM-3200 over USB from a development PC. The GUI tool ***Dtm3200Util*** is used to apply settings and observe status.

2.2 Configuration #1: Converting TSoIP to ASI

This set-up will receive a TSoIP stream and transmit the stream on the ASI interface.

2.2.1 Test set-up

For testing this configuration, a TSoIP transmitter[1] should be present on the network to generate a TSoIP test signal. To observe the output of the DTM-3200, an ASI receiver is helpful.

This tutorial assumes that the network dynamically assigns IP addresses through DHCP, and that the source generates a TSoIP stream with the following parameters:

Protocol	UDP
Destination address	224.1.1.1
UDP Port	5678

2.2.2 Configuring the TSoIP to ASI conversion

Using ***Dtm3200Util***, configure the DTM-3200 as shown in the table below. The receive process is first disabled before changing configuration parameters. When all parameters have been set, the DTM-3200 is enabled again.

	Category	Setting	Value
1	0x81: IP receive	2: Enable	0 = Off
2	0x81: IP receive	1: Addressing method	1 = Multicast
3	0x81: IP receive	4: FEC enable	0 = Off
4	0x81: IP receive	10: UDP port	5678
5	0x81: IP receive	11: IP-to-Output delay	100 = 100ms
6	0x81: IP receive	12: Multicast address	224.1.1.1
7	0x84: ASI/par. output	1: Packet size	0 = 188-byte MPEG-2 packets
8	0x81: IP receive	2: Enable	1 = On

[1] If you do not have a suitable TSoIP transmitter and/or ASI receiver, this functionality can for example be realized with a PC and a DekTec DTA-2160 I/O card in it. Please consult your local DekTec representative for more information.

The DTM-3200 will now receive a TSoIP stream and transmit this stream on the ASI port. The status LED will be green to indicate successful transmission of the TS on the ASI output. If no TSoIP stream is received, the status LED will be red.

2.3 Configuration #2: Converting ASI to TSoIP

This set-up will receive a stream on the ASI interface and transmit the stream over IP.

2.3.1 Test set-up

The equipment and tools are similar to the ones listed in §2.2.1 In this case we need an external ASI source and a TSoIP receiver on the network.

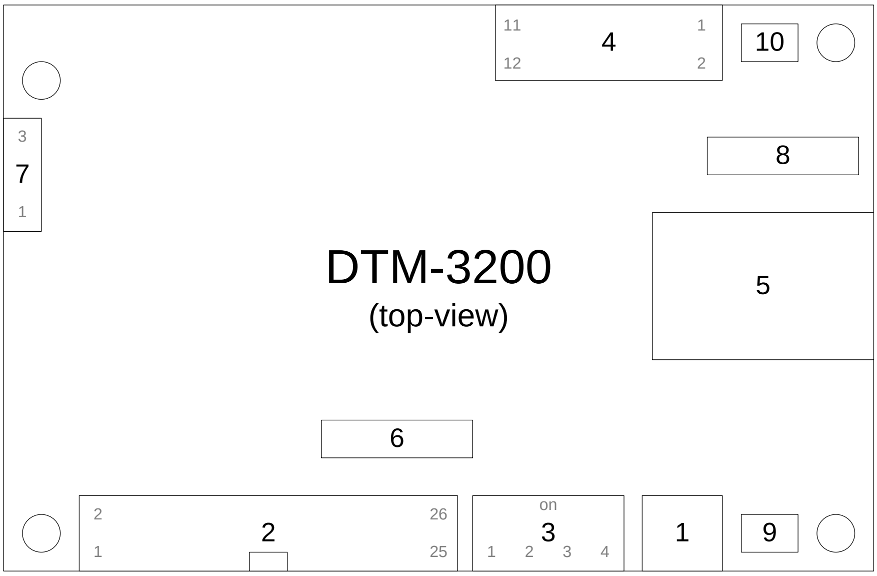
2.3.2 Configuring the ASI to TSoIP conversion

The settings below (sequential order) set up the DTM-3200 for ASI to IP conversion. No FEC will be generated in this example.

	Category	Setting	Value
1	0x82: IP transmit	1: Enable	0 = Off
2	0x82: IP transmit	3: FEC Enable	0 = Off
3	0x82: IP transmit	6: IP Address	224.1.1.1
4	0x82: IP transmit	8: UDP Port	5678
5	0x82: IP transmit	9: #TP per IP	3 = 3 Transport Packets per IP pckt
6	0x82: IP transmit	10: Protocol	0 = UDP
7	0x82: IP transmit	13: Time to Live	100
8	0x83: ASI/par. input	1: Packet Size	0 = 188-byte MPEG-2 packets
9	0x82: IP transmit	1: Enable	1 = On

3. Layout and Installation

3.1 Physical layout



1	ASI I/O	MCX 75Ω	ASI input or output
2	Parallel TS	26-pin box header 2.54mm pitch	Parallel Transport Stream input or output; Similar to DVB-SPI, but with fixed 27-MHz clock instead of TS clock
3	Switches	DIP	Selection of baud rate and control interface: RS-232 or RS-485/422
4	Control	12-pin header 2.54mm pitch	RS-232, RS-485/422 and I^{2}C interface for board control
5	Ethernet	RJ-45	Ethernet port for TS-over-IP transmission or reception
6	Identifier		Type and revision number
7	Power		Power and reset
8	Factory		Factory program connector; Not used in normal operation
9	Stream status		Stream status LED
10	DTM-3200 status		DTM-3200 Status LED

3.2 Mechanical dimensions

See Appendix A.

3.3 Hardware installation

3.3.1 Mechanical installation

The unit can be mounted onto a support plate by means of four 2.5 mm bolts and appropriate spacers. Ensure that there is sufficient airflow to provide cooling of the board.

3.3.2 ASI connector

The ASI connector (1) is an MCX connector with an impedance of 75 ohm.

3.3.3 Parallel-TS connector

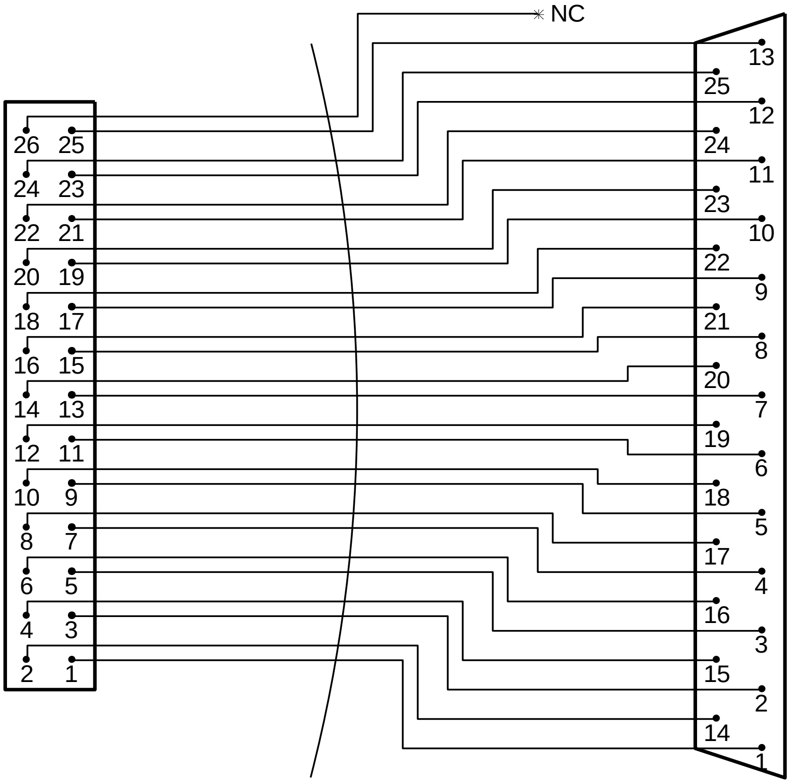
The pinning of the parallel Transport-Stream connector (2) is displayed in the table below. The signal levels and pin numbering the same as DVB-SPI.

Warning: Although the pinning is the same, the parallel interface is not compatible with DVB-SPI, because the clock of the DTM-3200's parallel interface is fixed to 27MHz (in DVB-SPI the clock is the TS rate in byte/sec).

Pin	Signal
1	DCLK+
3	GND
5	D7+
7	D6+
9	D5+
11	D4+
13	D3+
15	D2+
17	D1+
19	D0+
21	DVALID+
23	PSYNC+
25	GND

Pin	Signal
2	DCLK-
4	GND
6	D7-
8	D6-
10	D5-
12	D4-
14	D3-
16	D2-
18	D1-
20	D0-
22	DVALID-
24	PSYNC-
26	GND

The pin assignment of the pin header has been chosen in such a way that a flatcable with a sub-D male flatcable connector (25-way sub-D; ISO 2110) at the other end can be connected directly to the board.



3.3.4 DIP switches

The DIP switches permit configuration of I^2C device address, RS-232 or RS-485/422 mode and the baud rate. The state of the DIP switches is read at power up only. Changing the DIP switch settings while power is on has no effect.

Please note that it is not required to select between I^2C and RS-XXX: The DTM-3200 will automatically use the interface on which it detects activity.

DIP switches for RS-232/485/422

For RS-232 or RS-485/422, the DIP switches have the following meaning:

Switch #	Description
1	Device address bit 0 – LSB (off = 0, on = 1)
2	Device address bit 1 – MSB (off = 0, on = 1)
3	RS-232 or RS-485/422 (off = RS-232; on = RS-485/422)
4	Baud-rate (off = 9600; on = 115200)

The device address bits are used for RS-485/422 only. The device address is 0x40 + two LSBs as selected by the DIP switches. This means that the device address range is between 0x40 and 0x43.

DIP switches for I²C

For I²C communication, the DIP switches have the following meaning:

Switch #	Description
4, 3, 2, 1	Device address bit 3..0 (DIP switch off = 0, on = 1)

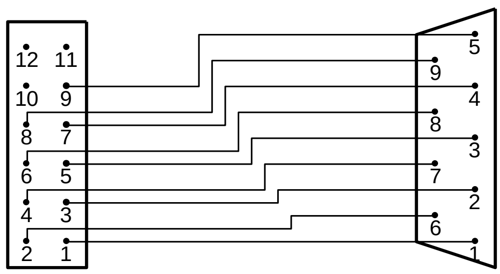
The I²C device address is 0x40 + four LSBs as selected by the DIP switches. This means that the I²C device address range is between 0x40 and 0x4F.

3.3.5 Control connector

The pinning of the control connector is shown in the table below. It's a single row pin header for connecting the RS-232, RS-485/422 or I²C control bus. The exact function of the signals depends on whether RS-232 or RS-485/422 mode is selected by the DIP switches.

Pin	RS-232 mode	RS-485/422 mode
1	NC	NC
2	NC	NC
3	TX	TX/RX-
4	CTS	NC
5	RX	NC
6	RTS	TX/RX+
7	NC	NC
8	NC	NC
9	GND	GND
10	NC	NC
11	SDA	SDA
12	SCL	SCL

The pinning of this connector has been chosen in such a way that a 9-way flatcable with a press-fit sub-D flatcable connector can be connected directly to pin 1 – 9, see the figure below.



12-pin female header 9-pin female sub-D

An I²C controller can be connected to SDA and SCL on pin 11 and 12, with signal ground on pin 9.

3.3.6 Ethernet connector

The Ethernet connector is a standard shielded RJ45 jack with two status LEDs.

Link speed LED	
Orange	1000Mbps
Green	10 or 100Mbps (half duplex)
Off	No signal

The link-activity LED flashes whenever an Ethernet packet is received or transmitted.

A standard Cat5E (or higher) patch cable can be used to connect the DTM-3200 to a network. Either a straight-through or cross-over network cable can be used; the type of cable will be automatically recognized (auto-MDIX operation). The DTM-3200 will automatically select the link speed of the connected network (10/100/1000Mbps).

3.3.7 Power connector

The DTM-3200 must be powered from an external source with a voltage between 5 and 24V DC. Power consumption is max. 5W. The pinning of the power connector is shown in the table below:

Pin	Description
1	Positive power connection
2	Ground
3	Reset connection

The board can be reset by connecting pin 3 to ground for at least 100ms. The connector type is the Molex KK series 2.54 mm pitch.

3.3.8 Stream status LED

This LED indicates the status of the ASI and parallel stream. The following colors are used for status indication:

ASI output mode	
Short red/green flashes	No output generated on ASI and parallel outputs
Red/green	Generating live output on ASI and parallel outputs

Note: When the 'Rate Estimation' (category 0x81, setting 0x16) is set to 3, bitrate estimation in the DTM-3200 is turned off and the stream status LED will always be Red/green.

ASI input mode	
Short green flashes	No carrier detected on ASI or parallel input
Long green flashes	Carrier detected but no data on ASI or parallel input
Green	Valid signal detected on ASI or parallel input
Red	Erroneous signal detected on ASI or parallel input

3.3.9 DTM-3200 status LED

This LED indicates the current status of the DTM-3200. The following colors are used for status indication:

Red	Running in failsafe mode
Red/green	Booting
Green	Running in operational mode

4. Device Configuration and Monitoring

4.1 Control interfaces

The DTM-3200 can be configured and monitored using I^2C, RS-232, RS-485 or RS-422. DIP switch #3 selects between RS-232 and RS-422 or RS-485. It is not required to select between I^2C and RS-XXX: The DTM-3200 will automatically use the interface on which it detects activity.

All control interfaces use the same command and response protocol that is described below.

4.2 Command protocol

Commands and responses are wrapped into a frame structure that contains address, category, setting, read/write, index (optional) and data (optional). The DTM-3200 accepts uppercase and lowercase characters, but will always respond in uppercase.

4.2.1 Command protocol on RS-232 / RS-422 / RS-485

Field	Format	Description
Start	ASCII character STX (0x02)	ASCII "start of text" character
Address	2 hex digits[2]	8-bit address, with the 6 MSBs fixed to 0x40 and the 2 LSBs configurable using DIP switches 1 and 2
Category	2 hex digits	Selects a "category" of settings
Setting	2 hex digits	Selects a setting within the selected category
Read/Write	ASCII character 'R' or 'W'	'R' for read and 'W' for write
Index	4 hex digits	(Optional) Provides an extra index parameter, e.g. to indicate the channel number[3]
Data	n hex digits / n ASCII characters[4]	The data written or read. The data length is variable for each setting. In case of a write operation, the data is a (negative) acknowledgement
Checksum	2 hex digits	This is the least significant byte of the two's complement[5] sum of all characters in the message, excluding the STX and ETX characters and the checksum itself
End	ASCII character ETX (0x03)	

[2] Hex digits are the ASCII characters 0…9 and A…F, concatenated to form a single hexadecimal value.
[3] The DTM-3200 supports a single channel only, so index is used as a channel number, it's always 0.
[4] Since firmware version 1.4 the firmware update setting uses ASCII characters 128 to 255 for sending the firmware data.
[5] Invert all bits and add one.

Figure 6 below shows the structure of a command written through the serial interface. If the command is a read-command, the data may be omitted. If the category does not require an index, the index must be omitted.

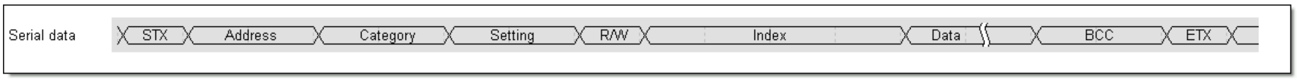


Figure 6. Command on an RS-XXX serial control interface

All commands successfully sent to the DTM-3200 are answered with a copy of the command including the data bytes.

When an incorrect checksum is detected, the DTM-3200 will not return an answer. When protocol errors are detected, e.g. a combination of a valid category with an invalid setting, the R/W byte of the reply is replaced with the ASCII character 'E' and the data is removed from the message.

4.2.2 Command protocol on I^2C

Field	Format	Description
Start	S	Standard I^2C start condition
Address	I^2C address byte	7-bit I^2C address followed by the I^2C R/W bit, which is set to 0 and 1 in the command- and response sequence respectively
Category	1 byte	Selects a "category" of settings
Setting	1 byte	Selects a setting within the selected category
Read/Write	1 byte	0x01 for read and 0x00 for write
Index	2 bytes	(Optional) Provides an extra index parameter, e.g. to indicate the channel number[6]
Data	n bytes	The data written or read. The data length is variable for each setting. In case of a write operation, the actual data is returned as a (negative) acknowledgement
Checksum	1 byte	This is the least significant byte of the two's complement of the sum of the 7-bit I2C slave address and all data-bytes in the I2C message (excluding the checksum). The I2C R/W bit is not included, an incorrect value of this bit would cause the checksum to be not received at all.
End	P	Standard I^2C stop condition. A repeated start condition can be used at all times to concatenate multiple I^2C read / write actions

[6] The DTM-3200 supports a single channel only. Whenever index is used as a channel number, it shall/will be set to 0.

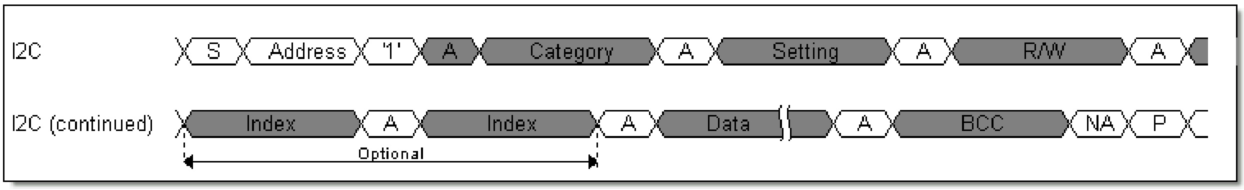
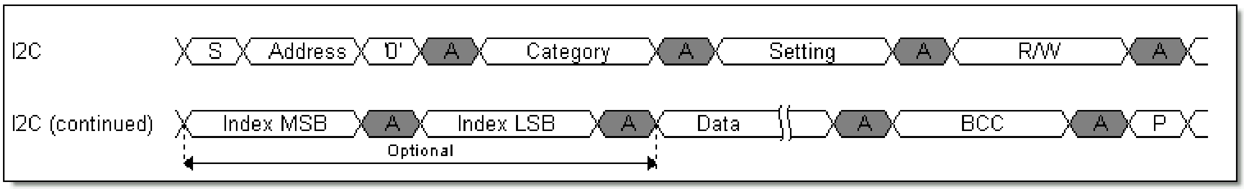


Figure 7 below shows the sequence to send a command over I²C to the DTM-3200. If the command is a read-command, the data may be omitted. If the category does not require an index, the index must be omitted.



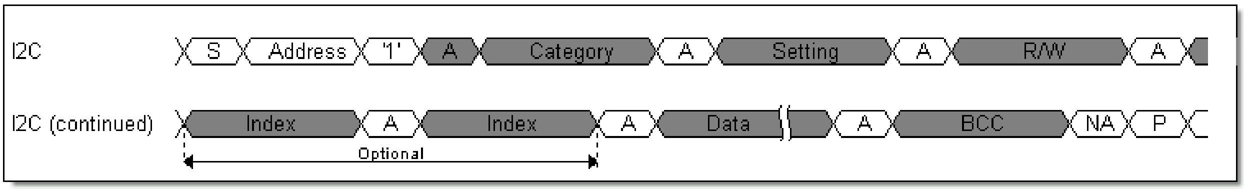


Figure 7. Command (upper sequence) and response (lower) sequence for I²C

When an incorrect checksum is detected, the DTM-3200 will not return an answer. When protocol-errors are detected, e.g. a combination of a valid category with an invalid setting, the R/W byte of the reply is replaced by the ASCII character 'E' and the data is removed from the message.

When a master starts writing to a device while the previous command is still being executed, this device replies with a NACK.

When a master starts reading from a device and there is no answer available (yet), the device will reply with the value 0x00. The value 0x00 will be returned until an answer is available and the master has initiated a new read transaction.

Note: In DTM-3200 package versions v1.3 and lower, in the case mentioned above, the device replies with a NACK instead of the value 0x00.

S and P are the standard I²C start and stop conditions. A repeated start condition can be used at all times to concatenate multiple I²C read / write actions.

4.3 Manageable items

The tables in this section provide lists of variables that can be configured and/or monitored using the I²C or RS-232/485/422 interface. The 'Access' column indicates whether the variable is Read Only (RO), Write Only (WO), Read/Write (R/W) or not applicable (n.a.).

When, in the 'Access' column, the text "(V)" is written after the access type, the variable can be stored either in a volatile or non-volatile manner, depending on the 'Volatile Storage' setting (category 0x02, setting 0x03). The volatility of the variables without "(V)" cannot be controlled.

4.4 Categories

Manageable Items – Categories			
Nr	Settings Category	Description	Index?
0x01	**Device**	Device properties	No
0x02	**Configuration**	Overall configuration	No
0x03	**Network**	Network settings	No
0x80	**Firmware update**	Firmware update	Yes
0x81	**IP receive**	IP receive settings	Yes
0x82	**IP transmit**	IP transmit settings	Yes
0x83	**ASI/par. input**	ASI/parallel input settings	Yes
0x84	**ASI/par. output**	ASI/parallel output settings	Yes

4.4.1 Data types

Manageable Items – Data types			
Type	Description	RS-232 / RS-485 / RS-422	I²C
int8	8-bit unsigned integer	2 hex digits	1 byte
int16	16-bit unsigned integer	4 hex digits	2 bytes
int32	32-bit unsigned integer	8 hex digits	4 bytes
int64	64-bit unsigned integer	16 hex digits	8 bytes
IP addr	IP address	8 hex digits	4 bytes
string	String of ASCII characters	Variable	string

All data types are sent with the most-significant byte first.

4.4.2 Device properties

Manageable Items - Category 0x01 – Device properties				
Nr	Variable	Description	Access	Type
0x01	**FPGA version**	Version number of the FPGA code on-board of the DTM-3200	RO	int8
0x02	**Hardware version**	Hardware version number	RO	int8
0x03	**Firmware version**	Firmware version: the major version is encoded in the tens, the minor version in the units, e.g. '10' indicates v1.0	RO	int32
0x04	**Serial number**	Unique serial number for this device, e.g. 3200.000.027	RO	int32
0x05	**Type**	Device type number, e.g. 3200	RO	int32

4.4.3 Overall configuration

The configuration settings are used to switch the mode and application of the DTM-3200. When settings 0x01 'Mode' or 0x02 'Application' are changed the DTM-3200 is rebooted automatically.

Manageable Items - Category 0x02 – Overall configuration				
Nr	Variable	Description	Access	Type
0x01	**Mode**	0 = ASI to TSoIP 1 = TSoIP to ASI	R/W	int8
0x02	**Application**	0 = Failsafe application 1 = Normal operation See section 4.5.3 for further information	R/W	int8
0x03	**Volatile storage**	0 = Disabled, settings stored in flash 1 = Enabled, settings stored in ram	R/W	int8

When the 'Volatile storage' variable is set to 1, any changes made to settings with access type R/W (V), will be lost after reboot.

4.4.4 Network settings

The network settings are used to configure the IP address, subnet mask and gateway for the network connection. The MAC address can be read from a read-only variable.

Manageable Items - Category 0x03 – Network settings				
Nr	Variable	Description	Access	Type
0x01	**IP address**	IP address of the DTM-3200	R/W	IP addr
0x02	**Subnet mask**	Subnet mask	R/W	IP addr
0x03	**Gateway**	Gateway	R/W	IP addr
0x04	**DHCP enabled**	0 = Use static IP address 1 = Use DHCP	R/W	int8
0x05	**Reboot**	1 = Reboot	WO	int8
0x06	**MAC-address**	MAC address of the DTM-3200	RO	int48

When the 'DHCP enabled' behavior variable is read, it indicates whether a static IP address or DHCP is used.

For DTM-3200 package versions up to **v1.3**, the following behavior applies:

- To switch from a statically assigned IP address to DHCP, write 1 to the 'DHCP enabled' variable. Effectively, this action causes 0.0.0.0 to be written to IP address, Subnet mask and Gateway. Writing 0.0.0.0 to IP address therefore has the same effect as writing 1 to 'DHCP enabled': In both cases, the DTM-3200 switches to a dynamically assigned IP address through DHCP (after a reboot).
- To switch from DHCP to a statically assigned IP address, write the new static IP address, subnet mask and gateway to the corresponding variables. It must be noted that switching to a static IP address cannot be achieved by writing 0 to the 'DHCP enabled' field.

Starting with DTM-3200 package version **v1.4**, the 'DHCP enabled' behavior has been altered, to provide the user with a more convenient way to enable or disable the DHCP client:

- The 0.0.0.0 IP address behavior mentioned before still applies, but the 'DHCP enable' setting can now be used to turn on and turn off the DHCP client.
  - When 1 is written to 'DHCP enabled', the IP address is set to 0.0.0.0 and (after a reboot) the DHCP client will start looking for a DHCP server to acquire an IP address.
  - When 0 is written to 'DHCP enabled', the DTM-3200 retrieves the most recently manually entered IP-address (fixed IP) and uses this address as its fixed IP-address (after a reboot).
- When a fixed IP-address has never been entered before, writing 0 to 'DHCP enabled' will cause the DTM-3200 to use a hardcoded factory default address (192.168.144.120 / 255.255.255.0) as its fixed IP address (after a reboot). Naturally this default IP address, selected by writing 0 to 'DHCP enabled', can be overridden by entering an IP address manually.

The DTM-3200 must be rebooted before a new IP-address assignment takes effect. Rebooting the device is initiated by writing 1 to the 'Reboot' setting, or by a power cycle.

4.4.5 Firmware update

To upgrade the DTM-3200, the new firmware has to be uploaded in "file parts". The firmware-update settings are used to upload the firmware file and control the programming process.

Manageable Items - Category 0x80 – Firmware update				
Nr	Variable	Description	Access	Type
0x01	**Part number**	Sequence number of most recently uploaded file part	RO	int16
0x02	**File part**	Data in file part; return message contains the file-part sequence number	R/W	1..1024 bytes
0x03	**Update device**	1 = Start device update and reboot	WO	int8
0x04	**Abort**	1 = Remove uploaded file	WO	int8
0x05	**Progress**	Programming progress (%)	RO	int8
0x06	**Error**	Error code (0 = no error, 1 = error)	RO	int8
0x07	**Number of parts**	Number of parts to upload the entire firmware file	R/W	int16
0x08	**Part size**	Number of bytes per file part	R/W	int16
0x09	**Remaining**	Number of bytes in the final file part	R/W	int16

This category requires an index to be sent with each command. For the "File part" setting, index should be the file part number. For the other settings, index is not used.

For communication through I^2C the parts may contain at most 250 data-bytes; For communication through RS-XXX the parts may contain at most 1024 data bytes.

For firmware version 1.3 and lower, each data byte was sent as 2 ASCII characters (1 character per nibble, using 0..9 and A..F), so 1024 data bytes were sent as 2048 ASCII characters.

To improve the firmware update speed this was changed into a different encoding.

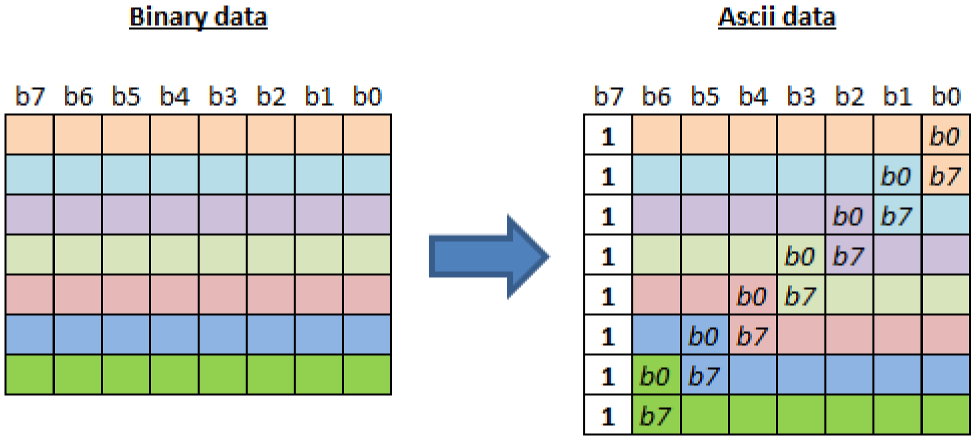


Figure 8. ASCII encoding for firmware update data

Starting from firmware version 1.4, for each 7 bits of data one 8-bit ASCII character is send, where the MSB of the ASCII character is set to 1 (extended ascii range). The translation is illustrated in Figure 8.

This encoding is only applied to the data-part of the "File part" message. (category 0x80, setting 0x02). The rest of the messages are encoded as before.

An example of a firmware update sequence and an example routine for the above translation can be found in paragraph 4.5.

4.4.6 IP receive settings

The IP receive settings are available only if the DTM-3200 is configured as IP-to-ASI converter.

Manageable Items - Category 0x81 – IP receive				
Nr	Variable	Description	Access	Type
0x01	**Addressing method**	0 = Unicast 1 = Any-source multicast	R/W (V)	int8
0x02	**Enable**	0 = Disable IP-to-ASI conversion 1 = Enable IP-to-ASI conversion	R/W (V)	int8
0x03	**FEC delay**	FEC delay in milliseconds	RO	int32
0x04	**FEC enable**	0 = Disable FEC decoding 1 = Enable FEC decoding	R/W (V)	int8
0x05	**FEC #columns**	0..20; #FEC columns	RO	int16
0x06	**FEC #rows**	0..20; #FEC rows	RO	int16
0x07	**IP jitter tolerance**	1..120ms; The time (in ms) that the DTM-3200 waits to receive 'late' IP packets	RO	int32
0x08	**IP lost after FEC**	Number of packets lost after FEC reconstruction	RO	int64
0x09	**IP lost before FEC**	Number of packets lost before FEC reconstruction	RO	int64

Manageable Items - Category 0x81 – IP receive				
Nr	Variable	Description	Access	Type
0x0A	**UDP port**	1..65535; UDP port number at which the DTM-3200 listens for the incoming TS	R/W (V)	int16
0x0B	**IP-to-output delay**	1..9942ms; Delay (ms) from IP input to ASI output in milliseconds	R/W (V)	int16
0x0C	**Multicast address**	When 'Addressing method' is 1, this is the multicast address at which the DTM-3200 listens for the incoming TS	R/W (V)	IP addr
0x0D	**#TP per IP**	Number of transport packets per IP packet	RO	int8
0x0E	**Status**	Current operational status: 0 = Channel is enabled; no errors detected 1 = Channel has been disabled 2 = Channel is enabled, but there is a problem with the processing of the received IP stream	RO	int8
0x0F	**Protocol**	Protocol used by the incoming IP stream: 0 = UDP 1 = RTP	RO	int8
0x10	**Index**	Channel index; always 1 for the DTM-3200 (because it supports a single channel only)	RO	int32
0x11	**Output type**	Identifies the type of output interface: 1 = ASI output 2 = parallel output	RO	int8
0x12	**Packet size**	0 = 188 bytes 1 = 204 bytes	RO	int8
0x13	**Bitrate**	Estimated bitrate (in bps@188) of the incoming TS	RO	int32
0x14	**PCR present**	0 = No PCRs in incoming TS 1 = PCRs found in incoming TS	RO	int8
0x15	**Rate-change counter**	Counter that keeps the number of bitrate changes detected on this channel. During normal operation this counter should remain constant	RO	int32

Manageable Items - Category 0x81 – IP receive				
Nr	Variable	Description	Access	Type
0x16	**Rate-estimation mode**	0 = Bitrate is estimated using PCRs if possible, otherwise the bitrate is estimated without PCRs 1 = Bitrate estimation is disabled; Packets are buffered, and transmitted with a fixed delay 2 = Bitrate is estimated without the use of PCRs 3 = Bitrate estimation is disabled; Packets are routed to the ASI port immediately	R/W (V)	int8
0x17	**Jitter error counter**	Counter that keeps the number of IP jitter errors for this channel. A jitter error occurs when an incoming IP packet has a jitter that exceeds the Jitter Tolerance. During normal operation this counter should remain constant	RO	int32
0x18	**Lock error counter**	Counter that keeps the number of channel restarts. A channel is restarted only under adverse circumstances, like large jitter of IP packets in combination with packet loss. A channel restart causes a signal interruption for this channel of about 1 second. During normal operation this counter should remain constant	RO	int32
0x19	**Delay factor**	The channel's delay factor in us. Delay factor is a measure of the maximum jitter on the IP packets received for this channel	RO	int32

This category requires an index to be sent with each command.

4.4.7 IP transmit settings

The IP transmit settings are available only if the DTM-3200 is configured as ASI-to-IP converter.

Manageable Items - Category 0x82 – IP transmit				
Nr	Variable	Description	Access	Type
0x01	**Enable**	0 = Disable 1 = Enable	R/W (V)	int8
0x03	**FEC enable**	0 = Disable generation of FEC packets 1 = Enable generation of FEC packets	R/W (V)	int8
0x04	**FEC #columns***	1..20; #FEC columns generated	R/W (V)	int16
0x05	**FEC #rows***	4..20; #FEC rows generated	R/W (V)	int16

Manageable Items - Category 0x82 – IP transmit				
Nr	Variable	Description	Access	Type
0x06	**IP address**	Destination IP address for the TSoIP packets	R/W (V)	IP addr
0x07	**IP address status**	0 = Destination IP address has been resolved to a MAC address 1 = Failed to resolve destination IP address	RO	int8
0x08	**UDP port**	0..65535; Destination UDP port number	R/W (V)	int16
0x09	**#TP per IP**	1..7; Number of transport packets to be generated per TSoIP packet	R/W (V)	int8
0x0A	**Protocol**	Protocol used for the TSoIP output: 0 = UDP 1 = RTP	R/W (V)	int8
0x0B	**Index**	Channel index; always 1 for the DTM-3200 (because it supports a single channel only)	RO	int32
0x0C	**Input type**	Identifies the type of input interface: Always 1 (= ASI input) for DTM-3200	RO	int8
0x0D	**Time to live**	0..255; The generated IP packets will use this TTL (=Time To Live)	R/W (V)	int8

* (FEC #columns × FEC #rows) ≤ 100

This category requires an index to be sent with each command.

4.4.8 ASI input settings

The ASI input settings are available only if the DTM-3200 is configured as ASI-to-IP converter.

Manageable Items - Category 0x83 – ASI input settings				
Nr	Variable	Description	Access	Type
0x01	**Packet size**	0 = Generate IP packets with 188-byte transport packets 1 = Generate IP packets with 204-byte transport packets 2 = Use size of incoming packets	R/W (V)	int8
0x02	**Physical port**	Physical port number of ASI input; always 1 for the DTM-3200	RO	int8
0x03	**Status**	0 = Valid ASI signal is received 1 = No carrier detected 2 = Failed to lock to ASI input 3 = Input rate is below minimum supported rate	RO	int8
0x04	**TS rate**	Transport stream rate (in bps@188) of the received ASI stream	RO	int32
0x05	**Valid count**	Number of received bytes	RO	int64
0x06	**Violation count**	Number of ASI code violation detected	RO	int64

This category requires an index to be sent with each command.

4.4.9 ASI output settings

The ASI output settings are available only if the DTM-3200 is configured as IP-to-ASI converter.

Manageable Items - Category 0x84 – ASI output settings				
Nr (hex)	Variable	Description	Access	Type
1	**ASI packet size**	0 = Generate 188-byte packets 1 = Generate 204-byte packets 2 = Use size of received packets	R/W (V)	int8
2	**Physical port**	Physical port number of ASI output; always 1 for the DTM-3200	RO	int8
3	**TS rate**	TS rate (in bps@188) of the generated ASI stream	RO	int32

This category requires an index to be sent with each command.

4.5 Firmware upgrade

4.5.1 Firmware upgrade - Phases

Updating the firmware of the device consists of three phases:

1. Uploading the file. The file has to be uploaded in "parts", one part at a time. This involves the following steps:
   a. Abort previous file upload, if any. This remove a previous uploads and gets the DTM-3200 into a default state.
   b. Set 'Number of parts': Write the number of parts of the file to upload.
   c. Set 'Part size': Write the size of each part in number of bytes.
   d. Set 'Remaining': Write the size of the last part, this is the size of the remainder of the file after all 'full' parts have been uploaded.
   e. Upload file parts: Check the index number of the last uploaded part and send the next part.
2. The actual flashing of the device firmware, as follows:
   a. Start the device update. The uploaded file is read from RAM and programmed in the flash memory of the DTM-3200.
   b. (Optional) Get progress. Read the progress of programming the flash as a percentage (optional).
   c. Check for update errors (category 0x80, setting 0x06: 0=no error, 1=error)

   Warning: Do not power off the device while flash programming is in progress.
3. When flashing the firmware is complete, the DTM-3200 should be rebooted to make the upgrade effective. A reboot of the device is initiated by writing 1 to the 'Reboot' setting (category 0x03, setting 0x05), or by a power cycle.

Note: In package versions up to **v1.3**, the DTM-3200 is rebooted automatically after completing a firmware update.

After the device has been updated and rebooted, the controller may check the status of the device/firmware using a combination of the following settings:

- Check the application. Category "***configuration settings***", setting "***application***" indicates whether the device is in normal operation mode or in ***failsafe*** mode. If the device was in normal operation mode and reboots in failsafe mode, the current application firmware is corrupt. See section 4.5.3 for more information about failsafe mode.
- Check the package version. Category "***device settings***", setting "***package version***" shows the version number of the current package.

4.5.2 Firmware upload – Example

If a file consisting of 1000 bytes must be sent in packets of 150 data-bytes, there are going to be 7 parts. The first six parts are 150 bytes long and the last part consists of 100 bytes. The table below shows the content of the communication messages. Each line represents a message and the lines are shown in chronological order.

	Category (hex)	Setting (hex)	Read/write (I^2C / RS-XXX)	Index (hex)	Data	Description
Abort	80	04	00 / 'W'	0000	1 (0x01)	
Reply	80	04	00 / 'W'	0000	1 (0x01)	
Number of parts	80	07	00 / 'W'	0000	7 (0x07)	
Reply	80	07	00 / 'W'	0000	7 (0x07)	
Part size	80	08	00 / 'W'	0000	150 (0x96)	
Reply	80	08	00 / 'W'	0000	150 (0x96)	
Remaining	80	09	00 / 'W'	0000	100 (0x64)	
Reply	80	09	00 / 'W'	0000	100 (0x64)	
File part	80	02	00 / 'W'	0001	File bytes: 0 - 149	
Reply	80	02	00 / 'W'	0001	1 (0x0001)	
File part	80	02	00 / 'W'	0002	File bytes: 150 - 299	
Reply	80	02	00 / 'W'	0002	2 (0x0002)	
File part	80	02	00 / 'W'	0003	File bytes: 300 - 449	
Reply	80	02	00 / 'W'	0003	3 (0x0003)	
File part	80	02	00 / 'W'	0004	File bytes: 450 - 599	
Reply	80	02	00 / 'W'	0004	4 (0x0004)	
File part	80	02	00 / 'W'	0005	File bytes: 600 - 749	
Reply	80	02	00 / 'W'	0005	5 (0x0005)	
File part	80	02	00 / 'W'	0006	File bytes: 750 - 899	
Reply	80	02	00 / 'W'	0006	6 (0x0006)	
File part	80	02	00 / 'W'	0007	File bytes: 900 - 999	
Reply	80	02	00 / 'W'	0007	7 (0x0007)	
Update device	80	03	00 / 'W'	0000	1	
Reply	80	03	00 / 'W'	0000	–	
Progress	80	05	01 / 'R'	0000	–	Update
Reply	80	05	01 / 'R'	0000	50 (0x00)	at 0%
Progress	80	05	01 / 'R'	0000	–	Update
Reply	80	05	01 / 'R'	0000	50 (0x32)	at 50%
Progress	80	05	01 / 'R'	0000	–	Update
Reply	80	05	01 / 'R'	0000	100 (0x64)	at 100%
Error	80	06	01 / 'R'	0000	–	Update
Reply	80	06	01 / 'R'	0000	0 (0x00)	successful

4.5.3 Binary data to Ascii data function

To convert the program data (File bytes) to ASCII the following function could be used.

```
unsigned int BinToAsc128(unsigned char* bin_data, unsigned int len, char* returnData)
{
  unsigned int bytes_todo, i;
  bytes_todo = len;

  while(bytes_todo > 0)
```

```
 {
  if(bytes_todo >= 7)
  {
   *returnData++ = 0x80 + (((*(bin_data + 0)) & (0x7F >> 0) ) << 0);
   *returnData++ = 0x80 + (((*(bin_data + 1)) & (0x7F >> 1) ) << 1) + ((*(bin_data + 0) & ~(0x7F >> 0)) >> 7);
   *returnData++ = 0x80 + (((*(bin_data + 2)) & (0x7F >> 2) ) << 2) + ((*(bin_data + 1) & ~(0x7F >> 1)) >> 6);
   *returnData++ = 0x80 + (((*(bin_data + 3)) & (0x7F >> 3) ) << 3) + ((*(bin_data + 2) & ~(0x7F >> 2)) >> 5);
   *returnData++ = 0x80 + (((*(bin_data + 4)) & (0x7F >> 4) ) << 4) + ((*(bin_data + 3) & ~(0x7F >> 3)) >> 4);
   *returnData++ = 0x80 + (((*(bin_data + 5)) & (0x7F >> 5) ) << 5) + ((*(bin_data + 4) & ~(0x7F >> 4)) >> 3);
   *returnData++ = 0x80 + (((*(bin_data + 6)) & (0x7F >> 6) ) << 6) + ((*(bin_data + 5) & ~(0x7F >> 5)) >> 2);
   *returnData++ = 0x80 +                                     ((*(bin_data + 6) & ~(0x7F >> 6)) >> 1);
   bytes_todo -= 7;
   bin_data += 7;
  }
  else
  {
   *returnData++ = 0x80 + (((*(bin_data + 0)) & (0x7F >> 0) ) << 0);
   i = 1;
   while(bytes_todo > 1)
   {
    *returnData++ = 0x80 + (((*(bin_data + i)) & (0x7F >> i) ) << i) + ((*(bin_data + (i-1)) & ~(0x7F >> (i-1))) >> (7-(i-1)));
    bytes_todo--;
    i++;
   }
   *returnData = 0x80 + ((*(bin_data + (i-1)) & ~(0x7F >> (i-1))) >> (7-(i-1))); //final byte
   bytes_todo--;
  }
 }
 if(((len*8)%7) > 0){
  return (((len*8)/7) + 1);
 }
 return ((len*8)/7);
}
```

4.6 Failsafe mode

The DTM-3200 supports a special "failsafe" mode to enable recovery from an erroneous configuration. In failsafe mode the unit has no ASI transcoding functionality and the user can only configure the IP address and load new firmware.

Failsafe mode is entered in the following cases:

- A firmware upgrade of the DTM-3200 has failed.
- The user selected the failsafe mode through configuration option 'application'

The normal operation mode can be selected again using configuration option 'application'.

5. Specifications

5.1 Network connection

	Min	Typ	Max	Unit / Remarks
Network Port				
Standard		IEEE 802.3a		
Data rate		100/1000 auto detect		Mbps
Connector		RJ-45 with LEDs		
Control				
Ethernet encapsulation		IEEE 802.2 SNAP and Ethernet II		
IP support		IPv4		
IP-address assignment		DHCP, link local or static		
Multicast support		IGMP v2		
Network management		Not supported		

5.2 DVB-ASI input/output

	Min	Typ	Max	Unit / Remarks
Standard				
DVB-ASI		EN50083-9		
Ports				
Number of channels		1		
Connector		1x 75-Ω MCX		Input or output
Return loss	15	17		dB
Error-free cable length	300			m
Transport-stream bitrate	0.01		214	Mbps
Packet size		188/204		bytes

5.3 Parallel port input/output

	Min	Typ	Max	Unit / Remarks
Interface port				
Connector		26 way boxed header 2.54 mm pitch		Pinning compatible with DVB-SPI 25-way sub D connector (ISO 2110)
Signals		Clock, Sync, Valid, 8-bit data		Fixed clock rate

	Min	Typ	Max	Unit / Remarks
Port used as input				
Interface standard		M-LVDS		
Clock rate (accepted)	10		30	MHz
Input impedance	90		132	Ω
Common mode voltage	0.625		1.875	V
Signal amplitude	100		2000	mV
Bitrate	0.01		240	Mbps
Port used as output				
Interface standard		M-LVDS		
Clock rate (fixed)		27		MHz
Output impedance			100	Ω
Common mode voltage		1.6		V
Signal amplitude			454	mV
Bitrate	0.01		214	Mbps

5.4 Transport-Stream input/output over IP

	Min	Typ	Max	Unit / Remarks
General				
TSoIP encapsulation		UDP or SMPTE 2022-2		
Input-to-IP delay			1	ms
IP-to-output delay	1		120[*1]	ms (programmable)
IP input jitter tolerance	1		120[*2]	ms (computed)
UDP				
TS-packet size		188 or 204		bytes
TS packets / IP packet	1		7	No long UDP packets
SMPTE 2022-2				
Transport protocol		RTP		
FEC		SMPTE 2022-1		
Packet size		188 or 204		bytes
TS packets / IP packet	1		7	
FEC Size: L	1		20	
FEC Size: D	4		20	
FEC Size: LxD	4		100	

*1 Maximum delay depends on input bitrate and #TS packets per IP packet
*2 Maximum jitter tolerance depends on programmed IP-to-output delay and on FEC delay

5.5 Serial control port

	Min	Typ	Max	Unit / Remarks
Interface port				
Connector		12-way pin header, 2.54 mm pitch		Pin 1..9 compatible with standard RS-232 sub D connector
Signals		TX/RX/RTS/CTS		
Serial Format				
Interface standard		RS-232/485/422		LVTTL RS-232 is a hardware option
Format		8 bit, one stop bit, no parity		
Handshaking		hardware flow control		
Speed		9600 or 115200		baud, selectable with DIP switch

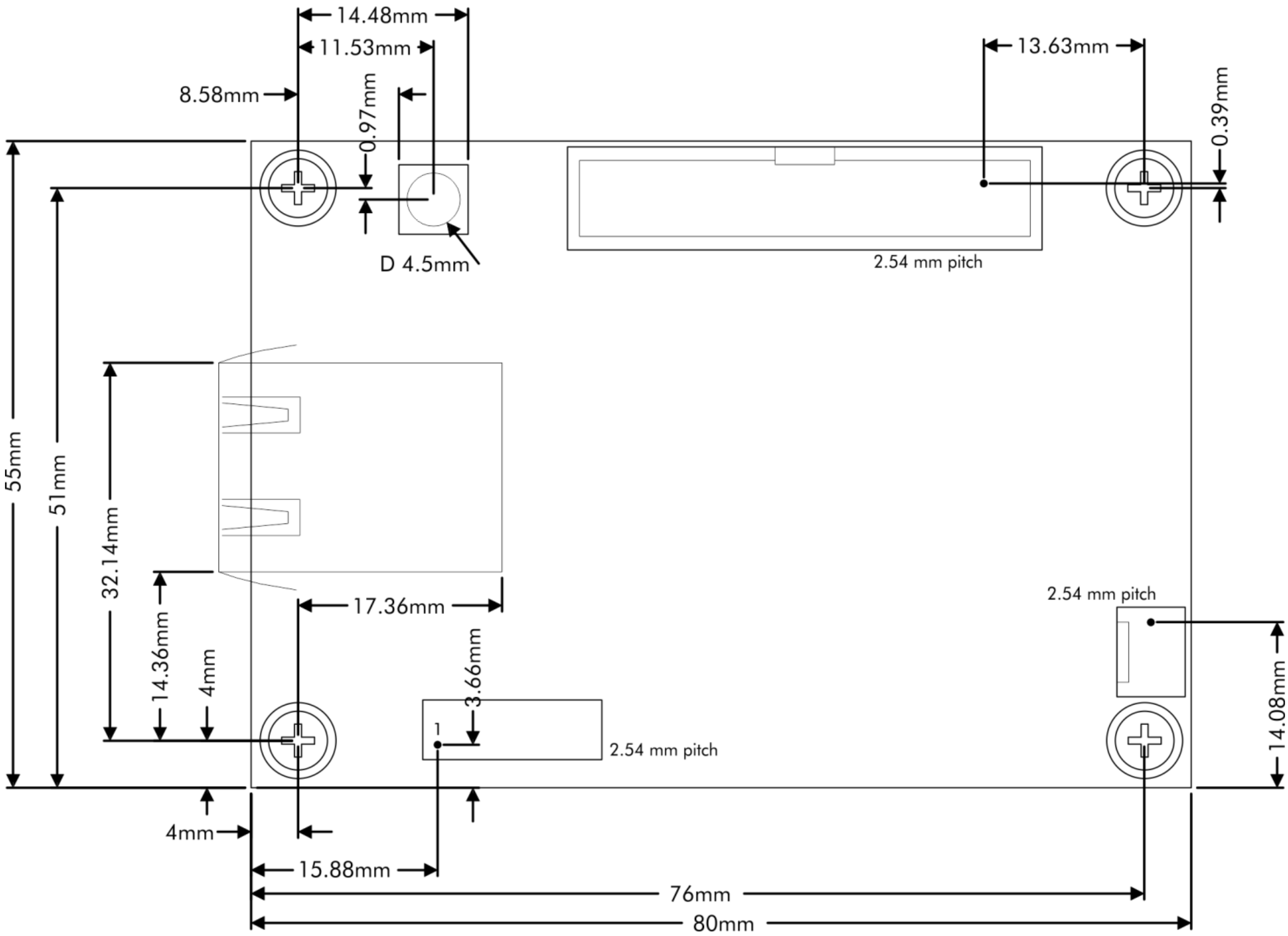
5.6 I²C control port

	Min	Typ	Max	Unit / Remarks
Interface port				
Connector		12-way pin header, 2.54 mm pitch		Signals available on pin 11-12 of serial port control connector
Signals		SDA/SCL		
I²C				
Interface voltage		3.3		V
Speed			400	kbit/s
Device address	0x40		0x4F	

5.7 Other specifications

	Min	Typ	Max	Unit / Remarks
Power				
Power supply voltage	5		24	V
Power consumption			5	W
Environmental				
Hazardous substances		RoHS compliant		
Flammability		UL-94 HB		
Operational Temperature	0		≥+45	°C
Mechanical				
Mounting hole diameter		2.7		mm (4 holes)
Dimensions W x H x D		80x55x14		mm (max)
Weight		39		g

Appendix A. Mechanical Dimensions



Appendix B. DTM-3200 Development Kit

B.1 DTM-3200 development kit – Contents

The DTM-3200 development kit contains the following items:

- DTM-3200 placed on four plastic studs
- DTM-3301 USB-interface board
- 10V/1.2A power supply
- USB cable type A to mini B
- MCX to BNC cable assembly (length = 130mm)
- DekTec USB flash drive containing DTM-3200 documentation and development tools (as well as documentation on DekTec's other products)

The development kit can be ordered from DekTec using type number DTM-3200-DEVKIT.

B.2 Using the DTM-3200 development kit

B.2.1 DIP switch setting

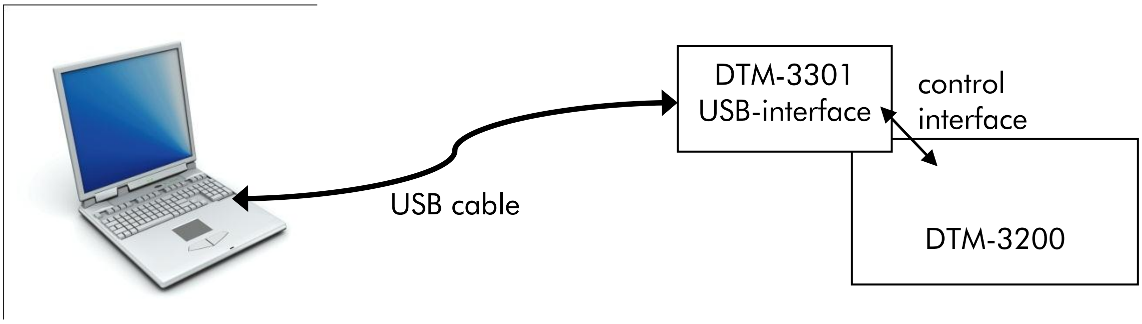
Before using the DTM-3200 it is essential to set the DIP switches on the DTM-3200 PCB into the right position. The required settings of the DIP switches are shown in the table below. For a description of the purpose of the individual DIP switches, refer to §3.3.4.

DIP switch	Setting	Description
1,2	off, off	Sets DTM-3200 address to 0x40
3	off	Selects RS-232 communication for control
4	on	Selects a baud rate of 115.2kBd

These DIP switch settings become active only after a system reboot.

B.2.2 Hardware installation

The DTM-3301 USB-interface board has to be plugged on the Control pin-header of the DTM-3200. The pin-1 indication of the DTM-3301 should match the pin-1 indication of the DTM-3200.



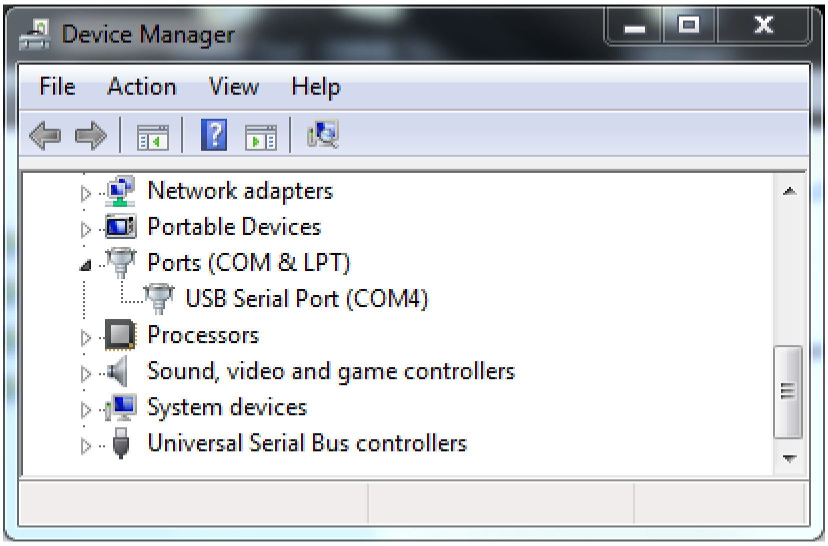
Connect the DTM-3200 to the power-supply using the power connector (see §3.2.7). The DTM-3200 will boot which will take a few seconds. During this time the two LEDs on the DTM-3200 are flashing in a start-up pattern. Wait until the DTM-3200 status LED turns green.

B.2.3 USB driver

Depending on your Windows version, it may or it may not be required to install a USB serial driver for the DTM-3301. On Windows 7 the driver is usually part of the Windows installation.

Connect the DTM-3301 to a USB port on the PC with the USB cable included in the development kit. For this step, it is not strictly necessary to power the DTM-3200 board, because the DTM-3301 is bus powered. After a while the USB connection to the interface board will become visible on the PC as a COM port in the Device Manager, section Ports (COM&LPT). The COM port number is displayed behind the USB Serial port entry, see the screen shot below.

If no USB COM port appears in the Device-Manager window, you can download and install the FTDI USB driver from http://www.ftdichip.com/Drivers/VCP.htm. After this step, you may also have to unplug and reconnect the USB cable.



In this case the COM port number is COM4.

B.2.4 Debugger

Dtm3200Util is a GUI tool to view status, control settings and upload firmware to the DTM-3200. The ***Dtm3200Util*** debugger tool can be found on the DekTec USB flash drive. It can also be downloaded from the DekTec website. ***Dtm3200Util*** is an executable that can be run from any directory on your PC.

When started, the debugger enumerates serial ports and lets the user select the serial port to which the DTM-3200 is connected. When a valid serial port is selected, all registers are read from the DTM-3200 and shown in the GUI. Blue fields can be edited and written to the DTM-3200 by clicking the pen symbol to the right of the edit fields. Yellow fields are read only; they are read when the refresh arrow is clicked.

Appendix C. Command-Line Debugger - DtmUart

DtmUart is a PC-based command-line tool to generate commands and to send those commands to the DTM-3200. This tool is useful while developing custom protocol handlers for the DTM-3200. *DtmUart* shows the detailed structure of commands and responses.

Before issuing commands with *DtmUart*, a text file named **`settings.txt`** must be created, containing only the name of the COM port. For example, when COM port 2 is used, **`settings.txt`** shall contain just one line with the text "COM2". The file shall be located in the same directory as *DtmUart*.

Open a DOS box in the directory containing the *DtmUart* executable and **`settings.txt`**. Each time *DtmUart* is run, a single command specified with the command-line arguments is executed on the DTM-3200.

Reading from the DTM-3200

The following command reads the setting 'category 1, setting 5':

```
DtmUart.exe -a=40 0105r -r
```

The parameters used in this command have the following meanings:

- -a=40 → Set the address of the DTM-3200 to 0x40. The address is used for all following commands, so '-a=40' can be omitted from further commands.
- 0105r → Send this command string to the DTM-3200 to read category 1 setting 5.
- -r → Read and show the response of the DTM-3200.

This command results into the following output:

```
Select COM2
COM-PORT connection opened
Device ID: 0x40
Send:       ☻400105R84♥ OK
Receive:    ☻400105R00000C80E9♥
COM-PORT connection closed
```

First the COM2 port is selected and opened. Next all arguments are executed sequentially. The device address is set to 0x40. The command is automatically completed by adding a start character, the device address (0x40), a checksum (0x84) and a stop character. The reply of the DTM-3200 is read and printed. From this reply it can be read that the data field contains the value 0x00000C80. Finally the COM port is closed and the program is terminated.

The start and stop characters in this example are displayed using a smiley and a heart respectively. These symbols may vary between different consoles.

Writing to the DTM-3200

To demonstrate the writing of a setting, we write 0x1234 to the setting "number of parts" (setting 7) of category "update firmware" (category 0x80). In addition to the usual arguments, this category also requires an index to be sent to the DTM-3200. The four index characters are placed after the read/write character and before the data characters.

```
DtmUart.exe 8007W00001234 -r
```

The parameters used in this command have the following meaning:

- `8007W00001234` → This command string consists of the category (0x80), the setting (0x07), R/W character (W), index (0x0000) and the 16 bit data value (0x1234).
- -r → Read and show the response of the DTM-3200.

As there is no explicit address selected, ***DtmUart*** will use the last-saved address 0x40. This command results into the following output:

```
Select COM2
COM-PORT connection opened
Send:       ☻408007W00001234EC♥ OK
Receive:    ☻408007W00001234EC♥
COM-PORT connection closed
```

Appendix D. Communication Example

In the examples below, grey areas in the timing diagrams are sent by the DTM-3200, while white areas are sent by the master. The address of the DTM-3200 in these examples is ***0x12***.

Write command on RS-XXX serial interface

Figure 9 shows the write command of the FEC-enable at the receive channel settings (category 0x81, setting 4, index 0). All values are displayed as ASCII characters.

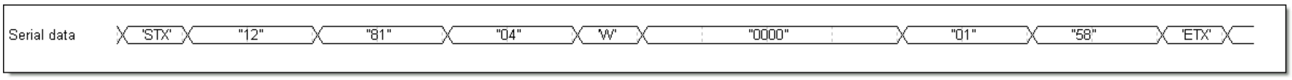


Figure 9: Write FEC enable in the receive channel settings

The command consists of the following parts:
- Start character 'STX'
- Two hexadecimal address characters ("12")
- Two hexadecimal category characters ("81")
- Two hexadecimal setting characters ("04")
- A write character 'W'
- Four hexadecimal index characters ("0000")
- Two hexadecimal characters ("01")
- Two hexadecimal checksum characters ("58", See Table 1)
- Stop character 'ETX'

Table 1: Checksum computation

Characters	ASCII value
STX	0x02
1	0x31
2	0x32
8	0x38
1	0x31
0	0x30
4	0x34
W	0x57
0	0x30
0	0x30
0	0x30
0	0x30
0	0x30
1	0x31
5	0x35
8	0x38
ETX	0x03
Sum:	0x2A8
Checksum:	0x58

Serial read command

Figure 10 shows the read command of the device type number (category 1, setting 5, no index). An index is not required for this category and the returned data consists of 4 bytes (int32). All values are displayed as ASCII characters.

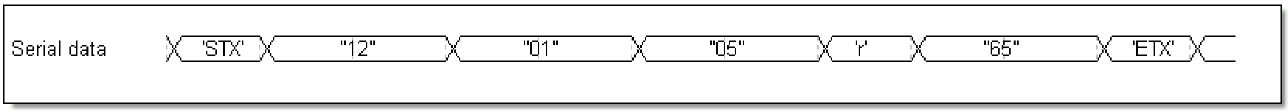


Figure 10: Read-command for the device type setting

The command consists of the following parts:

- Start character 'STX'
- Two hexadecimal address characters ("12")
- Two hexadecimal category characters ("01")
- Two hexadecimal setting characters ("05")
- A read character 'R'
- Two hexadecimal checksum characters ("85")
- End character 'ETX'

Figure 11 shows the two possible replies from the command in Figure 10. The replies are similar to the commands with the exception of the data-characters or the read character. On a successful command, the reply-data is set to the corresponding data (0x00000C80). When the received command cannot be executed, the read character is set to the ASCII character 'E'. In both cases the checksum is also updated.

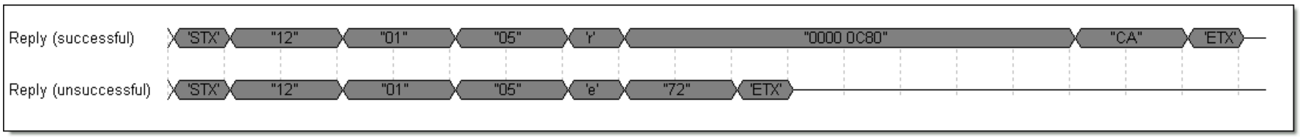


Figure 11: Reply after a device type read-command

I²C read command

Figure 12 shows the communication sequence used to issue a read subnet mask command (category 3, setting 2, no index). An index is not required for this category and the returned data consists of an IP-address.

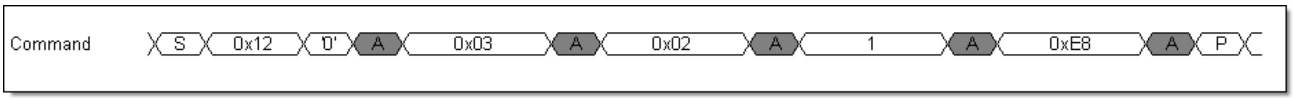


Figure 12: Send subnet-mask read-command

The command consists of the following bytes:

- Address and I²C write-bit (0x12 and '0')
- Category byte (0x03)
- Setting byte (0x02)
- Read byte (0x01)
- Checksum (0xE8, see Table 2). The checksum is computed with the address and without the I²C write-bit.

Figure 13 is the reply-sequence that may be executed after the read-command of Figure 12. After addressing this device, the bytes from the command are repeated followed with the 4-byte IP address.

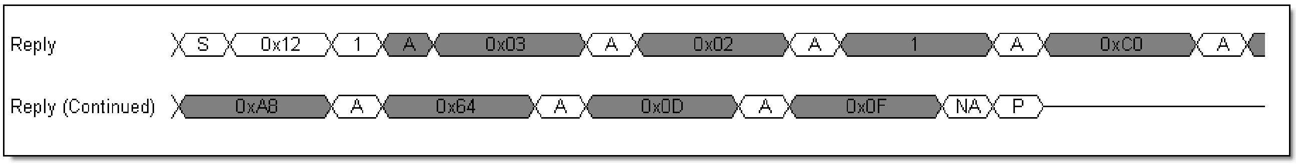


Figure 13: Read subnet-mask reply

The reply consists of the following bytes:

- Address and I²C write-bit (0x12 and '1')
- Category byte (0x03)
- Setting byte (0x02)
- Read byte (0x01)
- Four IP-address bytes (decimal 192.168.100.13 or hexadecimal C0.A8.64.0D)
- Checksum (0x0F, see Table 2)

Table 2: Checksum computation

	Command	**Reply**
Address	0x12	0x12
Category	0x03	0x03
Setting	0x02	0x02
R/W	'1'	'1'
Data byte 3	-	0xC0
Data byte 2	-	0xA8
Data byte 1	-	0x64
Data byte 0	-	0x0D
Sum:	0x18	0x1F1
Checksum:	0xE8	0x0F