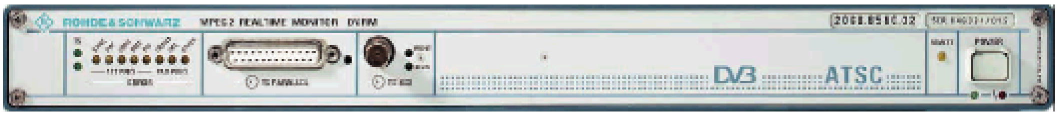
Broadcasting Division

Operating Manual

MPEG2 REALTIME MONITOR

DVRM

2068.8580...



Printed in Germany

ROHDE&SCHWARZ

EC Certificate of Conformity

CE

Certificate No.: 2000-06

This is to certify that:

Equipment type	Stock No.	Designation
DVRM	2068.8580.02	MPEG2 Real Time Monitor

complies with the provisions of the Directive of the Council of the European Union on the approximation of the laws of the Member States

- relating to electrical equipment for use within defined voltage limits (73/23/EEC revised by 93/68/EEC)
- relating to electromagnetic compatibility (89/336/EEC revised by 91/263/EEC, 92/31/EEC, 93/68/EEC)

Conformity is proven by compliance with the following standards:

EN60950 : 1997
EN50081-1 : 1992
EN50082-2 : 1995

Affixing the EC conformity mark as from 2000

ROHDE & SCHWARZ GmbH & Co. KG
Mühldorfstr. 15, D-81671 München

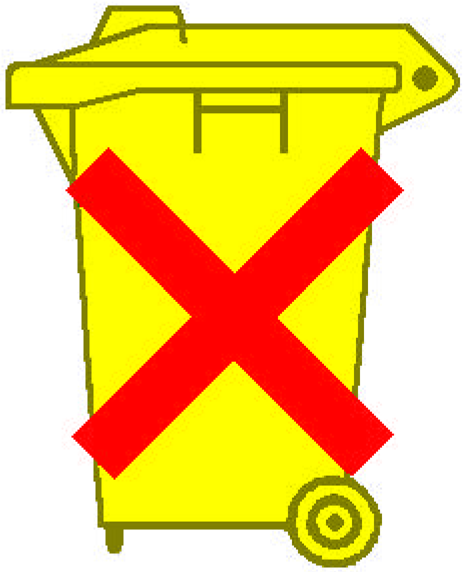
Munich, 2000-03-13

Central Quality Management FS-QZ / Becker

ROHDE&SCHWARZ

Kundeninformation zur Batterieverordnung (BattV)
Dieses Gerät enthält eine schadstoffhaltige Batterie.
Diese darf nicht mit dem Hausmüll entsorgt werden.

Nach Ende der Lebensdauer darf die Entsorgung nur über eine Rohde&Schwarz-Kundendienststelle oder eine geeignete Sammelstelle erfolgen.



Safety Regulations for Batteries (according to BattV)
This equipment houses a battery containing harmful substances that must not be disposed of as normal household waste.
After its useful life, the battery may only be disposed of at a Rohde & Schwarz service center or at a suitable depot.

Consignes de sécurité pour batteries (selon BattV)
Cet appareil est équipé d'une pile comprenant des substances nocives. Ne jamais la jeter dans une poubelle pour ordures ménagéres.
Une pile usagée doit uniquement être éliminée par un centre de service client de Rohde & Schwarz ou peut être collectée pour être traitée spécialement comme déchets dangereux.

PF 095.5140-0699

DIVIDER 1:

1 Introduction to Digital TV Transmission Technique 1.3

1.1 Definitions and Standards 1.3
1.1.1 MPEG-2 ..... 1.4
1.1.2 DVB 1.5
1.1.3 ATSC 1.5

1.2 Transmission Scenario for Digital TV 1.6

1.3 MPEG-2 Systems 1.7
1.3.1 Transport Stream Synchronization and Packet Identification 1.8
1.3.2 Contents of Transport Stream 1.9
1.3.3 Access to a Program 1.10
1.3.4 Descrambling 1.10
1.3.5 Program Synchronization 1.11
1.3.6 Service Information 1.12
1.3.7 Syntax Diagrams 1.13
1.3.7.1 Transport Packet 1.13
1.3.7.2 Adaptation Field 1.13
1.3.7.3 PES Header 1.14
1.3.7.4 Program Association Section 1.14
1.3.7.5 Program Map Section 1.15
1.3.7.6 Conditional Access Section 1.15
1.3.7.7 Private Section 1.16

1.4 Introduction to Video Quality Measurements 1.17
1.4.1 General 1.17
1.4.1.1 New Requirements through Digitization 1.17
1.4.1.2 Quality Problems caused by Encoding 1.17
1.4.1.3 Subjective Quality Measurements to ITU 1.19
1.4.2 Picture Quality Measurements with DVQ 1.20
1.4.2.1 Theory 1.20
1.4.2.2 Use 1.23

DIVIDER 2:

2 Description of Measurement Functions 2.1
2.1 Elements of Transport Stream Syntax 2.2
2.2 Overview of All Measurement Functions..2.3
2.3 TS_sync_loss - (1st priority)...2.4
2.4 Sync_byte_error - (1st priority) 2.7
2.5 PAT_error - (1st priority) 2.8
2.6 PMT_error - (1st priority)...2.9
2.7 Continuity_count_error - (1st priority) 2.10
2.8 PID_error - (1st priority) 2.11
2.9 Transport_error - (2nd priority) 2.12
2.10 CRC_error - (2nd priority) 2.13
2.11 PCR_error, PCR_accuracy_error - (2nd priority)..2.14
2.12 PTS_error - (2nd priority) 2.16
2.13 CAT_error - (2nd priority)...2.17
2.14 SI_repetition_error - (3rd priority)...2.18
2.16 BASE_PID_error - (3rd priority only in operating mode ATSC) 2.20
2.17 Unreferenced_PID - (3rd priority) 2.21
2.18 DATARATE_error 2.22
2.19 MULTIPLEX_error..2.22
2.20 SI_OTHER_error (only in operating mode DVB) 2.23
2.22 PARADIGM_error (only in operating mode ATSC) 2.25
2.23 MIP_error(only for DVB mode)..2.26
2.23.1 General description...2.26
2.23.2 Parameter overview...2.27
2.23.3 Parameter description..2.28
2.23.3.1 1. MIP_timing_error..2.28
2.23.3.2 2. MIP_structure_error 2.28
2.23.3.3 3. MIP_presence_error 2.29
2.23.3.4 4. MIP_pointer_error 2.29
2.23.3.5 5. MIP_periodicity_error...2.29
2.23.3.6 6. MIP_ts_rate_error..2.30

DIVIDER 3:

3	**Preparation for Use**	**3.1**
3.1	**Legend for Front and Rear View**	**3.1**
3.1.1	Front Panel	3.1
3.1.2	Rear Panel	3.3
3.2	**Putting into Operation**	**3.3**
3.2.1	Unpacking the Unit	3.4
3.2.2	Positioning the Unit	3.4
3.2.2.1	Desktop Model	3.4
3.2.2.2	Installation in a 19" Rack	3.4
3.2.3	EMC Safety Measures	3.5
3.2.4	AC Supply Voltage	3.5
3.3	**Function Test (Switch-on Test)**	**3.5**

DIVIDER 4:

4	**Remote Control**	**4.1**
4.1	**Brief Instructions**	**4.1**
4.2	**Setting the Transmission Parameters**	**4.1**
4.3	**Device-Dependent Messages (Commands and Responses)**	**4.2**
4.4	**Structure and Syntax of Device-Dependent Messages**	**4.2**
4.4.1	SCPI Introduction	4.2
4.4.2	Command Structure	4.3
4.4.3	Structure of a Program Message	4.5
4.4.4	Responses to Queries	4.6
4.4.5	Parameters	4.6
4.4.6	Overview of Syntax Elements	4.8
4.5	**Description of Commands**	**4.9**
4.5.1	Notation	4.9
4.5.2	Common Commands	4.10
4.5.3	Setting the Operating Modes	4.13
4.5.4	Commands of MONITORING Menu	4.14
4.5.5	Commands of DECODER Menu	4.25
4.5.6	Commands of TS INPUT Menu	4.30
4.5.7	Commands of PRINT Menu	4.31
4.5.8	Commands of STORE and RECALL CONFIG Menus	4.34
4.5.9	Commands of SETUP Menu	4.35
4.5.10	Commands for 'Trigger on Error'Mode	4.39
4.5.11	Commands for 'Dump' Mode	4.41
4.5.12	Commands of SCPI Registers	4.44
4.6	**Measurement Parameters**	**4.47**
4.6.1	Parameter Names for Monitoring	4.47
4.6.1.1	Parameter Names of 1st Priority	4.48
4.6.1.2	Parameter Names of 2nd Priority	4.49
4.6.1.3	Parameter Names of 3rd Priority	4.50
4.6.1.4	Parameter Names for Additional Settings	4.51
4.6.1.5	Error Numbers for Device-Internal Actions	4.52
4.6.2	Parameter Names for Monitoring Limits	4.54

4.7 Instrument Model and Command Processing....4.56
4.7.1 Input Unit....4.56
4.7.2 Command Identification....4.56
4.7.3 Data Set and Device Hardware....4.57
4.7.4 Status Reporting System....4.57
4.7.5 Output Unit....4.57
4.7.6 Command Sequence and Command Synchronization....4.57
4.8 Status Reporting System....4.58
4.8.1 Structure of an SCPI Status Register....4.58
4.8.2 Overview of Status Registers....4.60
4.8.3 Description of Status Registers....4.61
4.8.3.1 Status Byte (STB) and Service Request Enable Register (SRE)....4.61
4.8.3.2 Event Status Register (ESR) and Event Status Enable Register (ESE)....4.62
4.8.3.3 STATus:OPERation Register....4.63
4.8.3.4 STATus:QUEStionable Register....4.63
4.8.4 STATus:QUEStionable:DVRM Register....4.64
4.8.5 Use of the Status Reporting System....4.65
4.8.5.1 Service Request, Use of Hierarchical Structure....4.65
4.8.5.2 Queries....4.65
4.8.5.3 Error Queue Query....4.66
4.8.6 Resetting the Status Reporting System....4.66
4.9 Set of Commands....4.67
4.10 Error Messages of Remote-Control Interface....4.72
4.10.1 SCPI-Specific Error Messages....4.72
4.11 Programming Instructions....4.76
4.12 Programming Example....4.76
4.12.1 General....4.76
4.12.2 Definitions of Library Calls Supporting the V24 Interface....4.76
4.12.3 Examples....4.77
4.12.3.1 Example 1....4.77
4.12.3.2 Example 2....4.80

DIVIDER 5:

5 Maintenance and Checking....5.1
5.1 Maintenance....5.1
5.1.1 Cleaning the Outside....5.1
5.1.2 Storage....5.1
5.1.3 Replacing the Battery....5.2
5.2 Checking....5.6

DIVIDER 6:

Annex A Interfaces A.1

A.1 Synchronous Parallel Transport Stream Input - TS PARALLEL A.1
A.1.1 Pin Assignment of TS PARALLEL A.1
A.1.2 Transmission Formats - TS PARALLEL A.2
A.1.3 Data Rate Measurement A.3
A.2 Asynchronous Serial Transport Stream Input - TS ASI A.5
A.3 RS232Interface - COM1 .... A.6
A.3.1 Pin Assignment A.6
A.3.2 Transmission Parameters A.7
A.3.3 Interface Functions A.8
A.3.4 Handshake A.8
A.4 ALARM LINES Interface A.10
A.4.1 Pin Assignment A.10
A.5 Interfaces for Descrambling and Flash ROM Card A.10
A.5.1 Pin Assignment of Flash ROM Card A.11
A.5.2 Pin Assignment of Descrambling Card A.12
A.6 Parallel Printer Interface PRINTER A.13

DIVIDER 7:

Annex B Definition of Terms .... B.1 to B.9

DIVIDER 8:

Annex C Software MPEG2 Real Time Monitor

For your safety 2
Notes on the handbook 2

1. Chapter 1
How does the MPEG2 REALTIME MONITOR work 2
Background knowledge and system requirements 2

2. Chapter 1
System requirements 2
Hardware 2
Software 2
Equipment supplied 2

3. Chapter 1
Preparation 2
Connection to measuring instrument 2
Installation 3
To install the MPEG2 REALTIME MONITOR software proceed as follows: 3
Program start 5
Starting the software from the program group 5
Starting a program via the start menu 7
Deinstallation 9

4. Chapter 1
User interface 2
Main window 2
Menu bar 3
File 3
Mode 3
View 4
Options 4
Help 11
Toolbar 12
Status bar 12
Structure window 13
Tree Navigator 13
List Navigator 14
Parameter window 14
Statistics / Report 14
Data rates 17

DIVIDER 9:

INDEX

DIVIDER OVERVIEW

Introduction to Digital TV Transmission Technique	1
Description of Measurement Functions	2
Preparation for Use	3
Remote Control	4
Maintenance and Checking	5
Annex A: Interfaces	6
Annex B: Definition of Terms	7
Annex C: Software MPEG2 Real Time Monitor	8
	9
	10

1 Introduction to Digital TV Transmission Technique 1.3

1.1 Definitions and Standards 1.3
1.1.1 MPEG-2 .... 1.4
1.1.2 DVB 1.5
1.1.3 ATSC 1.5

1.2 Transmission Scenario for Digital TV 1.6

1.3 MPEG-2 Systems 1.7
1.3.1 Transport Stream Synchronization and Packet Identification 1.8
1.3.2 Contents of Transport Stream 1.9
1.3.3 Access to a Program 1.10
1.3.4 Descrambling 1.10
1.3.5 Program Synchronization 1.11
1.3.6 Service Information 1.12
1.3.7 Syntax Diagrams 1.13
1.3.7.1 Transport Packet 1.13
1.3.7.2 Adaptation Field 1.13
1.3.7.3 PES Header 1.14
1.3.7.4 Program Association Section 1.14
1.3.7.5 Program Map Section 1.15
1.3.7.6 Conditional Access Section 1.15
1.3.7.7 Private Section 1.16

1.4 Introduction to Video Quality Measurements 1.17
1.4.1 General 1.17
1.4.1.1 New Requirements through Digitization 1.17
1.4.1.2 Quality Problems caused by Encoding 1.17
1.4.1.3 Subjective Quality Measurements to ITU 1.19
1.4.2 Picture Quality Measurements with DVQ 1.20
1.4.2.1 Theory 1.20
1.4.2.2 Use 1.23

1 Introduction to Digital TV Transmission Technique

The units DVG (MPEG-2 generator), DVMD (MPEG-2 measurement decoder) and DVQ (digital video quality analyzer) are based on a completely new technique that is currently being phased-in and has become known as 'digital TV transmission technique'. Expectedly, many users of the above units will not have sufficient knowledge about this new technique. For this reason, this introductory chapter has been provided to give an outline of the subject matter and to deal in detail with aspects relevant for the units.

The first section gives an overview of relevant definitions and standards. The abbreviations MPEG, DVB and ATSC form the keywords.

The second section presents a possible transmission scenario. The fields of application of DVG, DVMD and DVQ are considered in particular.

The third section provides information about a special aspect of MPEG-2 (Part 1 / Systems), which is particularly interesting with regard to the operation of DVG, DVMD and DVQ.

The fourth section deals with picture quality measurements in general and with the analysis method implemented in DVQ.

1.1 Definitions and Standards

Some explanation is needed on the term 'digital TV'. What is understood by it is the transmission of information (vision, sound, data) by means of digital methods from the source to the sink. The essential aim is to duplicate the transmission capacity by minimizing the redundancy as well as to obtain flexibility in quality (transmission standards) and contents (any combination of vision, sound, data).

Digital TV is basically implemented in two stages:

Source coding and multiplexing:

Data reduction for vision and sound is performed first. Next, the compressed data streams together with additional data (eg teletext) have to be coded to a multiplex stream. Such multiplex streams are generated by DVG and evaluated by DVMD and DVQ. The required methods are defined by MPEG-2 (see 1.1.1). As for additional data, MPEG only defines the basic syntax. The additional data to be integrated into the multiplex stream and their form (see section 1.1.2) are stipulated by the European DVB project (see 0) or the North American ATSC (see 1.1.3).

Channel coding and transmission:

The transmission of compressed, almost redundancy-free data streams requires a high transmission quality or a bit error rate approaching zero to ensure decoding. That is why channel coding is performed before the digital modulation methods. A certain number of bit errors can then be corrected at the end of the transmission link. The methods for channel coding and transmission are defined by the European DVB and ATSC depending on the transmission medium. While Reed-Solomon coding is sufficient for cable transmission, convolutional coding is required in addition with satellite and terrestrial transmission to reduce the bit error rate.

Different modulation methods are used depending on the transmission medium and standard (see Table 1-1).

	DVB	ATSC
Satellite	QPSK	---
Cable	QAM (16/64/256)	16 VSB (4VSB/8VSB)
Terrestrial transmitter	OFDM (2k/8k)	8 VSB

Table 1-1 Modulation methods for DVB and ATSC

1.1.1 MPEG-2

MPEG stands for '**M**oving **P**ictures **E**xperts **G**roup'. This standard committee works on the coding of moving pictures and associated audio. MPEG-2 (ISO/IEC 13818) defines a corresponding standard documentation describing the compression of vision and sound data as well as the multiplexing of data streams. Moreover, there is MPEG-1, for example, which deals with the recording of video on CD in particular, or MPEG-4 for the transmission of pictures by means of very narrowband transmission channels. MPEG-3, which was supposed to define the distribution of high-resolution TV pictures has finally become part of MPEG-2.

MPEG-2, which is exclusively relevant for this manual, is subdivided into many sections. The parts concerning DVG, DVMD and DVQ are listed below.

Part 1 / Systems	ISO/IEC 13818-1	Multiplexing of several compressed vision and sound streams as well as of additional data streams to a transport multiplex
Part 2 / Video	ISO/IEC 13818-2	Compression of video data
Part 3 / Audio	ISO/IEC 13818-3	Compression of audio data
Part 4 / Compliance	ISO/IEC 13818-4	Test procedure for compressed streams (encoder) and decoder
Part 9 / Real-Time Interface Specification for Low Jitter Applications		Explanations regarding system-clock jitter during the distribution of transport streams

Table 1-2 MPEG-2 standards

The interface for DVG, DVMD and DVQ is the so-called transport stream (TS). Composition and structure of this multiplex stream are described in Part 1 / Systems, which will be dealt with further down in a separate section (see1.1.3).

1.1.2 DVB

In addition to the transmission methods based on the MPEG results, the European DVB (**D**igital **V**ideo **B**roadcasting) project laid down a number of additional definitions stipulated as a standard by ETSI / CENELEC (eg transmission of service information, integration of data streams, data encryption). The standards relevant for DVG / DVMD / DVQ are listed below:

ETS 300 468:	Specification for **S**ervice **I**nformation (SI) in Digital Video Broadcast (DVB) systems
ETS 300 472:	Specification for conveying ITU-R System B Teletext in Digital Video Broadcasting (DVB) bitstreams
ETR 162:	Allocation of Service Information (SI) codes for Digital Video Broadcasting (DVB) systems
ETR 211:	Guidelines on implementation and usage of Service Information
ETR 290:	Measurement guidelines for DVB systems
DVB Technical Module:	
Document A010:	Interfaces for CATV / SMATV Headends and similar Professional Equipment
TM 1341	Common Interface Specification for Conditional Access and other Digital Video Broadcasting Decoder Applications

Table 1-1 DVB standards

1.1.3 ATSC

Same as DVB, the Advanced Television Systems Committee (ATSC) employs the methods defined by MPEG-2 for video compression (MPEG-2 video) and for creating the TS (MPEG-2 systems). A separate standard was set up with AC-3 for audio compression, which specifies a different SI (service information) transmission.

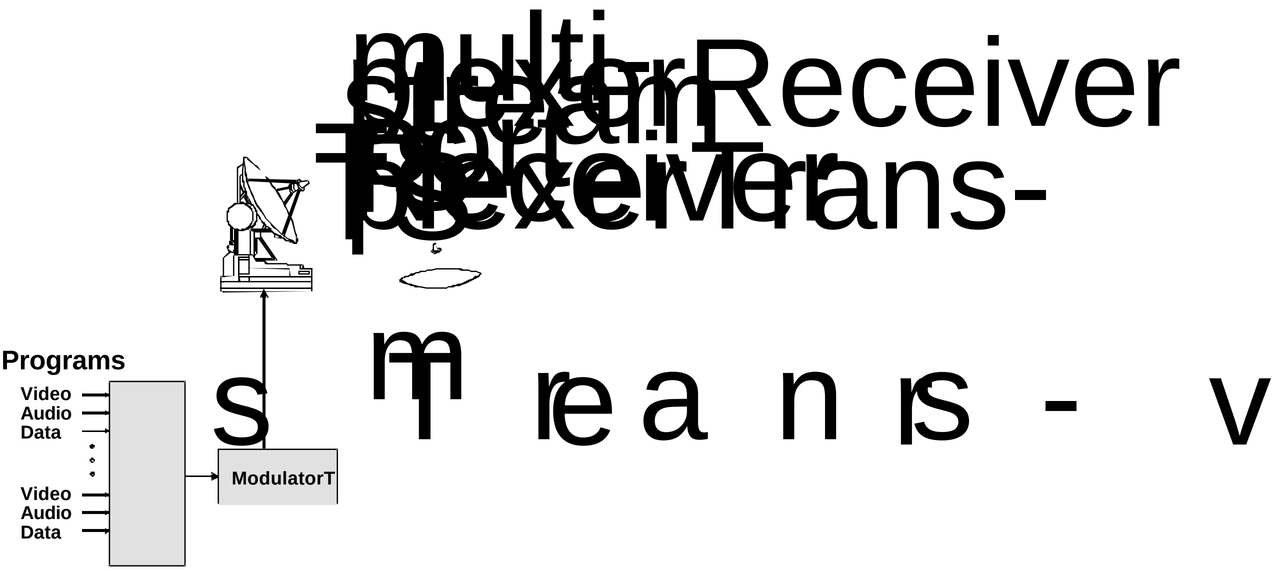
The standards relevant for DVG / DVMD / DVQ are listed below:

ATSC Document A/52	Digital Audio Compression (AC-3) Standard
ATSC Document A/53	ATSC Digital Television Standard
ATSC Document A/54	Guide to the Use of the ATSC Digital Television Standard
ATSC Document A/57	Program/Episode/Version Identification
ATSC Document A/58	Harmonization with DVB SI in the use of the ATSC Digital Television Standard
ATSC Document A/63	Standard for Coding 25/50 Hz Video
ATSC Document A/64	Transmission Measurement and Compliance Standard for Digital Television
ATSC Document A/65	Standard for Program and System Information Protocol for Terrestrial Broadcast and Cable

Table 1-2 ATSC standards

1.2 Transmission Scenario for Digital TV

Fig. 1-1 illustrates a possible transmission scenario for digital TV. TS indicates the interfaces for the transport stream. The transport stream is the output and input interface of DVG, DVMD and DVQ. At any of these interfaces, signals from DVG can be fed in or applied to DVMD and DVQ for analysis and decoding. TS analysis is of importance after every TS multiplexer. A program can run through several multiplexers if, for example, programs from different transmission paths are reorganized and sent on a new path.



1.3 MPEG-2 Systems

This section describes the structure of a transport stream starting with the decoding procedure. The individual steps required for decoding a program are illustrated and the relevant elements of the required data stream are explained. Fig. 1-2 presents an overview of the steps involved. The following subsections describe the decoding steps. The most important syntax elements are given in diagrams at the end of this chapter.

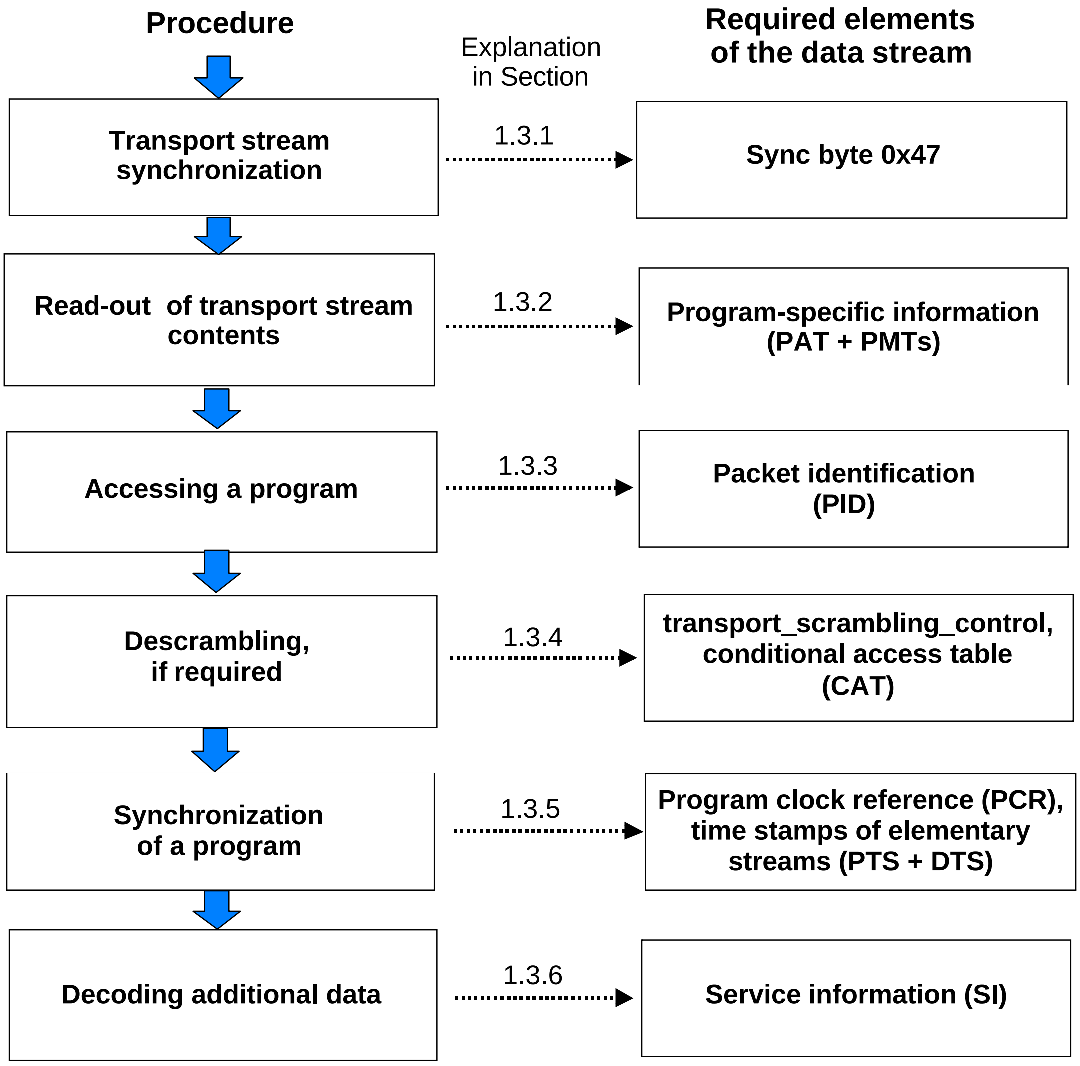


Fig. 1-2 Functions of a transport stream demultiplexer

1.3.1 Transport Stream Synchronization and Packet Identification

As a multiplex stream the transport stream has to receive data from different elementary streams. The beginning of a new packet is marked with a sync byte (47 hex). The packets of a transport stream have a fixed length of 188 bytes. The 47 hex value is not exclusively reserved for the beginning of a packet. Thus, this value does not always indicate a packet start.

To ensure stable synchronization it is necessary to check the repetitive occurrence of a sync byte at 188-byte intervals. The hysteresis parameters define how often the value 47 hex has to occur at 188-byte intervals for the packet synchronization to be considered locked or how often the sync byte has to be faulty for the packet synchronization to be declared lost.

The sync byte interval can also be 204 bytes (188 + 16 bytes) for DVB and 208 bytes for ATSC. In this case, the last 16 bytes (ATSC: 20 bytes) originate from channel coding (Reed-Solomon error protection). Channel decoding has already taken place at the transport stream level so that the 16 / 20 bytes at the end of each packet do not carry any useful information. Only the clock conversion from 204 / 208 to 188 bytes per packet duration has not been performed.

According to the American ATSC definition, the transport stream packet length may also be 208 bytes (188 + 20 bytes) instead of 204 bytes. Channel coding takes up about 20 bytes.

A header with a length of four bytes precedes each transport stream packet. As described above, the first byte of the header is the sync byte. If not all bit errors caused by transmission can be eliminated during channel decoding, the transport error indicator is set in the header of the packet concerned.

A useful information for identifying a packet is the PID (**P**acket **ID**entification). The PID is a field of 13 bits. It can thus assume 8196 different values. A PID is assigned to each substream, eg a video or audio stream (not to a program). Some PID values are fixed such as 0x0000 (hex) for the PAT (see section 1.3.7.4), 0x0001 (hex) for the CAT (see section 1.3.7.6) and 0x1FFF for the so-called null packets that do not contain useful data but only dummy bytes.

1.3.2 Contents of Transport Stream

The transport stream normally contains several programs consisting of several elementary streams. The contents of the transport stream is described in the **P**rogram-**S**pecific **I**nformation (PSI). Each transport stream contains a **P**rogram **A**ssociation **T**able (PAT) as well as one or several **P**rogram **M**ap **T**ables (PMTs).

The PAT is contained in the transport stream packets with the PID 0x0000. It refers to all the programs contained in the transport stream and indicates the program number and the corresponding PID for the program map table (PMT).

The elementary streams (video, audio, data) pertaining to the individual programs are described in a PMT. A PMT consists of one or several sections, each describing a program.

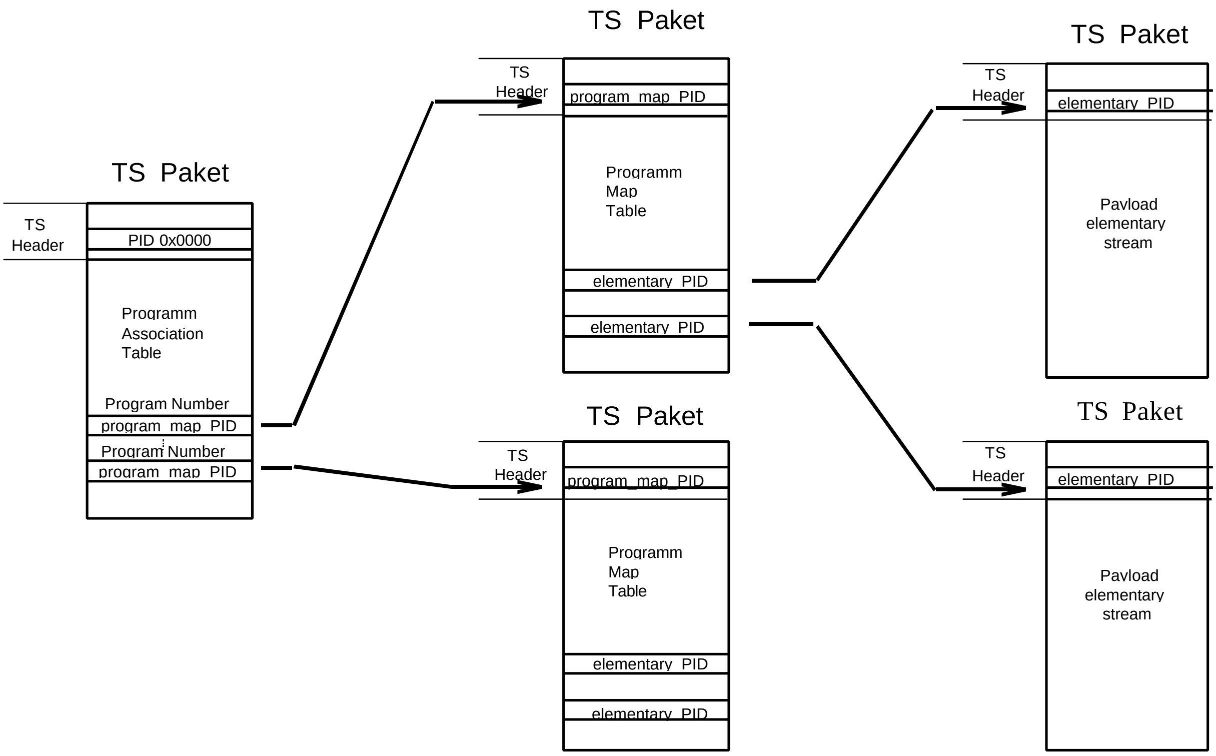


Fig. 1-3 PAT and PMT describe the contents of a transport stream

1.3.3 Access to a Program

The program to be decoded can be selected on the basis of the information on the contents of the transport stream. If the programs contains contending elementary streams, eg several audio streams, a selection has to be made. The packets of the selected elementary streams identified by the PID now have to be routed from the demultiplexer to the associated decoders.

By analyzing the continuity counter in the packet header it can be checked whether the individual packets of an elementary stream arrive completely and in the correct order. The continuity counter is represented by the four low-order bits of the last header byte, ie bit 29 to 32 of each packet. The value (0-15) is incremented with each new packet. The 15 is followed by a zero. Two exceptions are permissible:

- The discontinuity indicator (ie during a program step) is set in the adaptation field (see Fig. 1-5). The continuity counter may then assume any value.
- A packet may be transferred twice. The continuity counter must not be incremented.

1.3.4 Descrambling

The received data may be scrambled. There are two scrambling levels: the **T**ransport **S**tream (TS) or the **P**acketized **E**lementary **S**tream (PES) level.

The corresponding header information (PES or TS header) remains unscrambled. If scrambling is performed at the transport stream level, the PES headers are included while TS headers remain unscrambled in any case.

If a packet (TS or PES) is scrambled, the first bit of the scrambling control field in the header (TS or PES) is indicated. The second bit in the scrambling control field is required because the key code changes from time to time. This bit is set if a new key code is valid for the relevant packet.

Special control data must be available in the decoder for descrambling. This information is transferred in the **E**ntitlement **C**ontrol **M**essages (ECM) and **E**ntitlement **M**anagement **M**essages (EMM). The EMC contains the key codes whereas EMM serves for allocating the access rights to the receiver. ECM and EMM are contained or referenced in the program-specific information (CAT or PMT).

Descrambling is performed in a supplier-specific hardware which can be connected to the decoder via a standardized common interface.

1.3.5 Program Synchronization

A program normally contains several elementary streams (PIDs). A common clock reference is required to synchronize decoding and display (or output) of the individual elementary streams. This clock reference is carried along in the form of the **P**rogram **C**lock **R**eference (PCR) for each program in an elementary stream of the program. Every 40 ms at maximum, the 4-byte-long header of a transport packet is extended by a so-called adaptation field (see Fig. 1-5). In addition to a variety of other signalling information, which cannot be dealt with here, this adaptation field also contains the PCR. The adaptation field is always unscrambled.

The PCR value (43 bits long) corresponds to the state of a counter with a 27-MHz clock at the time of arrival of the first TS packet byte containing the PCR value. It is used in the decoder for controlling the 27 MHz system clock PLL. This ensures synchronization of the multiplexer at the transmitter end and of the demultiplexer at the receiver end.

The individual elementary streams contain time stamps such as the **D**ecoding **T**ime **S**tamps (DTS) and **P**resentation **T**ime **S**tamps (PTS) for synchronizing decoding and display of the individual elementary streams. The packetized elementary streams (PES) are transmitted in packets with a length of up to 64 Kbyte (more for video streams) and define a certain display period (eg a picture in the case of video streams). A header which also contains the DTS and PTS precedes each elementary stream packet (PES). If a transport packet contains the beginning of a PES packet, the 10th bit (payload unit start indicator) is set in the header of the transport packet.

The PTS / DTS value (33 bits) corresponds to the state of a 90 kHz counter and is referenced to the 33 high-order bits of PCR. The ratio of 27 MHz (PCR) to 90 kHz (PTS/DTS) is 300 and attained by the fact that the 9 low-order bits of PCR only count to 300 (instead of 512).

Two different time stamps (DTS and PTS) are provided as the order of arrival of the PES packets and their decoding do not always correspond to the order of the display, eg during the transmission of difference pictures in video streams. In many cases, PTS and DTS are of the same value or only one PTS is available.

1.3.6 Service Information

The tables defined by the DVB project or ATSC (see ETS 300 468 for DVB; Document A/65 for ATSC) are to be seen as service information. The information contained in these tables is not only required for decoding but also provides convenient access for the end user: it might be program information for the viewer or control information for the decoder and the units connected to it.

In many cases, PSI (**P**rogram-**S**pecific **I**nformation) is also mentioned in connection with SI (**S**ervice **I**nformation). PSI is already defined in MPEG-2 and partly contains the above-mentioned tables PAT (**P**rogram **A**ssociation **T**able), PMT (**P**rogram **M**ap **T**able), CAT (**C**onditional **A**ccess **T**able) and NIT (**N**etwork **I**nformation **T**able). The latter contains data provided by the network operator for tuning the receivers, eg orbit positions or transponder numbers. The CAT and NIT contents are not stipulated by MPEG but by the DVB project. ATSC does not use the NIT and transmits the respective data by means of the ATSC-defined TVCT (terrestrial) or CVCT (cable).

1.3.6.1 DVB Service information

Tables defined by the DVB project are as follows:

BAT (**B**ouquet **A**ssociation **T**able) contains information about the different programs of a supplier irrespective of the propagation paths of the programs.

SDT (**S**ervice **D**escription **T**able) describes the programs offered.

EIT (**E**vent **I**nformation **T**able) supplies the data base for an electronic TV guide with information on the type of program and age classification for the viewer.

RST (**R**unning **S**tatus **T**able) comprises status information about the individual programs and especially serves for controlling video recorders.

TDT (**T**ime and **D**ate **T**able) provides information about date and current time (UTC).

TOT (**T**ime **O**ffset **T**able) provides information about the local time offset in addition to date and time.

ST (**S**tuffing **T**able) has no relevant contents. It is generated when invalid tables are overwritten during transmission (eg at CATV headends).

The PIDs for the tables are permanently assigned. An exception are the PMTs, whose PIDs are defined in the PAT. Moreover, each type of table has a Table_ID, which is stated at the beginning of the table. The Table_ID is necessary to allow various tables to be transmitted with a single PID. The interrelation between table type, Table_ID and PID is shown by the table 1-5.

1.3.6.2 ATSC Service informationen

The MGT (**M**aster **G**uide **T**able) lists all other ATSC tables with service information except STT (**S**ystem **T**ime **T**able) by specifying the PID, version number and table length.

The TVCT (**T**errestrial **V**irtual **C**hannel **T**able) for terrestrial networks or CVCT (**C**able **V**irtual **C**hannel **T**able) for cable networks contains transmission parameters (eg carrier frequency, type of modulation) and thus replaces the NIT (network information table).

The RRT (**R**ating **R**egion **T**able) contains values for 'rating' the programs. The values are assigned to specific recommendations (eg 'suitable for all ages') and used in the EIT.

The EIT (**E**vent **I**nformation **T**able) contains information on the program content. Up to 128 different EITs (EIT-0 to EIT-127) can be transmitted. Each EIT covers a period of three hours starting from EIT-0 for the current three hours. All EITs above EIT-4 are optional.

The ETT (**E**xtended **T**ext **T**able) contains detailed descriptions for the program contents in the EIT. Each EIT (EIT-0 to EIT-127) can be assigned one of the optional ETTs. There are 'event ETTs' with information on programs and 'channel ETTs' with information on a transmission channel.

The STT (**S**ystem **T**ime **T**able) provides information on date and current time (UTC).

1.3.6.3 Including the Service Information in the Transport Stream

The PIDs for most of the tables are permanently assigned. An exception are the MPTs whose PIDs are defined in the PAT and the EITs and ETTs in the ATSC streams whose PMTs are defined in the MGT.

A Table_ID given at the beginning of the table is defined for each table type. The Table_ID is necessary to allow various tables to be transmitted and identified with a single PID. Please refer to Table 1-5 for interrelations of table type, Table_ID and PID. The ATSC document A/58 regulates the harmonization with DVB.

Table	PID	Table_ID	DVB / ATSC
PAT	0x0000	0x00	DVB/ATSC
PMT	0x0020...0x1FFA	0x02	DVB/ATSC
CAT	0x0001	0x01	DVB/ATSC
NIT	0x0010	0x40...0x41	only DVB
BAT	0x0011	0x4A	only DVB
SDT	0x0011	0x42, 0x46	only DVB
EIT	0x0012	0x4E...0x6F	only DVB
RST	0x0013	0x71	only DVB
TDT	0x0014	0x70	only DVB
TOT	0x0014	0x73	only DVB
ST	0x0010...0x0014	0x72	only DVB
MIP	0x0015	---	only DVB (*
MTG	0x1FFB	0xC7	only ATSC
TVCT	0x1FFB	0xC8	only ATSC
CVCT	0x1FFB	0xC9	only ATSC
RRT	0x1FFB	0xCA	only ATSC
EIT	0x20 ... 0x1FFA	0xCB	only ATSC
ETT	0x20 ... 0x1FFA	0xCC	only ATSC
SST	0x1FFB	0xCD	only ATSC

(* only for transmitters of single-frequency networks

Table 1-3 PID and table ID for PSI/SI/PSIP

1.3.7 Syntax Diagrams (MPEG2 Systems)

1.3.7.1 Transport Packet

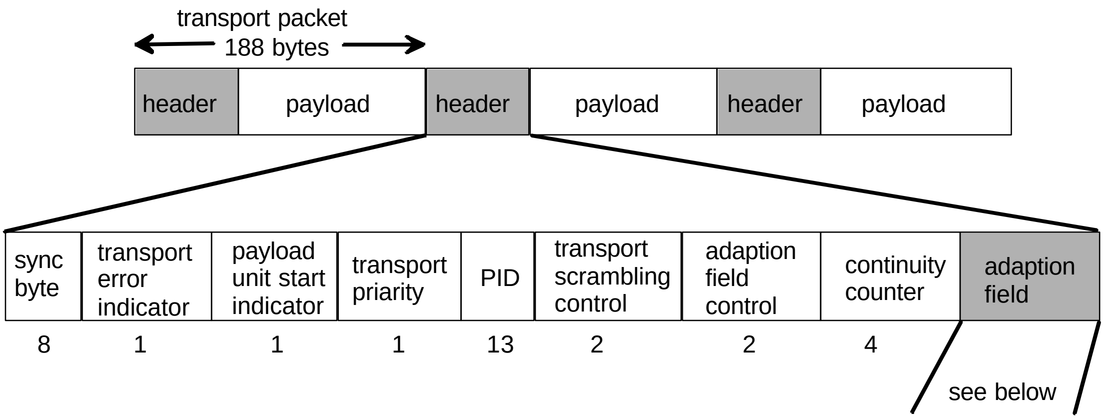


Fig. 1-4 Transport Packet

1.3.7.2 Adaptation Field

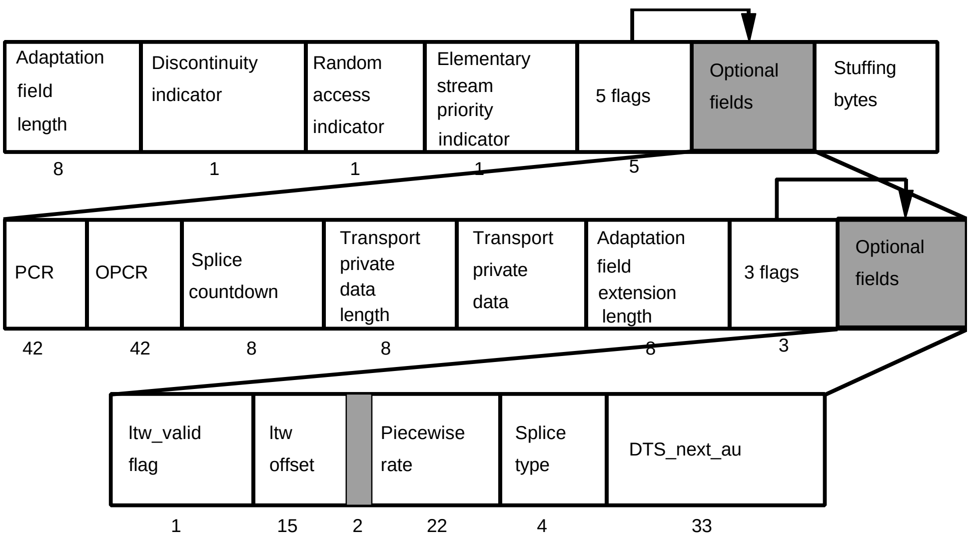


Fig. 1-5 Adaptation field

1.3.7.3 PES Header

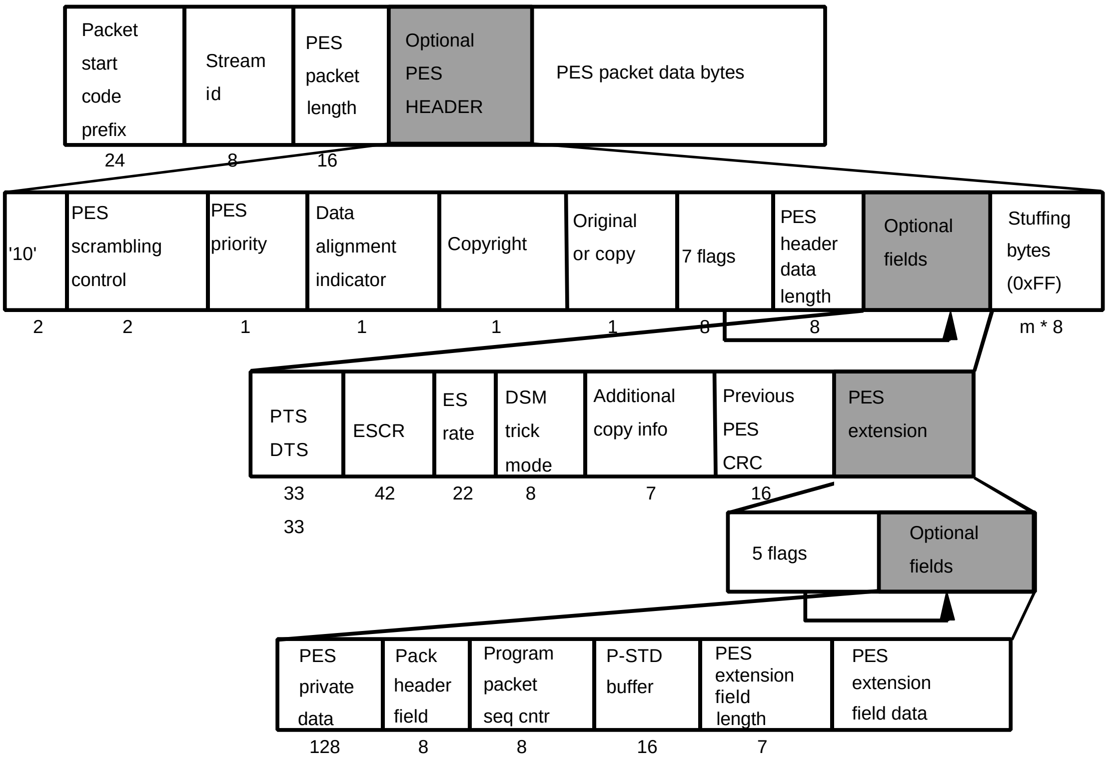


Fig. 1-6 PES header

1.3.7.4 Program Association Section

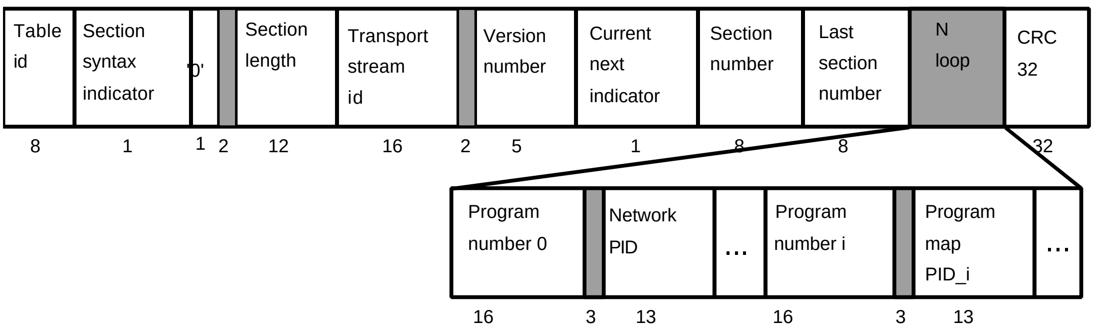


Fig. 1-7 Program association section (PAT)

1.3.7.5 Program Map Section

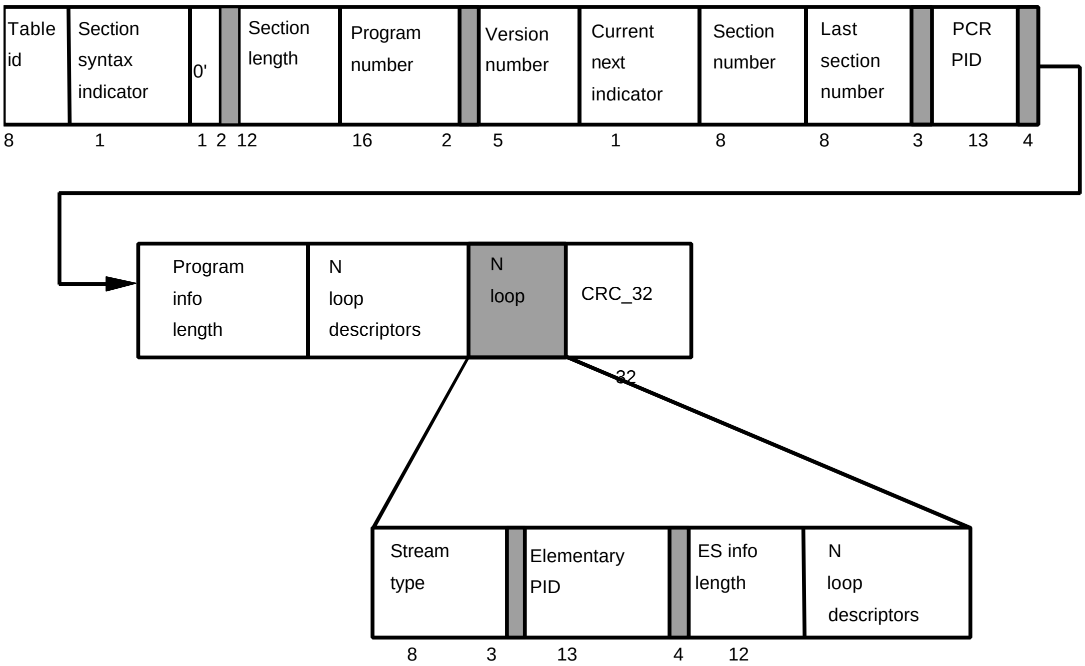


Fig. 1-8 Program map section (PMT)

1.3.7.6 Conditional Access Section

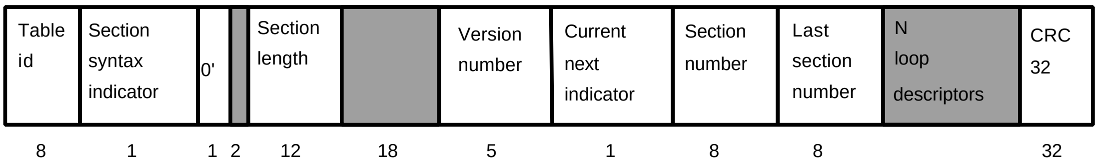


Fig. 1-9 Conditional Access Section (CAT)

1.3.7.7 Private Section

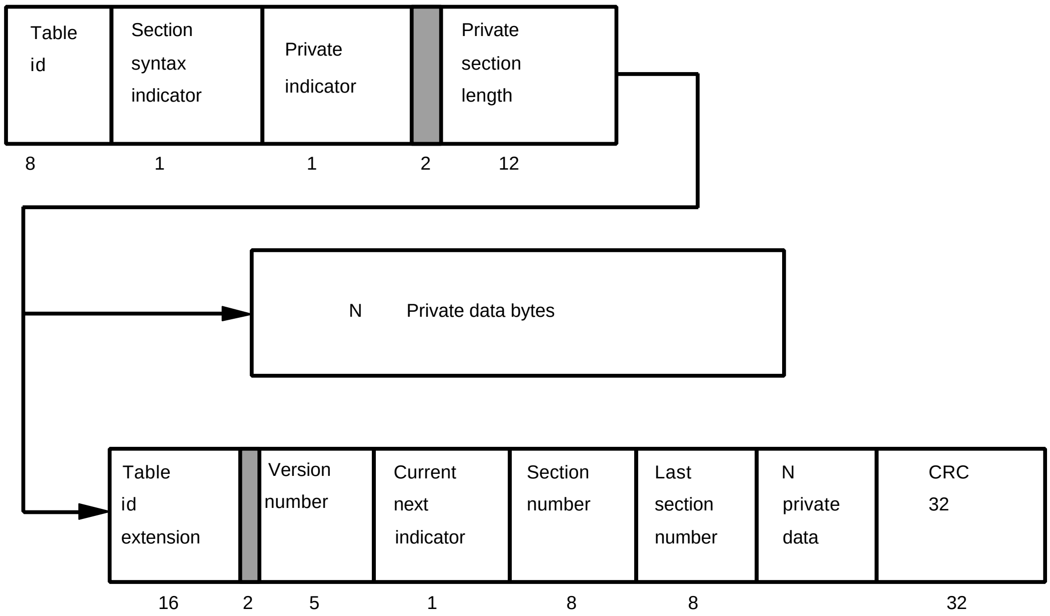


Fig. 1-10 Private section

1.4 Introduction to Video Quality Measurements

In this section basic principles of picture quality analysis are explained. Both general aspects and those specific to Video Quality Analyzer DVQ are dealt with.

1.4.1 General

1.4.1.1 New Requirements through Digitization

Colour television is in its thirties and the technique is more or less perfect. Why then think about picture quality? The answer is to be found in the transition from the analog TV set to the multimedia home platform. This means that TV terminals are going to allow the viewer to receive innumerable TV programs and work with interactive data services. The basis of this is digital video and sound broadcasting as defined by DVB and ATSC.

This change greatly affects the picture quality. In analog television, quality is determined by the length and quality of the transmission path. The possible degradations are familiar enough: noise, reflection and blurred pictures. The poorer the transmission path, the poorer the displayed picture. Not so with digital TV. Here the picture quality is essentially determined at the beginning of the transmission link through encoding and multiplexing several programs in a transmission channel (transport stream). Given errorfree transmission of the data signal, the picture quality remains unchanged over the entire path (see Fig. 1-11).

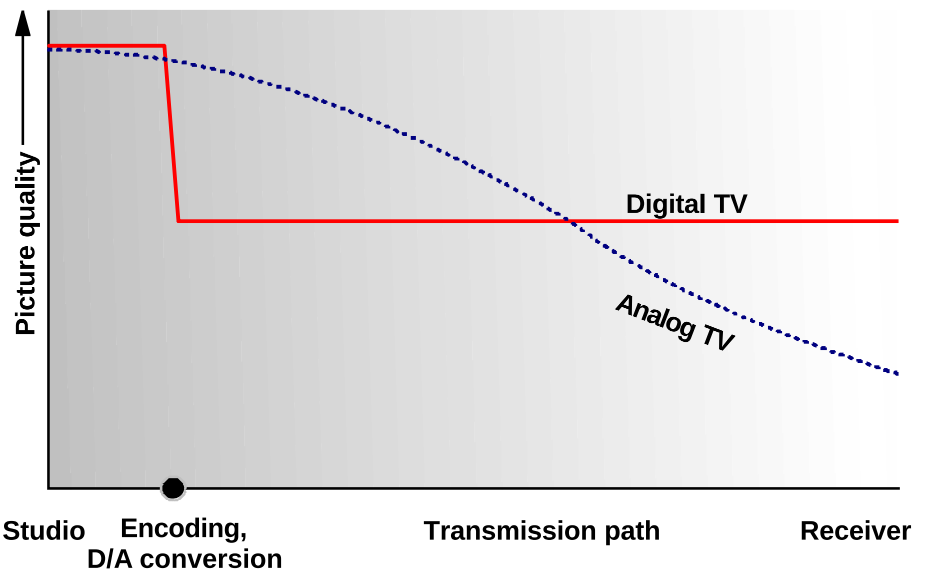


Fig. 1-11 Picture quality degradation along transmission path with analog and digital TV.

1.4.1.2 Quality Problems caused by Encoding

Encoding is performed in line with the MPEG-2 standard, which enables better utilization of transmission paths by drastically reducting the data rate of the digitized picture. The 270 Mbit/s source signal is converted in several steps to a signal of 5 Mbit/s or even less. Data compression of course affects the video picture. The art of encoding is to make the changes to the data stream so that they remain undetected by the human eye. This is more and more difficult to achieve with decreasing output data rate. The picture content itself also has an effect. The finer and more irregular the structures, the more difficult encoding becomes. The MPEG-2 standard merely describes the tools for data reduction and the syntax of the transmission signal. Outlay and quality of the implementation of a video encoder are left to the manufacturers. The achievable picture quality is therefore not only determined by the data rate and the source picture but also by the type of encoder used.

The changes caused in the picture by encoding differ from those encountered in analog transmission. The most clearly visible effect is blocking (see Fig. 1-15). The reason for this is that to perform data compression the picture is divided into DCT (**D**iscrete **C**osine **T**ransform) blocks of 8 x 8 pixels. Hence all the measurement techniques used so successfully for many years in analog TV for determining the picture quality have become redundant.

Fig. 1-12 Clearly visible blocking effects on digitally coded TV picture and – by comparison – picture without blocking.

1.4.1.3 Subjective Quality Measurements to ITU

How do you distinguish a good encoder from a less good one. How do you determine the minimum acceptable data rate? And finally, how can you monitor picture quality during transmission?

These requirements call for a test method that evaluates the picture signal itself and takes into account the perception capabilities of the human eye. The best way would be to include the viewer into the process. There is no accounting for tastes however, so the results obtained this way can only be subjective. For comparison and reproduction of results, ITU specified several test methods.

Two methods are used among others: With the DSCQS (**D**ouble **S**timulus **C**ontinuous **Q**uality **S**cale) method the test sequences to be judged and the original (eg before processing) are presented to the test person. A quality mark on a continuous scale is assigned to the two sequences, each of which is about 10 s long, and the difference is further evaluated. The employed scale ranging from 0 to 100 covers the quality levels excellent / good / fair / poor / bad as specified by ITU and all values in between (see Fig. 1-16). This method permits even very slight quality differences to be resolved.

With the second method, called SSCQE (**S**ingle **S**timulus **C**ontinuous **Q**uality **E**valuation), only the sequence to be assessed is displayed. During the presentation the test person moves a slider on a scale from 0 to 100 according to his/her subjective impression of picture quality. This value is sampled at a frequency of 2 Hz, thus yielding two quality values per second. This method can be used when no original sequence is available as a reference and thus corresponds better to the real-life situation of the TV viewer who cannot see the picture recorded in the studio.

Both methods take into account the subjective perception of the human eye. It was found, for instance, that quality degradation in fast-motion pictures or pictures showing very many details is not perceived to the same degree as in slow-motion pictures or pictures containing few details only (masking effect through high temporal and spatial activity).

Fig. 1-13 Quality scale for comparative (DSCQS) and absolute (SSCQE) subjective assessment of picture sequences.

1.4.2 Picture Quality Measurements with DVQ

1.4.2.1 Theory

During a research project of the Technical University of Braunschweig and Rohde & Schwarz it became apparent that the visual impression of the MPEG-2-inherent blocking structure has the greatest impact on the picture quality in an otherwise normal video stream.

To describe this effect, a parameter was defined, the **D**igital **V**ideo **Q**uality **L**evel, **W**eighted (DVQL-W).

The digital video quality level is computed from vectors containing information on the averaged differences between adjacent pixels (Fig. 1-17).

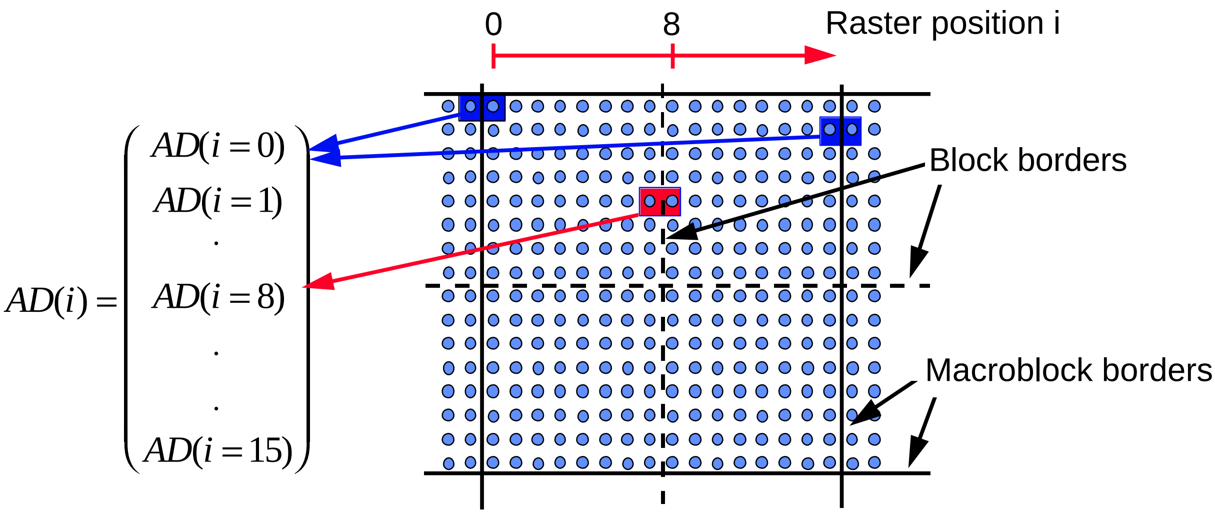


Fig. 1-14 Evaluation of amplitude differences of adjacent pixels

The MPEG-2 encoding process is based on blocks of 8 x 8 pixels and macroblocks of 16 x 16 pixels.

An analysis of the differences between all pairs of horizontally adjacent pixels shows an MPEG-2-specific characteristic.

Normally the differences between adjacent pixels are reduced by the encoding process. The exception are the pairs of pixels across the borders of blocks or macroblocks.

Fig. 1-15 shows the result of the calculation of the average differences carried out on the original 'Flowergarden' sequence.

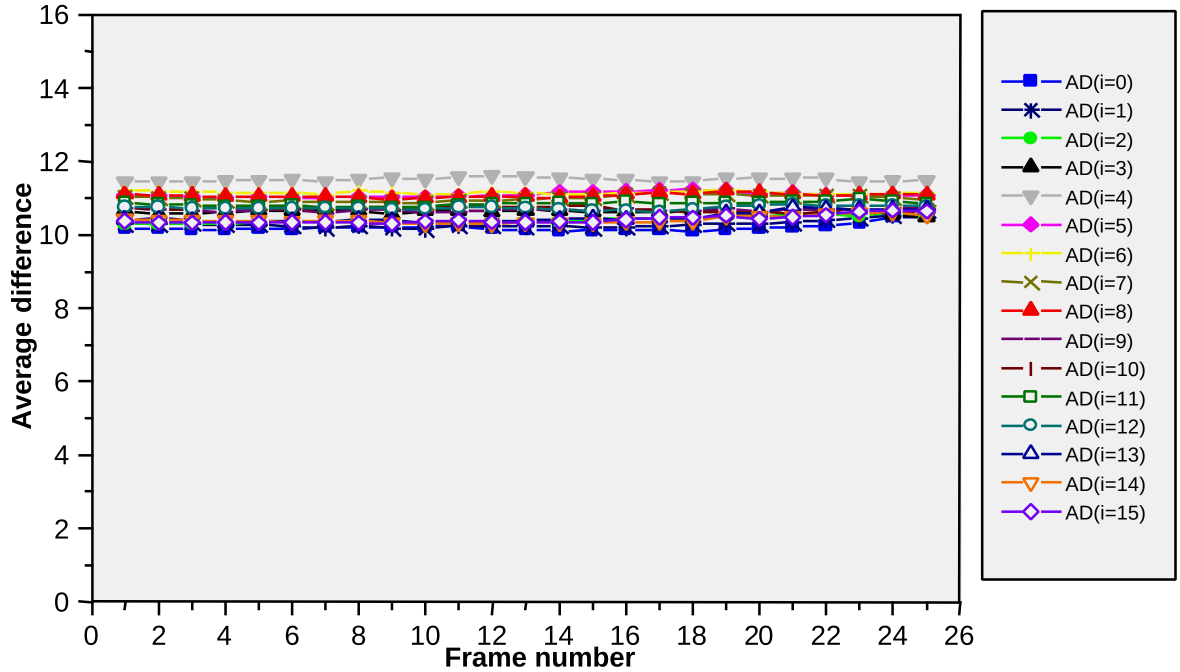


Fig. 1-15 Averaged pixel amplitude differences,
Example: Flowergarden (original)

The calculated values for all pairs of pixels (ie elements of the A/D vector) are very close.

After encoding the same sequence with 2 Mbyte/s and subsequent decoding, the average differences show a particular pattern (see Fig. 1-16).

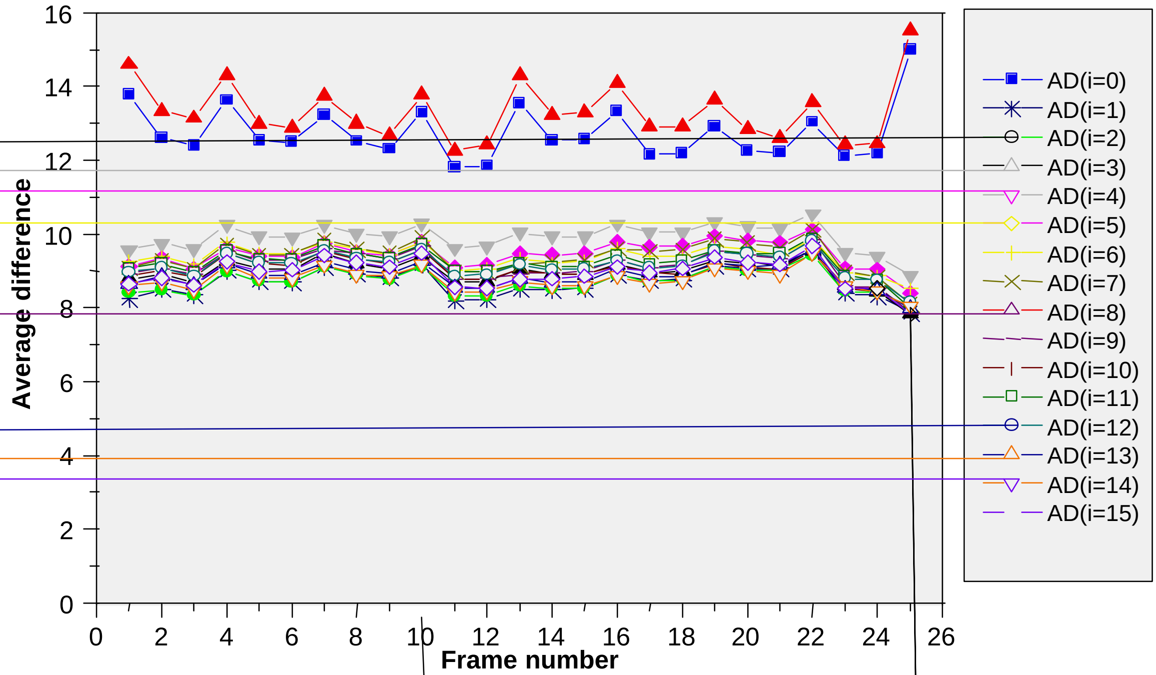


Fig. 1-16 Averaged pixel amplitude differences,
Example: Flowergarden (MPEG-2-coded/decoded with 2 Mbyte/s)

In this diagram the values for the element AD(i=0) and AD(i=8) are enhanced. They characterize the influence of the blocking structure in a MPEG-2 decoded picture. The digital video quality level is calculated from these values.

Since the computation of the DVQL-W parameter is mainly based on DCT-related effects, the same algorithm can be applied to other DCT-based compression systems.

As long as masking effects that may result from spatial activity and/or temporal activity are not considered, the whole parameter is very sensitive to any blocking structure even far below the threshold of visibility.

In this sense it can be compared with the measurement of the signal-to-noise ratio (S/N) as is applied on analogue video signals.

If the appropriate masking is incorporated, the DVQL-W delivers the equivalent of the **M**ean **O**pinion **S**core (MOS) but as a predicted value.

With the masking included, the algorithm shows an excellent correlation with subjective assessment results (Fig. 1-17).

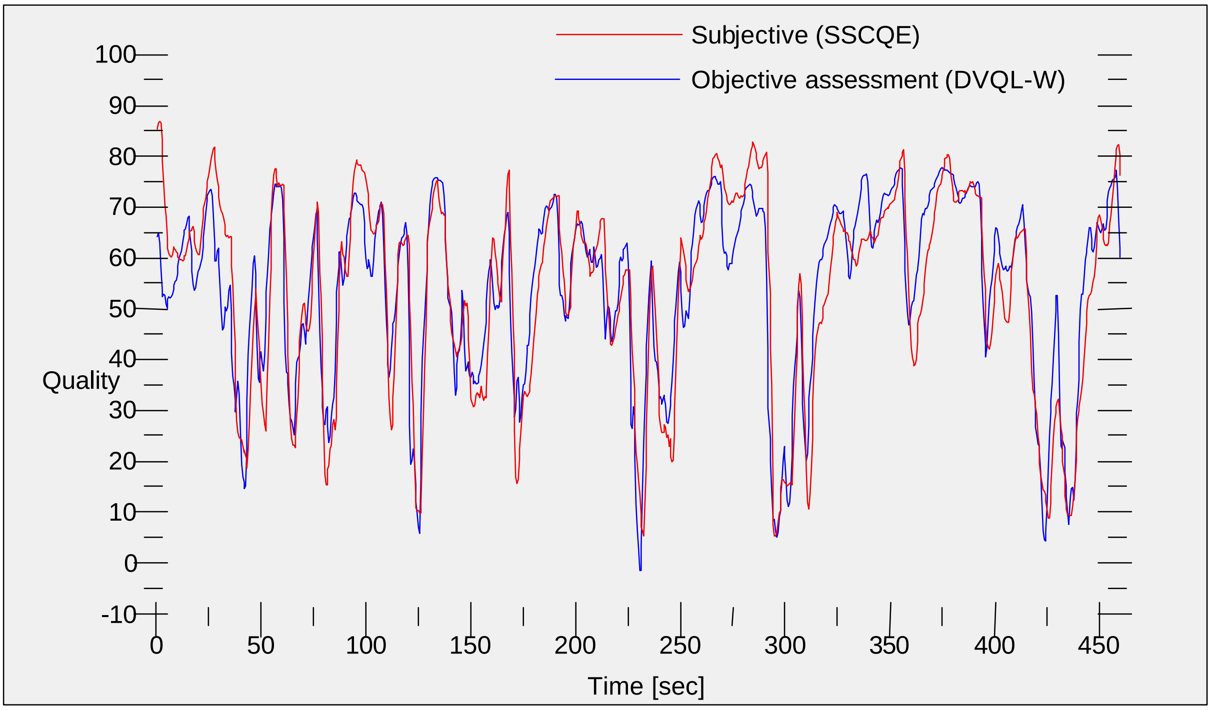


Fig. 1-17 Comparison of subjective and objective picture quality

These results were obtained by the single stimulus continuous quality evaluation (SSCQE) method. The compiled test sequence is approximately 8 minutes long and consists of 11 well-known test sequences such as 'Flowergarden', 'Mobile & Calendar', 'Table tennis', etc. The data rates for the sequences varied between 1 Mbit/s and 9 Mbit/s.

From the subjective assessment about 1000 measurement values were obtained. Their scaling factor was re-based and a fixed delay of 1 second was introduced.

With this optimization, an overall correlation of more than 94% was achieved.

1.4.2.2 Use

Nevertheless the instrument provides two inputs for video data streams in SDI (**S**erial **D**igital **I**nterface) standard. The analysis on both input signals is done in parallel, so that a comparison between two signals is equally possible in real time.

The same instrument also provides additional information on the decoded video signal such as 'Picture Freeze' and 'Picture Lost' as well as 'Audio Lost'.

In general, the instrument can determine the influence of the encoder and the influence of the transmission system on the picture quality.

If the transmission system does not produce any additional impairments, the measured picture quality at the output of the network should correspond to the picture quality at the input of the network.

An application that might be of interest for service providers and/or network operators is the control of encoder parameters such as bit rate and GOP (**G**roup **O**f **P**icture) length by the DVQL-W parameter. The same seems to work for the adjustment of bit rates in a statistical multiplexer.

Set-top box manufacturers may be interested in testing their products with impaired sequences. In such a case not only the decoders could be integrated in the tests but also the efficiency of any implemented concealment strategy could be verified.

2 Description of Measurement Functions 2.1
2.1 Elements of Transport Stream Syntax 2.2
2.2 Overview of All Measurement Functions..2.3
2.3 TS_sync_loss - (1st priority)..2.4
2.4 Sync_byte_error - (1st priority) 2.7
2.5 PAT_error - (1st priority)..2.8
2.6 PMT_error - (1st priority)..2.9
2.7 Continuity_count_error - (1st priority) 2.10
2.8 PID_error - (1st priority) 2.11
2.9 Transport_error - (2nd priority) 2.12
2.10 CRC_error - (2nd priority) 2.13
2.11 PCR_error, PCR_accuracy_error - (2nd priority)..2.14
2.12 PTS_error - (2nd priority)..2.16
2.13 CAT_error - (2nd priority)..2.17
2.14 SI_repetition_error - (3rd priority)..2.18
2.16 BASE_PID_error - (3rd priority only in operating mode ATSC) 2.20
2.17 Unreferenced_PID - (3rd priority) 2.21
2.18 DATARATE_error..2.22
2.19 MULTIPLEX_error..2.22
2.20 SI_OTHER_error (only in operating mode DVB)..2.23
2.22 PARADIGM_error (only in operating mode ATSC) 2.25
2.23 MIP_error(only for DVB mode)..2.26
2.23.1 General description..2.26
2.23.2 Parameter overview..2.27
2.23.3 Parameter description..2.28
2.23.3.1 1. MIP_timing_error..2.28
2.23.3.2 2. MIP_structure_error 2.28
2.23.3.3 3. MIP_presence_error 2.29
2.23.3.4 4. MIP_pointer_error 2.29
2.23.3.5 5. MIP_periodicity_error..2.29
2.23.3.6 6. MIP_ts_rate_error..2.30

2 Description of Measurement Functions

In this section you will find a comprehensive description of all measurement functions and error LEDs provided on the DVRM to monitor an MPEG2 transport stream. These measurement functions fully comply with the DVB guidelines for monitoring MPEG2 transport stream syntax (DVB Measurement Guidelines ETR 290). Provided they are transferable, the functions also find application with MPEG2 transport streams according to ATSC. Moreover, the data rate of null packets (PID 0x1FFF), the TS_ID (transport stream identity), the SI_OTHER tables (only DVB) as well as the parameters 'Program Paradigm' (only ATSC) can be monitored.

Moreover, the DVRM calculates the total transport stream data rate as well as the data rates of all programs contained in the transport stream and their elementary streams.

There are 10 LEDs on the front panel for displaying detected transport stream errors. Apart from this direct LED indication, the errors can be displayed with information on the type of error and processed with the aid of the 'Realtime Monitor' Windows software (see Annex C) supplied with the equipment. Furthermore, measurement parameters can be assigned to the 12 alarm contacts at the rear of the unit (see section 3.1.1). Refer also to the OPTION/INSTRUMENT SETTINGS/ALARM LINES menu of Realtime Monitor software.

2.1 Elements of Transport Stream Syntax

Fig. 2-1 gives a clear overview of the basic structure of a transport stream. Elements of the transport stream syntax which are relevant for monitoring such as packet header, adaptation field or PES (packetized elementary stream) header are shown in detail.

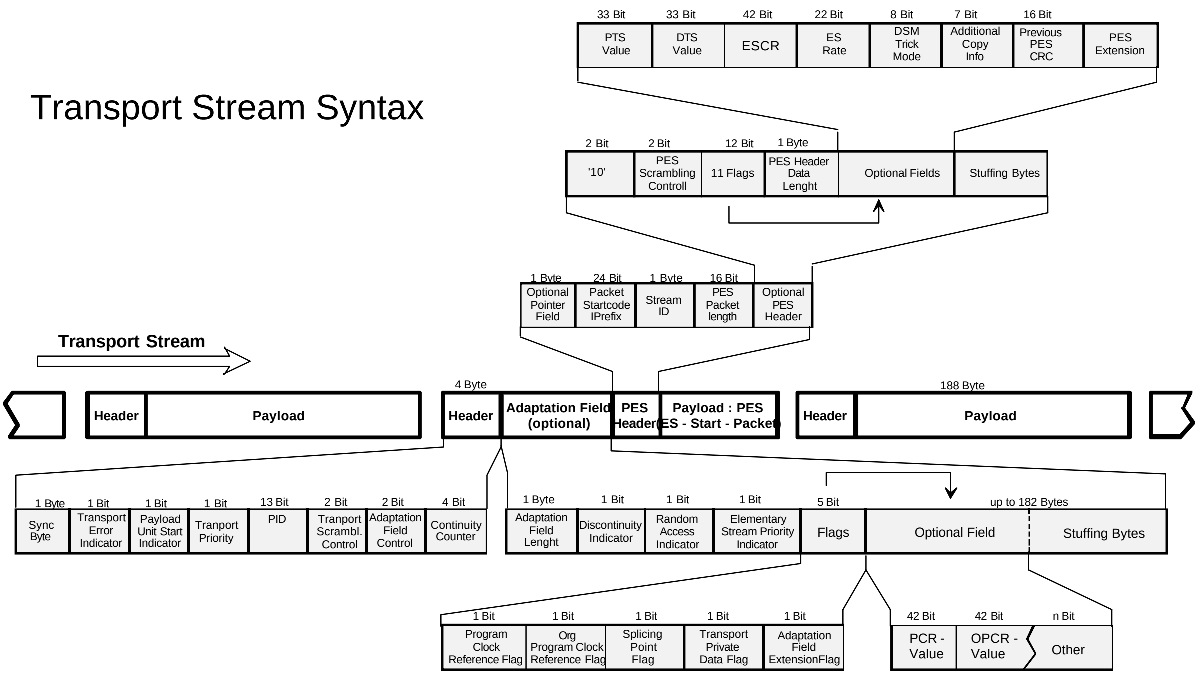


Fig. 2-1 Elements of transport stream syntax

2.2 Overview of All Measurement Functions

The measurement functions of the DVRM fully comply with the recommendations contained in the DVB Measurement Guidelines (ETR 290) for the analysis of MPEG2 transport stream syntax.

Table 2-1 gives an overview of all measurement functions of DVRM together with information on error indication either as a sum error (LED) or as a detailed error message in the STATISTICS/REPORT window of the Realtime Monitor software (see Annex C, page 4-13).

In addition to the measurement functions shown in Table 2-2 on the next page, the DVRM also calculates the following values from the transport stream multiplex:

- Total transport stream data rate [MBit/s]
- Data rates of all individual programs in the TS multiplex [Mbit/s]
- Data rates of all elementary streams of the individual programs in the TS multiplex [Mbit/s]
- Data rate for null packets
- Data rate for PSI/SI table

Note: *The data rates of programs and single streams are calculated by DVRM in two different ways depending on the setting in the OPTIONS/INSTRUMENT SETTINGS/SETUP menu of the Realtime Monitor software (see Annex C, page 4-6).*

With TOTAL selected, the data rates are the gross rates of the programs and single streams in the transport stream. Packet headers and possible adaptation fields are considered in the measured value.

With NET selected, only the payload of the programs and single streams is calculated without packet headers and possible adaptation fields (The transport stream data rate is always measured as the gross rate).

Measurement	Priority	Error indication			PID	Trigger	Error No.	DVB/ATSC
	ETR290	LED	Realtime Monitor software		info	on error	(ETR290)	
TS_sync_loss	1	◎ TS	Ts-Sync	Loss OK	- -	yes no	1.1	DVB/ATSC DVB/ATSC
Sync_byte_error	1	◎ SYNC	Sync Byte	Single Burst	no no	yes yes	1.2	DVB/ATSC DVB/ATSC
PAT_error	1	◎ PAT	PAT	Upper Distance Table ID Scrambled	yes yes yes	no yes yes	1.3	DVB/ATSC DVB/ATSC DVB/ATSC
Continuity_count _error	1	◎ CONT	Cont. Cnt	Packet Order More Than Twice Lost Packet	yes yes yes	yes yes yes	1.4	DVB/ATSC DVB/ATSC DVB/ATSC
PMT_error	1	◎ PMT	PMT	Upper Distance Scrambled	yes yes	no yes	1.5	DVB/ATSC DVB/ATSC
PID_error	1	◎ PID	PID Missing		yes	no	1.6	DVB/ATSC
Transport_error	2	◎ TRANS	Transport		yes	yes	2.1	DVB/ATSC

2.3 TS_sync_loss - (1st priority)

Each packet of the transport stream is preceded by a header consisting of four bytes. The first byte of the header is the synchronization byte (Sync Byte), whose content is always the hexadecimal value 0x47. In an MPEG2 decoder the sync byte serves for synchronization with the packetized transport stream. DVB recommendations define synchronism such that a sequence of at least five sync bytes has to be detected by an MPEG2 decoder. Synchronism is not attained or is lost if the sync bytes in at least three sequential TS packets are not detected as defined by the DVB recommendations. This status is referred to as TS_sync_loss. The synchronization hysteresis of 5/3 sync bytes recommended in the DVB guidelines is also a basic setting in the DVRM, although it may be modified for synchronization tests (see OPTIONS/INSTRUMENT SETTINGS/INPUT menu of the Realtime Monitor software, Annex C, page 4-5).

In the DVB Measurement Guidelines (ETR290) the preconditions for a TS_sync_loss message are set as follows:

TS_sync_loss is signalled if
• the content of the synchronization bytes in a sequence of three TS packets does not equal 0x47 (hexadecimal).

Two LEDs labelled TS and located one on top of the other at the very left of the front panel serve to display the synchronization status. An error (no synchronism) is indicated by the yellow LED (top), synchronism by the green LED (bottom) lighting up.

Measurement	Priority ETR290	Error indication			PID info	Trigger on error	Error No. (ETR290)	DVB/ATSC
		LED	Realtime Monitor software					
CRC_error	2	◎ CRC	CRC	PAT CAT PMT NIT EIT BAT SDT TOT MGT CVCT TVCT RRT STT CETT EIT-0 EIT-1 EIT-2 EIT-3 ETT-0 ETT-1 ETT-2 ETT-3	yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes	yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes	2.2	DVB/ATSC DVB/ATSC DVB/ATSC only DVB only DVB only DVB only DVB only DVB only ATSC only ATSC only ATSC only ATSC only ATSC only ATSC only ATSC only ATSC only ATSC only ATSC only ATSC only ATSC only ATSC only ATSC
PCR_error	2	◎ OTHER	PCR	Discontinuity Upper Distance	yes yes	yes no	2.3	DVB/ATSC DVB/ATSC
PCR_accuracy_err	2				yes	no	2.4	DVB/ATSC
PTS_error	2	◎ OTHER	PTS		yes	no	2.5	DVB/ATSC
CAT_error	2	◎ OTHER	CAT	Table ID Missing	yes yes	yes yes	2.6	DVB/ATSC DVB/ATSC
SI_repetition_error	3	◎ OTHER	SI REP	PAT Upp/Low Dist. CAT Upp/Low Dist. PMT Upp/Low Dist. NIT Upp/Low Dist. SDT Upp/Low Dist. BAT Upp/Low Dist. EIT Upp/Low Dist. RST Upp Dist. TDT Upp/Low Dist. TOT Upp/Low Dist. MGT Upp Dist. CVCT Upp Dist. TVCT Upp Dist RRT Upp Dist STT Upp Dist. EIT-0 Upp Dist.	yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes yes	no no no no no no no no no no no no no no no no	3.2	DVB/ATSC DVB/ATSC DVB/ATSC only DVB onlyDVB only DVB only DVB only DVB only DVB only DVB only ATSC only ATSC only ATSC only ATSC only ATSC only ATSC
NIT_error	3	◎ OTHER	NIT	Table ID NIT Upper Dist.	yes yes	yes no	3.1	only DVB only DVB
SDT_error	3	◎ OTHER	SDT	Table ID SDT Upper Dist.	yes yes	yes no	3.5	only DVB only DVB
EIT_error	3	◎ OTHER	EIT	Table ID EIT Upper Dist.	yes yes	yes no	3.6	only DVB only DVB
RST_error	3	◎ OTHER	RST	Table ID	yes	yes	3.7	only DVB
TDT_error	3	◎ OTHER	TDT	Table ID TDT Upper Dist.	yes yes	yes no		only DVB only DVB
BASE_PID_error	3	◎ OTHER	BASE_PID	Table_id	yes	yes	---	only ATSC
unreferenced_PID	3	◎ OTHER	Unref. PID		yes	yes	3.4	DVB/ATSC

Table 2-1 Overview of measurement functions from DVB Measurement Gudelines

Mesurement	Error indication LED	Realtime Monitor software		PID info	Trigger on error	DVB/ATSC
Datarate_error	◎ OTHER	DATARATE	Lower Limit Upper Limit	yes yes	no no	DVB/ATSC DVB/ATSC
Multiplex_error	◎ OTHER	MULTIPLEX	TS_ID	no	no	DVB/ATSC
SI_other_error	◎ OTHER	SI OTHER	NIT Upp/Low Dist. SDT Upp/Low Dist. EIT Upp/Low Dist.	yes yes yes	no no no	only DVB only DVB only DVB
NIT_error	◎ OTHER	NIT	Upper Dist	yes	no	only DVB
SDT_error	◎ OTHER	SDT	Upper Dist.	yes	no	only DVB
EIT_error	◎ OTHER	EIT	Upper Dist.	yes	no	only DVB
PARADIGM_error	◎ OTHER	PARADIGM	PMT PCR VIDEO AUDIO DATA	yes yes yes yes yes	no no no no no	only ATSC only ATSC only ATSC only ATSC only ATSC

Table 2-2 Additional measurement functions (not listed in ETR290)

2.4 Sync_byte_error - (1st priority)

Each packet of the transport stream is preceded by a header consisting of four bytes. The first byte of the header is the synchronization byte (sync byte), whose content is always the hexadecimal value 0x47. In the MPEG2 decoder the sync byte serves for synchronization with the packetized transport stream. If the sync byte is missing or contains errors too often, the decoder will not be able to synchronize to the transport stream.

DVRM checks the sync byte of every packet in the transport stream for correct contents.

In the DVB Measurement Guidelines (ETR290) the preconditions for a Sync_byte_error message are set as follows:

Sync_byte_error is signalled if
• the content of the synchronization byte in the TS header does not equal 0x47 (hexadecimal).

An error of this type is indicated by the LED labelled SYNC lighting up (line of LEDs at front panel). Additional information on the condition triggering the sync byte error message (SINGLE = single error or BURST = Burst error) can be obtained from the monitoring report of the Realtime Monitor software (see Annex C).

2.5 PAT_error - (1st priority)

The PAT (program association table) contains a list of all programs and PIDs contained in the transport stream and of associated PMTs (program map tables), which contain detailed program description. The PAT is therefore of key importance for decoding TV and audio programs. If the PAT is not available or contains an error, the MPEG2 decoder will not be able to select and decode a program from the transport stream multiplex.

The syntactic structure of a PAT is comprehensively defined in MPEG2 systems (ISO/IEC 13818-1). The PAT is exclusively transmitted in packets with 0x0000 as PID. The table may be divided up in several (max. 256) sections with the table index (table_id) of each section being 0x00.

In the DVB Measurement Guidelines (ETR290) the preconditions for a PAT_error message are set as follows (abbreviated designations in brackets are as text displayed in the monitoring report):

PAT_error is signalled if	
• PAT table index does not equal 0x00 (TABLE ID)	**or**
• PAT is transmitted in encrypted form (SCRAMBLED)	**or**
• Packets of PAT are not transmitted at least every 0.5 second (UPPER DISTANCE).	

An error of this type (sum error) is indicated by the LED labelled PAT lighting up (line of LEDs at front panel). Detailed information on the type of PAT error as listed in the above table can be obtained from the monitoring report of the Realtime Monitor software (Annex C).

2.6 PMT_error - (1st priority)

The PMT (program map table) is a table for detailed program descriptions referenced in the PAT. As essential information for the MPEG2 decoder it contains the PIDs of all packets of the individual TV, audio and data streams (elementary-stream PIDs) as well as the PIDs of packets serving for the transmission of PCR values associated with the program. Like the PAT, the PMT is therefore of key importance for decoding TV and audio programs. If PMT is not available or contains an error, the MPEG2 decoder will not be able to select and decode a program from the transport stream multiplex.

The syntactic structure of a PMT is defined in MPEG2 systems (ISO/IEC 13818-1). In contrast to the PAT, the PIDs of the individual PMTs are variable: MPEG2 permits values ranging from 0x0010 to 0x1 FFE. The table may be divided up in several (max. 256) sections with one section for each program. The table index (table_id) of each section must be 0x02.

In the DVB Measurement Guidelines (ETR290) the preconditions for a PMT_error message are set as follows (abbreviated designations in brackets are as text displayed in the monitoring report):

PMT_error is signalled if	
• PMT table index does not equal 0x02 (TABLE ID)	**or**
• PMT is transmitted in encrypted form (SCRAMBLED)	**or**
• Sections of PMT are not transmitted at least every 0.5 second (UPPER DISTANCE).	

An error of this type (sum error) is indicated by the LED labelled PMT lighting up (line of LEDs at front panel). Detailed information on the type of PMT error as listed in the above table can be obtained from the monitoring report of the Realtime Monitor software (see Annex C).

2.7 Continuity_count_error - (1st priority)

Each packet of the transport stream is preceded by a header consisting of four bytes. The fourth byte of the header contains the count of a four-bit continuity counter. The count must be increased by one for every packet of the transport stream that has the same PID. The count may consist of values ranging from 0 to 15; beyond 15, it will start from 0 again (modulo-16 counter). The continuity counter serves to recognize packets of a TV or audio program that are either missing or repeated more than once.

The MPEG2 standard also tolerates counter discontinuity, provided this is indicated by a so-called discontinuity indicator in the optional adaptation field (AF) of the same packet. This method is primarily used for the suppression of error messages when changing programs preceded by remultiplexing of the transport stream.

In case of so-called null packets, ie packets that do not contain any useful data but have a PID of 0x1FFF, continuity is not checked, as the value of the continuity counter in zero packets is not defined in the MPEG2 standard.

In the DVB Measurement Guidelines (ETR 290) the preconditions for a Continuity_count_error message are set as follows:

Continuity_count_error is signalled if	
• the same packet has been transmitted more than twice without discontinuity indicator (MORE THAN TWICE)	**or**
• a packet is missing, ie new count = old count + 2 without discontinuity indicator (LOST PACKET)	**or**
• there is a wrong sequence of packets, ie discontinuity without discontinuity indicator, without any of the above conditions present (PACKET ORDER)	

An error of this type (sum error) is indicated by the LED labelled CONT lighting up (line of LEDs at front panel). More detailed information on the type of Continuity_count error as listed in the above table can be obtained from the monitoring report of the Realtime Monitor software (see Annex C).

2.8 PID_error - (1st priority)

The PMT (Program Map Table) entries reveal the elementary-stream PIDs that are contained in the transport-stream multiplex. To decode a program with the corresponding PID, these packets must be contained in the transport stream, and for the MPEG2 decoder to function error-free these packets also need to be transmitted at certain intervals. The DVB Measurement Guidelines (ETR 290) speak of a 'user specified period', which means that it can be freely selected by the user. A value of 0.5 s is preset in the DVRM, but it can be modified in the OPTIONS/INSTRUMENT SETTINGS/LIMITS menu of the Realtime Monitor software according to individual requirements (see section 4.3.3.5).

In the DVB Measurement Guidelines (ETR 290) the preconditions for a PID_Error message are set as follows:

PID_error is signalled if
• the interval between two elementary-stream packets of the same PID > 0.5 seconds .

An error of this type (sum error) is indicated by the LED labelled PID lighting up (line of LEDs at front panel). More detailed information on the type of PID error as listed in the above table can be obtained from the monitoring report of the Realtime Monitor software (see Annex C).

2.9 Transport_error - (2nd priority)

The second byte of every packet header in the transport stream contains the so-called transport_error_indicator, which is a flag that serves to indicate bit errors in the following packet. This flag is generated and inserted by the Reed-Solomon decoder at the receiver end, if the decoder is no longer capable of correcting all bit and byte errors in the transport stream.

Because it is not possible in case of a set transport_error_indicator to predict which bit or byte contains an error, this packet must not be evaluated by an MPEG2 decoder. For this reason, in a case like this the DVRM only indicates the Transport_error, which means that the packet is not checked for further transport stream errors.

In the DVB Measurement Guidelines (ETR 290) the preconditions for a Transport_error message are set as follows:

Transport_error is signalled if
• a transport_error_indicator bit is set in the packet header.

An error of this type (sum error) is indicated by the LED labelled TRANS lighting up (line of LEDs at the front panel). As additional information, the PID of the packet containing the error can be obtained from the monitoring report of the Realtime Monitor software (Annex C).

Note: *The PID information contained in the monitoring report may be wrong when a transport error is indicated, if the bit error refers to the PID information of the packet header.*

2.10 CRC_error - (2nd priority)

If program-specific information (PSI tables) such as PAT, CAT, PMT, NIT, EIT, SDT, BAT and TOT is transmitted, a value for checking the check sum of this section is inserted at the end of each table section. The so-called CRC (cyclic redundancy check) is used for calculating the check sum at the transmitter and receiver end. Combined with the additionally transmitted CRC value, the check sum for each table section must be zero.

If the resulting check sum does not equal zero, the MPEG2 decoder must reject the information contained in this table.

If a CRC_error is detected, it cannot be predicted which part of the information contained in the table is not correct. In this case, the DVRM signals the CRC_error, but the transport stream is not checked for further errors which are derived from the faulty content of this table (eg search for PMT PIDs from a PAT or ES PIDs from a PMT).

In the DVB Measurement Guidelines (ETR 290) the preconditions for a CRC_error message are set as follows:

CRC_error is signalled if
• a packet with PAT, CAT, PMT, NIT, EIT, BAT, SDT or TOT and CR check of a section does not equal zero.

For the measurement on transport streams in line with ATSC the following applies:

CRC_error is signalled if
• a packet with PAT, CAT, PMT, MGT, TVCT, CVTC, RRT, STT, CETT, EIT-0, EIT-1, EIT-2, EIT-3, ETT-0, ETT-1, ETT-2 or ETT-3 and CRC check of a section does not equal zero.

An error of this type (sum error) is indicated by the LED labelled CRC lighting up (line of LEDs at front panel). As additional information, the PID of the packet containing the error can be obtained from the monitoring report of the Realtime Monitor software (see Annex C).

2.11 PCR_error, PCR_accuracy_error - (2nd priority)

In every transport stream, coded time values obtained from the system time are transmitted to enable the MPEG2 decoder to link its own clock to the system clock of the coder to allow decoding (PLL). Each program contained in the transport stream may have its own independent program system clock (referenced in PMT). The program map table (PMT) reveals for all programs contained in the transport stream in which packets (PIDs) the PCR (program clock reference) values are transmitted.

PCR values are transferred in the optional adaptation field with a width of 42 bits. The 42 bits consist of two parts: a PCR base with 33 bits and the PCR extension with 9 bits. The following formulae hold for the bit structure :

PCR base (i) = (system clock frequency * t(i) DIV 300) % 2^{33}

PCR extension (i) = (system clock frequency * t(i) DIV 1) % 300

PCR (i) = (PCR base (i) * 300) + PCR extension (i)

A 42 bit PCR value coded this way starts again from count 0 after the elapse of 2^{33} * 300 clocks (correponds to a period of approx. 26.5 hours at 27 MHz).

The MPEG2 standard also tolerates discontinuity of successive PCR values, provided this is indicated by the so-called discontinuity indicator in the optional adaptation field (AF) of the same packet. This method is primarily used for the suppression of PCR error messages when changing programs preceded by remultiplexing of the transport stream.

In the DVB Measurement Guidelines (ETR 290) the preconditions for a PCR_Error message are set as follows:

PCR_error is signalled if	
• the difference of two consecutive PCR values of a program > 100 ms and no discontinuity is indicated in the optional adaptation field (DISCONTINUITY)	**or**
• the interval between two packets with PCR values of a program > 40 ms (UPPER DISTANCE).	

The above mentioned intervals are given in the DVB Measurement Guidelines (ETR 290) and are preset in the DVRM, but they may be modified according to individual requirements in the OPTIONS/INSTRUMENT SETTINGS/LIMITS menu of the Realtime Monitor software (see Annex C).

The DVB Measurement Guidelines also recommend additional monitoring of the accuracy of the PCR values transmitted. Accuracy of PCR values may be impaired by inaccurate calculation of the 42 bit PCR word width or by errors during modification of PCR values in a remultiplex.

Note : *The term accuracy in this case does not refer to the absolute frequency accuracy of the 27MHz system clock, but to the fluctuation width of the PCR values of a program, which is caused by the above mentioned errors.*

The MPEG2 standard (ISO/IEC 13818-1) as well as the DVB Measurement Guidelines (ETR 290) prescribe a maximum tolerance of ±500 ns for PCR values. The MPEG2 standard (ISO/IEC 13818-4) also describes a method to be used for testing the so-called timing accuracy, which serves to monitor the compliance of these PCR tolerances. The description of this method contains an inequation, which must be fulfilled for all PCR values of a program. This inequation is as follows:

$$\frac{(i - i''-1)}{PCR(i) - PCR(i'') + \delta} \leq k \leq \frac{(i - i''+1)}{PCR(i) - PCR(i'') - \delta}$$

i being index of the byte, in which the current PCR value was transmitted

i'' being index of the byte, in which the previous PCR value was transmitted

δ being $27 + 810 * (PCR(i)-PCR(i'')) / 27E6$;

If the above inequation is not fulfilled for any consecutive two pairs of PCR values, (maximum value of the left side and minimum values of the right side are stored) the precondition for a PCR_accuracy_error is fulfilled.

PCR_accuracy_error is signalled if
• the PCR tolerance within a program >±500 ns.

A PCR_error and a PCR_accuracy_error are indicated as a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs at front panel). More detailed information on the type of PCR error as listed in the above tables can be obtained from the monitoring report of the Realtime Monitor software (see Annex C).

2.12 PTS_error - (2nd priority)

PTS values (presentation time stamps) in the PES headers are transmitted by transport stream packets of a program. They enable the MPEG2 decoder to identify the exact time when a transmitted data block (TV picture in case of video streams and beginning of an audio sequence in case of audio streams) is to be presented. The time stamps are transmitted with a word width of 33 bits and relate to the 27 MHz system clock transmitted in the transport stream together with the PCR values.

In the DVB Measurement Guidelines (ETR 290) the preconditions for a PTS_error message are set as follows:

PTS_error is signalled if
• the magnitude of the difference of two consecutive PTS values of a program is >700 ms.

The above-mentioned interval is given in the DVB Measurement Guidelines (ETR 290) and is preset in the DVRM, but it can also be modified according to individual requirements in the OPTIONS/INSTRUMENT SETTINGS/LIMITS menu of the Realtime Monitor software (see Annex C).

An error of this type is indicated in a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs at front panel). As an additional information, the PID of the program containing the faulty PTS values can be obtained from the monitoring report of the Realtime Monitor software.

2.13 CAT_error - (2nd priority)

If encrypted data are contained in a packet of the transport stream, this must be indicated in the packet header (2nd byte) in the field that is two bits wide and labelled transport_scrambling_control. The individual values indicate the following:

Value (binary)	Description
00	No encrypted data contained in the packet
01, 10, 11	Defined by user

If encrypted data are to be transmitted, the MPEG2 standard recommends the additional transmission of the tables containing the encryption data (conditional access table CAT) in separate packets with (0x0001) as PID and 0x01 as table index.

The MPEG2 standard prescribes that packet headers including the optional adaptation fields must not be transmitted in encrypted form. According to DVB specifications (ETS 300468), the same applies to tables containing service information (SI tables PAT, PMT, NIT, EIT, BAT, TDT, TOT and SDT). The only exception to this rule is the EIT (event information table) when program overviews are transmitted.

In the DVB Measurement Guidelines (ETR 290) the preconditions for a CAT_error message are set as follows :

CAT_error is signalled if	
• a packet contains encrypted data, but no CAT is found (MISSING)	**or**
• a packet with CAT-PID (0x0001) is found, but table index does not equal 0x01 (TABLE ID).	

An error of this type is indicated as a sum error indicating several single errors by the LED labelled OTHER lighting up (line of LEDs at front panel). Additional information on the type of error as listed in the above table can be obtained from the monitoring report of the Realtime Monitor software (see Annex C).

2.14 SI_repetition_error - (3rd priority)

The DVB standard (ETR211) prescribes minimum and maximum intervals for the repetition of individual packets and complete tables containing service information (SI) (see Table 2-3). These values are preset in the DVRM but each value can be modified according to individual requirements in the OPTIONS/INSTRUMENT SETTINGS/LIMITS menu of the Realtime Monitor software (see Annex C).

Service information	Max. interval (complete table)	Min. interval (individual sections)	DVB / ATSC
PAT	0.5 s	25 ms	DVB / ATSC
CAT	0.5 s)[1]	25 ms)[1]	DVB / ATSC
PMT	0.5 s	25 ms	DVB / ATSC
NIT	10 s)[2]	25 ms)[2]	only DVB
SDT	2 s)[3]	25 ms)[3]	only DVB
BAT	10 s)[1]	25 ms)[1]	only DVB
EIT	2 s)[4]	25 ms)[4]	only DVB
RST	---	25 ms	only DVB
TDT	30 s	25 ms	only DVB
TOT	30 s	25 ms	only DVB
MGT	0.15 s	---	only ATSC
TVCT	0.4 s	---	only ATSC
CVCT	0.4 s	---	only ATSC
RRT	60 s	---	only ATSC
STT	1 s	---	only ATSC
EIT-0	0.5 s	---	only ATSC

)[1] If present

)[2] Monitors table_id 0x40 (NIT actual). Concerning table_id 0x41 (NIT other) see parameter NIT OTHER

)[3] For the current transport stream multiplex only (table_id 0x42)

)[4] For the current transport stream multiplex only; present/following (table_id 0x4E)

Table 2-3 Repetition rates for service information according to DVB or ATSC

In the DVB Measurement Guidelines (ETS 300 468) the preconditions for a SI_repetition_error are set as follows , (*** is replaced by the abbreviation used in the appropriate SI table, eg NIT):

SI_repetition_error is signalled if	
• time difference between SI tables is too long (*** UPPER DISTANCE)	**or**
• time difference between SI sections is too short (*** LOWER DISTANCE).	

An error of this type is indicated as a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs at front panel). Additional information on the PID of the table, in which a repetition rate error was found, can be obtained from the monitoring report of the Realtime Monitor software (see Annex C).

2.15 NIT, SDT, EIT, RST and TDT_error - (3rd priority only in operating mode DVB)

These types of service information (SI) are also inserted into the transport stream as additional data (multiplex) and contain among other things the current date, time, description of the TV program, etc. Each of these tables is transmitted in the form of packets with a given packet number (PID) and must be contained in the transport stream at certain intervals according to the DVB specification. However, not every SI has a different PID, but packets for TDT and TOT as well as SDT and BAT have identical PIDs. These tables are differentiated by an entry in the table header, the so-called table index (table_id). This table_id enables an MPEG2 decoder working in compliance with the DVB standard to identify the type of service information it is dealing with.

gives an overview of the service information according to the DVB guideline ETS 300468:

Service Information	PID [hex]	Permitted table_id [hex]	Max. interval [sec]
NIT	0x0010	0x40, 0x41	10)[1]
SDT	0x0011	0x42, 0x46	2)[2]
BAT	0x0011	0x4A	10)[3]
EIT	0x0012	0x4E to 0x4F, 0x50 to 0x6F	2)[4]
RST	0x0013	0x71	-
TDT	0x0014	0x70	30
TOT	0x0014	0x73	30
Stuffing Table	0x0010 to 0x0013	0x72	-

)[1] NIT_error monitors table_id 0x40 (NIT actual) only. Concerning table_id 0x41 (NIT other) see parameter NIT OTHER

)[2] For the current transport stream multiplex only (table_id 0x42)

)[3] If present only

)[4] For the current transport stream multiplex only pesent/following (table_id 0x4E)

Table 2-4 Overview of service information according to DVB guideline ETS 300468

In the DVB Measurement Guidelines (ETR 290) the preconditions for NIT_error, SDT_error, EIT_error, RST_error or TDT_error messages are set as follows , (*** is replaced by the abbreviation used in the appropriate SI table, eg NIT):

NIT_error, SDT_error, EIT_error, RST_error or TDT_error are signalled if	
• a packet with the PID of an SI packet NIT, SDT, EIT, RST, TDT or TOT, but wrong table index is contained in the transport stream (TABLE ID)	**or**
• time span between SI tables of NIT, SDT, EIT, TDT or TOT is too long (*** UPPER DISTANCE).	

An error of this type is indicated as a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs at front panel). As additional information, the table_id of the erroneous packet or a possible repetition rate error can be obtained from the monitoring report of the Realtime Monitor software (Apendix C).

2.16 BASE_PID_error - (3rd priority only in operating mode ATSC)

In the so-called BASE_PID (0x1FFB) the following tables are transferred according to ATSC document A/65:

MGT	Tab_id: 0xC7
TVCT	Tab_id: 0xC8
CVCT	Tab_id: 0xC9
RRT	Tab_id: 0xCA
STT	Tab_id: 0xCD

In the BASE_PID no tables with other Tab_id may be present.

The following applies to the monitoring of BASE_PID:

BASE_PID_error is signalled if:
• a packet with the PID 0x1FFB contains a table with a table_id other than 0xC7 to 0xCA or 0xCD.

An error of this type is indicated in a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs on front panel).

2.17 Unreferenced_PID - (3rd priority)

The PMT (program map table) contains a complete list of all program definitions contained in the transport stream. In turn, each program definition contains the information on all PIDs of the elementary stream packets of this program. This means that by evaluating the PMT one obtains the PIDs of all transport stream packets which are permitted to convey the useful data of the program. Besides these referenced packets, the transport stream multiplex is only permitted to contain packets with program-specific information (PSI tables) such as PAT, CAT, CA-PID, PMT, NIT, BAT, SDT, TDT, TOT, EIT, RST or having packet numbers reserved by the MPEG2 standard.

Only if the program is changed (new PMT) a non-referenced PID may be contained in the transport stream during 0.5 s according to the DVB Measurement Guidelines (ETR 290). This interval of 0.5 s is preset in the DVRM to comply with DVB, but it can be modified according to individual requirements in the OPTIONS/INSTRUMENT SETTINGS/LIMITS menu of the Realtime Monitor software (see Annex C).

In the DVB Measurement Guidelines (ETR 290) the preconditions for an Unreferenced_PID_error message are set as follows:

Unreferenced_PID_error is signalled if
• the transport stream contains a packet with a PID which is not the PID of PAT, CAT, CA-PID, PMT, NIT, BAT, SDT, TDT, TOT, EIT or RST and which is not referenced in a PMT that is at least 0.5 second old (ES-PID or PCR-PID).

An error of this type is indicated as a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs at front panel). As additional information, the PID of the packet containing the error can be obtained from the monitoring report of the Realtime Monitor software (see Annex C).

2.18 DATARATE_error

The DVRM monitors the data rate of the null packets, which have the packet identity (PID) 0x1FFF. An upper and a lower limit can be set (see OPTIONS/INSTRUMENT SETTINGS/LIMITS menu of Realtime Monitor software, Annex C). Null packets are TS packets by which the transport stream can be filled up to obtain a specific data rate. Every change of the data rate of a single stream belonging to the transport stream causes the change the data rate of the null packet the other way round (not with Statistical Multiplex). Adding or loosing of a single stream produces the same result.

DATARATE_error is signalled if	
• the data rate of the null packets (PID 0x1FFF) is higher than the specified upper limit.	**or**
• the data rate of the null packets (PID 0x1FFF) is lower than the specified lower limit.	

An error of this type is indicated in a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs on front panel).

2.19 MULTIPLEX_error

The DVRM monitors the identity of the transport stream (TS_ID). A fixed value or range of valid values can be set (see OPTIONS/INSTRUMENT SETTINGS/LIMITS menu of the Realtime Monitor software, Annex C). This way it can be checked if the right transport stream is monitored.

MULTIPLEX_error is signalled if
• the transport stream identity is outside the specified values

An error of this type is indicated in a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs on front panel).

2.20 SI_OTHER_error (only in operating mode DVB)

Additional to the monitoring of the SI table intervals according to ETR290 the intervals of the SI_OTHER tables are watched. Each limit can be set according to the individual requirements (see OPTIONS/INSTRUMENT SETTINGS/LIMITS menu of Realtime Monitor software, Annex C). The values of the table below follows DVB guidelines (ETS 400 368).

Service information	Table_id	Max. interval (complete table)	Min. interval (individual sections)
NIT_OTHER	0x41	10 s	25 ms
SDT_OTHER	0x46	2 s	25 ms
NIT_OTHER	0x4F*	2 s	25 ms

***** EIT OTHER; present/following

Table 2-5 Repetition rates for service information concernig other networks (NIT_OTHER) or other transport streams (SDT_OTHER, EIT_OTHER) according to DVB

SI_OTHER_error is signalled if	
• time difference between SI_OTHER tables is too long (*** UPPER DISTANCE)	**or**
• time difference between SI_OTHER sections is too short (*** LOWER DISTANCE).	

*** is replaced by the abbreviation used in the appropriate SI table, eg NIT.

An error of this type is indicated in a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs on front panel).

2.21 NIT_OTHER_error, SDT_OTHER_error, EIT_OTHER_error (only in operating mode DVB)

Additional to the monitoring of the SI table intervals according to ETR290 the intervals of the SI_OTHER tables are watched. Each limit can be set according to individual requirements (see OPTIONS/INSTRUMENT SETTINGS/LIMITS menu of Realtime Monitor software, Annex C). The values of the table below follows DVB guidelines (ETS 400 368).

Service Information	PID [hex]	Permitted table_id [hex]	Max. interval [sec]
NIT_OTHER	0x0010	0x40, 0x41	10
SDT_OTHER	0x0011	0x42, 0x46	2
EIT_OTHER	0x0012	0x4F, 0x60 to 0x6F	2*

*only EIT OTHER; present/following (table id: 0x4F) is monitored

Table 2-6 Service information concerning other networks (NIT_OTHER) or other transport streams (SDT_OTHER, EIT_OTHER) according to DVB guideline ETS 300468

NIT_OTHER_error, SDT_OTHER_error, EIT_OTHER_error are signalled if
• time span between SI tables of NIT other, SDT other or EIT other is too long (*** UPPER DISTANCE).

*** is replaced by the abbreviation used in the appropriate SI table, eg NIT.

An error of this type is indicated in a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs on front panel).

2.22 PARADIGM_error (only in operating mode ATSC)

The ATSC document A/53 defines the 'Program Paradigm‘. According to this document, the PIDs of all elementary streams of a program (eg video, audio) should have a reference to the program number:

A base_PID can be defined for each program in the transport stream (permissible program numbers 2 to 255) according to the following formula:

- base_PID = program number << 4

The PIDs of PMT, the video elementary stream, the audio elementary stream and the data stream are derived from the base_PID plus a fixed differential value. The PCR should be contained in the video elementary stream (see error condition).

The program number 1 is excluded to ensure compatibility to DVB streams in which PIDs 0x10 to 0x14 are reserved for service information.

PARADIGM_error is signalled if:	
• the PID with the value of base_PID does not contain the PMT of the program	**or**
• the PID with the value (base_PID + 0x01) does not contain a video elementary stream	**or**
• the PID with the value (base_PID + 0x01) does not contain the PCR of the program	**or**
• the PID with the value (base_PID + 0x04) does not contain an audio elementary stream	**or**
• the PID with the value (base_PID + 0x0A) does not contain a data stream	

An error of this type is indicated in a sum error signalling several single errors by the LED labelled OTHER lighting up (line of LEDs on front panel).

Exception:
If 'Program_Paradigm' is not met, the most significant bit of all the PIDs (b12) that belong to the program should be set to '1'.

2.23 MIP_error(only for DVB mode)

The MIP (megaframe initialization packet) is included in the transport streams of single-frequency networks. The MIP parameter is only tested if the transport stream contains the PID 0x15 (MIP PID).

2.23.1 General description

The following block diagram shows a complete SFN system network:

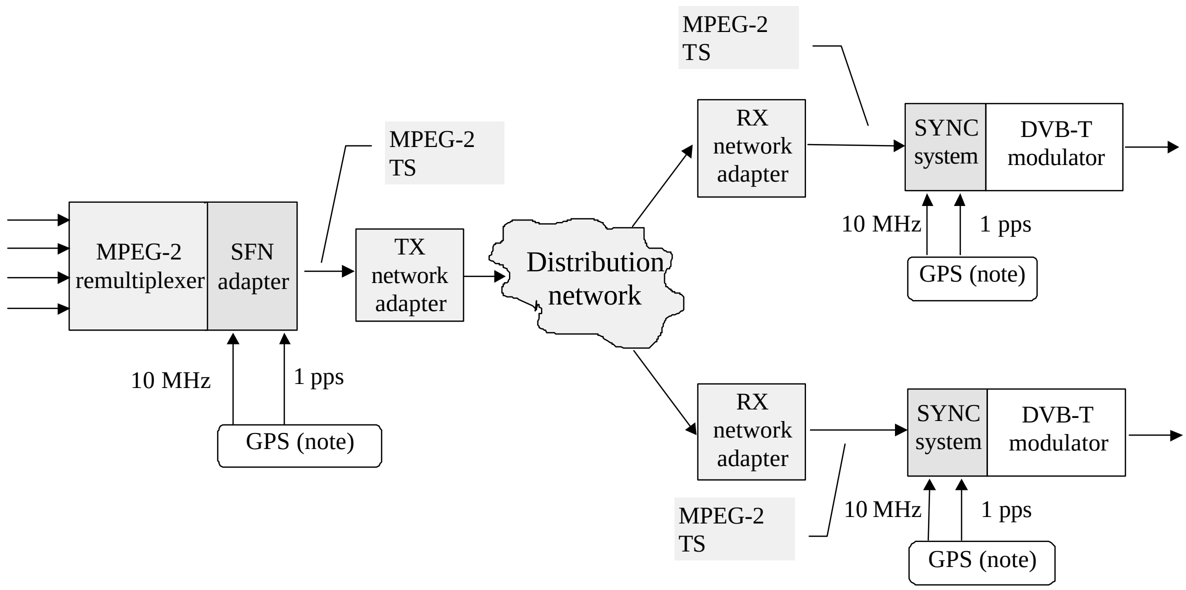


Fig. 2-2 SFN system network

Major components of the network are the SFN adapter and the SYNC system.

The **SFN adapter** forms a mega-frame consisting of n TS packets, and inserts a Mega-frame Initialization Packet (MIP) with a dedicated PID value. The MIP inserted anywhere within a mega-frame allows to uniquely identify the starting point (i.e. the first packet) of the mega-frame. The time difference between the latest pulse of the one-pulse-per-second reference derived for example from GPS that precedes the start of the mega-frame M+1 and the actual start (i.e. first bit of first packet) of the mega-frame M+1 is copied into the MIP_M. This parameter is called Synchronization Time Stamp (STS).

The **SYNC system** provides a propagation time compensation by comparing the inserted STS with the local time reference and calculates the extra delay needed for SFN synchronization.

For further information refer to TS 101 191: "Digital Video Broadcasting (DVB); DVB mega-frame for Single Frequency Network (SFN) synchronization".

2.23.2 Parameter overview

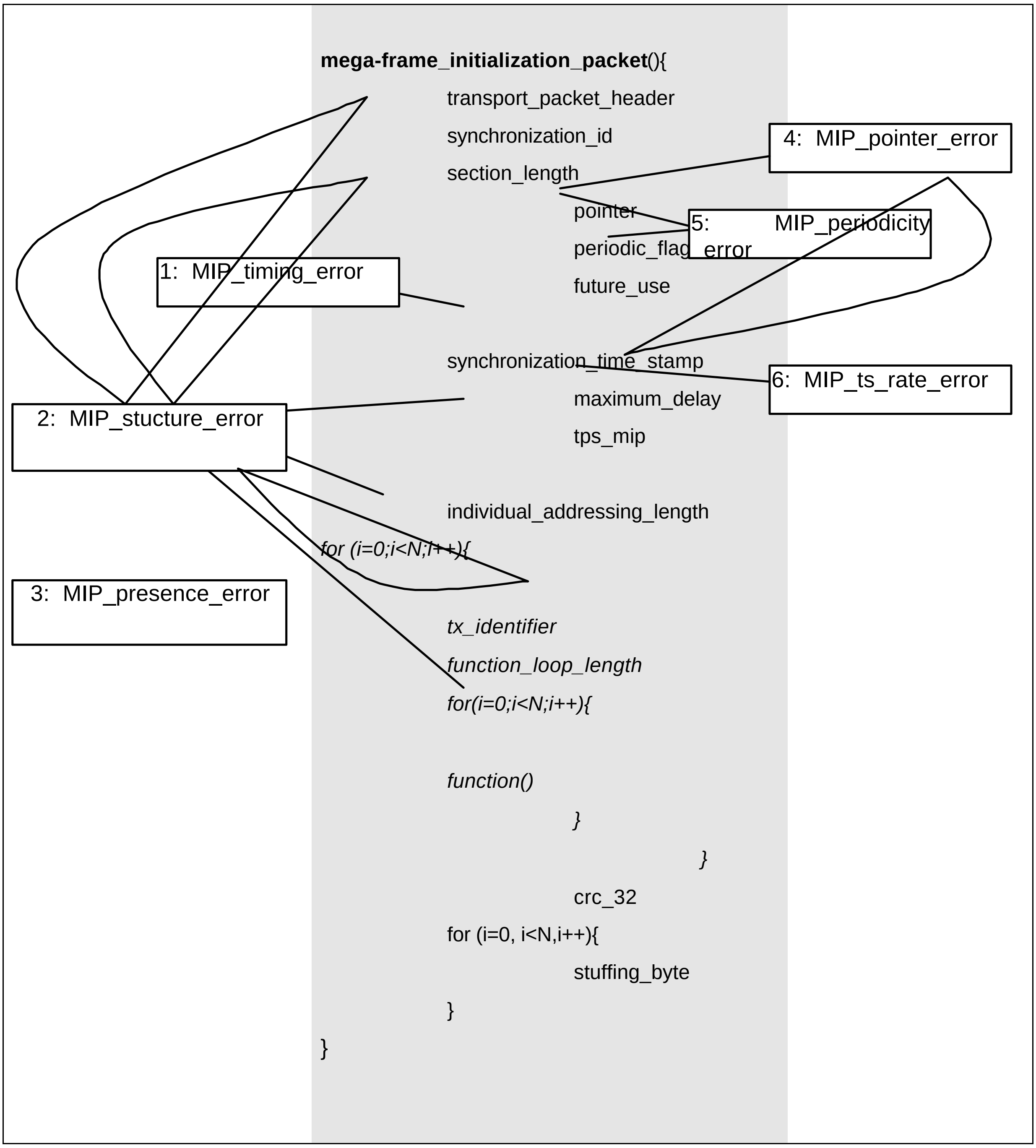


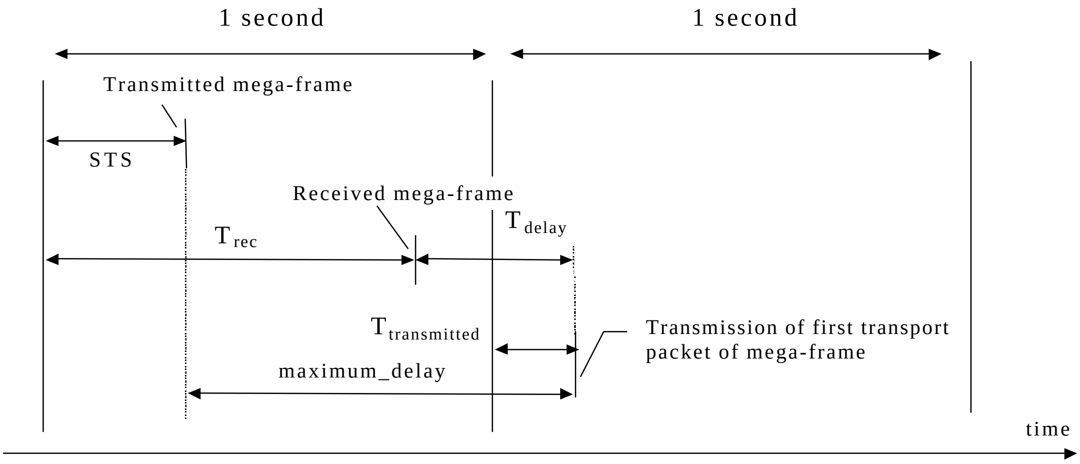
Fig. 2-3 Parameter overview

2.23.3 Parameter description

MIP - ERROR set if:	
• MIP_timing_error set (see 2.23.3.1)	**or**
• MIP_structure_error set (see 2.23.3.2)	**or**
• MIP_presence_error set (see 2.23.3.3)	**or**
• MIP_pointer_error set (see 2.23.3.4)	**or**
• MIP_periodicity_error set (see 2.23.3.5)	**or**
• MIP_ts_rate_error set (see 2.23.3.6)	

2.23.3.1 1. MIP_timing_error

The synchronization_time_stamp (STS) of MIP_M contains the time difference, expressed as a number of 100 ns steps, between the latest pulse of the one-pulse-per-second reference derived for example from GPS that precedes the start of the mega-frame M+1 and the actual start (i.e. first bit of first packet) of the mega-frame M+1.



This test verifies that successive STS values are self-consistent within a user defined limit

2.23.3.2 2. MIP_structure_error

The structure of the MIP is defined by TS 101 191. At the transmitter end, the SYNC system and the modulator require a compliant structure.

This test verifies that the syntax of the MIP complies with the specification in TS 101 191

The following tests are performed:

a. Header	Does the TS header comply with TS101 191?
b. Field length	Are all field lengths consistent to provide a proper length packet?
c_1. Max_delay	Is the maximum_delay in a range of 0x0 to 0x98967F?
c_2. STS	Is the synchronization_time_stamp in a range of 0x0 to 0x98967F?
d. CRC	Does the CRC_32 field match the CRC calculated from the MIP data?

2.23.3.3 3. MIP_presence_error

Each mega-frame contains exactly one Mega-frame Initialization Packet (MIP). The actual position may vary in an arbitrary way from mega-frame to mega-frame.

This test verifies that exactly one MIP is inserted into each mega-frame

The following tests are performed:

a. Extra MIP	Does every mega-frame contain not more than one MIP?
b. Missing	Does every mega-frame contain at least one MIP?

2.23.3.4 4. MIP_pointer_error

The pointer – a 2-byte binary integer – indicates the number of transport packets between the MIP and the first packet of the succeeding mega-frame.

This test verifies that the mega-frame size indicated by the pointer matches the mega-frame size calculated from the tps_mip.

2.23.3.5 5. MIP_periodicity_error

The periodic_flag indicates if a periodic or an aperiodic insertion of the MIP is performed. Periodic insertion means that the value of the pointer is not time varying. A "0" indicates aperiodic mode and a "1" indicates periodic mode. All SFN SYNC systems shall be able to handle both aperiodic and periodic modes.

With the periodic_flag set to ”1”, this test verifies that the pointer value and the mega_frame size remain constant.

The following tests are performed:

a. pointer Does the pointer value remain constant?

b. MF size Does the mega-frame size calculated from the packets between each MIP remain constant?

2.23.3.6 6. MIP_ts_rate_error

In a SFN network the modulator settings are transmitted by the tps_mip (see TS 101 191, chapter 6, Table 3). These settings determine the transmission channel and thus the bit rate of the transport stream.

This test verifies that the actual transport stream data rate matches the modulator setting given by the tps_mip.

3 Preparation for Use ... 3.1

3.1 Legend for Front and Rear View ... 3.1
3.1.1 Front Panel ... 3.1
3.1.2 Rear Panel ... 3.3

3.2 Putting into Operation ... 3.3
3.2.1 Unpacking the Unit ... 3.4
3.2.2 Positioning the Unit ... 3.4
3.2.2.1 Desktop Model ... 3.4
3.2.2.2 Installation in a 19" Rack ... 3.4
3.2.3 EMC Safety Measures ... 3.5
3.2.4 AC Supply Voltage ... 3.5

3.3 Function Test (Switch-on Test) ... 3.5

3 Preparation for Use

3.1 Legend for Front and Rear View

3.1.1 Front Panel

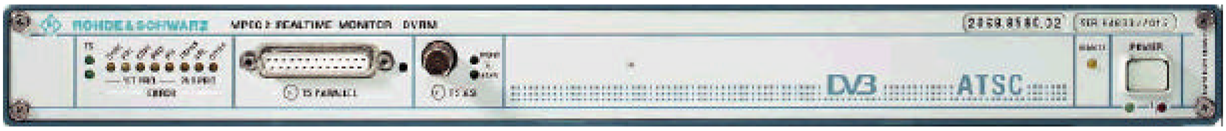
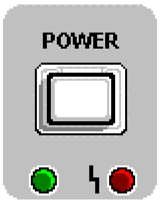


Fig. 3-1 Front view

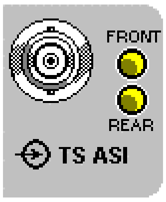


POWER

After the unit has been connected to the AC supply and the power switch has been pressed, the green LED lights signalling that the unit functions properly.

The red LED signals a defect. The unit has to be switched off and disconnected from AC supply to undergo repair.

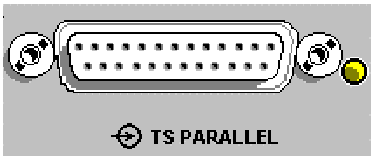
Quick flashing of the red LED when the unit is switched on or off does not indicate a defect.



TS ASI

Input for an MPEG2 transport stream to DVB DOCUMENT A010: ASYNCHRONOUS SERIAL INTERFACE

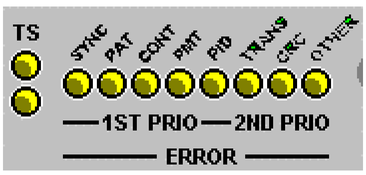
Another input of this type is provided at the rear of the unit. If one of the two inputs is selected, the respective LED lights.



TS PARALLEL (LVDS)

Input for an MPEG2 transport stream to DVB DOCUMENT A010: SYNCHRONOUS PARALLEL INTERFACE

If this input is selected, the LED is on.



ERROR INDICATION

The two TS LEDs indicate whether the instrument identifies an MPEG2 transport stream at the selected input. The user-selectable hysteresis parameters in the OPTIONS/INSTRUMENT SETTINGS/INPUT menu of the Realtime Monitor software (see Annex XXX) are decisive. If a transport stream is identified, the green LED (bottom) lights up, otherwise the yellow LED (top) is on.

The other eight LEDs signal protocol errors in the transport stream in line with DVB TM 1601. The following errors are signalled.

SYNC	Sync_byte_error
PAT	PAT_error
CONT	Continuity_count_error
PMT	PMT_error
PID	PID_error
TRANS	Transport_error
CRC	CRC_error
OTHER	<u>DVB standard</u>: PCR_error, PCR_accuracy_error, PTS_error, CAT_error, SI_repetition_error, NIT_error, SDT_error, EIT_error, RST_error, TDT_error, unreferenced_PID Datarate_error, Multiplex_error, SI_other_error, NIT_other_error, SDT_other_error, EIT_other_error <u>ATSC standard</u>: PCR_error, PCR_accuracy_error, PTS_error, CAT_error, SI_repetition_error, BASE_PID_error, unreferenced_PID, Datarate_error, Multiplex_error, PARADIGM_error.

The error messages are indicated at intervals of one second.

3.1.2 Rear Panel

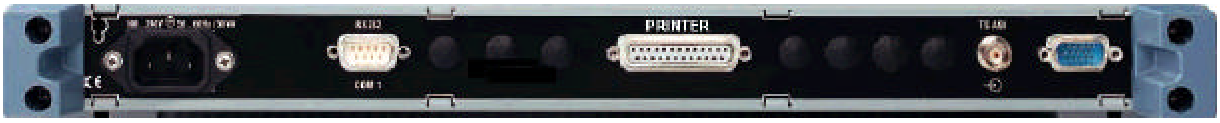
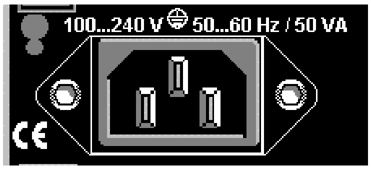


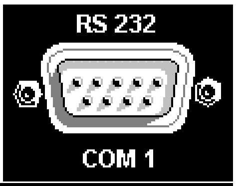
Fig. 3-2 Rear view



POWER

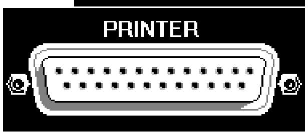
Power connector

The attached flap may be used to prevent the power cable from becoming disconnected.



RS 232

The serial interface is for remote control.



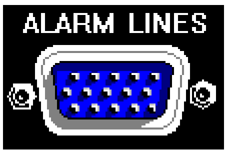
PRINTER

When option B5 is fitted, a 25-contact connector with parallel CENTRONICS interface is provided for connecting a PC printer.

TS ASI

Input for an MPEG2 transport stream to DVB DOCUMENT 010: ASYNCHRONOUS SERIAL INTERFACE

When this input is selected, the front-panel LED marked TS ASI REAR lights.



ALARM LINES

When option B5 is fitted, a 15-contact connector with the outputs of the 12 alarm lines is provided.

3.2 Putting into Operation

Caution:

Prior to putting the DVRM into operation, make sure that

- the power connector is provided with protective earth,
- the ventilation openings are not obstructed,
- the signal voltage levels applied to the inputs are not above the permissible limits,
- the outputs of the instrument are not overloaded or connected incorrectly.

Any non-compliance with these precautions may cause the instrument to be damaged.

3.2.1 Unpacking the Unit

After the unit has been unpacked, check whether the delivery is complete:

- MPEG2 Realtime Monitor DVRM
- Power cable
- This operating manual
- Realtime Monitor software
- Modem bypass cable for RS-232 connection between PC and DVRM

Carefully inspect the unit for mechanical damage. Should you detect any damage, immediately inform the forwarding agent that delivered the unit. In this case, make sure to keep the cardboard box and the packing material.

3.2.2 Positioning the Unit

3.2.2.1 Desktop Model

DVRM is intended for indoor use. The site should meet the following requirements:

- The ambient temperature should be between + 5 and + 45 °C.
- The ventilation openings and the air outlet on top of the unit should not be obstructed.

For use in the lab or on a desk, fold out the feet at the bottom of the unit.

3.2.2.2 Installation in a 19" Rack

Caution:

When installing the unit in a rack, make sure that the air inlet and outlet are not obstructed.

To install the DVRM in a 19" rack, an Adapter ZZA-91 (Order No. 0396.4870.00) is required. Installation instructions are supplied with the adapter.

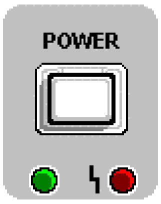
3.2.3 EMC Safety Measures

To prevent electromagnetic interference, the unit should be closed when in use. Only suitable shielded signalling and control cables are to be used.

3.2.4 AC Supply Voltage

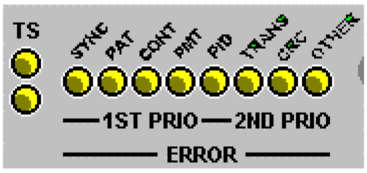
DVRM may be operated on an AC supply from 100 to 240 V at frequencies of 47 to 63 Hz. The power connector is located at the rear of the unit (see above).

3.3 Function Test (Switch-on Test)



The unit is switched on by pressing the POWER switch. DVRM then performs a built-in test which is completed after a few seconds. At the beginning of the boot phase all LEDs on the front panel light up and a memory test is carried out. In the case of a memory error, all LEDs will be blinking. If there is no error, the LEDs are switched off and booting is continued. The individual steps of the boot phase are indicated by LEDs. The SYNC LED corresponds to the least significant bit, the PAT LED to the next higher bit, etc. If an error occurs during the boot phase, the current status indicated by the LEDs will be retained (see Table 3-1). Contact your local Rohde & Schwarz service center in this case.

Otherwise the instrument is set to the previously used instrument status and the status menu is displayed.



The green LEDs next to the front-panel inputs indicate at which input the instrument expects a signal. If a valid MPEG2 signal is applied to this input, the instrument must synchronize to it. After synchronization has been completed, the lower green LED below TS at the very left comes on.

LED code	Boot phase
0x01	Check update
0x02	Loading of main program
0x03	Starting of main program
0x32	Reset of MPEG2 chipset
0x35, 0x36, 0x37	Initialization and loading of MPEG2 demultiplexer
0x38	Loading of MPEG2 audio decoder
0x39	Loading of MPEG2 video decoder
0x3A	Loading of video DAC
0x3B	Starting of MPEG2 demultiplexer and MPEG2 video decoder
0x3C	Initialization of On Screen Display function
0x3D	Loading of FPGA for protocol analysis
0x3F	Loading of DSP for protocol analysis

Table 3-1 LED codes during booting of equipment

4 Remote Control 4.1

4.1 Brief Instructions **4.1**

4.2 Setting the Transmission Parameters **4.1**

4.3 Device-Dependent Messages (Commands and Responses) **4.2**

4.4 Structure and Syntax of Device-Dependent Messages **4.2**

4.4.1 SCPI Introduction 4.2
4.4.2 Command Structure 4.3
4.4.3 Structure of a Program Message 4.5
4.4.4 Responses to Queries 4.6
4.4.5 Parameters 4.6
4.4.6 Overview of Syntax Elements 4.8

4.5 Description of Commands **4.9**

4.5.1 Notation 4.9
4.5.2 Common Commands 4.10
4.5.3 Setting the Operating Modes 4.12
4.5.4 Commands of MONITORING Menu 4.13
4.5.5 Commands of DECODER Menu 4.23
4.5.6 Commands of TS INPUT Menu 4.28
4.5.7 Commands of PRINT Menu 4.29
4.5.8 Commands of STORE and RECALL CONFIG Menus 4.32
4.5.9 Commands of SETUP Menu 4.33
4.5.10 Commands for 'Trigger on Error'Mode 4.37
4.5.11 Commands for 'Dump' Mode 4.39
4.5.12 Commands of SCPI Registers 4.42

4.6 Measurement Parameters **4.45**

4.6.1 Parameter Names for Monitoring 4.45
4.6.1.1 Parameter Names of 1st Priority 4.46
4.6.1.2 Parameter Names of 2nd Priority 4.47
4.6.1.3 Parameter Names of 3rd Priority 4.48
4.6.1.4 Parameter Names for Additional Settings 4.49
4.6.1.5 Error Numbers for Device-Internal Actions 4.50
4.6.2 Parameter Names for Monitoring Limits 4.52

4.7 Instrument Model and Command Processing **4.54**

4.7.1 Input Unit 4.54
4.7.2 Command Identification 4.54
4.7.3 Data Set and Device Hardware 4.55
4.7.4 Status Reporting System 4.55
4.7.5 Output Unit 4.55
4.7.6 Command Sequence and Command Synchronization 4.55

4.8 Status Reporting System **4.56**

4.8.1 Structure of an SCPI Status Register 4.56
4.8.2 Overview of Status Registers 4.58
4.8.3 Description of Status Registers 4.59
4.8.3.1 Status Byte (STB) and Service Request Enable Register (SRE) 4.59
4.8.3.2 Event Status Register (ESR) and Event Status Enable Register (ESE) 4.60
4.8.3.3 STATus:OPERation Register 4.61
4.8.3.4 STATus:QUEStionable Register 4.61
4.8.4 STATus:QUEStionable:DVRM Register 4.62
4.8.5 Use of the Status Reporting System 4.63
4.8.5.1 Service Request, Use of Hierarchical Structure 4.63
4.8.5.2 Queries 4.63
4.8.5.3 Error Queue Query 4.64
4.8.6 Resetting the Status Reporting System 4.64

4.9 Set of Commands **4.65**

4.10 Error Messages of Remote-Control Interface 4.70
4.10.1 SCPI-Specific Error Messages 4.70
4.11 Programming Instructions 4.74
4.12 Programming Example 4.74
4.12.1 General 4.74
4.12.2 Definitions of Library Calls Supporting the V24 Interface 4.74
4.12.3 Examples 4.75
4.12.3.1 Example 1 .... 4.75
4.12.3.2 Example 2 .... 4.77

4 Remote Control

Introduction

DVRM is fitted with an RS-232-C interface as standard. The connector is located at the rear of the instrument and permits an external controller to be connected. The instrument supports the SCPI version 1995.0 (Standard Commands for Programmable Instruments). The SCPI standard is based on the IEEE 488.2 standard and aims at a standardization of device-specific commands, error handling and status registers (see section 4.4.1 SCPI Introduction).

Basic knowledge of SCPI programming and operation of the controller is required for a clear understanding of this chapter.

The requirements of the SCPI standard regarding command syntax, error handling and configuration of status registers are described in detail in the relevant sections. Tables provide a quick overview of the commands implemented in the instrument and the assignment of the bits in the status registers. The tables are supplemented by a detailed description of commands and status registers. To understand the description of the commands a basic knowledge of DVRM manual operation is required.

All program examples for control via RS-232 are written in C for the program package V24.Tools Plus from Langner.

4.1 Brief Instructions

The following brief and simple operating instructions allow the user to put the instrument quickly into operation and to set the basic functions.

- Connect the DVRM to the controller via a modem bypass cable.
- Set DVRM (menu item SETUP/RS232) to 9600 bit/s, 8 bits, parity NONE, 1 stop bit and pace NONE.
- Start a terminal program on the controller and set it to the same protocol as the DVRM.
- Enter the following command sequence:
  `DISP:OSD:MODE ON↵`
  `SYST:DISP:CONT STAT↵`
  `CONF:MON:CONT CLEA↵`
  `CONF:MON:CONT START↵`
  The MONITORING REPORT is cleared and monitoring restarted. The OSD (On Screen Display) shows the MONITORING/STATISTICS menu.

4.2 Setting the Transmission Parameters

To ensure error-free and correct data transmission, the same transmission parameters must be set on the DVRM and the controller. The parameters can be varied either manually in the SETUP/RS232 menu or via remote control using the command `SYSTem:COMMunicate:SERial: to .`

The transmission parameters of the interface are factory-set as follows: baud rate = 9600, bits = 8, stop bits = 1, parity = NONE and pace = NONE.

4.3 Device-Dependent Messages (Commands and Responses)

Device-dependent messages are transmitted on the data lines of the remote-control interface, the ASCII/ISO code being used. Device-dependent messages are differentiated according to the direction in which they are sent via the interface:

Commands are messages sent by the controller to the device. They control the device functions and request information. The commands are classified as follows:

1. According to the effect they have on the device:

   Setting commands trigger device settings, eg resetting of the instrument.

   Queries cause data to be provided for output via the IEC/IEEE bus, eg for device identification or query of the active input.

2. According to their definition as per IEEE 488.2 standard:

   Common commands are precisely defined in the IEEE 488.2 standard as regards their function and notation. They concern functions as, for instance, the management of the standardized status registers, resetting and selftest.

   Device-specific commands relate to functions that depend on device characteristics such as frequency setting. Part of these commands is standardized by the SCPI Consortium (see section 4.4.1).

Responses are messages sent by the device to the controller following a query. They may contain results, device settings or information about the device status (see section 4.4.4).

Structure and syntax of the device-dependent messages are described in section 4.4. Commands are listed and explained in section 4.5. The description of the commands requires basic knowledge of the manual operation of DVRM.

4.4 Structure and Syntax of Device-Dependent Messages

4.4.1 SCPI Introduction

SCPI (Standard Commands for Programmable Instruments) describes a standardized command set for the programming of instruments regardless of the type of instrument or manufacturer. The objective of the SCPI Consortium is to standardize device-specific commands to a large extent. For this purpose an instrument model has been developed which defines identical functions within an instrument or of different instruments. Command systems have been generated and assigned to these functions so that it is possible to address identical functions by the same commands. The command systems have a hierarchical structure. Fig. 4-1 shows this tree structure, using a detail from the SYSTem command system for controlling parts of the instrument setup. The other examples of command syntax and structure are taken from this command system.

SCPI is based on the IEEE 488.2 standard, ie it uses the same syntax elements as well as the "common commands" defined in IEEE 488.2. The syntax of the responses is partly subjected to stricter rules than laid down in IEEE 488.2 (see section 4.4.4).

4.4.2 Command Structure

The commands consist of a header and usually one or several parameters. Header and parameters are separated by a "white space" (ASCII code 0 to 9, 11 to 32 decimal, eg space). The headers may be composed of several keywords. The query form is generated by appending a question mark directly to the header.

Note: *The commands used in the examples below are not necessarily implemented in the instrument.*

Common Commands

Common commands consist of an a header preceded by an asterisk "*". The header may be followed by one or several parameters.

Examples:

`*RST`	RESET, resets the device
`*ESE 235`	EVENT STATUS ENABLE, sets the bits of the Event Status Enable Register
`*ESR?`	EVENT STATUS QUERY, queries the contents of the Event Status Register.

Device-specific commands

Hierarchy: Device-specific commands have a hierarchical structure (see Fig. 4-1). The various levels are represented by compound headers. Headers of the highest level (root level) have one keyword only. This keyword stands for a whole command system.

Example: `SYSTem` This keyword denotes the command system `SYSTem`.

For lower-level commands the full path has to be specified, starting with the highest level in the left-most position. The individual keywords are separated by a colon ":".

Example: `SYSTem:COMMunicate:SERial:RECeive:BAUD 9600`

This command is at the fifth level of the system SYSTem and sets the baud rate to 9600.

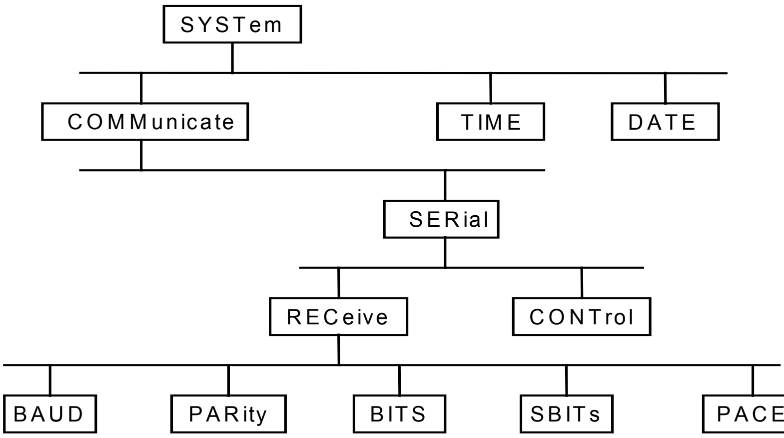


Fig. 4-1 Tree structure of SCPI command systems, SYSTem being shown as an example

Long and short form: The keywords have a long and a short form. They may be entered in the short or the long form, other abbreviations are not allowed.

Example: `SYSTem:COMMunicate:SERial:BAUD 9600`

The following command has the same effect.

`SYST:COMM:SER:BAUD 9600`

Note: *The short form uses upper case characters, the long form gives the whole keyword. Upper case and lower case letters are only used for identification in the instrument manual, the instrument itself does not differentiate between upper case and lower case.*

Parameter: The parameter must be separated from the header by a "white space". If a command contains several parameters, these have to be separated by a comma ",". Some of the queries allow the specification of the parameters MINimum, MAXimum and DEFault. For a description of the various types of parameter see section 4.4.4.

Example: `SYSTem:COMM:SER:REC:BAUD? MAXimum` Response: `115200`

This query returns the maximum value for the baud rate.

Numeric suffix: If an instrument has several identical functions or features, eg inputs, the desired function can be selected by a suffix to the command. Commands given without suffix are interpreted as having suffix 1.

Example: `SYSTem:COMMunicate:SERial2:BAUD 4800`

This command sets the baud rate of the serial interface SERial2 to 4800 baud.

Optional keywords: In some command systems it is possible to insert or to omit certain keywords in the header. These keywords are shown in the manual in brackets. For reasons of compatibility with the SCPI standard, the instrument must be able to recognize the full command length. Some of the commands become considerably shorter when the optional keywords are omitted.

Example: `SYSTem:COMMunicate:SERial[:RECeive]:BAUD 4800`

This command sets the baud rate of the SERial interface to 4800 baud. The following command has the same effect.

`SYSTem:COMMunicate:SERial:BAUD 4800`

Note: *An optional keywor not be omitted if its effect is specified in more detail by a numeric suffix.*

4.4.3 Structure of a Program Message

A pram message may contain one oreveral commands. It is terminated by a <Carriage Return><New Line> via the RS-232 interface.

Several commands in a pram message are separated by a semicolon ";". If the next command belongs to a different command system, the semicolon is followed by a colon.

Example: `SYSTem:TIME 20,30,00;:SENSe:PROGram 10`

This pram message contains two commands. The first command belongs to the SYSTem system and is used to set the time of the system clock. The second command belongs to the SENSe system, and selects the pram to be decoded.

If the successive commands belong to the same system and therefore have one orseveral common levels, the pram message may be shorhe second command following the semicolon then starts at the level that is below the common levels (see also Fig. 4-1). The colon after the semicolon has to be omitted in this case.

Example: `SYSTem:COMM:SERial:BAUD 4800;:SYSTem:COMM:SERial:BITS 8`

This pram message contains two commands separated by a semicolon. Both commands belong to the SYSTem system and its `COMMunicate:SERial` subsystem, ie they have three common levels.
In the shoram message the second command startsat the level below `SYSTem:COMMunicate:SERial`. The colon after the semicolon has to be omitted.

The shoror of the pram message is:

`SYST:COMM:SER:BAUD 4800;BITS 8`

A new pram message always starts with the full path however.

Example:
```
SYSTem:COMMunicate:SERial:BAUD 4800
SYSTem:COMMunicate:SERial:BITS 8
```

4.4.4 Responses to Queries

Unless otherwise expressly specified, a query is defined for each setting command. The query is generated by appending a question mark to the associated command. The SCPI rules valid for the query responses are somewhat stricter than those of the IEEE 488.2 standard:

1. The required parameter is sent without header.
   Example: `SYSTem:COMMunicate:SERial:BAUD?` Response: `4800`
2. Maximum and minimum values can be queried by special text parameters.
   Example: `SYSTem:COMMunicate:SERial:BAUD? MINimum` Response: `1200`
3. Numeric values are output without unit. Physical quantities refer to the basic units.
   Example: `READ:TS:BITRate?` Response: `38E6` for `38 MBit/s`
4. Boolean values are returned as 0 (for OFF) and 1 (for ON).
   Example: `ROUTe:AUDio:RIGHt?` Response: `1`
5. Character data used for a parameter are returned in short form (see also section 4.4.5).
   Example: `ROUTe:VIDeo?` Response: `PALN`

4.4.5 Parameters

Most commands require the specification of a parameter. The parameters must be separated from the header by a "white space". Parameters may be specified as numeric values, Boolean parameters, character data, character strings and block data. The type of parameter required for the specific command as well as the permissible range of values are described together with the commands (see section 4.5).

Numeric values Numeric values may be entered in any customary form, ie with sign, decimal point and exponent. If the values exceed the resolution of the instrument, they will be rounded off. The mantissa may comprise up to 255 characters, the exponent must be in the range between -32 000 and 32 000. The exponent is denoted by an "E" or "e". The exponent alone must not be used. Physical quantities may be stated with the unit. Permissible prefixes for the unit are G (giga), MA (mega, MOHM and MHZ are also allowed), K (kilo), M (milli), U (micro) and N (nano). If no unit is specified, the basic unit will be used.

Example: `SOURce:FREQuency 1.5 kHz`
= `SOURce:FREQuency 1.5E3`

Special numeric values

The parameters MINimum, MAXimum, and DEFault are interpreted as special numeric values.

Upon a query, the numeric value will be returned.

Example: Setting command: `SYST:COMM:SER:BAUD MAX`
Query: `SYST:COMM:SER:BAUD?` Response: `115200`

MIN/MAX — MINimum and MAXimum denote the minimum and maximum value.

DEF — DEFault denotes a preset value stored in the EPROM. This value coincides with the basic setting called up by the *RST command.

NAN — Not A Number (NAN) is represented as 9.91E37. NAN is only sent as a response. This value is not defined. NAN is typically returned when dividing by zero, subtracting infinity from infinity and representing missing data.

Boolean parameters

Boolean parameters represent two states. The ON state (true condition) is represented by ON or a numeric value other than 0. The OFF state (false condition) is represented by OFF or the numeric value 0. Queries provide 0 or 1.

Example: Setting command: `ROUTe:AUDio:RIGHt ON`
Query: `ROUTe:AUDio:RIGHt?` Response: `1`

Character data

Character data follow the syntax rules for keywords, ie they also have a short and a long form. Like any other parameters, they must be separated from the header by a 'white space'. A query returns the short form of the character data.

Example: Setting command: `ROUTe:VIDeo PALNtsc`
Query: `ROUTe:VIDeo?` Response: `PALN`

Character strings

Strings must always be given in single or double quotes.

Example: `SYSTem:LANGuage "SCPI"` or
`SYSTem:LANGuage 'SCPI'`

Block data

The block data format is suitable for the transmission of large data volumes. A command with a block data parameter has the following structure:
Example: `HEADer:HEADer #45168xxxxxxxx`

The ASCII character # denotes the beginning of the data block. The next numeral specifies the number of subsequent digits defining the length of the data block. In the example above, the four digits specify a length of 5168 bytes. The 5168 data bytes follow next. During the transmission of these data bytes all terminators and other control characters are ignored until all 5168 bytes have been transmitted.
If block data are returned upon a query, they will always be sent in the above format (<DEFINITE LENGTH ARBITRARY RESPONSE DATA>).

In the DVRM, block data are always transmitted as a byte stream. Special formats are either fixed to the respective command or described in the data stream. Formatting of the commands is described together with the commands (see section 4.5).

Note: *The transmission of block data via the RS-232 interface is subject to certain restrictions (see Appendix A.5.4, RS-232-C interface.*

4.4.6 Overview of Syntax Elements

The following list provides an overview of the syntax elements.

':' The colon separates the keywords of a command. In a program message, the colon following a semicolon identifies the highest command level.

';' The semicolon separates two commands in a program message.

',' The comma separates several parameters of a command.

'?' The question mark forms a query.

'*' The asterisk identifies a common command.

'"' Quotation marks denote the beginning of a character string and terminate it.

'#' The number sign denotes the beginning of block data.

' ' A "white space" (ASCII code 0 to 9, 11 to 32 decimal, eg space) separates header and parameters.

4.5 Description of Commands

4.5.1 Notation

In the following sections, all commands implemented in the DVRM are tabulated according to the operating mode and the command system, and described in detail. The notation is largely in line with the SCPI standard. The SCPI conformity information is given in a table in section 4.9.

Section 4.9 lists all commands implemented in the device in tabular form.

Command table

Command: The command column gives an overview of the commands and their hierarchical relationships (see indentations).

Parameter: The parameter column states the required parameters and their range of values.

Unit: The unit column shows the basic unit of the physical parameters.

Notes: The notes column indicates
– whether the command has a query form,
– whether the command is only in the form of a query,
– whether the command is implemented in a specific instrument option only.

Indentations

The various levels of the SCPI command hierarchy are shown in the table by indentations to the right. The lower the level, the greater the indentation to the right. It should be noted that the complete notation of the command always includes the higher levels too.

The command description shows the hierarchy. This means that for each command all keywords listed above the command through to the left-hand margin have to be taken into consideration as well.

Upper/lower case

Upper case/lower case characters are used to differentiate between the long form and the short form of the keywords of a command in the command description (see section 4.4.2). The instrument itself does not differentiate between upper case and lower case.

Special characters **|** A vertical bar in the notation of the parameters is used to separate alternative options and is to be seen as "or". The effect of the command differs according to the parameter stated.

Example: Selection of parameters for the command
`SYSTem:COMM:SER:PAR NONE | EVEN | ODD`

[ ] Keywords in brackets may be omitted in compound headers (see section 4.4.2, optional keywords). For reasons of compatibility with the SCPI standard, the instrument must be able to recognize the full length of the command.
Parameters in brackets may optionally be inserted into the command or omitted.

{ } Parameters in braces may be included in the command zero, one or more times.

< > A name given in angle brackets is a summarized description of character data that have been combined in a table because of their volume.

Example: `CONFigure:MONITORING:PARameter <parameter name>`

4.5.2 Common Commands

The common commands are based on the IEEE 488.2 (IEC 625.2) standard. A specific command has the same effect in different instruments. The headers of these commands consist of an asterisk "*" followed by three letters. Many common commands refer to the status reporting system described in detail in section 4.8.

Command	Parameter	Unit	Notes
***CLS**			No query
***ESE**	0 to 255		
***ESR?**			Query only
***IDN?**			Query only
***OPC**			
***PSC**	0 | 1		
***RST**			Query only
***SRE**	0 to 255		
***STB?**			Query only
***TST?**			Query only
***WAI**			

***CLS**

CLEAR STATUS sets the status byte (STB), the Standard Event Register (ESR) and the EVENt part of the QUEStionable and of the OPERation Register to zero. The command has no effect on the enable and transition parts of the registers. It clears the output buffer.

***ESE** 0 to 255

EVENT STATUS ENABLE sets the Event Status Enable Register to the defined value. The query *ESE? returns the contents of the Event Status Enable Register in decimal form.

***ESR?**

STANDARD EVENT STATUS QUERY returns the contents of the Event Status Register in decimal form (0 to 255) and then clears the register.

***IDN?**

IDENTIFICATION QUERY for identification of the instrument.
The response is for example: "Rohde&Schwarz, DVRM,0,2.00"

0 = serial number
2.00 = firmware version number

***OPC**

OPERATION COMPLETE sets bit 0 in the Event Status Register if all preceding commands have been executed. This bit may be used to trigger a service request (see section 4.6).

***OPC?**

OPERATION COMPLETE QUERY writes the message "1" into the output buffer as soon as all preceding commands have been executed (see section 4.6).

***PSC** 0 | 1

POWER ON STATUS CLEAR determines whether the contents of the ENABle Registers is retained or cleared upon power-up.

*PSC = 0 causes the status registers to retain their contents. With an adequate configuration of the ESE and SRE status registers, a service request may be triggered upon power-up.

*PSC $\neq$ 0 clears the registers.

The query *PSC? reads out the contents of the power-on-status-clear flags. The response may be 0 or 1.

***RST**

RESET sets the device to a defined default state. The default setting is given together with the description of the commands.

***SRE** 0 to 255

SERVICE REQUEST ENABLE sets the Service Request Enable Register to the defined value. This command determines the conditions under which a service request will be triggered. The query *SRE? outputs the contents of the Service Request Enable Register in decimal form. The value of the unused bit 6 (MSS bit) is always 0.

***STB?**

READ STATUS BYTE QUERY outputs the contents of the status byte in decimal form.

***TST?**

SELF TEST QUERY triggers all selftests of the device and outputs an error code in decimal form.

***WAI**

WAIT-to-CONTINUE allows processing of commands only after all preceding commands have been executed and all signals are settled (see also section 7.6 and "*OPC").

4.5.3 Setting the Operating Modes

In the remote-control mode the MPEG2 Realtime Monitor DVRM features three additional operating modes.

Command	Parameter	Unit	Notes
SENSe **:FUNCtion** **[:ON][?]**	MONitoring | TRERror | DUMP | MEASure		Setting the operating mode

SENSe
:FUNCtion
[:ON][?] MONitoring | TRERror | DUMP | MEASure *RST value: MONitoring

Selects one of the following four operating modes of the decoder:

MONitoring Activates the monitoring mode. An applied transport stream is monitored.

TRERror Activates the 'Trigger on Error' function (see section 4.5.10). If the error condition set with `CONFigure:TRERror:TRIGger` has occurred, a maximum of 800 transport stream packets from the transport stream region where the error has occurred can be read with `READ:TRERror?`.

DUMP Activates the 'Dump' function (see section 4.5.11). If the filter condition set with `CONFigure:DUMP:STATe` and `to :TRIGger:PID` has occurred, a maximum of 1394 transport stream packets can be read with `READ:DUMP?`.

MEASure The 'Measure' function is only available via the Stream Explorer option B1.

Example: *SENS:FUNC MON*

Note:

The TRERror, DUMP and MEASure modes can only be activated by remote control. Upon switchover from MONitoring to TRERror, DUMP or MEASure, monitoring of the transport stream is stopped (corresponds to command CONF:MON:CONT STOP). When a switchover is performed from remote to manual control (see section 4.1) the MONitoring mode is automatically selected. If monitoring was active before the switchover to TRERror, DUMP or MEASure, monitoring will be continued when the MONitoring mode is selected again.

On power-up DVRM is always in the MONitoring mode.

4.5.4 Commands of MONITORING Menu

The MONITORING mode is activated by means of the `SENSe:FUNCtion:MONitoring` command.

Command	Parameter/Response	Unit	Notes
READ **[:SCALar]** **:MONitoring?**	<parameter name> / Year, month, day, hour, minute, second, parameter status		Reads the current parameter status, -1 = parameter not active 0 = OK 1 = error
:ALL?	- / Year, month, day, hour, minute, second, parameter status 1, to , parameter status 19		Reads the parameter states of the three priorities -1 = parameter not active 0 = OK 1 = error
	EXTended / Year, month, day, hour, minute, second, parameter status 1, to , parameter status 25		Reads all parameter states

Command	Parameter/Response	Unit	Notes
READ **[:SCALar]** **:MONitoring** **:REPort?**	- / 0 | 1, year, month, day, hour, minute, second <error number>, PID, detail | | -1, year, month, day, hour, minute, second, <error number>, PID, detail		Reads one line of the test report; the read cursor is incremented, 0 = read cursor == write cursor 1 = line in test report -1 = entries overwritten
:LINE?	0 to 999 / 0 | 1, year, month, day, hour, minute, second <error number>, PID, detail | -1, year, month, day, hour, minute, second <error number>, PID, detail		Reads one line of the test report. 0 = last line 1 = last but one line, etc Response: 0 = line not in report 1 = line in test report - -1 = entries overwritten
:MOMent?	<parameter name> 1980 to 2079, 1 to 12, 1 to 31, 0 to 23, 0 to 59, 0 to 59 / 0 | 1, <error number>, PID {, to } | -1, <error number>, PID {, to }		Reads an entry in the test report for a specific parameter at a specific time. Response: 0 = no entry in test report 1 = line in test report -1 = entries overwritten
:EXTMoment?	<parameter name>, 1980 to 2079, 1 to 12, 1 to 31, 0 to 23, 0 to 59, 0 to 59 / 0 | 1, <error number> , PID, detail {, to } | -1, <error number> , PID, detail {, to }		Reads an extended entry in the test report for a specific parameter at a specific time. Response: 0 = no entry in test report 1 = line in test report -1 = entries overwritten
:DURation?	0, day, hours, minutes, seconds | 1, day, hours, minutes, seconds		Evaluated period 0 = measurement stopped 1 = measurement on
:PROGram **:COUNt?**	- / noProgram, noPMT		Reads maximum possible number of programs to be monitored.

Command	Parameter/Response	Unit	Notes
CONFigure **:MONitoring** **:CONTrol[?]**	STARt | STOP | CLEar		Starts and stops monitoring, clears measured values
:PARameter[?]	<parameter name>, ON | OFF		Includes specific parameters in or excludes them from monitoring, see section 4.6.1
:ALL	ON | OFF		Includes all parameters in the monitoring or clears all, no query
:LIMit **:UPPer[?]**	<parameter name>, 100 to 60000 ms	S	Sets upper limit value, see section 4.6.2
:LOWer[?]	<parameter name>, 1 to 100 ms	S	Sets lower limit value, see section 4.6.2
:MIPTiming[?]	100 to 99900 ns	ns	Sets upper limit value
:PIDR[?]	ALLPid | VAONly		One Limit for al PID or for Video/Audio extra
:PDOR[?]	ON | OFF		Limit for PID with typ Data/Other on/off
:PROGram **:MODE[?]**	AUTO | MANual		Selection of monitoring mode
:SELect[?]	1 to 65535, ON | OFF		Selection of programs to be monitored
:BITRate **:LIMit** **:UPPer[?]**	NULL, 1000 to 54000000 bit/s	bit/s	Sets upper limit value
:LOWer[?]	NULL, 1000 to 54000000 bit/s	bit/s	Sets lower limit value
:MULTiplex **:LIMit** **:UPPer[?]**	TSID, 0 to 65535		Sets highest permissible transport stream ID
:LOWer[?]	TSID, 0 to 65535		Sets lowest permissible TS-ID
:ALARmlines[?]	<parameter name>, 0 to 12 | STOP, 0 to 12		Assigns an alarm line to a specific parameter
:ALL	0 to 12		Assigns alarm lines to all parameters

READ
[:SCALar]
:MONitoring? <parameter name>

This command queries the current status of a parameter in the MONitoring mode. In all other modes NAN is returned.

Parameter name: see section 4.6.1.

Response: year, month, day,
hour, minute, second,
parameter status

Parameter status : -1 parameter not active
0 OK, no error occurred
1 an error occurred

Example: *READ:MON? CCNT* Response: *1996,05,06,20,15,00,1*

:ALL?

Queries the current status of parameters of the three priorities in the MONitoring mode. In all other modes NAN is returned.

Response: year, month, day,
hour, minute, second,
parameter status of

TSSL,	SBE,	PATE,	CCOE,	PMTE,	PIDE,	
TPEE,	CRCE,	PCRE,	PCRA,	PTSE,	CATE,	
NITE,	SIRE,	PIDU,	SDTE,	EITE,	RSTE,	TDTE

for operating mode DVB
or
parameter status of

TSSL,	SBE,	PATE,	CCOE,	PMTE,	PIDE,
TPEE,	CRCE,	PCRE,	PCRA,	PTSE,	CATE,
SIRE,	PIDU,	BASE,	PDGM,		
BITRate,	MULTiplex				

for operating mode ATSC

Parameter status : -1 parameter not active
0 OK, no error occurred
1 an error occurred

Example: *READ:MON:ALL?* Response: *1996,05,06,20,15,00,1,0,0 to ,0*

:ALL? EXTended

Queries the current status of all parameters in the MONitoring mode. In all other modes NAN is returned.

Response: year, month, day,
hour, minute, second,
parameter status of

TSSL,	SBE,	PATE,	CCOE,	PMTE,	PIDE,	
TPEE,	CRCE,	PCRE,	PCRA,	PTSE,	CATE,	
NITE,	SIRE,	PIDU,	SDTE,	EITE,	RSTE,	TDTE
SIOR,	NITO,	SDTO,	EITO,			
BITRate,	MULTplex		MIPE			

for operating mode DVB
or
parameter status of

TSSL,	SBE,	PATE,	CCOE,	PMTE,	PIDE,
TPEE,	CRCE,	PCRE,	PCRA,	PTSE,	CATE,
SIRE,	PIDU,	BASE,	PDGM,		
BITRate,	MULTiplex				

for operating mode ATSC

Parameter status : -1 parameter not active
0 OK, no error occurred
1 an error has occurred

Example: *READ:MON:ALL? EXT* Antwort: *1996,05,06,20,15,00,1,0,0 to ,0*

READ
[:SCALar]
:MONitoring?
:ERRSeconds? <parameter name>

Queries the error seconds of a parameter.
Parameter name: see section 4.6.1
Response: year, month, day,
hour, minute, second,
error seconds

Error seconds :
-1 parameter not active
0 OK, no error occurred
1 to 999 number of error seconds

Example: *READ:MON:ERRS? CCNT* Response: *1996,05,06,20,15,00,17*

:ALL?

Queries the error seconds of parameters of the three priorities.
Response: year month day,
hour, minute, second,
error seconds of
TSSL, SBE, PATE, CCOE, PMTE, PIDE,
TPEE, CRCE, PCRE, PCRA, PTSE, CATE,
NITE, SIRE, PIDU, SDTE, EITE, RSTE, TDTE
for operating mode DVB
or
error seconds of
TSSL, SBE, PATE, CCOE, PMTE, PIDE,
TPEE, CRCE, PCRE, PCRA, PTSE, CATE,
SIRE, PIDU, BASE, PDGM,
BITRate, MULTiplex
for operating mode ATSC

Error seconds :
-1 parameter not active
0 OK, no error occurred
1 to 999 number of error seconds

Example: READ:MON:ERRS:ALL? Response: 1996,05,06,20,15,00,17,3,0, to ,0,11

:ALL? EXTended

Queries the error seconds of all parameters.
Response: year month day,
hour, minute, second,
error seconds of
TSSL, SBE, PATE, CCOE, PMTE, PIDE,
TPEE, CRCE, PCRE, PCRA, PTSE, CATE,
NITE, SIRE, PIDU, SDTE, EITE, RSTE, TDTE
SIOR, NITO, SDTO, EITO
BITRate, MULTplex, MIPE
for operating mode DVB
or
error seconds of
TSSL, SBE, PATE, CCOE, PMTE, PIDE,
TPEE, CRCE, PCRE, PCRA, PTSE, CATE,
SIRE, PIDU, BASE, PDGM,
BITRate, MULTiplex
for operating mode ATSC

Error seconds :
-1 parameter not active
0 OK, no error occurred
1 to 999 number of error seconds

Example: READ:MON:ERRS:ALL? EXT Response: 1996,05,06,20,15,00,17,3,0, to ,0,11

READ
[:SCALar]
:MONitoring
:REPort?

Reads an entry in the MONITORING/REPORT and sets the corresponding read cursor to the next entry in the report.

Response: Report status,
year, month, day,
hour, minute, second,
<error number>, PID, detail

Report status :	0	No new entry in the report. Read cursor == write cursor. Only the report status is output.
	1	There are entries in the report.
	-1	Entries in the report not read with `READ:MONitoring:REPort?` were overwritten.
Error number :		see section 4.6.1
PID:		PID at which the error occurred
Detail:		see section 4.6.1

Example: *READ:MON:REP?* Response: *1,1996,05,06,20,15,00,101,1250,-1000*

:LINE? 0 to 999

Reads a specific entry in the MONITORING/REPORT. The location of the entry is given with respect to the last entry.

Response: Report status,
year, month, day,
hour, minute, second,
<error number>, PID, detail

Report status :	0	The desired entry is not in the report. Only the report status is output.
	1	The desired entry is in the report.
	-1	Entries in the report not read with `READ:MONitoring:REPort?` were overwritten.
Error number :		see section 4.6.1
PID:		PID at which an error occurred
Detail:		see section 4.6.1

Example: *READ:MON:REP:LINE 5* Response: *1,1996,05,06,20,15,00,101,1250*

READ
[:SCALar]
:MONitoring
:REPort
:MOMent? <parameter name>, year, month, day, hour, minute, second

Reads the entries in the MONITORING/REPORT for the selected parameter and time.

Parameter name: see section 4.6.1

Response: Report status,
<error number>, PID {, <error number> , PID }

Report status :	0	No entry in the report for the selected parameter and time. Only the report status is output.
	1	The entry is in the report.
	-1	Entries in the report not read with `READ:MONitoring:REPort?` were overwritten.
Error number :		see section 4.6.1
PID:		PID at which an error occurred

Example: *READ:MON:REP:MOM? CCNT,1996,05,07,20,15,20* Response: *1,101,1250*

:EXTMoment? < parameter name >, year, month, day, hour, minute, second

Reads the extended entries in the MONITORING/REPORT for the selected parameter and time.

Parameter name: see section 4.6.1

Response: Report status,
< error number >, PID {, < error number > , PID, detail }

Report status :	0	No entry in the report for the selected parameter and time. Only the report status is output.
	1	The entry is in the report.
	-1	Entries in the report not read with `READ:MONitoring:REPort?` were overwritten.
Error number:		see section 4.6.1
PID:		PID at which the error occurred
Detail:		see section 4.6.1

Example: *READ:MON:REP:MOM? CCNT,1996,05,07,20,15,20*

Response:

READ
[:SCALar]
:MONitoring
:DURation?

Reads the monitoring period (value in the ELAPSED TIME field in the OSD)

Response: Monitoring status,
day, hours, minutes, seconds

Monitoring status : 0 monitoring disabled
1 monitoring active

Example: *READ:MON:DUR?* Response: *1,0,5,45,06*

:PROGram
:COUNt?

The command reads the maximum possible number of monitored programs. Two values are returned. The first value is for the number of programs, the second the maximum number of PMT-PIDs.

Example: *READ:MON:PROG:COUN?* Response*: 64,10*

CONFigure
:MONitoring
This node defines the commands for configuration and control of MPEG2 transport stream monitoring.

:CONTrol STARt | STOP | CLEAR *RST value: STARt
This command controls monitoring.
STARt starts monitoring
STOP stops monitoring
CLEAR clears error report and error seconds
Example: *CONF:MON:CONT STOP*

:PARameter <parameter name>, ON | OFF *RST value: <parameter name>, ON
Switches monitoring of the specified parameter on or off.
Parameter name: see section 4.6.1
Example: *CONF:MON:PAR CCNT,ON*

:ALL ON | OFF
Switches monitoring of all parameters on or off.
Example: *CONF:MON:PAR:ALL? ON*

:LIMit
:UPPer <parameter name>, 100 to 60000 ms
Sets the upper limit for the specified parameter.
Parameter name: see section 4.6.1
Limit values: see section 4.6.2
Example: *CONF:MON:LIM:UPP PATR, 600 ms*

:LOWer <parameter name>, 1 to 100 ms
Sets the lower limit for the specified parameter.
Parameter name: see section 4.6.1
Limit values: see section 4.6.2
Example: *CONF:MON:LIM:LOW PATR, 30 ms*

:MIPTiming 100 to 99900 ns
This command sets the upper limit for the MIP Timing parameter.
Limit values: see section 4.6.2
Example: *CONF:MON:LIM:MIPT 100 ns*

:PIDR ALLPid | VAONly
This command refers to the PID_error parameter and determines whether the same PID_distance limit is applicable to all PIDs or whether a different limit value is applicable to the PIDs with video or audio elementary streams.
Example: *CONF:MON:LIM:PIDR VAONly*

:PDOR ON|OFF
This command refers to the PID_error parameter and determines whether PIDs with elementary streams of the Data or Other type are monitored. Only effective if CONF:MON:LIM:PIDR is on VAONly.
Example: *CONF:MON:LIM:PDOR ON*

CONFigure
:MONitoring
:PROGram
:MODE AUTO | MANual *RST value: AUTO
The command determines whether the programs to be monitored are selected automatically (AUTO) or manually (MANual)
Example: *CONF:MON:PROG:MODE MAN, -*

:SELect 1 to 65535, ON | OFF
The command selects one of the programs to be monitored in the MANual mode.
If the selected program is not contained in the transport stream or the AUTO operating mode ('CONF:MON:PROG:MODE AUTO) is set, an SCPI error is generated.
Example: *CONF:MON:PROG:SEL 9001,ON*

:BITRate
:LIMit
:UPPer NULL, 1000 to 54000000 bit/s *RST value: 54000000
This command sets the upper limit for the data rate of the NULL packets.
Example: *CONF:MON:BITR:LIM:UPP NULL, 5000000*

:LOWer NULL, 1000 to 54000000 bit/s *RST value: 1000
This command sets the lower limit for the data rate of the NULL packets.
Example: *CONF:MON:BITR:LIM:LOW NULL, 15000*

:MULTiplex
:LIMit
:UPPer TSID, 0 to 65535 *RST value: 65535
Sets the upper limit value for the transport stream ID.
Example: *CONF:MON:MULT:LIM:UPP TSID, 27800*

:LOWer TSID, 0 to 65535 *RST value: 0
Sets the lower limit value for the transport stream ID.
Example: *CONF:MON:MULT:LIM:LOW TSID, 27000*

:ALARmlines <parameter name>, 0 to 12 | STOP, 0 to 12
If option B5 is fitted, the command selects the alarm line for the specified parameter.
Parameter name: see section 4.6.1
With configuration `CONFigure:MONitoring:CONTrol STOP`, the parameter STOP activates the alarm line.
Alarm line: 0 The parameter is not assigned any alarm line.
1 to 12 The parameter is assigned alarm line 1 to 12.
Example: *CONF:MON:ALARM PAT,5*

:ALL 0 to 12
Selects an alarm line for all parameters.
Alarm line: 0 No alarm line assigned
1 to 12 Alarm line 1 to 12 assigned
Example: *CONF:MON:ALARM:ALL 5*

4.5.5 Commands of DECODER Menu

Command	Parameter/response	Unit	Notes
READ [:SCALar] :TS :BITRate?	- / 0.00 to 99.999 E 6	bit/s	Queries the bit rate of the transport stream.
:ID?	- / 0 to 65535		Queries the ID of the transport stream
:PROGram :NAME?	1 to 65535 / 'name'		Queries a program name.
:BITRate?	1 to 65535 / 0.0 to 99.999E6	bit/s	Queries the bit rate of a program depending on the SYSTem:FORMat:BITRate setting
:CONDaccess?	1 to 65535 / 1 | 0		Queries whether a program is scrambled.
READ :ARRay :PROGram?	- / 1 to 65535 {, 1 to 65535}		Reads the number of the programs contained in the transport stream.
:PID?	1 to 65535 / 0 to 8191 {,0 to 8191}		Queries the PID numbers of all program elements. The first number is the PID of the PMT, the second is the PCR PID.
READ [:SCALar] :PID :TYPE?	0 to 8191 / VIDeo | AUDio | DATA | PSI | SI		Queries the type of an element.
:BITRate?	0 to 8191 / 0.0 to 99.999E6	bit/s	Queries the bit rate of an element depending on the SYSTem:FORMat:BITRate setting
:CONDaccess?	0 to 8191 / 1 | 0		Queries whether an element is scrambled.

Command	Parameter/response	Unit	Notes
READ :ARRay :NIT?			Reads the information from the Network Information Table (NIT)
	- / net_id, ‘networkname’, table_id, {, net_id, ‘networkname’, table_id }		Reads the list of network IDs
	net_id / ts_id, org_net_id, system_type, frequency {, ts_id, org_net_id, system_type, frequency }		Reads the list of transport streams present in the specified network
READ :ARRay :SDT?			Reads information from the Service Description Table (SDT)
	- / ts_id, ‘networkname’, org_net_id, table_id, {, ts_id, ‘networkname’, org_net_id, table_id }		Reads the list of transport stream IDs
	ts_id, org_net_id / service_id, ‘servicename’, ‘serviceprovidername’ {, service_id, ‘servicename’, ‘serviceprovidername’}		Reads the list of programs/services present in the specified transport stream.
READ :ARRay :EIT?			Reads information from the Event Information Table (EIT)
	- / service_id, ‘servicename’, ts_id, org_net_id, table_id, {, service_id, ‘servicename’, ts_id, org_net_id, table_id }		Reads the list of program/service IDs
	service_id, ts_id, org_net_id / event_id, section_nr, start_date, start_time, duration, ‘eventname’, ‘eventdescription’ {, event_id, section_nr, start_date, start_time, duration, ‘eventname’, ‘eventdescription’ }		Reads the events of the specified program/service.

READ
[:SCALar]
:TS
:BITRate?
Queries the bit rate of the transport stream in the MONitoring mode. In all other modes NAN is returned.
Example: *READ:TS:BITR?* Response: *38.014e6*
The response is 38.014 Mbit/s

:ID?
Queries the ID of the transport stream.
Beispiel: *READ:TS:ID?* Antwort: *27020*

:PROGram
:NAME? 1 to 65535
Queries the program name.
Example: *READ:PROG:NAME? 1* Response: *ARD*

:BITRate? 1 to 65535
Queries the bit rate of a program in the MONitoring mode. The value returned (total/net) depends on the `SYSTem:FORMat:BITRate` setting. In all other modes NAN is returned.
Example: *READ:PROG:BITR? 1* Response: *5.3e6*
The response is 5.3 Mbit/s

:CONDaccess? 1 to 65535
Queries whether the program is scrambled.
Example: *READ:PROG:COND? 1* Response: *0*

READ
:ARRay
:PROGram?
Queries the program numbers contained in the transport stream.
Example: *READ:ARR:PROG?* Response: *1,2,3,5,7*

:PID? 1 to 65535
Queries the PIDs of the program elements. The first PID identifies the PMT to which the program is referred, the second the stream containing the PCR values. The following PIDs identify the program elements.
Example: *READ:ARR:PROG:PID?* Response: *90,100,101,102*

READ
[:SCALar]
:PID
If the PID is not contained in the transport stream or if operating mode TRERror, DUMP or MEASure is set, NAN is returned.

TYPE? 0 to 8191
Queries the type of PID.

xx, VIDeo	PID contains video stream of type xx (ISO/IEC 13818-1)
xx, AUDio	PID contains audio stream of type xx (ISO/IEC 13818-1)
xx, DATA	PID contains teletext of type xx (ISO/IEC 13818-1)
xx, OTHer	PID contains data of type xx (ISO/IEC 13818-1)
-1, PSI	PID contains PSI tables
-1, .SI	PID contains SI tables
-1, UNDef	PID contains unknown data
-1, NULL	PID contains null packets

Example: *READ:PID:TYPE? 100* Response: 2.*VIDEo*

:BITRate? 0 to 8191
Queries the bit rate of an element in the MONitoring mode. The value returned (total/net) depends on the `SYSTem:FORMat:BITRate` setting.
Example: *READ:PID:BITR? 100* Response: *4.0e6*
The response is 4.0 Mbit/s.

:CONDaccess? 0 to 8191
Queries whether the element is scrambled.
Example: *READ:PID:COND? 100* Response: *1*

READ
:ARRay
:NIT?
Queries the list of network IDs.
'SCPI error' is returned in the ATSC mode.
Response: net_id, ‘networkname’, table_id
{, net_id, ‘networkname’, table_id }

net_id :	0 to 65535	network ID
networkname:		network name
table_id:	64	table describing the current network
	65	table describing another network

Example: *READ:ARR:NIT?* Response: *1,’ASTRA’,64*

:NIT? net_id
Queries the list of transport streams in the specified network.
'SCPI error' is returned in the ATSC mode.
Response: ts_id, org_net_id, system_type, frequency
{, ts_id, org_net_id, system_type, frequency }

ts_id :	0 to 65535	transport stream ID
org_net_id:	0 to 65535	original network ID
system_type:	0	transmission system not known
	1	satellite
	2	cable
	3	terrestrial transmission

frequency:
Example: *READ:ARR:NIT? 1* Response: *1080,1,1,12.2345E9*

READ
:ARRay
:SDT?

Queries the list of transport stream IDs.
'SCPI error' is returned in the ATSC mode.

Response: ts_id, ‘networkname’, org_net_id, table_id
{, ts_id, ‘networkname’, org_net_id, table_id }

ts_id :	0 to 65535	transport stream ID
networkname:		network name
org_net_id:	0 to 65535	original network ID
table_id:	66	table describing the current transport stream
	70	table describing another transport stream

Example: *READ:ARR:SDT?* Response: *27020,’ASTRA’,1,66*

:SDT? ts_id, org_net_id

Queries the list of programs/services present in the specified transport stream.
'SCPI error' is returned in the ATSC mode.

Response: service_id, ‘servicename’, ‘serviceprovidername’
{, service_id, ‘servicename’, ‘serviceprovidername’ }

service_id :	1 to 65535	program number/service ID
servicename:		name of program/service
serviceprovidername:		name of program/service provider

Example: *READ:ARR:SDT? 1080,1* Response: *27020,’ARD Muxx’,‘ARD’*

:EIT?

Queries the list of program/service IDs.
'SCPI error' is returned in the ATSC mode.

Response: service_id, ‘servicename’, ts_id, org_net_id, table_id
{, ts_id, ‘networkname’, org_net_id, table_id }

service_id :	1 to 65535	program number/service ID
servicename:		name of program/service
ts_id :	0 to 65535	transport stream ID
org_net_id:	0 to 65535	original network ID
table_id:	78	table describing the current transport stream
	79	table describing another transport stream

Example: *READ:ARR:EIT?* Response: *27020,’ARD Muxx’,1080,1,78*

:EIT? service_id, ts_id, org_net_id

Queries the list of events of the specified program/service.
'SCPI error' is returned in the ATSC mode.

Response: event_id, section_nr, start_date, start_time, duration,
‘eventname’, ‘eventdescription’
{, event_id, section_nr, start_date, start_time, duration,
‘eventname’, ‘eventdescription’ }

event_id :	0 to 65535	event ID
section_nr:	0	present event
	1	following event
start_date:		aear, month, day
start_time:		hours, minutes, seconds
duration:		hours, minutes, seconds
eventname:		name of event
eventdescription:		description of event

Example: *READ:ARR:EIT? 27020,1080,1*
Response: *12345,0,20,1998,12,15,20,00,00,00,14,40,’News’,‘with weatherforcast’*

4.5.6 Commands of TS INPUT Menu

In the TS INPUT menu, the input is selected and the hysteresis for synchronization of the DVRM is set.

Command	Parameter	Unit	Notes
ROUTe **:INPut[?]**	RSERial | FSERial | FPARallel		Selects signal input
SENSe **:SCONdition** **:LOCK[?]**	1 to 31		Number of consecutive valid sync words received before locking is detected.
:DROP[?]	1 to 7		Number of consecutive invalid sync words received before unlocking is detected.

ROUTe
:INPut RSERial | FSERial | FPARallel *RST value: FSERial
The command selects the signal input for the transport stream
RSERial serial, asynchronous input (T link) at the rear
FSERial serial, asynchronous input (T link) at the front
FPARallel parallel input (LVDS) at the front
Example: *ROUT:INP FSER*

SENSe
:SCONdition
This node provides the commands for setting the hysteresis for synchronization to the MPEG2 transport stream.

:LOCK 1 to 31 *RST value: 5
This command sets the number of consecutive valid sync words required before DVRM detects synchronization.
Example: *SENS:SCON:LOCK 5*

:DROP 1 to 7 *RST value: 3
Sets the number of consecutive invalid sync words required before DVRM detects a drop-out.
Example: *SENS:SCON:DROP 2*

4.5.7 Commands of PRINT Menu

Command	Parameter	Unit	Notes
HCOPy :MONitoring :DATA?			Prints the MONITORING/STATISTICS contents as block data via the RS-232 interface
:REPort :DATA?			The lines of the MONITORING/REPORT added since the last report output are printed as block data via the RS-232 interface
:LINes :DATA?	0 to 999, 0 to 999		Prints line n to line m of the MONITORING/REPORT as block data via the RS-232 interface
:SETTings :DATA?			Prints the current instrument settings as block data via the RS-232 interface
:PROGram :DATA?			Prints the contents of the applied transport stream as block data via the RS-232 interface
HCOPy :MONitoring [:IMMediate]			Outputs the contents of MONITORING/STATISTICS via the parallel printer interface (option B5)
:REPort [:IMMediate]			The lines of the MONITORING/REPORT added since the last report printout are output via the parallel printer interface (option B5)
:REPort LINes [:IMMediate]	0 to 999, 0 to 999		Outputs line n to line m of the MONITORING/REPORT via the parallel printer interface (option B5)
:SETTings [:IMMediate]			Outputs the current instrument settings via the parallel printer interface (option B5)
:PROGram [:IMMediate]			Outputs the contents of the applied transport stream via the parallel printer interface (option B5)
:ABORt			Aborts printing (option B5)
:STATe?	- \ IDLE | RUNN | ERR		Queries printer status (option B5)

HCOPy

This node provides the commands for printing data as they are sent to the printer in manual operation using the functions of the PRINT menu. When the serial interface is used, the data are output as block data of defined length. After removing the block data header and the terminator, the data can be sent from the controller direct to the printer. If option B5 is fitted, the data are sent direct to the printer via the parallel interface using the appropriate commands.

:MONitoring

:DATA?

This command outputs the data of the MONITORING/STATISTICS report in the form of block data.
Example: *HCOP:MON:DATA?*
Response: *#41345MPEG2 REALTIME MONITOR V261 DVB...*

:REPort

All commands under this node output the report entries in the form of block data.

:DATA?

The data of the report in the MONITORING/REPORT menu lines added since the last report printout are output in the form of block data.
Example: *HCOP:REP:DATA?*
Response: *#512345MPEG2 REALTIME MONITOR V261 DVB...*

:LINes

:DATA? 0 to 999, 0 to 999

The data of the report for the specified range of the MONITORING/REPORT menu are output in the form of block data. The two parameters determine the first and last report entry to be output.
Example: *HCOP:REP:LIN:DATA? 15,45*
Response: *#512345MPEG2 REALTIME MONITOR V261 DVB...*

:SETTings

:DATA?

The command reads out the data of the current instrument settings in the form of block data
Example: *HCOP:SETT:DATA?*
Response: *#512345MPEG2 REALTIME MONITOR V261 DVB...*

:PROGram

:DATA?

The command outputs the data of the contents of the applied transport stream in the form of block data.
Example: *HCOP:PROG:DATA?*
Response: *#512345MPEG2 REALTIME MONITOR V261 DVB...*

HCOPy

:MONitoring

[:IMMediate]

This command outputs the data of the MONITORING/STATISTICS report via the parallel printer interface.
Example: *HCOP:MON*

:REPort

The commands under this node output the report entries.

[:IMMediate]

The data of the report in the MONITORING/REPORT menu lines added since the last report printout are output via the parallel printer interface.
Example: *HCOP:REP*

:LINes

[:IMMediate] 0 to 999, 0 to 999

The data of the report for the specified range of the MONITORING/REPORT menu are output via the parallel printer interface. The two parameters determine the first and last report entry to be output.
Example: *HCOP:REP:LINE*

:SETTings

[:IMMediate]

Outputs the current instrument settings via the parallel printer interface.
Example: *HCOP:SETT*

:PROGram

[:IMMediate]

Outputs the contents of the applied transport stream via the parallel printer interface.
Example: *HCOP:PROG*

:ABORt

This command allows printing to be aborted (option B5).
Example: *HCOP:ABOR*

:STATe?

Queries the printer status (option B5).

IDLE	Printer function not active
RUNN	Printing via the parallel printer interface
ERRR	An error has occurred during printing via the parallel printer interface

Example: *HCOP:STAT?* Response: *IDLE*

4.5.8 Commands of STORE and RECALL CONFIG Menus

Command	Parameter	Unit	Notes
MEMory :SETTings1..9 :STORe	'name' (max. 8 characters)		Stores the instrument configuration in memory locations 1 to 9 No query
:NAME?	'name'		Queries the name of the stored configuration.
MEMory :SETTings1..10 :RECall			Calls up an instrument configuration from memory locations 1 to 10. Memory location 10 contains the PRESET setting. No query

MEMory
:SETTings1..9
:STORe 'NAME'
This command stores the configuration under 'name' in memory locations 1 to 9.
Example: *MEM:SETT1:STOR 'DVRMTEST'*

:NAME?
Queries the name of a configuration.
Example: *MEM:SETT1:NAME?* Response: *DVRMTEST*

MEMory
:SETTings1..10
:RECall
Calls up the configuration from memory locations 1 to 10.
Memory location 10 contains the PRESET setting.
Example: *MEM:SETT1:REC*

4.5.9 Commands of SETUP Menu

The SETUP menu is for setting the operating parameters (serial interface, system clock, etc) for the DVRM.

Command	Parameter	Unit	Notes
SYSTem :COMMunicate :SERial [:RECeive] :BAUD[?]	1200 | 2400 | 4800 | 9600 | 19200 |38400 | 57600 | 115200		Sets the serial interface Data rate
:PARity[?] [:TYPE[?]]	EVEN | ODD | NONE		Type of parity
:BITS[?]	8		Number of data bits
:SBITs[?]	1		Number of stop bits
:PACE[?]	XON | ACK | NONE		Protocol
:PRINter :CHANnel[?]	SERial | CENTronics		Selects the printer interface for manual operation (option B5)
:DATE[?]	1980 to 2079, 1 to 12, 1 to 31		Date
:TIME[?]	0 to 23, 0 to 59, 0 to 59		Time
:ERRor?			Queries the error queue
:FORMat :PID[?]	DECimal | HEXadecimal		Sets the format for the PIDs
:BITRate[?]	TOTal | NET		Sets the format for the data rates
:IDN[?]	'kennung' (max. 8 characters)		Sets device ID
:ALARmlines :POLarity[?]	NORMal | INVerted		Sets the active state of the alarm lines
:VIEW :TABLe[?]	ON | OFF		Switches reading of tables NIT, SDT other and EIT in the REMOTE mode on and off.

SYSTem
:COMMunicate
:SERial

This node provides the commands for setting the communication parameters of the RS-232-C interface.

[:RECeive]
:BAUD 1200 | 2400 | 4800 | 9600 | 19200 | 38400 | 57600 | 115200
This command sets the baud rate.
Example: *SYST:COMM:SER:BAUD 9600*

:PARity
[:TYPE] EVEN | ODD | NONE
This command sets the parity bit.
Example: *SYST:COMM:SER:PAR ODD*

:BITS 8
This command sets the number of data bits.
Example: *SYST:COMM:SER:BITS 8*

:SBITs 1
This command sets the number of stop bits.
Example: *SYST:COMM:SER:SBIT 1*

:PACE NONE | XON | ACK
This command sets the transmission protocol.
NONE no handshake
XON software handshake
ACK hardware handshake RTS/CTS
Example: *SYST:COMM:SER:PACE NONE*

:PRINter
:CHANnel SERial | CENTronics *RST value: SER
If option B5 is fitted, this command sets the printer interface for manual operation. In case of remote control data are input as block data via the RS-232 interface using the query `HCOPy: to :DATA?` or they are output to the parallel printer interface using command `HCOPy: to [:IMMediate].`
Example: *SYST:COMM:PRIN:CHAN CENT*

:DATE 1980 to 2079, 1 to 12, 1 to 31
Sets the date of the system clock.
Example: *SYST:DATE 1996,08,01*

:TIME 0 to 23, 0 to 59, 0 to 59
Sets the time of the system clock.
Example: *SYST:TIME 10,20,00*

SYSTem

:ERRor?

Queries the oldest entry in the error queue. Positive error numbers denote device-specific errors, negative error numbers error messages specified by SCPI (see section 4.10).
Example: *SYST:ERRor?* Response: *-221, "Settings conflict"*

:VERSion?

Queries the number of the SCPI version supported by the device.
Example: *SYST:VERS?* Response: *1995.0*

:FORMat

:PID[?] DECimal | HEXadecimal *RST value: DEC

Sets the format for displaying the PIDs.

DECimal	PIDs printed as decimal numbers
HEXadecimal	PIDs printed as hexadecimal numbers

This setting only affects the output to the printer.
Example: *SYST:FORM:PID HEX*

:FORMat

:BITRate[?] TOTal | NET *RST value: TOT

Sets the format for the data rates.

TOTal	total data rate output
NET	net data rate output

This setting has an effect on the data rate to be output using `READ[:SCALar]:PROGram:BITRate?` and `READ[:SCALar]:PID:BITRate?`.
Example: *SYST:FORM:BITR NET*

:IDN[?] 'KENNUNG'

This command stores the text 'KENNUNG' for device identification.
Example: *SYST:IDN 'CHANNEL2'*

:ALARmlines

:POLarity NORMal | INVerted *RST value: NORMal

Selects the active state of the alarm lines.

NORMal	If an error occurs, the relay contact is closed and the alarm line taken to ground.
INVerted	If an error occurs, the relay contact is opened and the alarm line disconnected from ground.

Example: *SYST:ALAM:POL NORM*

:VIEW

:TABLe ON | OFF *RST value: ON

Switches the reading of tables NIT, SDT other and EIT on and off.

OFF	Tables NIT, SDT other and EIT are not read when the device is in the REMOTE mode.

Example: *SYST:VIEW:TABL OFF*

4.5.10 Commands for 'Trigger on Error'Mode

With the 'Trigger on Error' mode the faulty part of the transport stream can be read out from the internal memory via the remote-control interface if one or several of the settable trigger conditions occur (see section 4.6).The mode is switched on by means of the `SENSe:FUNCtion:TRERror` command.

Command	Parameter	Unit	Notes
CONFigure **:TRERror** **:STATe[?]**	ON | OFF		Starts the 'Trigger on Error' mode
:TRIGger[?]	<parameter name>, ON | OFF		Adds individual parameters to the trigger condition or erases them
:ALL	ON | OFF		Enters or clears all trigger condition parameters
READ **[:SCALar]** **:TRERror?**	-400 to +400		Reads a packet, 0 = packet in which the error occurred
:TRIGger			Reads out trigger condition that has occurred

CONFigure
:TRERror
:STATe[?] ON | OFF

The command activates and deactivates the trigger for the conditions set with `CONFigure:TRERror:TRIGer`. When the trigger condition occurs, bit 5 is set in the OPERation:STATus register and, if desired, a service request is initiated. In addition, the setting for `CONFigure:TRERror:STATe` is reset to `OFF`. If the Trigger on Error mode is not on, an error message will be generated.

Example: *CONF:TRER:STAT ON*

:TRIGger[?] <parameter name>, ON | OFF

This command includes the parameter in the trigger condition or clears it.

Parameter name: see section 4.6.1

Example: *CONF:TRER:TRIG CCNT, ON*

:ALL ON | OFF

This command includes all parameters in the trigger condition or clears them.

Example: *CONF:TRER:TRIG:ALL ON*

READ
[:SCALar]
:TRERror? -400 to +400

After the trigger condition has occurred, the data of one of the stored transport stream packets are output. They are returned as block data with a constant length of 188 bytes. If the trigger condition has not occurred or if the MONitoring, DUMP or MEASure mode is set, NAN is returned.

Example: *READ:TRER 5* Response: *#3188G...*

:TRIGger?

The command reads out the trigger condition that occurred. The error number of section 4.6.1 is returned. If the trigger condition has not occurred or if the MONitoring, DUMP or MEASure mode is set, NAN is returned.

Example: *READ:TRER:TRIG?* Response: *121..*

4.5.11 Commands for 'Dump' Mode

With the aid of the 'Dump' mode a selectable number of transport stream packets corresponding to a set filter condition can be stored in the unit and read out via the remote-control interface. The mode is set by means of the command `SENSe:FUNCtion DUMP`.

Command	Parameter	Unit	Notes
CONFigure **:DUMP** **:STATe[?]**	PID | AFPid| AF PUSPid| PUS APPid APUS PSI | TABid | HEADer | ALL OFF		Reads packet with PID - Packet with PID and adaptation field - Reads all packets with adaptation field - Reads packet with payload unit start indicator and PID - Reads all packets with PUS - Reads packet with PID and adaptation field and payload unit start indicator - Reads packet with adaptation field and payload unit start indicator - Reads packet with PSI table - Packets with PID comprising sections with the table_id. - Reads all packet headers - Reads all packets - Stops reading
:TRIGger **:PID[?]**	0 to 8191		Sets PID for to :STATe PID, AFPid, PUSPid and TABid
:TABid[?]	0 to 255		Sets table_id for to :STATe TABid
:COUNt[?]	1 to 1394		Number of packets to be read into the trace buffer
READ **[:SCALar]** **:DUMP?**	1 to 1394		Reads a transport packet from the trace buffer With HEADer a packet is made up of 47 headers
:COUNt?			Reads number of transport packets currently read in

CONFigure
:DUMP
:STATe[?]PID | AFPid | AF | PUSPid | PUS | APPid | APUS | PSI | TABid | HEADer | ALL | OFF

This command activates and deactivates the trigger with the specified filter condition:

PID	Only transport packets with a PID set with `CONFigure:TRIGger:PID` are read.
AFPid	Only transport packets with an adaptation field and a PID set with `CONFigure:TRIGger:PID` are read.
AF	All transport packets with an adaptation field are read.
PUSPid	Transport packets with payload unit start indicator and PID set under `CONFigure:TRIGger:PID` are read in.
PUS	All transport packets with payload unit start indicator are read in.
APPid	Only transport packets with adaptation field and payload unit start indicator and a PID set with `CONFigure:TRIGger:PID` are read in.
APUS	All transport packets with adaptation field and payload unit start indicator are read in.
PSI	All transport packets with PSI tables are read in. With PMTs maximum 10 different PIDs are read in.
TABid	Only transport stream packets with the PID set under `CONFigure:TRIGger:PID` and containing sections of the tables with the table_id set under `CONFigure:TRIGger:TABid` are read.
HEADer	The header (4 bytes) of each transport packet is read.
ALL	All transport packets are read.
OFF	The trigger is disabled or reading is interrupted.

After the number of transport packets set with `CONFigure:DUMP:COUNt` has been read into the dump memory, bit 5 is set in the OPERation:STATus register and, if desired, a service request is activated. In addition, the setting for `CONFigure:DUMP:STATe` is reset to `OFF`. If the DUMP mode is not on, an error message will be generated.

Example: CONF:DUMP:STAT PID

:TRIGger
:PID[?] 0 to 8191 *RST value: 0

Sets the PID for the filter conditions PID and AF.

Example: *CONF:DUMP:TRIG:PID 100*

:TABid[?] 0 to 255 *RST value: 0

The command sets the TAbid for the filter condition TABid.

Example: *CONF:DUMP:TAB 40*

:COUNt[?] 1 to 1394 *RST value: 1394

Sets the number of transport packets to be read, which correspond to the filter condition set with `CONFigure:DUMP:STATe`. With `CONFigure:DUMP:STATe HEADer` 47 packet headers of 4 byte each are read for each unit of `to :COUNt`

Example: *CONF:DUMP:COUN 250*

READ
[:SCALar]
:DUMP? 1 to 1394

This commands reads the data of a transport stream packet once the number of transport packets set with `CONFigure:DUMP:COUNt` is available in the dump memory. The data are output in the form of block data with a constant length of 188 bytes each. With `CONFigure:DUMP:STATe HEADer`, 47 4-byte packet headers are read per unit. If the desired packet is not in the memory, eg packet 124 should be read but the set `..:COUNt` was 120 only, or if triggering was switched off with command `CONF:DUMP:STAT OFF` before the desired number of packets could be read into the memory, NAN is returned. NAN is also returned when the MONitoring, TRERror or MEASure mode is active.

Example: *READ:DUMP 5* Response: *#3188G to*

:COUNt?

The command reads the number of currently read-in transport packets. NAN will be returned when the MONitoring, TRERror or MEASure mode is active.

Example: *READ:DUMP:COUN?* *Response: 200*

4.5.12 Commands of SCPI Registers

Commands for controlling SCPI-defined and device-specific STATus registers. The functions of the individual registers are described in detail in section 4.8.

Command	Parameter	Unit	Notes
STATus :QUEue [:NEXT]?			Queries the error/event queue
:OPERation [:EVENt]?			Query only
:CONDition?			Query only
:ENABle	0 to 32767		
:PTRansition	0 to 32767		
:NTRansition	0 to 32767		
:QUEStionable [:EVENt]?			Query only
:CONDition?			Query only
:ENABle	0 to 32767		
:PTRansition	0 to 32767		
:NTRansition	0 to 32767		
:QUEStionable :DVRM [:EVENt]?			Query only
:CONDition?			Query only
:ENABle	0 to 32767		
:PTRansition	0 to 32767		
:NTRansition	0 to 32767		

STATus
:QUEue
[:NEXT]?

This query returns the next item from the error/event queue and removes it from the queue. Positive error numbers denote device-specific errors, negative error numbers denote error messages defined by SCPI (see section 4.10). If the queue is empty, DVRM returns 0, ie "No error". If the queue has overflown, DVRM returns -350 = "Too many errors".

Example: *STAT:QUE?* Response: *0, "No Error"*

STATus

:OPERation

This node provides the commands for controlling the SCPI-STATus:OPERation register.

[:EVENt]?

Queries the contents of the EVENt register of the STATus:OPERation register. Reading the EVENt register clears it.
Example: *STAT:OPER?*

:CONDition?

This query returns the contents of the CONDition register of the STATus:OPERation register. Reading the CONDition register does not clear it.
Example: *STAT:OPER:COND?*

:ENABle 0 to 32767

Sets the bits in the ENABle register of the STATus:OPERation register. This register selects and enables the individual bits of the EVENt register for the summary bit in the status byte. The numeric value after loading the initial state (PRESET) is 0.
Example: *STAT:OPER:ENAB 32*

:PTRansition 0 to 32767

Sets the positive transition filters of the CONDition bits of the STATus:OPERation register. If a PTRansition bit is 1, the associated bit in the EVENt register is set upon transition of the CONDition bit from 0 to 1.
The numeric value after loading the initial state (PRESET) is 32767.
Example: *STAT:OPER:PTR 32*

:NTRansition 0 to 32767

Sets the negative transition filters of the CONDition bits of the STATus:OPERation register. If an NTRansition bit is 1, the associated bit in the EVENt register is set upon transition of the CONDition bit from 1 to 0.
The numeric value after loading the initial state (PRESET) is 0.
Example: *STAT:OPER:NTR 0*

:QUEStionable

This note provides the commands for controlling the SCPI-STATus:QUEStionable register.

[:EVENt]?

This query returns the contents of the EVENt register of the STATus:QUEStionable register. Reading the EVENt register clears it.
Example: *STAT:QUES?*

:CONDition?

Queries the contents of the CONDition register of the STATus:QUEStionable register. Reading the CONDition register does not clear it.
Example: *STAT:QUES:COND?*

:ENABle 0 to 32767

This command sets the bits of the ENABle registers of the STATus:QUEStionable register. This register selects and enables the individual bits of the EVENt register for the summary bit of the status byte. The numeric value after loading the initial state (PRESET) is 32767.
Example: *STAT:QUES:ENAB 128*

STATus

:QUEStionable

:PTRansition 0 to 32767

Sets the positive transition filters of the CONDition bits of the STATus:QUEStionable register. If a PTRansition bit is 1, the associated bit in the EVENt register is set upon transition of the CONDition bit from 0 to 1.
The numeric value after loading the initial state (PRESET) is 32767.
Example: *STAT:QUES:PTR 128*

:NTRansition 0 to 32767

Sets the negative transition filters of the CONDition bits of the STATus:QUEStionable register. If an NTRansition bit is 1, the associated bit in the EVENt register is set upon transition of the CONDition bit from 1 to 0.
The numeric value after loading the initial state (PRESET) is 0.
Example: *STAT:QUES:NTR 0*

:DVRM

This node provides the commands for controlling the device-specific STATus:QUEStionable:DVRM register. This register shows questionable device states in the DVRM.

[:EVENt]?

This query returns the contents of the EVENt register of the STATus:QUEStionable:DVRM register. Reading the EVENt register clears it.
Example: *STAT:QUES:DVRM?*

:CONDition?

Returns the contents of the CONDition register of the STATus:QUEStionable:DVRM register. Reading the CONDition register does not clear it.
Example: *STAT:QUES:DVRM:COND?*

:ENABle 0 to 32767

This command sets the bits in the ENABle register of the STATus:QUEStionable:DVRM register. This register selects and enables the individual bits of the EVENt register for the summary bit of the STATus:QUEStionable register.
The numeric value after loading the initial state (PRESET) is 32767.
Example: *STAT:QUES:DVRM:ENAB 128*

:PTRansition 0 to 32767

Sets the positive transition filters of the CONDition bits of the STATus:QUEStionable:DVRM register. If a PTRansition bit is 1, the associated bit in the EVENt register is set upon transition of the CONDition bit from 0 to 1.
The numeric value after loading the initial state (PRESET) is 32767.
Example: *STAT:QUES:DVRM:PTR 128*

:NTRansition 0 to 32767

Sets the negative transition filters of the CONDition bits of the STATus:QUEStionable:DVRM register. If an NTRansition bit is 1, the associated bit in the EVENt register is set upon transition of the CONDition bit from 1 to 0.
The numeric value after loading the initial state (PRESET) is 0.
Example: *STAT:QUES:DVRM:NTR 0*

4.6 Measurement Parameters

4.6.1 Parameter Names for Monitoring

The parameter names apply to the queries:

- READ:MONitoring?
- READ:MONitoring:ERRSeconds?
- CONFigure:MONitoring:PARameter[?]
- CONFigure:MONitoring:ALARmlines[?]
- CONFigure:TRERror:TRIGger[?]
  (only the parameters marked in the TRIGGER ON ERROR column)

The unit or the name for additional information is specified in the '**Detail Info**' column:

s	seconds a value > 999.999 s is indicated as 1000.000
Mbit/s	data rate in Mbit/s
TS ID	transport stream ID
S ID	service ID
MeCnt	count of extra MIP's 1 to 29999 30000 : > 29999
MHdr	MIP structur TS header 1 : PUSI != ,1' 2 : priority != ,1' 3 : AF != ,01' 4 : scrambled
MLen	MIP structur length 1 : section length 2 : individual addressing length 3 : function loop length 4 : function length
MDly	MIP max delay in s 1.0000000 to 1.6777000
MSTS	MIP synchronization time stamp in s 1.0000000 to 1.6777000
OPckt	MIP offset in TS packets –29999 to +29999 –30000 : < -29999 +30000 : > +29999 0 : no additional information available
OTime	MIP offset in s –0.4999000 to +0.4999000 –0.5000000 : < -0.4999000 +0.5000000 : > +0.4999000 0.0000000 : no additional information available
ODrate	MIP offset in MBit/s –29.999 to +29.999 –30.000 : < -29.999 +30.000 : > +29.999 0.000 : no additional information available

-1.000 is output for report entries without additional information.

The 'DVB/ATSC' column indicates whether a parameter is only valid for DVB or ATSC or for both operating modes (see command SYSTem:STANdard[?]).

4.6.1.1 Parameter Names of 1st Priority

Parameter name	Parameter	OSD Menus MONITORING	Reason	PID info	Detail info	TRIGGER ON ERROR	DVB/ATSC	Error number
TSSL	TS_sync_loss	TS SYNC	Loss			X	DVB/ATSC	100
			OK				DVB/ATSC	101
SBE	Sync_byte_error	SYNC BYTE	Single			X	DVB/ATSC	110
			Multib			X	DVB/ATSC	111
PATE	PAT_error	PAT	Upper Dist.	X	s		DVB/ATSC	120
			Tab_id	X		X	DVB/ATSC	121
			Scrambled	X		X	DVB/ATSC	122
CCOE	Continuity_count_error	CONT COUNT	Pack_order	X		X	DVB/ATSC	130
			More_than twice	X		X	DVB/ATSC	131
			Lost_packt	X		X	DVB/ATSC	132
PMTE	PMT_error	PMT	Upper Dist.	X	s		DVB/ATSC	140
			Scrambled	X		X	DVB/ATSC	141
PIDE	PID_error	PID		X		X	DVB/ATSC	150

4.6.1.2 Parameter Names of 2nd Priority

Parameter name	Parameter	OSD Menus MONITORING	Reason	PID info	Detail info	TRIGGER ON ERROR	DVB/ATSC	Error number
TPEE	Transport_error	TRANSPORT		X		X	DVB/ATSC	200
CRCE	CRC_error	CRC	PAT	X		X	DVB/ATSC	210
			PMT	X		X	DVB/ATSC	211
			CAT	X		X	DVB/ATSC	212
			NIT	X		X	only DVB	213
			EIT	X		X	only DVB	214
			BAT	X		X	only DVB	215
			SDT	X		X	only DVB	216
			TOT	X		X	only DVB	217
			MIP	X			only DVB	800
			MGT	X		X	only ATSC	700
			CVCT	X		X	only ATSC	701
			TVCT	X		X	only ATSC	702
			RRT	X		X	only ATSC	703
			STT	X		X	only ATSC	704
			EIT-0	X		X	only ATSC	705
			EIT-1	X		X	only ATSC	706
			EIT-2	X		X	only ATSC	707
			EIT-3	X		X	only ATSC	708
			CETT	X		X	only ATSC	710
			ETT-0	X		X	only ATSC	711
			ETT-1	X		X	only ATSC	712
			ETT-2	X		X	only ATSC	713
			ETT-3	X			only ATSC	714
PCRE	PCR_error	PCR	Discont	X		X	DVB/ATSC	220
			PCR_U	X	s		DVB/ATSC	221
			PCR_L	X	s		DVB/ATSC	222
PCRA	PCR_accuracy_error	PCR ACCURACY		X			DVB/ATSC	230
PTSE	PTS_error	PTS		X			DVB/ATSC	240
CATE	CAT_error	CAT	Tab_id	X		X	DVB/ATSC	250
			Scramb-ling without CAT	X		X	DVB/ATSC	251

4.6.1.3 Parameter Names of 3rd Priority

Parameter name	Parameter	OSD Menus MONITORING	Reason	PID info	Detail info	TRIGGER ON ERROR	DVB/ATSC	Error number
NITE	NIT_error	NIT	Tab_id	X		X	only DVB	300
			NIT_U	X	s		only DVB	301
SIRE	SI_repetition_error	SI REPEAT	PAT_U	X	s		DVB/ATSC	310
			PAT_L	X	s		DVB/ATSC	311
			CAT_U	X	s		DVB/ATSC	312
			CAT_L	X	s		DVB/ATSC	313
			PMT_U	X	s		DVB/ATSC	314
			PMT_L	X	s		DVB/ATSC	315
			NIT_U	X	s		only DVB	316
			NIT_L	X	s		only DVB	317
			SDT_U	X	s		only DVB	318
			SDT_L	X	s		only DVB	319
			BAT_U	X	s		only DVB	320
			BAT_L	X	s		only DVB	321
			EIT_U	X	s		only DVB	322
			EIT_L	X	s		only DVB	323
			RST_L	X	s		only DVB	324
			TDT_U	X	s		only DVB	325
			TDT_L	X	s		only DVB	326
			TOT_U	X	s		only DVB	327
			TOT_L	X	s		only DVB	328
			MGT_U	X	s		only ATSC	720
			VCT_U	X	s		only ATSC	721
			TVCT_U	X	s		only ATSC	722
			RRT_U	X	s		only ATSC	723
			STT_U	X	s		only ATSC	724
			EIT_O_U	X	s		only ATSC	725
PIDU	unreferenced_PID	UNREF PID		X		X	DVB/ATSC	340
SDTE	SDT_error	SDT	Tab_id	X		X	only DVB	350
			SDT_U	X	s		only DVB	351
EITE	EIT_error	EIT	Tab_id	X		X	only DVB	360
			EIT_U	X	s		only DVB	361
RSTE	RST_error	RST	Tab_id	X		X	only DVB	370
TDTE	TDT_error	TDT	Tab_id	X		X	only DVB	380
			TDT_U	X	s		only DVB	381
BASE	BASE_PID_error	BASE_PID	Tab_id	X		X	only ATSC	730

4.6.1.4 Parameter Names for Additional Settings

Parameter name	Parameter	OSD Menus MONITORING	Reason	PID info	Detail info	TRIGGER ON ERROR	DVB/ATSC	Error number
SIOR	SI_other repetition_error	SI OTHER REPEAT	NITOU	X	s		only DVB	600
			NITOL	X	s		only DVB	601
			SDTOU	X	s		only DVB	602
			SDTOL	X	s		only DVB	603
			EITOU	X	s		only DVB	604
			EITOL	X	s		only DVB	605
NITO	NIT_other_error	NITO	NIT_U	X	s		only DVB	610
SDTO	SDT_other_error	SDTO	SDT_U	X	s		only DVB	620
EITO	EIT_other_error	EITO	EIT_L	X	s		only DVB	630
BITRate	Datarate_error	DATARATE	NUL U	X	Mbit/s		DVB/ATSC	640
			NUL L	X	Mbit/s		DVB/ATSC	641
MULTiplex	Multiplex_error	MULTIPLEX	TSID		TS ID		DVB/ATSC	650
PDGM	PARADIGM_error	PARADIGM	PMT	X	S ID		only ATSC	740
			VIDEO	X	S ID		only ATSC	741
			PCR	X	S ID		only ATSC	742
			AUDIO	X	S ID		only ATSC	743
			DATA	X	S ID		only ATSC	744
MIP	MIP_error	MIP	PRESENT: EXTRA	X	MeCnt		only DVB	810
			PRESENT: MISSING	X			only DVB	811
			STRUCT: TS HEAD	X	MHdr		only DVB	820
			STRUCT: LENGTH	X	MLen		only DVB	821
			STRUCT: MAX DLY	X	MDel		only DVB	822
			STRUCT: STS	X	MSTS		only DVB	823
			STRUCT: CRC	X			only DVB	824
			POINTER	X	OPckt		only DVB	830
			PERIOD: POINTER	X	OPckt		only DVB	840
			PERIOD: MF SIZE	X	OPckt		only DVB	841
			TIMING	X	OTime		only DVB	850
			TS RATE	X	ODrate		only DVB	860

4.6.1.5 Error Numbers for Device-Internal Actions

In addition to the errors detected, changes of device settings affecting future report entries are also entered in the MONITORING/REPORT menu (eg monitoring of CRCE was enabled).

MONITORING\REPORT	Cause		Error number
POWER OFF	DVRM was switched off -		400
POWER ON	DVRM was switched on		401
NEW DATE	SYSTem :DATE	 1980 to 2079, 1 to 12, 1 to 31	402
NEW TIME	SYSTem :TIME	 0 to 23, 0 to 59, 0 to 59	403
START	CONFigure :MONitoring :CONTrol	 STARt	410
STOP		STOP	411
CLEAR		CLEAr	412
SERIAL REAR	ROUTe :INPut	 RSERial	420
SERIAL FRONT		FSERial	421
PARALLEL	FPARalle		422
PARAMETER GROUP	CONFigure :MONitoring :PARameter OFF :ALL	 <parameter name>, ON | ON | OFF	430
MONIT. SET PROGRAM	Number of program		440
AUTO SELECT	CONFigure :MONitoring :PROGram :MODe	 AUTO	450
MANUAL SELECT		MANual	451
MODE MONITORING	SENSe :FUNCtion	 MONitoring	460
MODE TRIGGER ON ERR		TRERror	461
MODE DUMP		DUMP	462
MODE MEASURE		MEASure	463

MONITORING\REPORT	Cause	Error number
LIMITS	CONFigure :MONitoring :LIMit :UPPer < parameter name>, 100 to 60000 ms :LOWer <parameter name>, 1 to 100 ms :MIPTiming 100 to 99900 ns :PIDR ALLPid | VAONly :PDOR ON | OFF	470
RECALL	MEMory :SETTings1 to 10 :RECall	480
DATARATE LIMITS	CONFigure :MONitoring :BITRate :LIMit :UPPer NULL, 1000 to 54000000 bit/s :LOWer NULL, 1000 to 54000000 bit/s	501
MULTIPLEX LIMITS	CONFigure :MONitoring :MULTiplex :LIMit :UPPer TSID, 0 to 65535 :LOWer TSID, 0 to 65535	510

4.6.2 Parameter Names for Monitoring Limits

The parameter names are valid for commands:

- CONFigure:MONitoring:LIMit:UPPer[?] (see column MAX)
- CONFigure:MONitoring:LIMit:LOWer[?] (see column MIN)

Parameter-name	MONITORING\LIMITS	LIMit :LOWer	LIMit :UPPer	Settable limits MIN	Settable limits MAX	To ETR 290 (DVB) or to Doc A/65 (ATSC) MIN	To ETR 290 (DVB) or to Doc A/65 (ATSC) MAX	DVB/ATSC
PATR	PAT Distance	X	X	0 ms to 100 ms	0.1 s to 60.0 s	25 ms	0.5 s	DVB/ATSC
CATR	CAT Distance	X	X	0 ms to 100 ms	0.1 s to 60.0 s	25 ms	0.5 s	DVB/ATSC
PMTR	PMT Distance	X	X	0 ms to 100 ms	0.1 s to 60.0 s	25 ms	0.5 s	DVB/ATSC
NITR	NIT Distance	X	X	0 ms to 100 ms	0.1 s to 60.0 s	25 ms	10 s	only DVB
SDTR	SDT Distance	X	X	0 ms to 100 ms	0.1 s to 60.0 s	25 ms	2 s	only DVB
BATR	BAT Distance	X	X	0 ms to 100 ms	0.1 s to 60.0 s	25 ms	10 s	only DVB
EITR	EIT Distance	X	X	0 ms to 100 ms	0.1 s to 60.0 s	25 ms	2 s	only DVB
RSTR	RST Distance	X	---	0 ms to 100 ms	---	25 ms	---	only DVB
TDTR	TDT Distance	X	X	0 ms to 100 ms	0.1 s to 60.0 s	25 ms	30 s	only DVB
TOTR	TOT Distance	X	X	0 ms to 100 ms	0.1 s to 60.0 s	25 ms	30 s	only DVB
PCRR	PCR Distance	X	X	0 ms to 10 ms	0.01 s to 1.00 s	0 ms	0.04 s	DVB/ATSC
PCRD	PCR Discontinuity	---	X	---	0.01 s to 1.00 s	---	0.1 s	DVB/ATSC
PTSR	PTS Distance	---	X	---	0.1 s to 60.0 s	---	0.7 s	DVB/ATSC
PIDR	PID Distance ALL PID / V+A	---	X	---	0.1 s to 60.0 s	---	0.5 s	DVB/ATSC
PDOR	PID Distance D+O	---	X	---	0.1 s to 60.0 s	---	---	DVB/ATSC

Parameter-name	MONITORING\LIMITS	

4.7 Instrument Model and Command Processing

The instrument model shown in Fig 4-2 has been configured for processing remote-control commands. The individual components operate independently of each other and simultaneously. They communicate with each other by means of messages.

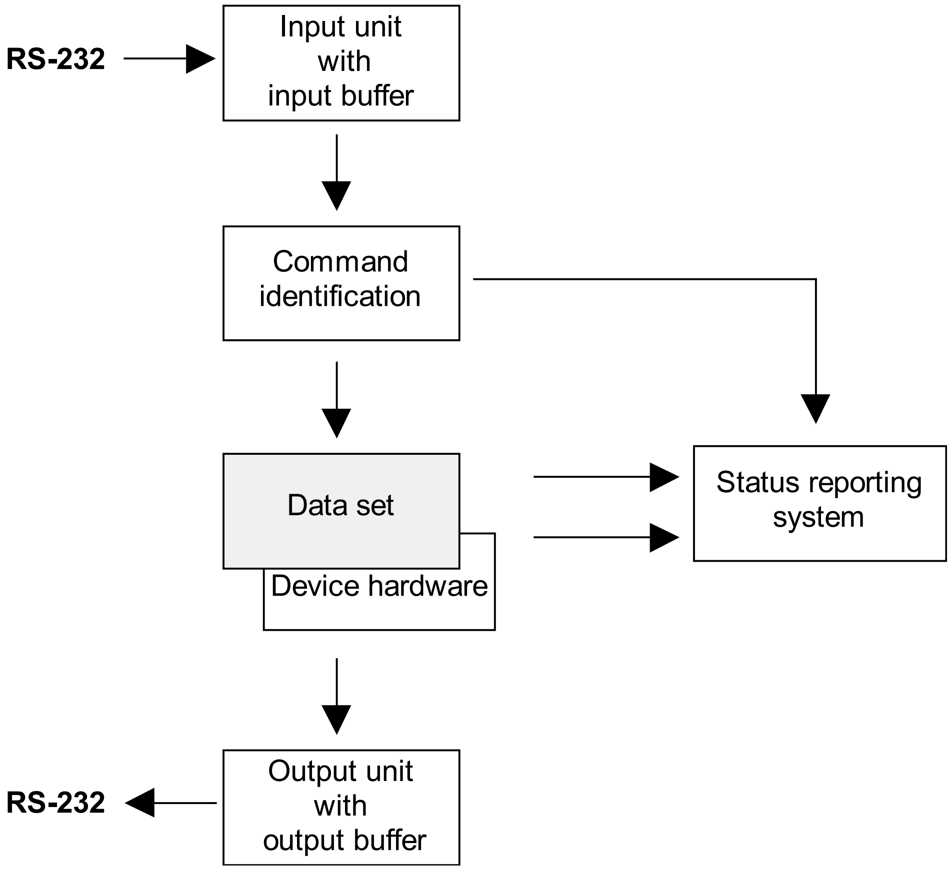


Fig. 4-2 Instrument model with remote control via RS-232 interface

4.7.1 Input Unit

The input unit receives the commands in the form of characters from the remote-control interface and collects them in the input buffer. The input buffer has a capacity of 512 characters. The input unit sends a message to the command identification as soon as the input buffer is full or as soon as it receives a terminator (see table A8 of Annex A)

If the input buffer is full or a terminator is received, data reception is stopped and the data received so far are processed. Data reception is then continued.

4.7.2 Command Identification

The command identification analyzes the data received from the input unit, proceeding in the order in which the data are received. Syntax errors in the command are recognized and passed on to the status reporting system. Following the syntax test, the range of values of the data is verified and the setting made accordingly. Only when the command has been completely executed will the next command be processed by the command identification.

4.7.3 Data Set and Device Hardware

The term "device hardware" refers to that part of the device which performs the measurement function (measurements, etc).

The data set contains all parameters required for setting the device hardware.

Setting commands cause a modification in the data set. Before the data are entered in the data set, they are verified for compatibility both with the other data and with the device hardware. If it turns out that the setting is not possible, an "execution error" message will be sent to the status reporting system and the setting ignored. After successful completion of the verification, setting will immediately be performed. Prior to the hardware setting, the SETTling bit is set in the STATus:OPERation register (see section 4.8.3.3). The hardware carries out the settings and as soon as the settled state is reached the bit is reset. The SETTling bit is only set if hardware settings take more time than processing by the command identification. This bit may be used for synchronization of the command processing.

Queries cause the data-set management to send the desired data to the output unit.

4.7.4 Status Reporting System

The status reporting system collects information about the device status and makes is available to the output unit on request. Structure and function are described in detail in section 4.8.

4.7.5 Output Unit

The output unit collects the information requested by the controller from the data set management. It processes the information in line with the SCPI rules and makes it available in the output buffer.

4.7.6 Command Sequence and Command Synchronization

All commands are immediately executed. There is no overlapping command processing. The user therefore can determine the sequence of execution. The `*WAI` command has no effect on the sequence of the command execution.
If a longer program message is terminated by `*OPC` or `*OPC?`, the end of command processing will be signalled to the controller by a suitably programmed message.

Table 4-1 Synchronization with *OPC, *OPC?

Command	Action after hardware setting	Programming of controller
*OPC	Setting the Operation Complete bit in the ESR	- Setting bit 0 in the ESE - Setting bit 5 in the SRE - Waiting for service request (SRQ)
*OPC?	Writing a "1" into the output buffer	Addressing the device as a talker

An example of command synchronization is given in section 4.11.

4.8 Status Reporting System

The status reporting system (see Fig. 4-4) stores all information on the current operating status of the instrument (eg a new measurement is being performed) and on errors. The information is stored in the status registers and the error queue. The contents of the status registers and the error queue can be queried via remote control.

The information is hierarchically structured. The topmost level is formed by the status byte (STB) defined by IEEE 488.2 and the associated Service Request Enable (SRE) Register. The STB receives its data from the Standard Event Status Register (ESR) also defined in IEEE 488.2 and the associated Standard Event Status Enable (ESE) Register as well as from the SCPI-defined STATus:OPERation and STATus:QUEStionable registers which contain detailed information on the instrument.
The output buffer contains the messages returned by the device to the controller. It is not part of the status reporting system, but since it determines the value of the MAV bits in the STB, it is also shown in Fig. 4-4.

4.8.1 Structure of an SCPI Status Register

Each SCPI register consists of five registers of 16 bit each and with different functions (see Fig. 4-3). The individual bits are independent of each other, ie each hardware status is assigned a bit number which is the same for all five registers. Bit 54 of the STATus:OPERation register, for instance, is assigned to "Waiting for TRIG" in all five registers. Bit 15 (the most significant bit) is set to zero in all SCPI status registers. Thus the contents of the registers can be processed by the controller program as a positive integer.

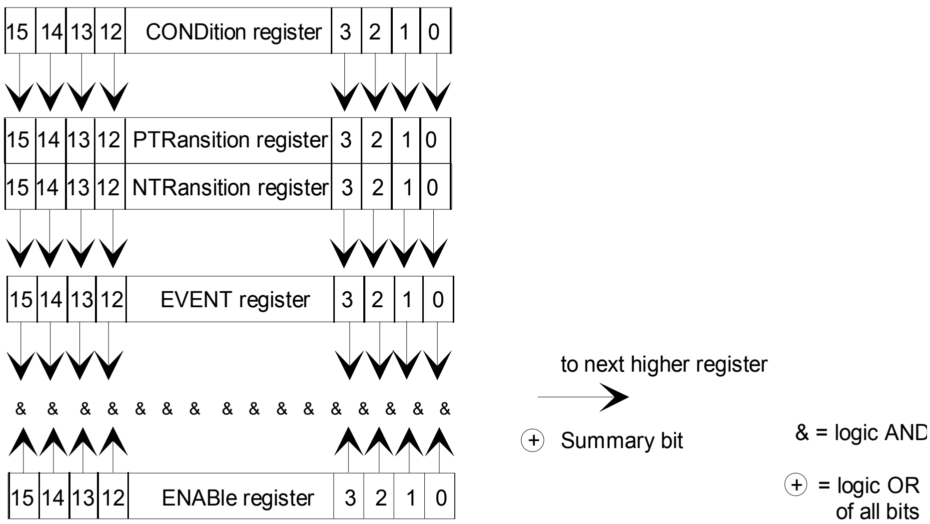


Fig. 4-3 Status Register model

CONDition register The CONDition register is directly written to by the hardware or the summary bit of the next lower register. Its contents reflects the current device status. This register can only be read, but neither written nor cleared. Reading the register does not change its contents.

PTRansition register The Positive TRansition register acts as a transition filter. Upon transition of a bit of the CONDition register from 0 to 1, the associated PTR bit decides whether the EVENt bit will be set to 1
PTR bit = 1: the EVENt bit is set.
PTR bit = 0: the EVENt bit is not set.
This register can be written and read. Reading the register does not change its contents.

NTRansition register The Negative TRansition register also acts as a transition filter. Upon transition of a bit of the CONDition register from 1 to 0, the associated NTR bit decides whether the EVENt bit will be set to 1.
NTR bit = 1: the EVENt bit is set.
NTR bit = 0: the EVENt bit is not set.
This register can be written and read. Reading the register does not change its contents.

With the aid of these two transition filter registers the user can define the status change of the CONDition register (none, 0 to 1, 1 to 0 or both) that is to be reported in the EVENt register.

EVENt register The EVENt register reports whether an event has occurred since its last reading, ie it is the "memory" of the CONDition register. It only indicates events that have been transmitted by the transition filters. The EVENt register is continuously updated by the device. It can only be read by the user. Reading this register clears its contents. This register is frequently referred to as the overall register.

ENABle register The ENABle register determines whether the EVENt bit influences the summary bit (see below). Each bit of the EVENt register is ANDed (symbol '&') with the associated ENABle bit. The events of all logical operations of this register are ORed (symbol '+') and passed on to the summary bit.
ENAB bit = 0: the associated EVENt bit has no influence on the summary bit.
ENAB bit = 1: if the associated EVENt bit is "1", the summary bit is also set to "1".
This register can be written and read by the user. Reading the register does not change its contents.

Summary bit As stated above, the summary bit for each register is derived from the EVENt and ENABle registers. The result is entered into a bit of the CONDition register of the next higher register.
The device automatically generates the summary bit for each register. An event, eg a missing input signal, may thus cause a service request through all hierarchical levels.

4.8.2 Overview of Status Registers

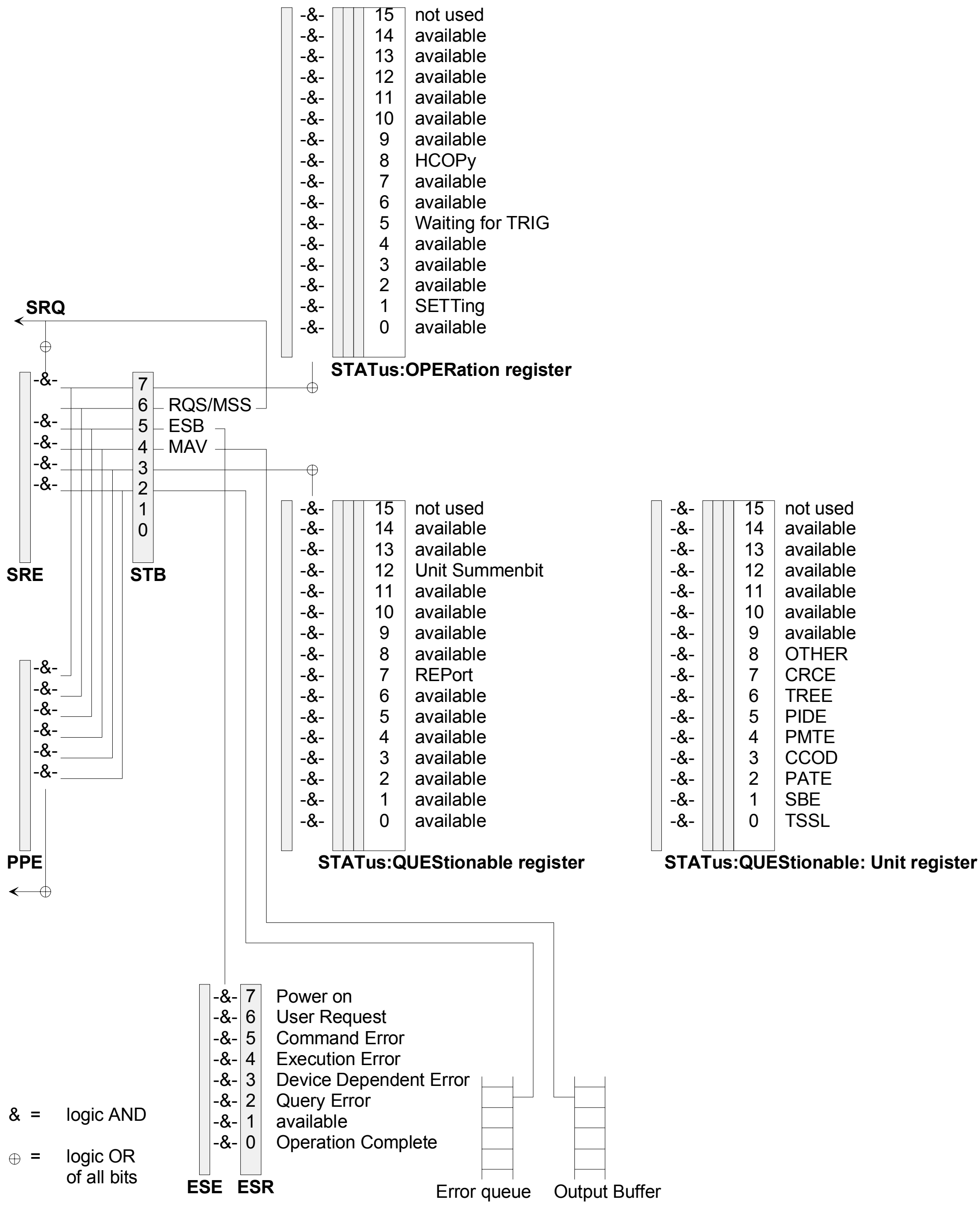


Fig. 4-4 Overview of Status Registers

4.8.3 Description of Status Registers

4.8.3.1 Status Byte (STB) and Service Request Enable Register (SRE)

The status byte is already defined in IEEE 488.2. It is the root of the SCPI status register tree. The previously defined bits of the IEEE 488.2 standard remain unchanged. Bits 3 and 7 are new, ie the summary bits of the QUEStionable and OPERationable status registers. Although the STB is integrated into the SCPI hierarchy, there are some historical differences.

The function of the Service Request Enable Register SRE corresponds to that of the STB ENABle register. The summary bit of the STB is its own bit 6. The STB has no EVENt register, it directly represents the device status in the CONDition register.

PTRansition and NTRansition registers therefore have no significance and are not defined.

Each bit of the STB is assigned a bit in the SRE. Bit 6 of the SRE is ignored. If a bit is set in the SRE and the associated bit in the STB changes from 0 to 1, a service request (SRQ) will be generated which triggers an interrupt in the controller, provided that the controller has been configured accordingly, and can be further processed by the controller.
The status byte is read out by the query `*STB?`. The SRE can be set by the command `*SRE` and read out by the query `*SRE?`.

Table 4-2 Meaning of bits used in status byte

Bit No.	Meaning
2	**Error queue not empty** This bit is set when the error queue receives an entry. If this bit is enabled by the SRE, each entry in the error queue will cause a service request. An error can thus be recognized and specified in detail by querying the error queue. The query yields a conclusive error message.
3	**QUEStionable Status Register summary bit** This bit is set if an EVENt bit is set in the QUEStionable Status Register and the associated ENABle bit is set to 1. A set bit denotes a questionable device status which can be specified in greater detail by querying the QUEStionable Status Register.
4	**MAV bit** (message available) This bit is set if a readable message is in the output buffer.
5	**ESB bit** Summary bit of the Event Status Register. This bit is set if one of the bits in the Event Status Register is set and enabled in the Event Status Enable Register (ENABle bit set to '1'). Setting this bit denotes a serious error which can be specified in greater detail by querying the Event Status Register.
6	**MSS bit** (master status summary bit) This bit is set if the instrument triggers a service request, which happens if one of the other bits of this register is set together with its enable bit in the Service Request Enable Register (SRE).
7	**OPERation Status Register summary bit** This bit is set if an EVENt bit is set in the OPERation Status Register and the associated ENABle bit is set to 1. A set bit denotes that an action is being performed by the instrument. Information on the type of action can be obtained by querying the OPERation Status Register.

4.8.3.2 Event Status Register (ESR) and Event Status Enable Register (ESE)

Although the ESR is already defined in the IEEE 488.2 standard, it fits quite well into the SCPI register mode. It is comparable to the EVENt register of an SCPI register. The ESE forms the associated ENABle register. All PTRansition bits are fixed to logic '1' and all NTRansition bits to logic '0'. The Event Status Register can be read out by the query `*ESR?`. The ESE can be set by the command `*ESE` and read out by the query `*ESE?`.

Table 4-3 Meaning of bits used in Event Status Register

Bit No.	Meaning
0	**Operation Complete** Upon reception of the *OPC command this bit is set as soon as all previous commands have been executed.
2	**Query Error** This bit is set if the controller wants to read data from the device without prior query. A frequent cause is a faulty query which cannot be executed.
3	**Device-dependent Error** This bit is set if a device-dependent error occurs. An error message with a number between -300 and -399 or a positive error number denoting the error in greater detail (see section 4.10, Error Messages of Remote-Control Interface) will be entered into the error queue.
4	**Execution Error** This bit is set if the syntax of the command received is correct but the command cannot be executed for various reasons. An error message with a number between -200 and -300 describing the error in greater detail (see section 4.10, Error Messages of Remote-Control Interface) will be entered into the error queue.
5	**Command Error** This bit is set if an undefined command or a command with incorrect syntax is received. An error message with a number between -100 and -200 describing the error in greater detail (see section 4.10, Error Messages of Remote-Control Interface) will be entered into the error queue.
6	**User Request** This bit is set upon transition from [REMOTE] to [LOCAL], ie when the instrument is switched to manual operation by means of the LOCAL key.
7	**Power On** (AC supply on) This bit is set upon power-on of the instrument.

4.8.3.3 STATus:OPERation Register

The OPERation Status Register contains information on the operations currently performed by the instrument.
It can be read by the queries `STATus:OPERation:CONDition?` or `STATus:OPERation[:EVENt]?`.

Table 4-4 Meaning of bits used in STATus:OPERation Register

Bit No.	Meaning
1	**SETTling** This bit is set as long as settling takes place following a new setting command. It is only set if the settling time is longer than the command processing time.
5	**Waiting for TRIG** This bit is set if the device is active in the Trigger on Error or DUMP mode.
8	**HCOPy** This bit is set when the print function is active.

4.8.3.4 STATus:QUEStionable Register

This register contains information on questionable device states which may for instance occur if the instrument is operated out of specifications. The register can be read by the queries `STATus:QUEStionable:CONDition?` or `STATus:QUEStionable[:EVENt]?`.

Table 4-5 Meaning of bits used in STATus:QUEStionable Register

Bit No.	Meaning
9	**REPort** This bit is set if the MONITORING/REPORT contains entries that have not yet been read using READ:MONitoring:REPort?.
13	**DVRM summary bit** This summary bit is set if an EVENt bit is set in the QUEStionable:DVRM Register and the associated ENABle bit is set to 1.

4.8.4 STATus:QUEStionable:DVRM Register

This register contains information on questionable DVRM-specific device states.
It can be read with the queries `STATus:QUEStionable:DVRM:CONDition?` or `STATus:QUEStionable:DVRM[:EVENt]?`.
The bit assignment corresponds to the LED display on the front panel.

Table 4-6 Meaning of bits used in STATus:QUEStionable:DVRM Register

Bit No.	Meaning
0	**TSSL** This bit is set if a TS SYNC error is indicated in the MONITORING/STATISTICS menu.
1	**SBE** This bit is set if a SYNC BYTE error is indicated in the MONITORING/STATISTICS menu.
2	**PATE** This bit is set if a PAT error is indicated in the MONITORING/STATISTICS menu.
3	**CCOE** This bit is set if a CONT_COUNT error is indicated in the MONITORING/STATISTICS menu.
4	**PMTE** This bit is set if a PMT error is indicated in the MONITORING/STATISTICS menu.
5	**PIDE** This bit is set if a PID error is indicated in the MONITORING/STATISTICS menu.
6	**TPEE** This bit is set if a TRANSPORT error is indicated in the MONITORING/STATISTICS menu.
7	**CRCE** This bit is set if a CRC error is indicated in the MONITORING/STATISTICS menu.
8	**OTHER** This bit is set if an error is indicated in the MONITORING/STATISTICS menu for parameters not mentioned above.

4.8.5 Use of the Status Reporting System

For an efficient use of the status reporting system, the information contained has to be transferred to the controller program and processed. There are various methods which are described in the following. Detailed program examples are given section 4.11.

4.8.5.1 Service Request, Use of Hierarchical Structure

Under certain conditions, the instrument may send a service request (SRQ) to the controller. This service request usually causes an interrupt at the controller to which the controller program can respond as appropriate. As shown in Fig. 4-3 (section 4.8.2) an SRQ will always be triggered if one or several of the bits 2, 3, 4, 5 or 7 of the status byte have been set and enabled in the SRE. Each of these bits combines the information from another status register, from the error queue or the output buffer. If the ENABle registers of the status registers are set accordingly, any bit in any status register will be able to trigger a SRQ. To utilize the possibilities of the service request, the corresponding bits in the enable registers must be set to "1".

Example (see also Fig. 4-3, section 4.8.2 and section 4.12.3.2):

Using the command `*OPC` to generate an SRQ

- Set bit 0 (Operation Complete) in the ESE
- Set bit 5 (ESB) in the SRE
  The instrument generates an SRQ upon completion of its settings.

Indicating the end of a trigger event by an SRQ on the controller.

- Set bit 7 (sum bit of STATus:OPERation register) in the SRE
- Set bit 54 (Waiting for TRIG) in STATus:OPERation:ENABle.
- Set bit 54 in STATus:OPERation:NTRansition so that the transition of the Waiting for TRIG bit from 1 to 0 (end of measurement) is also marked in the EVENt register.
  The device generates an SRQ after the data have been read in.

The SRQ is the only way for the instrument to become active of its own. A controller program should set the instrument so that a service request will be triggered in case of malfunctions. The program should suitably respond to the service request. A detailed example of a service request routine is given in 4.12.3.2.

4.8.5.2 Queries

Each part of a status register can be read out by queries. The queries are given in the detailed description of the registers in section 4.8.3. The queries always yield a number representing the bit pattern of the queried register. This number is evaluated by the controller program.
Queries to read out the status registers are mainly used after an SRQ to obtain detailed information about the cause for the SRQ.

4.8.5.3 Error Queue Query

Each error in the instrument causes an entry in the error queue. The entries in the error queue are detailed error messages in plain text which can be read out via remote control by the query `SYSTem:ERRor?`. Each `SYSTem:ERRor?` query is answered by an entry from the error queue. If there are no more error messages in the error queue, 0 = "No error" is returned by the instrument.
The error queue should be queried in the controller program whenever a precise description of the error cause is required after a SRQ. In particular in the test phase of a controller program, the entries in the error queue provide a valuable support, since they also contain faulty commands from the controller to the instrument.

4.8.6 Resetting the Status Reporting System

Table 4-7 contains the various commands and events causing a reset of the status reporting system. None of the commands, with the exception of *RST affects the functional device settings..

Table 4-7 Resetting device functions

Event Effect	Switching on AC supply voltage Power On Status Clear		*RST	*CLS
	0	**1**		
Clears STB, ESR	—	yes	—	yes
Clears SRE, ESE	—	yes	—	—
Clears EVENt registers	—	yes	—	yes
Clears ENABle registers of all OPERation and QUESTionable registers, fills ENABle registers of all other registers with "1".	—	yes	—	—
Fills PTRansition registers with "1", clears NTRansition registers	—	yes	—	—
Clears error queue	yes	yes	—	yes
Clears output buffer	yes	yes	1)	1)
Clears command processing and input buffer	yes	yes	—	yes

1) Any command that is the first one in a program message, ie which immediately follows a <PROGRAM MESSAGE TERMINATOR>, clears the output buffer.

4.9 Set of Commands

The instrument supports the SCPI version 1995.0.
The list shows all commands of DVRM in alphabetical order. SCPI-defined commands are identified by "SCPI" in the column "SCPI info".

Command	Parameter	SCPI Info	Page
*CLS		SCPI	4.11
*ESE[?]	0 to 255	SCPI	4.11
*ESR?		SCPI	4.11
*IDN?		SCPI	4.11
*OPC[?]		SCPI	4.11
*PSC	0 | 1	SCPI	4.11
*RST		SCPI	4.11
*SRE[?]	0 to 255	SCPI	4.11
*STB?		SCPI	4.11
*TST?		SCPI	4.11
*WAI		SCPI	4.11
CONFigure:DUMP:COUNt[?]	1 to 1394		4.38
CONFigure:DUMP:STATe[?]	PID | AFPid | AF | PUSPid | PUS | APPid | APUS | PSI | TABid | HEADer | ALL | OFF		4.38
CONFigure:DUMP:TRIGger:PID[?]	0 to 8191		4.38
CONFigure:DUMP:TRIGger:TABid[?]	0 to 255		4.38
CONFigure:MONitoring:ALARmlines:ALL[?]	0 to 12		4.22
CONFigure:MONitoring:ALARmlines[?]	<parameter name>, 0 to 12 | STOP, 0 to 12		4.22
CONFigure:MONitoring:BITRate:LIMit:LOWer[?]	NULL, 1000 to 54000000 bit/s		4.22
CONFigure:MONitoring:BITRate:LIMit:UPPer[?]	NULL, 1000 to 54000000 bit/s		4.22
CONFigure:MONitoring:CONTrol[?]	STARt | STOP | CLEar		4.21
CONFigure:MONitoring:LIMit:LOWer[?]	<parameter name>, 0 to 100 ms		4.21
CONFigure:MONitoring:LIMit:MIPTiming[?]	100 to 99900 ns		4.21
CONFigure:MONitoring:LIMit:PDOR[?]	ON | OFF		4.21

Command	Parameter	SCPI Info	Page
CONFigure:MONitoring:LIMit:PIDR[?]	ALLPid | VAONly		4.21
CONFigure:MONitoring:LIMit:UPPer[?]	<parameter name>, 100 to 60000 ms		4.21
CONFigure:MONitoring:MULTiplex:LIMit:LOWer[?]	TSID, 0 to 65535		4.22
CONFigure:MONitoring:MULTiplex:LIMit:UPPer[?]	TSID, 0 to 65535		4.22
CONFigure:MONitoring:PARameter:ALL	ON | OFF		4.21
CONFigure:MONitoring:PARameter[?]	<parameter name>, ON | OFF		4.21
CONFigure:MONitoring:PROGram:MODE[?]	AUTO | MANual,		4.22
CONFigure:MONitoring:PROGram:SELect[?]	1 to 65535, ON | OFF		4.22
CONFigure:TRERror:STATe[?]	ON | OFF		4.36
CONFigure:TRERror:TRIGger:ALL	ON | OFF		4.36
CONFigure:TRERror:TRIGger[?]	<parameter name>, ON | OFF		4.36
HCOPy:ABORt			4.31
HCOPy:MONitoring:DATA?			4.30
HCOPy:MONitoring[:IMMediate]			4.31
HCOPy:PROGram:DATA?			4.30
HCOPy:PROGram[:IMMediate]			4.31
HCOPy:REPort:DATA?			4.30
HCOPy:REPort:LINes:DATA?	0 to 999, 0 to 999		4.30
HCOPy:REPort:LINes[:IMMediate]	0 to 999, 0 to 999		4.31
HCOPy:REPort[:IMMediate]			4.31
HCOPy:SETTings:DATA?			4.30
HCOPy:SETTings[:IMMediate]			4.31
HCOPy:STATe?			4.31
MEMory:SETTing1...10:NAME?			4.32
MEMory:SETTing1...10:RECall			4.32
MEMory:SETTing1...9:STORe	’NAME’		4.32
READ:ARR:EIT?			4.27
READ:ARR:EIT?	1 to 65535, 0 to 65535, 0 to 65535		4.27

Command	Parameter	SCPI Info	Page
READ:ARR:NIT?			4.26
READ:ARR:NIT?	0 to 65535		4.26
READ:ARR:SDT?			4.27
READ:ARR:SDT?	0 to 65535, 0 to 65535		4.27
READ:ARRay:PROGram:PID?	1 to 65535		4.25
READ:ARRay:PROGram?			4.25
READ[:SCALar]:DUMP:COUNt[?]			4.39
READ[:SCALar]:DUMP?	1 to 1394		4.39
READ[:SCALar]:MONitoring:ALL?			4.16
READ[:SCALar]:MONitoring:ALL?	EXTended		4.16
READ[:SCALar]:MONitoring:DURation?			4.20
READ[:SCALar]:MONitoring:ERRSeconds:ALL?			4.17
READ[:SCALar]:MONitoring:ERRSeconds:ALL?	EXTended		4.17
READ[:SCALar]:MONitoring:ERRSeconds?	<parameter name>		4.17
READ[:SCALar]:MONitoring:PROGram:COUNt?			4.20
READ[:SCALar]:MONitoring:REPort:EXTMoment?	<parameter name>, 1900 to 2100, 1 to 12, 1 to 31, 0 to 23, 0 to 59, 0 to 59		4.19
READ[:SCALar]:MONitoring:REPort:LINE?	0 to 999		4.18
READ[:SCALar]:MONitoring:REPort:MOMent?	<parameter name>, 1900 to 2100, 1 to 12, 1 to 31, 0 to 23, 0 to 59, 0 to 59		4.19
READ[:SCALar]:MONitoring:REPort?			4.18
READ[:SCALar]:MONitoring?	<parameter name>		4.16
READ[:SCALar]:PID:BITRate?	0 to 8191		4.26
READ[:SCALar]:PID:CONDaccess?	0 to 8191		4.26
READ[:SCALar]:PID:TYPE?	0 to 8191		4.26
READ[:SCALar]:PROGram:BITRate?	1 to 65535		4.25
READ[:SCALar]:PROGram:CONDaccess?	1 to 65535		4.25
READ[:SCALar]:PROGram:NAME?	1 to 65535		4.25
READ[:SCALar]:TRERror:TRIGger?			4.36

Command	Parameter	SCPI Info	Page
READ[:SCALar]:TRERror?	-400 to +400		4.36
READ[:SCALar]:TS:BITRate?			4.25
READ[:SCALar]:TS:ID?			4.25
ROUTe:INPut[?]	RSERial | FSERial | FPARallel		4.28
SENSe:FUNCtion[:ON][?]	MONitoring | TRERror | DUMP | MEASure		4.12
SENSe:SCONdition:DROP[?]	1 to 7		4.28
SENSe:SCONdition:LOCK[?]	1 to 31		4.28
STATus:OPERation:CONDition?		SCPI	4.41
STATus:OPERation:ENABle[?]	0 to 32767	SCPI	4.41
STATus:OPERation:NTRansition[?]	0 to 32767	SCPI	4.41
STATus:OPERation:PTRansition[?]	0 to 32767	SCPI	4.41
STATus:OPERation[:EVENt]?		SCPI	4.41
STATus:QUEStionable:CONDition?		SCPI	4.41
STATus:QUEStionable:DVRM:CONDition?		SCPI	4.42
STATus:QUEStionable:DVRM:ENABle[?]	0 to 32767	SCPI	4.42
STATus:QUEStionable:DVRM:NTRansition[?]	0 to 32767	SCPI	4.42
STATus:QUEStionable:DVRM:PTRansition[?]	0 to 32767	SCPI	4.42
STATus:QUEStionable:DVRM[:EVENt]?		SCPI	4.42
STATus:QUEStionable:ENABle[?]	0 to 32767	SCPI	4.41
STATus:QUEStionable:NTRansition[?]	0 to 32767	SCPI	4.42
STATus:QUEStionable:PTRansition[?]	0 to 32767	SCPI	4.42
STATus:QUEStionable[:EVENt]?		SCPI	4.41
STATus:QUEue[:NEXT]?		SCPI	4.40
SYSTem:ALARmlines:POLarity[?]	NORMal | INVerted		4.35
SYSTem:COMMunicate:PRINter:CHANnel[?]	SERial | CENTronics	SCPI	4.34
SYSTem:COMMunicate:SERial[:RECeive]:BAUD[?]	1200 | 2400 | 4800 | 9600 | 19200 | 38400 | 57600 | 115200	SCPI	4.34
SYSTem:COMMunicate:SERial[:RECeive]:BITS[?]	8	SCPI	4.34

Command	Parameter	SCPI Info	Page
SYSTem:COMMunicate:SERial[:RECeive]:PACE[?]	XON | ACK | NONE	SCPI	4.34
SYSTem:COMMunicate:SERial[:RECeive]:PARity[:TYPE][?]	EVEN | ODD | NONE	SCPI	4.34
SYSTem:COMMunicate:SERial[:RECeive]:SBITs[?]	1	SCPI	4.34
SYSTem:DATE[?]	1980 to 2079, 1 to 12, 1 to 31	SCPI	4.34
SYSTem:ERRor?		SCPI	4.35
SYSTem:FORMat:BITRate[?]	TOTal | NET		4.35
SYSTem:FORMat:PID[?]	DECimal | HEXadecimal		4.35
SYSTem:IDN[?]	’KENNUNG’		4.35
SYSTem:TIME[?]	0 to 23, 0 to 59, 0 to 59	SCPI	4.34
SYSTem:VERSion?		SCPI	4.35
SYSTem:VIEW:TABLe[?]	ON | OFF		4.35

4.10 Error Messages of Remote-Control Interface

The list below contains all possible error messages. Negative error numbers are defined in the SCPI standard, positive error numbers identify device-specific errors.

The error code is queried with **SYSTem:ERR?**

The left column of the table below gives the error code. In the right column, the text of the error message entered in the error/event queue is printed in bold. An additional explanation is given below this text.

4.10.1 SCPI-Specific Error Messages

Table 4-8 No error

Error number	**Error text returned upon queue query** Explanations
0	**No error** This message is displayed if the error queue is empty.

Table 4-9 Command Errors to cause bit 5 in the ESR register to be set

Error number	**Error text returned upon queue query** Explanations
-100	**Command Error** Command faulty or invalid.
-101	**Invalid Character** The command comprises an invalid character. Example: a header contains an AND sign, "`SENS&`".
-102	**Syntax error** The command is invalid. Example: the command contains block data which the device does not accept.
-103	**Invalid separator** The command contains an illegal character instead of a delimiter. Example: no semicolon after command.
-104	**Data type error** The command contains an illegal data element. Example: ON is received instead of a numeric value.
-105	**GET not allowed** Group Execute Trigger (GET) within a program message.
-108	**Parameter not allowed** The command contains too many parameters. Example: the command `SENSe:PROGram` allows only one program number.
-109	**Missing parameter** The command contains fewer parameters than required. Example: the command `SENSe:PROGram:VIDeo` requires a PID to be specified.

Table 4-9 Command Errors to cause bit 5 in the ESR register to be set

Error number	**Error text returned upon queue query** Explanations
-111	**Header separator error** The header contains an illegal delimiter. Example: header not followed by a "white space", "*ESE255"
-112	**Program mnemonic too long** The header contains more than 12 characters.
-113	**Undefined header** The header is not defined for the device. Example: *XYZ is not defined for any device.
-114	**Header suffix out of range** The header contains an illegal numeric suffix. Example: CONFigure:MONitoring:PROGram:LIST30 does not exist.
-120	**Numeric data error** The command contains an erroneous numeric parameter.
-121	**Invalid character in number** A numeric value contains an invalid character. Example: "A" in a decimal numeric value or "9" in an octal number.
-123	**Exponent too large** The absolute value of the exponent is >32000.
-124	**Too many digits** The numeric value consists of too many digits.
-128	**Numeric data not allowed** The command contains numeric data not allowed at this position.
-131	**Invalid suffix** The entered suffix is not valid for this device. Example: nHz is not defined.
-134	**Suffix too long** The suffix contains more than 12 characters.
-138	**Suffix not allowed** The suffix used is not allowed for this command or at this position of the command. Example: the command SENSe:PROGram does not allow a suffix to be specified.
-141	**Invalid character data** The character data element either contains an invalid character or is not valid for this command. Example: spelling mistake in parameter specification.
-144	**Character data too long** The character data element contains more than 12 characters.
-148	**Character data not allowed** The character data element used is not allowed for this command or at this position of the command. Example: the command SENSe:PROGram requires a numeral to be specified.
-151	**Invalid string data** The command contains invalid string data. Example: an END message was received before the terminal quote character.
-158	**String data not allowed** The command contains legal string data at a position where they are not allowed. Example: character data element in quotation marks.

Table 4-9 Command Errors to cause bit 5 in the ESR register to be set

Error number	**Error text returned upon queue query** Explanations
-168	**Block data not allowed** The command contains legal block data at a position where they are not allowed.
-171	**Invalid expression** The command contains an invalid mathematical expression. Example: the expression contains unmatched parentheses.
-178	**Expression data not allowed** The command contains a mathematical expression at a position where it is not allowed.

Table 4-10 Execution Errors to cause bit 4 in the ESR register to be set

Error number	**Error text returned upon queue query** Explanations
-200	**Execution error** Error encountered upon command execution (eg printer has run out of paper).
-221	**Settings conflict** There is a setting conflict between two parameters. Example: In the DUMP mode, monitoring cannot be started when MONITORING is selected.
-222	**Data out of range** The data element was outside the legal range as defined by the device. Example: the command `SENSe:PROGram` allows entries only in the range 1 to 65535.
-223	**Too much data** The command contains too much data. Example: the memory of the device does not have enough storage capacity.
-224	**Illegal parameter value** The parameter value is illegal Example: the parameter value does not comply with the list of possible values.
-230	**Data corrupt or stale** Data incomplete or invalid. Example: a measurement was aborted by the device.
-240	**Hardware error** The command cannot be executed because of hardware problems in the device.
-241	**Hardware missing** The command cannot be executed because of missing device hardware. Example: an option is not fitted.

Table 4-11 Device-Specific Errors to cause bit 3 in the ESR register to be set

Error number	**Error text returned upon queue query** Explanations
-300	**Device-specific error** Device-specific error that cannot be defined more precisely.
-310	**System error** Indicates that some device-internal error has occurred. Please contact the R&S service center.
-311	**Memory error** An error was detected in the device's memory.
-313	**Calibration memory lost**
-314	**Save/recall memory lost**
-315	**Configuration memory lost**
-330	**Selftest failed**
-350	**Queue overflow** Error code entered in the queue instead of the code that caused the error when the queue is full. The error was not recorded. Five of these codes are accepted in the queue.

Table 4-12 Query Errors to cause bit 2 to be set in the ESR register

Error number	**Error text returned upon queue query** Explanations
-400	**Query error** Generic query error that cannot be defined more precisely.
-410	**Query INTERRUPTED** The query was interrupted. Example: a query is followed by new data before the response was completely sent.
-420	**Query UNTERMINATED** The query is incomplete. Example: the device receives incomplete data.
-430	**Query DEADLOCKED** The query cannot be processed. Example: both input buffer and output buffer are full and the device cannot continue.

4.11 Programming Instructions

With baud rates >9600 baud special precautions have to be taken to avoid data losses.

- The RS232 interface of the controller has to be rated for these baud rates.
- The controller must be able to process the received data in time.
- Hardware handshake has to be used to synchronize data transmission.
- The connection from controller to DVRM has to be made by means of the null modem cable required for hardware handshake (for wiring plan see Annex A.5).
- Please note the instructions regarding your RS232 control program to avoid data losses.

4.12 Programming Example

The example illustrates programming of the device and may be used as a basis for solving more complex programming problems.

4.12.1 General

The programming language is Borland C. The serial interface is supported by the program packet "V.24 Tools Plus" from Langner GmbH.

4.12.2 Definitions of Library Calls Supporting the V24 Interface

Function	Definitions
v24open v24setparams v24sethandshake v24settimeout v24setbuffer v24flush	These functions initialize the serial interface.
v24puts	This function outputs a string to the serial interface. This string is not automatically terminated with CR/LF.
v24gets	This function reads a string from the serial interface.
v24read	This function reads an array of defined length from the serial interface.

4.12.3 Examples

4.12.3.1 Example 1

The example comprises a general routine by which the connection to DVRM is made, the DVRM signal input is selected and the presence of a transport stream is checked.
Then, two routines are provided for reading out the error statistics (MONITORING mode) and the transport stream data (DUMP mode).

```
#include <stdio.h>
#include "v24tools.h"

int main()
{
#define ANSWERBUFFER_LEN    256
  char answerbuffer[ANSWERBUFFER_LEN + 1];
  int  v24_port;
  unsigned int v24_timeout;
  int  qstate, qcount, qerror;

  printf("\nINIT COM PORT\n");
  /* open COM port */
  if ((v24_port = v24open("COM1")) < 0)
  {
    printf("   ***COM port open error\n");
    return(v24_port);
  }
  /* set communication parameter */
  v24setparams(v24_port, 9600L,       /* baudrate */
                             8,       /* databits */
                             N,       /* parity none */
                             1);      /* stopbits */
  /* set receive timeout */
  v24_timeout = 18;                   /* 18 = 1 second */
  v24settimeout(v24_port, v24_timeout);
  /* define receive buffer and transmit buffer */
  v24setbuffer(v24_port, SND, 256);   /* transmit buffer */
  v24setbuffer(v24_port, RCV, 256);   /* receive buffer */
  v24flush(v24_port, SND);            /* clear */
  v24flush(v24_port, RCV);            /* clear */
  /* set communication handshake */
  v24sethandshake(v24_port, V24RTSCTS);
  DTR(v24_port,EIN);  /* DTR line on */

  printf("CLEAR DVRM STATUS AND READ IDENTIFICATION\n");
  v24puts(v24_port, "*cls\r\n");   /* \r\n = command terminator */
  v24puts(v24_port, "*idn?\r\n");
  if (v24gets(v24_port, &answerbuffer[0], ANSWERBUFFER_LEN,'\n'))
    printf("   %s\n", &answerbuffer[0]);

  printf("START MONITORING ON INPUT SERIAL FRONT\n");
  v24flush(v24_port, RCV);   /* flush receive buffer */
  v24puts(v24_port, "sense:function:on monitoring\r\n");
  v24puts(v24_port, "route:input fserial\r\n");
  /* test sync ok */
  do
  {
    v24flush(v24_port, RCV);
    v24puts(v24_port, "status:questionable:DVRM?\r\n");
    qstate = 0x0001;
    if (v24gets(v24_port, &answerbuffer[0], ANSWERBUFFER_LEN,'\n'))
    {
      if (sscanf(&answerbuffer[0], "%d", &qstate) == 1)
        qstate &= 0x0001;
    }
```

```
  } while (qstate == 0x0001);

  printf("READ MONITORING STATISTIC CRC ERROR\n");
  v24flush(v24_port, RCV);   /* flush receive buffer */
  v24puts(v24_port, "read:monitoring:errseconds? crce\r\n");
  if (v24gets(v24_port, &answerbuffer[0], ANSWERBUFFER_LEN,'\n'))
  {
    if (sscanf(&answerbuffer[0], "%d,%d", qstate, &qerror) == 2)
    {
      switch (qstate)
      {
        case -1:
          printf("   [no CRC monitoring]\n");
          break;
        case 0:
          printf("   [no] CRCerrors\n");
          break;
        case 1:
          printf("   [%d] CRCerrors\n", qerror);
          break;
      }
    }
  }

  printf("DUMP ALL AND READ FIRST PACKET\n");
  v24puts(v24_port, "sense:function:on dump\r\n");
  v24puts(v24_port, "conf:dump:count 10\r\n");
  v24puts(v24_port, "conf:dump:state all\r\n");
  /* wait dump ready */
  do
  {
    v24flush(v24_port, RCV);
    v24puts(v24_port, "conf:dump:state?\r\n");
    qstate = 1;
    if (v24gets(v24_port, &answerbuffer[0], ANSWERBUFFER_LEN,'\n'))
    {
      if (sscanf(&answerbuffer[0], "%d", &qstate) != 1)
        qstate = 1;
    }
  } while (qstate == 1);
  /* read dump packet 1 */
  v24flush(v24_port, RCV);
  v24puts(v24_port, "read:dump? 1\r\n");
  qcount = 0;
  while (v24getc(v24_port, &answerbuffer[qcount], v24_timeout))
    qcount++;
  printf("   %d bytes read\n", qcount);

  printf("PROGRAM END\n\n");
  return(0);
}
```

Follows the output of this program:

```
INIT COM PORT
CLEAR DVRM STATUS AND READ IDENTIFICATION
   "ROHDE&SCHWARZ,DVRM,0,31.22"
START MONITORING ON INPUT SERIAL FRONT
READ MONITORING STATISTIC CRC ERROR
   [no] CRCerrors
DUMP ALL AND READ FIRST PACKET
   194 bytes red
PROGRAM END
```

4.12.3.2 Example 2

The following example demonstrates the monitoring of DVRM status registers by means of SRQ (service request). The programming language is BORLAND C3.1 and the program can be run on an IBM-compatible PC using the operating system MS-DOS. A detailed comment is given on the individual program sections. The program serves as a basis for more complex remote-control programs.

The program is compiled for COM1 with 9600 baud, no parity, 8 data bits and 1 stop bit. The parameters are defined in #define COM_PORT and #define COM_PARMS. The program initializes the DVRM and enables the SRQ for the STATus:QUEStionable:DVRM register.

The interrupt service routine denominated comm_isr() stores all received characters except the SRQ characters in a ring buffer. If the read character is an SRQ character (ASCII 0x02), it is registered in a counter variable.

After initialization, the main program waits for a keystroke and then performs the selected functions.

The program is terminated with the key combination Alt-X.
The article 'Serial communications with Turbo C' by Greg Chursenoff in 'The C Users Journal' has been used as a basis.

```
/************************************************************************
*************************************************************************
**
**      MODULE    SERIAL.C
**
**      PROJECT   DVRM
**
**      COMPILER  BORLAND BC 3.1
**
**      LANGUAGE  C
**
**      DATE      21.08.98
**
**      AUTHOR    R & S
**
**      ABSTRACT  Demonstration of monitoring the status registers
**                in the DVRM over serial communication with SRQ.
**                The program is compiled for COM1 with 9600 baud
**                no parity, 8 databits and 1 stopbit.
**                The parameters are defined in (#define COM_PORT) and
**                (#define  COM_PARMS).
**                To change the comm parameters change the two defines
**                COM_PORT and COM_PARAMS and make a new compilation.
**
**                The program initialize the DVRM generating a SRQ
**                when a bit in the status:questionable:DVRM register
**                change.
**                The COMx port is initialized working with interrupts.
**                The interrupt service routine comm_isr() response
**                whenever an interrupt occurs at the COMx port.
**                The handler saves the incoming character other than the
**                SRQ character in a circular buffer.
**                Is the incoming character the SRQ character (0x02),
**                the handler counts them in the srq.detect variable.
**
**                The main program begins by initializing the comm port
**                and its own interrupt handler comm_isr(). A cleanup
**                routine comm_cleanup() restores the previous state
**                when the program terminates. Once the initialization
**                is complete, all the main program must do is:
**                 - wait for keys to be hit and select the implemented
**                   functions print comm status, toggle SRQ and
**                   display status:questionable:DVRM.
**                 - call the routine DVRM_check_srq_event().
```

```
**                 The routine handles the DVRM SRQ event, if a new
**                 SRQ is detected.
**
**              This process repeats endlessly until you type Alt-X.
**              The program then executes comm_cleanup() and
**              returns to DOS.
**
**              Description of the 8250 Asynchronous Communication Elements.
**              ------------------------------------------------------------
**                Note: The port addresses are shown as COM1 first,
**                      then COM2.
**                Receiver/Transmitter Data Register(port 0x3f8, 0x2f8).
**                  0-7     8-bit data character
**                Interrupt Enable Register (port 0x3f9, 0x2f9)
**                  0       Received data available
**                  1       Transmitter holding register empty
**                  2       Receiver Line Status
**                  3       Modem status
**                  4-7     (unused)
**                Interrupt Identification Register (port 0x3fa, 0x2fa)
**                  0       Interrupt pending (0 = true, 1 = false)
**                  1-2     Interrupt type
**                            11 = Receiver line status (OE, PE, FE, BI)
**                            10 = Received data available (DR)
**                            01 = Transmitter holding register empty (THRE)
**                            00 = Modem status (DCTS, DDSR, RI, DDCD)
**                  3-7     (unused)
**                Line Control Register (port 0x3fb, 0x2fb)
**                  0       WLSO   Word length select
**                  1       WLS1
**                  2       STB    Number of stop bits
**                  3       PEN    Parity enable
**                  4       EPS    Even parity select
**                  5       SP     Stick parity
**                  6       BRK    Set break
**                  7       DLAS   Divisor Latch access bit
**                Modem Control Register (port 0x3fc, 0x2fc)
**                  0       DTR     Data terminal ready
**                  1       RTS     Request to send
**                  2       OUT1    (not connected)
**                  3       OUT2    Enable interrupt buffer
**                  4       LOOP    Loopback (CTS, DSR, DCD, RI = DTR,
**                                           RTS, OUT1, OUT2)
**                  5-7     (unused)
**                Line Status Register (port 0x3fd, 0x2fd)
**                  0       DR      Data ready
**                  1       OE      Overrun error
**                  2       PE      Parity error
**                  3       FE      Framing Error
**                  4       BI      Break interrupt
**                  5       THRE    Transmitter holding register empty
**                  6       TSRE    Transmitter shift register empty
**                  7       (always zero)
**                Modem Status Register (port 0x3fe, 0x2fe)
**                  0       DCTS   Delta clear to send
**                  1       DDSR   Delta data set ready
**                  2       TERI   Trailing edge ring indicator
**                  3       DDCD   Delta data carrier detect
**                  4       CTS    Clear to send
**                  5       DSR    Data set ready
**                  6       RI     Ring indicator
**                  7       DCD    Data carrier detect
**              ---------------------------------------------------------
**
*************************************************************************
************************************************************************/


/* INCLUDE FILES *******************************************************/
```

```
#include  <bios.h>
#include  <dos.h>
#include  <conio.h>
#include  <stdio.h>
#include  <stdlib.h>
#include  <string.h>


/* GLOBAL VARIABLES DEFINITION *****************************************/
/* GLOBAL CONSTANTS DEFINITION *****************************************/
/* LOCAL DEFINES *******************************************************/
/* comm port definition */
#define  COM_PORT               1   /* serial port definitions for COMx */
#define  COM_PARMS    (0xe0 | 0x03)  /* comm. parameters for 9600-n-8-1 */

/* data declarations and constants */
#define DISABLE  asm pushf; asm cli  /* disable/enable interrupt flag */
#define ENABLE   asm popf

#define  INPUT_QUEUELEN       1024   /* size of serial input buffer */
#define  S_EOF                  -1   /* return value for sgetch() */
                                     /* if no characters are available */
#define  SRQ_CHAR             0x02   /* SRQ character */

/* 8259A PIC register addresses and commands */
#define  PIC_CTL_REG          0x20   /* 8259A PIC control register */
#define  PIC_INT_MASK_REG     0x21   /* 8259A PIC interrupt mask reg */
#define  NON_SPEC_EOI         0x20   /* non-specific end of interrupt */

/* 8250 ACE register addresses and bit definitions */
#if  (COM_PORT == 1)                 /* definitions for COM1 */
#define  ACE_DATA_REG        0x3f8   /* data register */
#define  ACE_INT_ENB_REG     0x3f9   /* interrupt enable register */
#define  ACE_INT_IDENT_REG   0x3fa   /* interrupt identification reg */
#define  ACE_LINE_CTL_REG    0x3fb   /* line control register */
#define  ACE_MODEM_CTL_REG   0x3fc   /* modem control register */
#define  ACE_LINE_STAT_REG   0x3fd   /* line status register */
#define  ACE_MODEM_STAT_REG  0x3fe   /* modern status register */
#define  COM_INT_NUM            12   /* interrupt number for CON1 */
#define  IRQ_MASK             0xef   /* IRQ mask for IRQ4 (11101111) */

#elif  (COM_PORT == 2)               /* definitions for CON2 */
#define  ACE_DATA_REG        0x2f8   /* data register */
#define  ACE_INT_ENB_REG     0x2f9   /* interrupt enable register */
#define  ACE_INT_IDENT_REG   0x2fa   /* interrupt identification reg */
#define  ACE_LINE_CTL_REG    0x2fb   /* line control register */
#define  ACE_MODEM_CTL_REG   0x2fc   /* modem control register */
#define  ACE_LINE_STAT_REG   0x2fd   /* line status register */
#define  ACE_MODEM_STAT_REG  0x2fe   /* modem status register */
#define  COM_INT_NUM            11   /* interrupt number for COM2 */
#define  IRQ_MASK             0xf7   /* IRQ mask for IRQ3 (11110111) */
#endif

#define  THRE                 0x20   /* transmit holding reg empty */
#define  CTS                  0x10   /* clear to send */
#define  DSR                  0x20   /* delta set ready */
#define  RI                   0x40   /* ring indicator */
#define  DCD                  0x80   /* data carrier detect */
#define  PE                      4   /* parity error */
#define  FE                      8   /* framing error */
#define  OE                      2   /* overrun error */
#define  BI                   0x10   /* break interrupt */
#define  DCTS                    1   /* delta clear to send */
#define  DDSR                    2   /* delta data set ready */
#define  TERI                    4   /* trailing edge ring dete */
#define  DDCD                    8   /* delta data carrier detect */


/* LOCAL TYPES DECLARATION *********************************************/
```

```
/* LOCAL VARIABLES DEFINITION ******************************************/
static struct comm_s
{
  void interrupt (*old_vector)(); /* place to save old int. vector */
  struct
  {
    int  index;                 /* input index of comm buffer */
    int  rd_index;              /* output index of comm buffer */
    char queue[INPUT_QUEUELEN]; /* input buffer for input chars. */
  } input;
  struct
  {
    int break_detect;           /* inc on break detect */
    int char_overrun;           /* set true on character overrun */
    int frame_error;            /* count of framing errors */
    int parity_error;           /* count of parity errors */
  } status;
} comm;

struct srq_s
{
  int mode;                     /* SRQ on/off */
  int detect;                   /* inc if SRQ character detect */
  int rd_detect;                /* inc if SRQ detect checked */
  int overflow;                 /* set true on srq detect overflow */
} srq;


/* LOCAL CONSTANTS DEFINITION ******************************************/


/* LOCAL FUNCTIONS DEFINITION ******************************************/
/* FUNCTION ************************************************************/
static void menu_create(void)
/*
SPECIFICATION:
Creats the menu windows.
**********************************************************************/
{
  #define W_LEFT  10
  #define W_TOP    7
  int ch;

  /* clear the screen */
  textbackground(BLACK);
  clrscr();
  /* create frame window */
  window(W_LEFT-2, W_TOP-1, W_LEFT+62, W_TOP+12);
  textcolor(BLACK);
  textbackground(WHITE);
  clrscr();
  cprintf("                 D V M D  SRQ demonstration.\r\n");
  cprintf("\n");
  cprintf("       This is a demonstration of monitoring the DVRM\r\n");
  cprintf("       STATus:QUEStionable:DVRM register with SRQ.\r\n");
  cprintf("\n");
  cprintf("       The DVRM must be connected to COM%1d, with\r\n", COM_PORT);
  cprintf("       9600 baud, no parity, 8 databits and 1 stopbit.\r\n");
  cprintf("\n");
  cprintf("       !!! This program doesn't work under WINDOWS NT. !!!\r\n");
  cprintf("\n\n\n");
  cprintf("       >>> Press any key to continue\r\n");

  /* wait key pressed */
  ch = getch();
  ch = ch;
```

```
  cprintf("   Alt-S (toggle SRQ); Alt-E (COM%1d status);       Alt-X (exit)",
COM_PORT);
  window(W_LEFT-1, W_TOP, W_LEFT+61, W_TOP+11);
  textbackground(BLUE);
  clrscr();

  /* create message window */
  window(W_LEFT, W_TOP, W_LEFT+60, W_TOP+11);
  textcolor(WHITE);
  textbackground(BLUE);
  clrscr();
}

/* FUNCTION ***********************************************************/
static void menu_draw_comm_status(void)
/*
SPECIFICATION:
Displays the serial port status.
**********************************************************************/
{
  int ch;

  clrscr();

  if (srq.mode)
  {
    if (srq.overflow)
      cprintf("SRQ on: count = %d (overflow).\r\n", srq.detect);
    else
      cprintf("SRQ on: count = %d.\r\n", srq.detect);
  }
  else
    cprintf("SRQ off.\r\n");

  ch = inportb(ACE_MODEM_STAT_REG);   /* get current status */
  cprintf("COM%1d status\r\n", COM_PORT);
  cprintf("  DCD=%1d DSR=%1d CTS=%1d\r\n",
           (ch & DCD) ? 1:0, (ch & DSR) ? 1:0, (ch & CTS) ? 1:0);
  if (comm.status.break_detect ||
      comm.status.frame_error  ||
      comm.status.char_overrun ||
      comm.status.parity_error)
  {
    cprintf("  Errors:\r\n");  /* error report */
    cprintf("    overrun       = %4d\r\n", comm.status.char_overrun);
    cprintf("    frame error   = %4d\r\n", comm.status.frame_error );
    cprintf("    break detect  = %4d\r\n", comm.status.break_detect);
    cprintf("    parity error  = %4d\r\n", comm.status.parity_error);
    comm.status.break_detect =
    comm.status.frame_error =
    comm.status.char_overrun =
    comm.status.parity_error = 0;
  }
  else
  {
    cprintf("  No errors.\r\n");
  }
}

/* FUNCTION ***********************************************************/
static void menu_draw_status_questionable_DVRM(int status)
/*
SPECIFICATION:
Displays the DVRM status:questionable:DVRM register.
**********************************************************************/
{
  clrscr();

  if (srq.mode)
```

```
  {
    if (srq.overflow)
      cprintf("SRQ detected: count = %d (overflow).\r\n", srq.detect);
    else
      cprintf("SRQ detected: count = %d.\r\n", srq.detect);
  }
  else
    cprintf("SRQ off.\r\n");

  cprintf("STATus:QUEStionable:DVRM\r\n");
  cprintf("  %1d : TS SYNC\r\n",    ((status & 0x0001) ? 1:0));
  cprintf("  %1d : SYNC BYTE\r\n",  ((status & 0x0002) ? 1:0));
  cprintf("  %1d : PAT\r\n",        ((status & 0x0004) ? 1:0));
  cprintf("  %1d : CONT COUNT\r\n", ((status & 0x0008) ? 1:0));
  cprintf("  %1d : PMT\r\n",        ((status & 0x0010) ? 1:0));
  cprintf("  %1d : PID\r\n",        ((status & 0x0020) ? 1:0));
  cprintf("  %1d : TRANSPORT\r\n",  ((status & 0x0040) ? 1:0));
  cprintf("  %1d : CRC\r\n",        ((status & 0x0080) ? 1:0));
  cprintf("  %1d : OTHER\r\n",      ((status & 0x0100) ? 1:0));
}

/* FUNCTION *************************************************************/
static void menu_cleanup(void)
/*
SPECIFICATION:
Routine resets the serial port at exit time.
*************************************************************************/
{
  /* clear the whole screen */
  window(1, 1, 80, 25);
  textcolor(WHITE);
  textbackground(BLACK);
  clrscr();
  printf("\nprogram terminated.\n");
}

/* FUNCTION *************************************************************/
static void interrupt comm_isr(void)
/*
SPECIFICATION:
Interrupt service routine, executed when receive data is available,
or when line status or modem status changes.
*************************************************************************/
{
  char ch;
  int temp_index;

  enable();                         /* re-enable the other interrupts */

  ch = inportb(ACE_INT_IDENT_REG);  /* examine IIR for cause of int. */
  switch (ch)
  {
    case 0:   /* modem status interrupt */
      ch = inportb(ACE_MODEM_STAT_REG);   /* read modem status reg. */
      if (ch & DCTS)       /* clear to send changed */
      {
        /* a routine to handle CTS changes may go here */
      }
      if (ch & DDSR)       /* data set ready changed */
      {
        /* a routine to handle DSR changes may go here */
      }
      if (ch & TERI)       /* trailing edge ring indicator */
      {
        /* a routine to handle ring indicator may go here */
      }
      if (ch & DDCD)       /* data carrier detect changed */
      {
        /* a routine to handle DCD changes may go here */
```

```
      }
      break;

    case 4:   /* received data available */
      ch = inportb(ACE_DATA_REG);          /* read character from the ACE */
      if (ch == SRQ_CHAR)                  /* srq character ? */
      {
        srq.detect++;
        if (srq.detect < 0)
        {
          srq.detect = 0;
          srq.overflow = 1;
        }
      }
      else
      {                                    /* read character */
        temp_index = comm.input.index + 1;
        if (temp_index >= INPUT_QUEUELEN)  /* test queue end */
          temp_index = 0;
        if (temp_index != comm.input.rd_index)   /* test for queue overflow */
        {                                   /* if overflow, characters lost */
          comm.input.queue[temp_index] = ch;    /* put character into queue */
          comm.input.index = temp_index;
        }
      }
      break;

    case 6:   /* Line status interrupt */
      ch= inportb(ACE_LINE_STAT_REG);   /* read line status register */
      if (ch & PE)
        comm.status.parity_error++;
      if (ch & OE)
        comm.status.char_overrun++;
      if (ch & FE)
        comm.status.frame_error++;
      if (ch & BI)
        comm.status.break_detect++;
      inportb(ACE_DATA_REG);            /* read the data reg. to empty it */
      break;
    }

    /* finally, send the nonspecific EOI to re-enabte the PIC */
    outportb(PIC_CTL_REG, NON_SPEC_EOI);
}

/* FUNCTION ***********************************************************/
static void comm_install(void)
/*
SPECIFICATION:
Installs the interrupt service routine and initializes the 8250 for
active communications.
*********************************************************************/
{
  char ch;

  bioscom(0, COM_PARMS, COM_PORT - 1); /* set communications params. */

    /* enable ACE modem status, tine status & receive data interrupts */
  outportb(ACE_INT_ENB_REG, 0xd);      /* enable modem/line/rcvdata ints */
  inportb(ACE_DATA_REG);               /* empty receive data register */
  inportb(ACE_LINE_STAT_REG);          /* clear the line status register */

  comm.old_vector = getvect(COM_INT_NUM);  /* save previous int. vector */
  setvect(COM_INT_NUM, *comm_isr);         /* set vector to our routine */

  ch = inportb(PIC_INT_MASK_REG);      /* get current interrupt mask */
  ch &= IRQ_MASK;                      /* unmask our IRQ (set it to 0) */
  outportb(PIC_INT_MASK_REG, ch);      /* and write it back to the PIC */
```

```
  outportb(ACE_MODEM_CTL_REG, 0xb);    /* set RTS, DTR to enable modem */
  /* and turn on OUT2 to enable the 8250's IRQ interrupt to system */
  cprintf("COM%d initialized.\r\n", COM_PORT);   /* sign-on message */
}

/* FUNCTION ***********************************************************/
static void comm_cleanup(void)
/*
SPECIFICATION:
Routine resets the serial port at exit time.
**********************************************************************/
{
  char ch;
  outportb(ACE_INT_ENB_REG, 0);     /* disable all 8250 interrupts */
  outportb(ACE_MODEM_CTL_REG, 0);   /* clear RTS, DTR to disable modem */
  /* and turn off OUT2 to disable the 8250's IRQ int. to system */

  ch = inportb(PIC_INT_MASK_REG);   /* get current interrupt mask */
  ch |= !IRQ_MASK;                  /* mask our IRQ (set it to 1) */
  outportb(PIC_INT_MASK_REG, ch);   /* and write it back to the PIC */

  setvect(COM_INT_NUM, comm.old_vector);  /* restore previous int. vector */
}

/* FUNCTION ***********************************************************/
static int comm_getch(void)
/*
SPECIFICATION:
Read a character from the serial input buffer.

PARAMETERS:
RETURN VALUES:
Received Character.
**********************************************************************/
{
  char ch;
  int temp_index;   /* temporary index value avoids a conflict with */
                    /* interrupt routine updating the same variable */

  if (comm.input.rd_index != comm.input.index)
  {
    temp_index = comm.input.rd_index + 1; /* increment temporary index */
    if (temp_index >= INPUT_QUEUELEN)     /* test queue end */
      temp_index = 0;

    ch = comm.input.queue[temp_index];    /* read character from input queue */
    comm.input.rd_index = temp_index;

    return(ch);                           /* character is return value */
  }

  return(S_EOF);
}

/* FUNCTION ***********************************************************/
static int comm_putch(char ch)
/*
SPECIFICATION:
Output a character to serial transmitter.

PARAMETERS:
ch:  character

RETURN VALUES:
0 = ok.
**********************************************************************/
{
  #define CP_TIMEOUT  100  /* 1 = 10ms */
  int timeout;
```

```
  timeout = CP_TIMEOUT;
  while (timeout)
  {
    if ((inportb(ACE_LINE_STAT_REG) & THRE) &&   /* buffer empty? */
        (inportb(ACE_MODEM_STAT_REG) & CTS))     /* clear to send */
    {
      break; /* ok */
    }
    delay(10);  /* ms */
    timeout--;
  }
  if (timeout)
  {
    outportb(ACE_DATA_REG, ch);    /* send char to the 8250 */
    return(0);
  }

  return(1);
}

/* FUNCTION *************************************************************/
static int DVRM_write_command(char *cmdstring)
/*
SPECIFICATION:
Send to serial transmitter the DVRM cmdstring.

PARAMETERS:
RETURN VALUES:
0 = ok:
***********************************************************************/
{
  DISABLE;
  comm.input.rd_index = comm.input.index;   /* clear input buffer */
  ENABLE;

  while (*cmdstring)
  {
    if (comm_putch(*cmdstring) != 0)
      return(1);
    cmdstring++;
  }
  comm_putch('\r');
  comm_putch('\n');

  return(0);
}

/* FUNCTION *************************************************************/
static int DVRM_query_int_command(char *cmdstring, int *pValue)
/*
SPECIFICATION:
Send to serial transmitter the query cmdstring and wait for response.

PARAMETERS:
RETURN VALUES:
0 = ok
***********************************************************************/
{
  #define DQIC_TIMEOUT  1000   /* 1 = 10ms */
  int i, error, timeout;
  char ch, qstring[12];

  *pValue = 0;
  error = 0;
  qstring[0] = 0;

  error = DVRM_write_command(cmdstring);
```

```
  /* read query data */
  timeout = DQIC_TIMEOUT;
  i = 0;
  ch = 0;
  while(timeout)
  {
    ch = comm_getch();
    switch (ch)
    {
      case S_EOF:
        delay(10);
        timeout--;
        break;
      case '\n':
        qstring[i++] = 0;
        timeout = 0;
        break;
      default:
        qstring[i++] = ch;
        timeout = DQIC_TIMEOUT;
        break;
    }
  }

  if (ch == '\n')
  {
    if (sscanf(qstring, "%d", pValue))
      error = 0;
    else
      error = 1;
  }

  return(error);
}

/* FUNCTION *************************************************************/
static void DVRM_setup_status_questionable_srq(void)
/*
SPECIFICATION:
Routine enables the status:questionable SRQ.
*************************************************************************/
{
  DVRM_write_command("*SRE 0");
  DVRM_write_command("*ESE 0");                    /* mask, not used */
  DVRM_write_command("STAT:OPER:ENAB 0");          /* mask, not used */

  DVRM_write_command("STAT:QUES:NTR 0; PTR 32767");
  DVRM_write_command("STAT:QUES:ENAB 8192");

  DVRM_write_command("STAT:QUES:DVRM:NTR 32767; PTR 32767");
  DVRM_write_command("STAT:QUES:DVRM:ENAB 511");

  DVRM_write_command("*CLS");

  srq.detect = srq.rd_detect = srq.overflow = 0;
}

/* FUNCTION *************************************************************/
static void DVRM_srq_on(void)
/*
SPECIFICATION:
Routine enables the SRQ monitoring.
*************************************************************************/
{
  DVRM_write_command("*SRE 8");
}

/* FUNCTION *************************************************************/
static void DVRM_srq_off(void)
```

```
/*
SPECIFICATION:
Routine disables the SRQ monitoring.
***********************************************************************/
{
  DVRM_write_command("*SRE 0");
}

/* FUNCTION ************************************************************/
static void DVRM_check_srq_event(void)
/*
SPECIFICATION:
Routine signals an DVRM SRQ event, if a new SRQ is detected.
***********************************************************************/
{
  int status, error, success;

  error =
  success = 0;
  if (srq.detect != srq.rd_detect)
  {
    DVRM_srq_off();
    /* read STATUS BYTE */
    if ((error = DVRM_query_int_command("*STB?", &status)) == 0)
    {
      if (status & 0x48)
      {
        /* read STATUS:QUESTIONABLE:EVENT */
        if ((error = DVRM_query_int_command("STAT:QUES?", &status)) == 0)
        {
          if (status & 0x2000)
          {
            /* read STATUS:QUESTIONABLE:DVRM:EVENT */
            if ((error = DVRM_query_int_command("STAT:QUES:DVRM?", &status)) == 0)
            {
              if (status)
              {
                /* draw STATUS:QUESTIONABLE:DVRM:CONDITION */
                if ((error = DVRM_query_int_command("STAT:QUES:DVRM:COND?", &status))
== 0)
                {
                  menu_draw_status_questionable_DVRM(status);
                  success = 1;
                }
              }
            }
          }
        }
      }
    }
    if (!success)
    {
      clrscr();
      if (srq.overflow)
        cprintf("unknown SRQ detected: count = %d (overflow).\r\n", srq.detect);
      else
        cprintf("unknown SRQ detected: count = %d.\r\n", srq.detect);
      if (error)
        cprintf("QueryError\r\n");
    }
    srq.rd_detect = srq.detect;
    if (srq.mode)
      DVRM_srq_on();
  }
}

/* FUNCTION ************************************************************/
static int DVRM_search_device(void)
/*
```

```
SPECIFICATION:
Reading the Idnstring from the DVRM.

RETURNVALUE:
0:= DVRM found.
1:= no DVRM found.
************************************************************************/
{
  #define DSD_TIMEOUT  100   /* 1 = 10ms */
  int i, error, timeout;
  char ch, qstring[32];

  error = 1;
  qstring[0] = 0;

  cprintf("searching DVRM:\r\n");
  if (DVRM_write_command("*IDN?") == 0)
  {
    /* read query data */
    timeout = DQIC_TIMEOUT;
    i = 0;
    ch = 0;
    while(timeout)
    {
      ch = comm_getch();
      switch (ch)
      {
        case S_EOF:
          delay(10);
          timeout--;
          break;
        case '\n':
          qstring[i++] = 0;
          timeout = 0;
          break;
        default:
          qstring[i++] = ch;
          timeout = DSD_TIMEOUT;
          break;
      }
    }
    if (ch == '\n')
    {
      if (strlen(&qstring[0]) > 0)
      {
        cprintf("found         : %s\r\n", &qstring[0]);
        error = 0;
      }
    }
  }

  if (error)
    cprintf("!!!no DVRM found!!!\r\n");

  return(error);
}

/* FUNCTION ************************************************************/
static void main_init(void)
/*
SPECIFICATION:
Initialize the global variables.
************************************************************************/
{
  comm.old_vector = NULL;
  comm.input.index = 0;
  comm.input.rd_index = 0;
  comm.input.queue[0] = 0;
  comm.status.break_detect = 0;
```

```
  comm.status.char_overrun = 0;
  comm.status.frame_error = 0;
  comm.status.parity_error = 0;

  srq.mode = 0;
  srq.detect = 0;
  srq.rd_detect = 0;
  srq.overflow = 0;
}

/* FUNCTION ************************************************************/
static void main_switch_srq_mode(int mode)
/*
SPECIFICATION:
Handles the switch of SRQ mode.
PARAMETERS:
mode:           0:= SRQ mode off
                1:= SRQ mode on
                2:= SRQ mode toggle
***********************************************************************/
{
  int changeFlag = 0;

  switch (mode)
  {
     case 0:  /* off */
       if (srq.mode)
       {
         srq.mode = 0;
         changeFlag = 1;
       }
       break;
     case 1:  /* on */
       if (!srq.mode)
       {
         srq.mode = 1;
         changeFlag = 1;
       }
       break;
     case 2:  /* toggle */
       srq.mode = !srq.mode;
       changeFlag = 1;
       break;
  }

  if (changeFlag)
  {
    clrscr();
    if (srq.mode)
    {
      cprintf("SRQ on.\r\n");
      DVRM_setup_status_questionable_srq();
      DVRM_srq_on();
    }
    else
    {
      cprintf("SRQ off.\r\n");
      DVRM_srq_off();
    }
  }
}


/* GLOBAL FUNCTIONS DEFINITION ******************************************/
/* FUNCTION ************************************************************/
main()
/*
SPECIFICATION:
```

```
Main program initiatizes the hardware, then enters the main loop.
*************************************************************************/
{
  int ch, iValue;

  main_init();

  menu_create();
  comm_install();         /* initiatize COM port */

  atexit(menu_cleanup);  /* post removal routine to execute at program end */
  atexit(comm_cleanup);

  if (DVRM_search_device())
  {
    /* wait key pressed */
    ch = getch();
    exit(0);      /* no DVRM found, terminate the program */
    return;
  }

  while(1)              /* main program Loop */
  {
    if (kbhit())        /* check if key has been hit */
    {
      ch = getch();
      if (ch == 0)      /* = 0 next is scan code */
        ch = getch();
      switch (ch)
      {
        case 0x12:      /* Alt-E: print comm status */
          main_switch_srq_mode(0);
          menu_draw_comm_status();
          break;
        case 0x1f:      /* Alt-S toggle SRQ */
          main_switch_srq_mode(2);
          break;
        case 0x20:      /* Alt-D display status:questionable:DVRM */
          main_switch_srq_mode(0);
          if (!DVRM_query_int_command("STAT:QUES:DVRM:COND?", &iValue))
            menu_draw_status_questionable_DVRM(iValue);
          else
          {
            clrscr();
            cprintf("QueryError\r\n");
          }
          break;
        case 0x2d:      /* Alt-X  EXIT */
          main_switch_srq_mode(0);
          exit(0);      /* terminate the program */
        default:
          break;
      }
    }

    DVRM_check_srq_event();
  }
}
/* end of program */
```

5 Maintenance and Checking 5.1
5.1 Maintenance 5.1
5.1.1 Cleaning the Outside 5.1
5.1.2 Storage 5.1
5.1.3 Replacing the Battery 5.2
5.2 Checking 5.6

5 Maintenance and Checking

5.1 Maintenance

The instrument requires no regular maintenance. Generally, maintenance is confined to cleaning the instrument and possibly changing the battery. Make sure in particular that air inlets and outlets are not contaminated and clean them at regular intervals.

5.1.1 Cleaning the Outside

The outside of the instrument should best be cleaned with a soft, lint-free cloth or a brush. In the case of heavier contamination, use spirit or mild soap suds for cleaning.

Caution: *In no case use solvents like nitro thinners, acetone, etc, to avoid damaging of front-panel labels or plastic parts.*

Furthermore, any dust collected inside the instrument should be removed at regular intervals to ensure unimpaired cooling (approx. every 1 to 2 years depending on the daily on-time of the equipment and the amount of dust in the environment).
Cleaning the Inside

5.1.2 Storage

Storage temperature of instrument:

- 40 to +70 °C.

When stored for an extended period of time the instrument should be protected against dust.

5.1.3 Replacing the Battery

Caution:

The battery of the instrument is a high-power lithium cell. Avoid by all means short circuiting and charging the battery, a this might cause the battery to explode. Do not open discharged batteries and dispose them as HAZARDOUS WASTE.

The instrument comprises a lithium battery for storing the selected status and for operation of the real-time clock. The battery has a lifetime of approx. 5 years and should be replaced as required (eg after a longer storage period at high temperature). Replace the discharged battery by one of the same type.

For replacing the battery the instrument must be opened.

Tools required:

- Phillips screwdrivers, size 0, 1 and 2
- Soldering iron

Material required:

- Lithium battery, 3 V (eg: SAFT LS3 CNA R&S Order No. 0565.1687)
- Cable tie (width 2 mm)

Opening the instrument

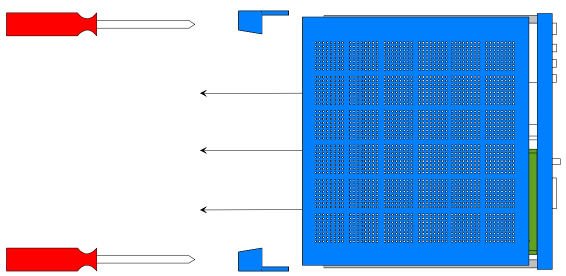


Fig. 5-1 Opening the instrument

Caution:

The instrument contains electrostatic-sensitive components. Any work required should therefore be carried out on a grounded work position.

Disconnect power plug before opening the instrument

- ➢ Switch off instrument and disconnect power plug.
- ➢ Unscrew and remove the two rear-panel feet (2 Philips screws each).
- ➢ Take off top and bottom covers towards the rear.

Removing the front panel:

- Undo the 2 trim screws on the front panel.
- Undo the 4 fixing screws.
- Remove front panel.

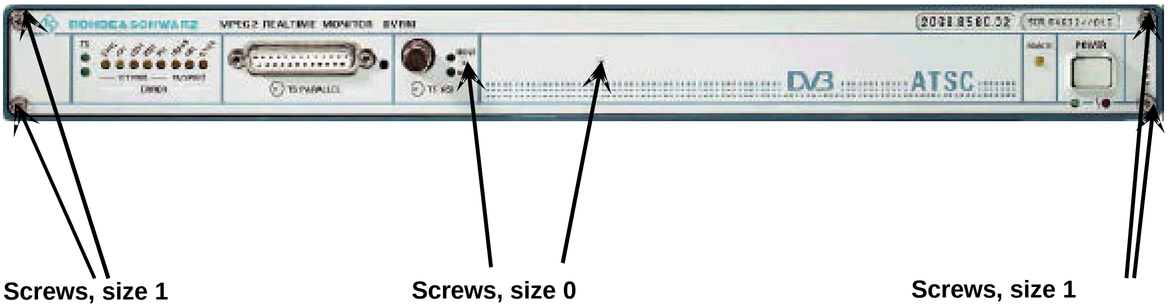


Fig. 5-2 Front-panel screws

Removing the module:

- Disconnect the two flat cables from X8 of the motherboard (W3, wide cable) and from the cable link (W2, narrow cable).
- Push module out of the frame towards the front.

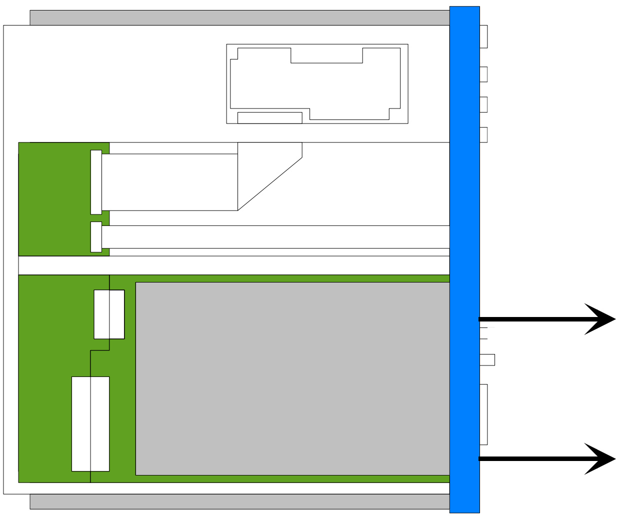


Fig. 5-3 Removing the module

Opening the module

- Unscrew top RF cover of generator board. Lift off the shielding cover and disconnect the fan cable from multipoint connector X28.
- Turn instrument upside down.
- Unscrew lower RF cover of generator board and remove it.
- Turn instrument round again.

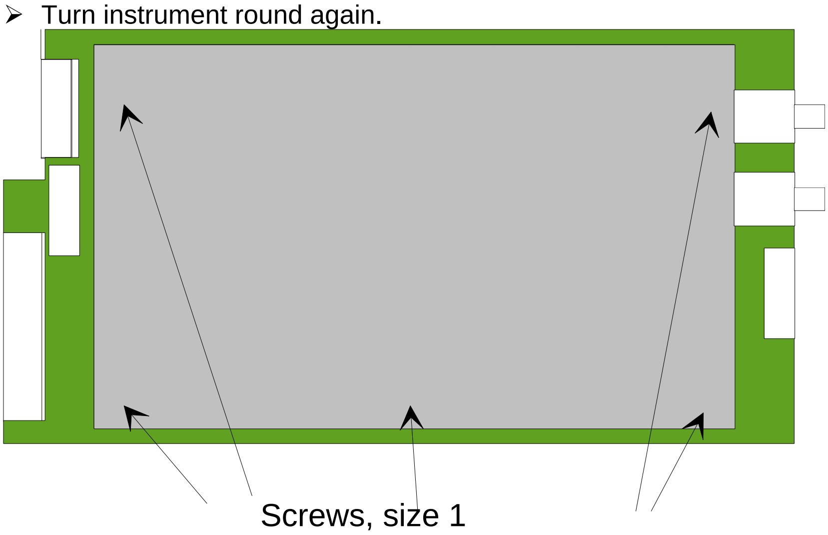


Fig. 5-4 Fixing RF cover

Removing the plug-on 'Alarm contacts' PCB

- Evenly undo the four screws of the 'Alarm contacts' PCB. (see Fig. 6-5).

Caution: *When the screws are evenly loosened, the PCB automatically rises out of its position. The PCB will be twisted if only one screw is tight.*

Tighten screws evenly to refit the PCB.

- Pull off plug-on PCB completely from the multipoint connector.

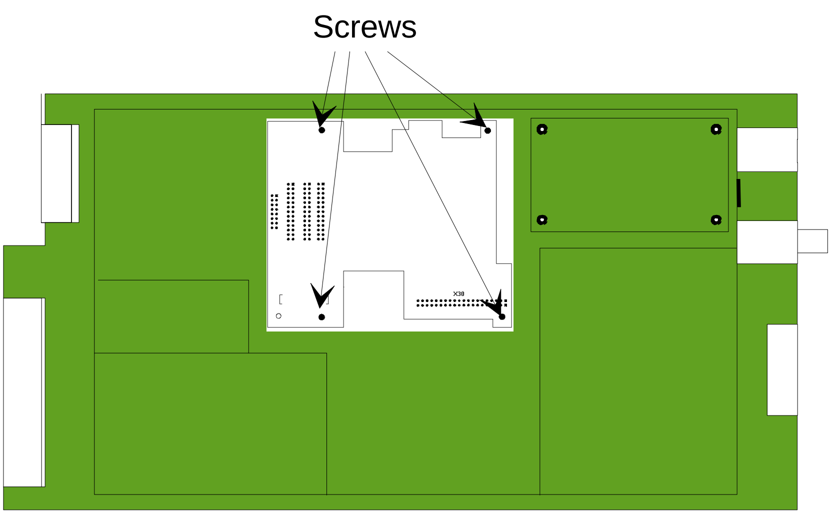


Fig. 5-5 Removing the 'Alarm contacts' PCB

Replacing the battery

Replace the battery properly (solder lugs and fix with cable tie). Make sure that the poles are correctly connected. They are marked on the battery with "+" and "-". For position and polarity of the battery on the decoder board see drawing below.

- Open the cable tie holding the battery using side cutters.
- Unsolder battery terminals (negative pole first).
- Cut terminals of the new battery as required and solder them (positive pole first).
- Insert new cable tie through openings and tighten.

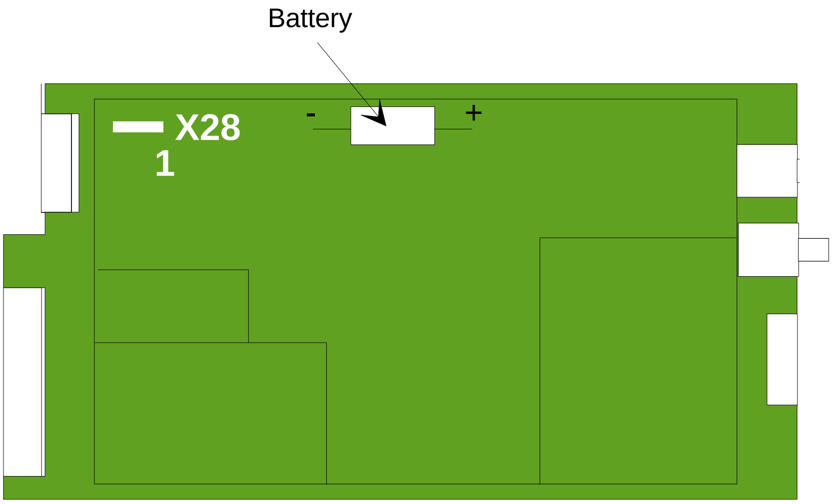


Fig. 5-6 Position and poles of battery

Closing the instrument:

- Position the 'Alarm contacts' PCB and tighten the four screws evenly (see Fig. 6-5).
- Connect fan cable (fixed to RF cover) to multipoint connector X28 (blue to pin 5 / red to pin 7 / pin 1 next to label X28).
- Put on upper RF cover of generator module and screw it on. Make sure that the fan cable is not caught between the RF cover and a shielding strip.
- Insert the module from the front into the frame and connect the two flat cables to X8 of the motherboard (W3, wide cable) and to the cable link (W2, narrow cable).
- Slide on top cover from the rear.
- Put instrument upside down.
- Put lower RF cover onto decoder board and screw it on.
- Slide on bottom cover from the rear.
- Put back feet on rear panel and screw them on.

After this date and time have to be newly set.

➢ Switch on instrument.
➢ Set date and time in the SETUP menu.

```
SETUP    |-14-MAR-00
DATE+TIME| 10:32:21
```

⇑⇓	selecting DATE and TIME
⇐⇒	selecting individual elements
ENT	entry mode
ESC	return to main menu

5.2 Checking

Upon power-up, the instrument starts a BOOT procedure with a memory and hardware check. If a fault is detected, it is indicated by the LEDs at the front panel (see section 2.3, Switch-on Test).

In the case of a total failure send the instrument to your local Rohde & Schwarz service center.

Annex A Interfaces A.1

A.1 Synchronous Parallel Transport Stream Input - TS PARALLEL A.1
A.1.1 Pin Assignment of TS PARALLEL A.1
A.1.2 Transmission Formats - TS PARALLEL A.2
A.1.3 Data Rate Measurement A.3

A.2 Asynchronous Serial Transport Stream Input - TS ASI A.5

A.3 RS232Interface - COM1 .... A.6
A.3.1 Pin Assignment A.6
A.3.2 Transmission Parameters A.7
A.3.3 Interface Functions A.8
A.3.4 Handshake A.8

A.4 ALARM LINES Interface A.10
A.4.1 Pin Assignment A.10

A.5 Interfaces for Descrambling and Flash ROM Card A.10
A.5.1 Pin Assignment of Flash ROM Card A.11
A.5.2 Pin Assignment of Descrambling Card A.12

A.6 Parallel Printer Interface PRINTER A.13

Annex A Interfaces

A.1 Synchronous Parallel Transport Stream Input - TS PARALLEL

A 25-pin sub-D connector designated TS PARALLEL is provided on the front panel of the unit for connecting an MPEG2 transport stream with synchronous parallel data transmission (8-bit parallel).

A.1.1 Pin Assignment of TS PARALLEL

The pin assignment is in line with DVB DOCUMENT A010 (synchronous parallel interface) :

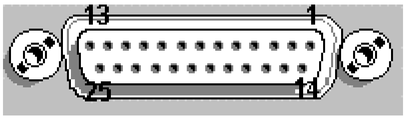


Fig. A-1 Pin assignment of TS PARALLEL

Pin	Abbreviation	Signal
1	CLOCK A	Clock for data word
2	GND	Ground
3	DATA BIT 7 A (MSB)	Data bit 7 (most significant bit)
4	DATA BIT 6 A	Data bit 6
5	DATA BIT 5 A	Data bit 5
6	DATA BIT 4 A	Data bit 4
7	DATA BIT 3 A	Data bit 3
8	DATA BIT 2 A	Data bit 2
9	DATA BIT 1 A	Data bit 1
10	DATA BIT 0 A (LSB)	Data bit 0 (least significant bit)
11	DVALID A	Data word valid
12	PSYNC A	Packet sync
13	GND	Ground

Pin	Abbreviation	Signal
14	CLOCK B	Clock for data word, inverted
15	GND	Ground
16	DATA BIT 7 B (MSB)	Data bit 7 inverted (most significant bit)
17	DATA BIT 6 B	Data bit 6 inverted
18	DATA BIT 5 B	Data bit 5 inverted
19	DATA BIT 4 B	Data bit 4 inverted
20	DATA BIT 3 B	Data bit 3 inverted
21	DATA BIT 2 B	Data bit 2 inverted
22	DATA BIT 1 B	Data bit 1 inverted
23	DATA BIT 0 B (LSB)	Data bit 0 inverted (least significant bit)
24	DVALID B	Data word valid, inverted
25	PSYNC B	Packet sync inverted

Table A-1 Pin assignment of TS PARALLEL (in line with DVB DOCUMENT A010)

A.1.2 Transmission Formats - TS PARALLEL

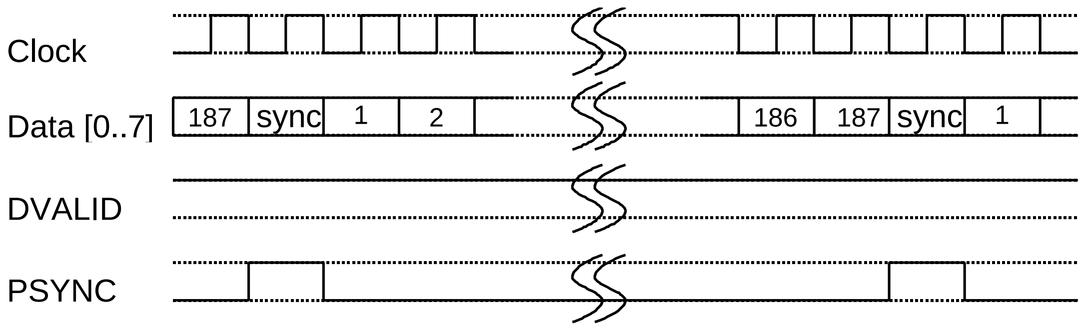


Fig. A-2 Transmission format with packets of 188 bytes

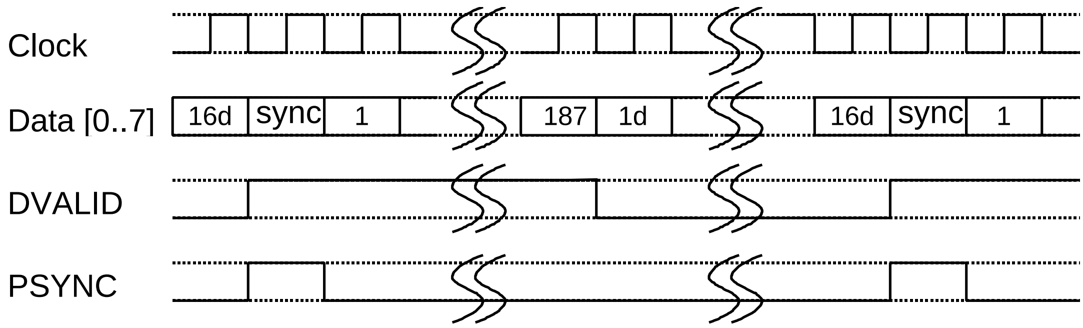


Fig. A-3 Transmission format with packets of 204 bytes (188 bytes payload and 16 dummy bytes)

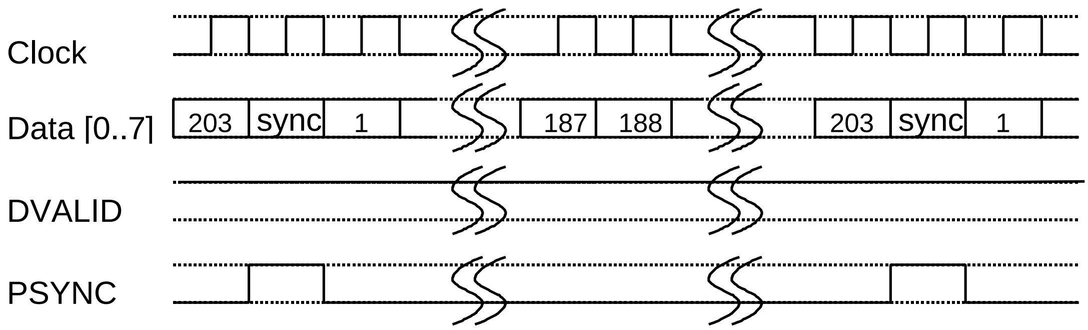


Fig. A-4 Transmission format with packets of 204 bytes (188 bytes payload and 16 extra bytes)

The three transmission formats shown in Figs. A-2 to A-4 are also supported for partial transport streams (see ETR 154 rev 2, section 4.2.1.). There are gaps between byte 187 (A-2) or 203 (A-3, A-4) and sync byte, these gaps being marked by the deactivated signal DVALID (zero level).

A.1.3 Data Rate Measurement

The following explanations are valid for the measurement of the TS data rate as well as of the gross elementary streams (total, see MENU SETUP/DATA RATE FORMAT). The TS packet length does not have any influence on the measurement of the net data rate.
DVRM supports the three data formats specified in EN50083-9 (DVB-A010) (see Figs A5 to A9) and also the format of the partial TS described in ETR154 (section 4.2.1) (see Figs A8 and A9). Fig. A6 represents a special case of Fig. A8 (X = 16). Therefore, a receiver supporting partial TS cannot recognize the transmission format shown in Fig. 2 as a format with a packet length of 204 bytes but as a partial TS with a packet length of 188 bytes.

DVRM takes into account all valid data indicated by the DVALID signal in all transmission formats for the calculation of the TS data rate.

Note concerning TS Generator DVG (R&S 2068.8600):
DVG provides the transmission format as shown in Fig. 2 at the parallel output in the 204 bytes mode and takes into account 204 bytes for each TS packet on determining the data rate. Since DVRM measures the data rate as explained above because of the support of partial TS a deviation by the factor 188/204 is obtained in this case.

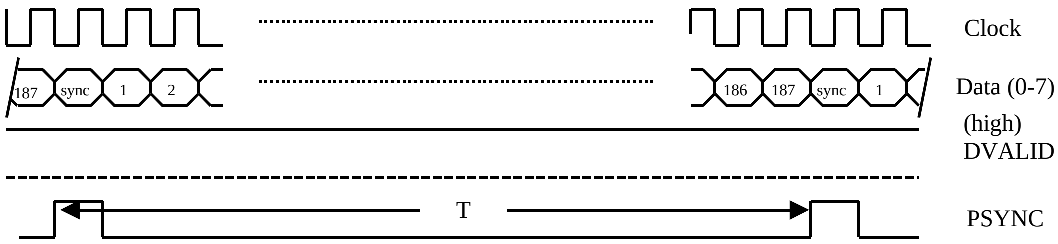


Fig. A-5 Transmission format with a packet length of 188 bytes
TS rate $_{DVRM, DVG}$ = 188 x 8 bis / T

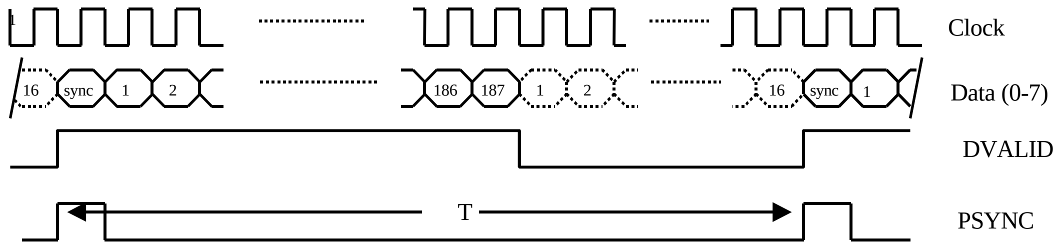


Fig. A-6 Transmission format with a packet length of 204 bytes
(188 data bytes and 16 dummy bytes)
TS rate $_{DVRM}$ = 188 x 8 bits / T TS rate $_{DVG}$ = 204 x 8 bits / T

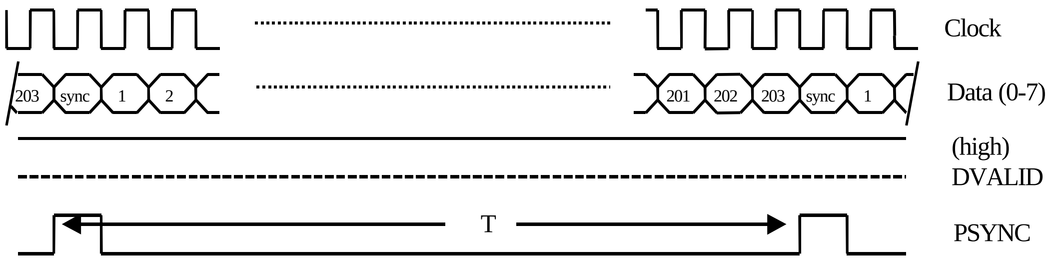


Fig. A-7 Transmission format with a packet length of 204 bytes
(188 data bytes and 16 valid additional bytes)
TS rate $_{DVRM}$ = 204 x 8 bits / T (Format for DVRM not applicable)

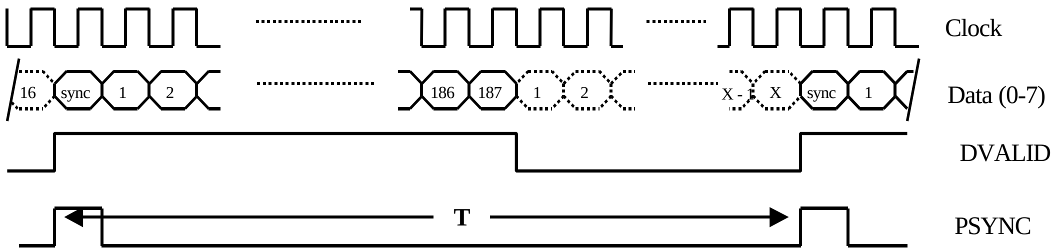


Fig. A-8 Partial TS with a packet length of 188 bytes and x dummy bytes between TS packets
TS rate $_{DVRM}$ = 188 x 8 bits / T (Format for DVRM not applicable)

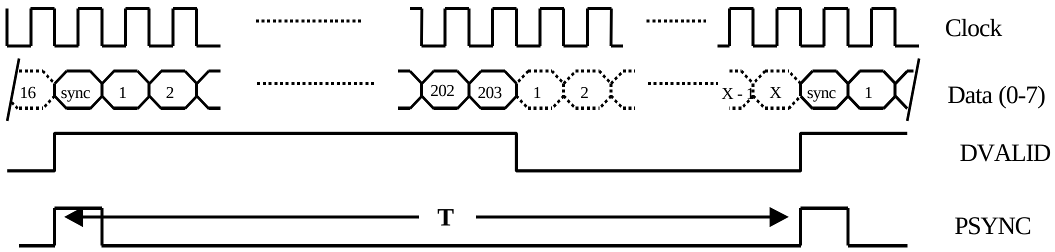


Fig. A-9 Partial TS with a packet length of 204 bytes and x dummy bytes between TS packets
TS rate $_{DVRM}$ = 204 x 8 bits / T (Format for DVRM not applicable)

Definition of signals:

Data (0 to 7): TS data word (8 bits: data 0 to data 7). Data 7 is the most significant bitt.

DVALID: High active, indicates valid data. This signal is constantly high in the 188 bytes mode In the 204 bytes mode as well as with partial TS, the signal with level 0 indicates invalid data.

PSYNC: High active, indicates the beginning of a TS packet.

A.2 Asynchronous Serial Transport Stream Input - TS ASI

A BNC connector each designated by TS - ASI in line with DVB DOCUMENT A010 (Asynchronous Serial Interface) is provided at the front and rear panel of the unit for connecting an MPEG2 transport stream with asynchronous serial data transmission.

The asynchronous serial transport stream interface has a constant data rate of 270 Mbit/s. Data bytes (8 bits) are transmitted via this interface at a data rate of max. 27 Mbyte/s. According to a standard table, two bits are added to each byte for marking non-relevant data bytes (dummy bytes) required to obtain the 27-Mbyte/s data rate as well as for preventing DC components in the serial signal.

The BNC connector has an input impedance of 75 Ω. The input level of a standard-conformant signal is 800 mV (±10%).

For the calculation of the TS data rate as well as of the gross elementary streams (total, see 4.3.10.5) 188 bytes or 204 bytes are taken into account for each TS packet depending on the TS packet format. The TS packet length does not have any influence on the measurement of the net data rate.

A.3 RS232Interface - COM1

The serial interface COM1 of DVRM is either used for remote control or for the printer.

A.3.1 Pin Assignment

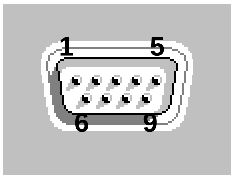


Fig. A-10 Pin assignment of COM1

Pin	Abbreviation	Signal
1	CD	CARRIER DETECT
2	RXD	RECEIVE DATA
3	TXD	TRANSMIT DATA
4	DTR	DATA TERMINAL READY
5	GND	GROUND
6	DSR	DATA SET READY
7	RTS	REQUEST TO SEND
8	CTS	CLEAR TO SEND
9	-	not assigned

Table A-2 Pin assignment of COM1

A.3.2 Transmission Parameters

The transmission parameters are set in the SETUP / RS232 menu (see section 3.4.8.2).

The following interface parameters can be selected:

Transmission rate (baud)
1 200
2 400
4 800
9 600
19 200
38 400
57 600
115 200

Table A-3 Setting the transmission rate for COM1

Parity	Meaning
none	No parity check
even	Even parity
odd	Odd parity

Table A-4 Setting the parity of COM1

Number of data bits
8

Table A-5 Data bit setting for COM1 (fixed)

Number of stop bits
1

Table A-6 Stop bit setting for COM1 (fixed):

Pace	Meaning
none	No specific interface protocol
XON/XOFF	XON/XOFF protocol
RTS/CTS	Hardware handshake

Table A-7 Pace setting for COM1

A.3.3 Interface Functions

To control the interface, some control characters are taken from the ASCII code range 0 to 20 hex. The control characters are transmitted via the interface.

Name	Control character	Function
SRQ_char	<Ctrl B> 02 hex	Service request (SRQ) If bit 6 of the status byte is set, this character will be sent. Receipt of SRQ_char will be ignored.
XON_char	<Ctrl Q> 11 hex	Enable character output XON_char informs the remote station that it may again transmit data.
XOFF_char	<Ctrl S> 13 hex	Stop character output XOFF_char informs the remote station that it is not to send data any more.
END_char	0Dhex, 0Ahex	Terminator <CR><LF>

Table A-8 Control characters of the RS-232 interface

A.3.4 Handshake

Software handshake

In case of software handshake data transmission is controlled via the two control characters XON_char / XOFF_char. DVRM signals its ready-to-receive state by means of the control character XON_char. If the receive buffer is full, DVRM informs the controller by means of control character XOFF_char that it is not to send more data. The controller will interrupt the data output until it receives another XON_char from DVRM. The controller signals its ready-to-receive state to DVRM in the same way.

During the transmission of block data a software handshake cannot take place because of the above-mentioned reasons.

Interconnections with local controller in case of software handshake

In case of software handshake, DVRM is connected to the controller according to the following wiring diagram. The wiring diagram applies to a controller with 9-pin or 25-pin connector.

DVRM 9 pins		Controller 9 pins
1		1
2	------------RxD / TxD---------------	3
3	------------TxD / RxD---------------	2
4		6
5	------------GND / GND-------------	5
6		4
7		8
8		7
9		9

DVRM 9 pins		Controller 25 pins
1		8
2	------------RxD / TxD---------------	2
3	------------TxD / RxD---------------	3
4		6
5	------------GND / GND-------------	7
6		20
7		5
8		4
9		22

Fig. A-11 Wiring for software handshake

Hardware handshake

In case of hardware handshake, DVRM signals its ready-to-receive state via the RTS lines. A logic '0' (active) means "ready" and a logic '1' means "not ready".

DVRM is informed about the ready-to-receive state of the remote station via the CTS line. A logic '0' activates the output of data, a logic '1' tells DVRM to stop sending data. The DTR line remains active (logic '0') as long as the serial interface is operated as a remote-control interface.

Interconnections with local controller in case of hardware handshake

DVRM is connected to the controller by means of a null modem cable. When this cable is used, the data, control and signalling lines have to be crossed. The following wiring diagram applies to a controller with 9-pin or 25-pin connector.

DVRM 9 pins		Controller 9 pins
1		1
2	------------RxD / TxD--------------	3
3	------------TxD / RxD--------------	2
4	------------DTR /DSR--------------	6
5	------------GND / GND------------	5
6	------------DSR / DTR-------------	4
7	------------RTS / CTS--------------	8
8	------------CTS / RTS--------------	7
9		9

DVRM 9 pins		Controller 25 pins
1		8
2	------------RxD / TxD--------------	2
3	------------TxD / RxD--------------	3
4	------------DTR /DSR--------------	6
5	------------GND / GND------------	7
6	------------DSR / DTR-------------	20
7	------------RTS / CTS--------------	5
8	------------CTS / RTS--------------	4
9		22

Fig. A-12 Wiring for hardware handshake

A.4 ALARM LINES Interface

A.4.1 Pin Assignment

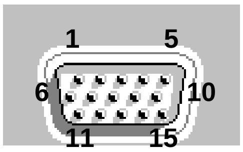


Fig. A-13 Pin assignment of ALARM LINES

Pin	Abbreviation	Signal
1	A01	Alarm line 1
2	A02	Alarm line 2
3	A03	Alarm line 3
4	A04	Alarm line 4
5	A05	Alarm line 5
6	A06	Alarm line 6
7	A07	Alarm line 7
8	A08	Alarm line 8

Pin	Abbreviation	Signal
9	A09	Alarm line 9
10	A10	Alarm line 10
11	A11	Alarm line 11
12	A12	Alarm line 12
13	GND	Ground
14	GND	Ground
15	VCC	+5 V into 25 Ω

Table A-9 Pin assignment of ALARM LINES

A.5 Interfaces for Descrambling and Flash ROM Card

These two interfaces can be accessed by opening the unit. The flash ROM card is always inserted into the bottom one of the two slots immediately on top of each other. This card contains the complete firmware of DVRM. The descrambling card is optionally inserted in the top slot. The programs or elementary streams contained in a transport stream may be scrambled. After the descrambling card required for these programs has been fitted, the instrument is able to decode the scrambled programs and deliver them at the appropriate video outputs. The interfaces for the descrambling card and the flash ROM card are in line with the PCMCIA standard (Personal Computer Memory Card International Association).

A.5.1 Pin Assignment of Flash ROM Card

Pin	Signal	Function
1	GND	Ground
2	D3	Data bit 3
3	D4	Data bit 4
4	D5	Data bit 5
5	D6	Data bit 6
6	D7	Data bit 7
7	CE1	Card Enable 1
8	A10	Address bit 10
9	OE	Output Enable
10	A11	Address bit 11
11	A9	Address bit 9
12	A8	Address bit 8
13	A13	Address bit 13
14	A14	Address bit 14
15	WE/PGM	Write Enable
16	RDY/BSY	Ready/Busy (EEPROM)
17	VCC	Supply voltage +5V
18	VPP1	Programming voltage 1
19	A16	Address bit 16
20	A15	Address bit 15
21	A12	Address bit 12
22	A7	Address bit 7
23	A6	Address bit 6
24	A5	Address bit 5
25	A4	Address bit 4
26	A3	Address bit 3
27	A2	Address bit 2
28	A1	Address bit 1
29	A0	Address bit 0
30	D0	Data bit 0
31	D1	Data bit 1
32	D2	Data bit 2
33	WP	Write Protect
34	GND	Ground

Pin	Signal	Function
35	GND	Ground
36	CD1	Card Detect 1
37	D11	Data bit 11
38	D12	Data bit 12
39	D13	Data bit 13
40	D14	Data bit 14
41	D15	Data bit 15
42	CE2	Card Enable 2
43	RFSH	Refresh (DRAM)
44	RFU	Reserved
45	RFU	Reserved
46	A17	Address bit 17
47	A18	Address bit 18
48	A19	Address bit 19
49	A20	Address bit 20
50	A21	Address bit 21
51	VCC	Supply voltage +5V
52	VPP2	Programming voltage 2
53	A22	Address bit 22
54	A23	Address bit 23
55	A24	Address bit 24
56	A25	Address bit 25
57	RFU	Reserved
58	RFU	Reserved
59	RFU	Reserved
60	RFU	Reserved
61	REG	Register Select
62	BVD2	Battery Voltage Detect 2
63	BVD1	Battery Voltage Detect 1
64	D8	Data bit 8
65	D9	Data bit 9
66	D10	Data bit 10
67	CD2	Card Detect 2
68	GND	Ground

Table A-10 Pin assignment of flash ROM card

A.5.2 Pin Assignment of Descrambling Card

Pin	Signal	Function
1	GND	Ground
2	D3	Data bit 3
3	D4	Data bit 4
4	D5	Data bit 5
5	D6	Data bit 6
6	D7	Data bit 7
7	CE1	Card Enable 1
8	A10	Address bit 10
9	OE	Output Enable
10	A11	Address bit 11
11	A9	Address bit 9
12	A8	Address bit 8
13	A13	Address bit 13
14	A14	Address bit 14
15	WE/PGM	Write Enable
16	IREQ	Interrupt Request
17	VCC	Supply voltage +5V
18	VPP1	Programming voltage 1
19	MIVAL	MPEG Input Data Valid
20	A15	Address bit 15
21	A12	Address bit 12
22	A7	Address bit 7
23	A6	Address bit 6
24	A5	Address bit 5
25	A4	Address bit 4
26	A3	Address bit 3
27	A2	Address bit 2
28	A1	Address bit 1
29	A0	Address bit 0
30	D0	Data bit 0
31	D1	Data bit 1
32	D2	Data bit 2
33	RFU	Reserved
34	GND	Ground

Pin	Signal	Function
35	GND	Ground
36	CD1	Card Detect 1
37	MDO3	MPEG Data Out Bit 3
38	MDO4	MPEG Data Out Bit 4
39	MDO5	MPEG Data Out Bit 5
40	MDO6	MPEG Data Out Bit 6
41	MDO7	MPEG Data Out Bit 7
42	CE2	Card Enable 2
43	MCLK	MPEG Clock
44	IORD	I/O Read
45	IOWR	I/O Write
46	MISTRT	MPEG Input Data Start
47	MDI0	MPEG Data In Bit 0
48	MDI1	MPEG Data In Bit 1
49	MDI2	MPEG Data In Bit 2
50	MDI3	MPEG Data In Bit 3
51	VCC	Supply voltage +5V
52	VPP2	Programming voltage 2
53	MDI4	MPEG Data In Bit 4
54	MDI5	MPEG Data In Bit 5
55	MDI6	MPEG Data In Bit 6
56	MDI7	MPEG Data In Bit 7
57	RFU	Reserved
58	RESET	Card Reset
59	WAIT	Prolongation of bus cycles
60	INPACK	Input Port Acknowledge
61	REG	Register Select
62	MOVAL	MPEG Data Out Valid
63	MOSTRT	MPEG Data Out Start
64	MDO0	MPEG Data Out Bit 0
65	MDO1	MPEG Data Out Bit 1
66	MDO2	MPEG Data Out Bit 2
67	CD2	Card Detect 2
68	GND	Ground

Table A-11 Pin assignment of descrambling card

A.6 Parallel Printer Interface PRINTER

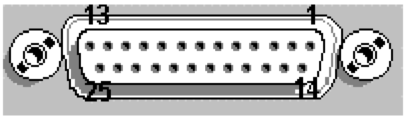


Fig. A-14 Pin assignment of PRINTER

Pin	Abbreviation	Signal
1	STROBE	Clock for data word
2	DATA BIT 0	Data bit 0 (least significant bit)
3	DATA BIT 1	Data bit 1
4	DATA BIT 2	Data bit 2
5	DATA BIT 3	Data bit 3
6	DATA BIT 4	Data bit 4
7	DATA BIT 5	Data bit 5
8	DATA BIT 6	Data bit 6
9	DATA BIT 7	Data bit 7 (most significant bit)
10	ACKNOWLEDGE	Acknowledgement
11	BUSY	Not ready
12	PAPER END	Error message: paper out
13	SELECT	

Pin	Abbreviation	Signal
14	AUTO LINE FEED	
15	ERROR	Error message
16	INITIALIZE PRINTER	
17	SELECT IN	
18	GND	Ground
19	GND	Ground
20	GND	Ground
21	GND	Ground
22	GND	Ground
23	GND	Ground
24	GND	Ground
25	GND	Ground

Table A-12 Pin assignment of PRINTER

Annex B: Definition of Terms

ASI	®	**Asynchronous Serial Interface**
ATSC	®	**Advanced Television Systems Committee**
BAT	®	**Bouquet Association Table**
CA	®	**Conditional Access**
CAT	®	**Conditional Access Table**
CI	®	**Common Interface**
CRC	®	**Cyclic Redundancy Check**
CVCT	®	**Cable Virtual Channel Table**
DTS	®	**Decoding Time Stamp**
DVB	®	**Digital Video Broadcasting**
ECM	®	**Entitlement Control Messages**
EIT	®	**Event Information Table**
EMM	®	**Entitlement Management Messages**
ES	®	**Elementary Stream**
ETT	®	**Extended Text Table**
IRD	®	**Integrated Receiver Decoder**
LVDS	®	**Low Voltage Differential Signalling**
MGT	®	**Master Guide Table**
MIP	®	**Mega-frame Initialization Paclet**
NIT	®	**Network Information Table**
OFDM	®	**Orthogonal Frequency Division Multiplex**
PAT	®	**Program Association Table**
PCR	®	**Program Clock Reference**
PCMCIA	®	**PCMCIA**
PES	®	**Packetized Elementary Stream**
PID	®	**Packet Identity**
PMT	®	**Program Map Table**
Profile	®	**MP@ML**
PS	®	**Program Stream**
PSI	®	**Program Specific Information**
PSIP	®	**Program and System Information Protocol**
PTS	®	**Presentation Time Stamp**
QAM	®	**Quadrature Amplitude Modulation**
QPSK	®	**Quadrature Phase Shift Keying**
RRT	®	**Rating Region Table**
SDT	®	**Service Description Table**
SI	®	**Service Information**
ST	®	**Stuffing Table**
STD	®	**System Target Decoder**
STT	®	**System Time Table**
TDT	®	**Time and Date Table**
TOT	®	**Time Offset Table**
TS	®	**Transport Stream**
TVCT	®	**Terrestrial Virtual Channel Table**
VSB	®	**Vestigial Sideband Modulation**

Adaptation Field
The adaptation field is an extension of the *TS header* and contains ancillary data for a program. The *program clock reference (PCR)* is of special importance. The adaptation field must never be scrambled when it is to be transmitted (see *Conditional Access*).

Asynchronous Serial Interface (ASI)
The ASI is an interface for the *transport stream.* Each byte of the transport stream is expanded to 10 bits (energy dispersal) and is transmitted with a fixed bit clock of 270 MHz (asynchronous) irrespective of the data rate of the transport stream. The fixed data rate is obtained by adding dummy data without information content. Useful data is integrated into the serial data stream either as individual bytes or as whole *TS packets.* This is necessary to avoid *PCR jitter*. A variable buffer memory at the transmitter end is therefore not permissible.

Advanced Television Systems Committee (ATSC)
North American Standard Committee which determined the standard of the same name for the digital transmission of TV signals. Like *DVB,* ATSC is also based on *MPEG2* systems as far as transport stream multiplexing is concerned and on *MPEG2* video for video compression. However, standard AC-3 instead of *MPEG2* is used for audio compression. ATSC specifies the terrestrial and cable-conducted transmission while the emission via satellite is not taken into account.

Bouquet Association Table (BAT)
The BAT is an *SI* table (DVB). It contains information about the different programs (bouquet) of a broadcaster. It is transmitted in *TS packets* with *PID* 0x11 and indicated by *table_ID* 0x4A.

Cable Virtual Channel Table (CVCT)
CVCT is a *PSIP* table (*ATSC*) which comprises the characteristic data (eg channel number, frequency, modulation type) of a program (= virtual channel) in the cable (terrestrial transmission → *TVCT*). TVCT is transmitted with the *PID* 0x1FFB in *TS packets* and indicated by the *table_id* 0xC9.

Common Interface (CI)
The CI is an interface at the receiver end for a broadcaster-specific exchangeable *CA* plug-in card. This interface allows scrambled programs from different broadcasters to be descrambled with the same hardware despite differences in CA systems.

Conditional Access (CA)
The CA is a system allowing to scramble programs and to provide access to these programs at the receiver end only to authorized users. Broadcasters can thus charge fees for programs or individual broadcasts. Scrambling can be performed at one of the two levels provided by an MP*EG2* multiplex stream, ie the *transport stream* or the *packetized elementary stream* level. The relevant headers remain unscrambled. The *PSI* and *SI* tables also remain unscrambled except for the *EIT*.

Conditional Access Table (CAT)
The CAT is a *PSI* table (MPEG2) and comprises information required for descrambling programs. It is transmitted in *TS packets* with *PID* 0x0002 and indicated by *table_ID* 0x01.

Continuity Counter
A continuity counter for each *elementary stream (ES)* is provided as a four-bit counter in the fourth and last byte of each *TS header*. It counts the *TS packets* of a *PES*, determines the correct order and checks whether the packets of a PES are complete. The counter (15 is followed by zero) is incremented with each new packet of the PES. Under certain circumstances, exceptions are permissible (see 1.3.3.).

Cyclic Redundancy Check (CRC)
The CRC serves for verifying whether data transmission was error-free. To this effect, a bit pattern is calculated in the transmitter based on the data to be monitored. This bit pattern is added to the corresponding data in such a way that an equivalent computation in the receiver yields a fixed bit pattern in case of error-free transmission after processing the data. Every *transport stream* contains a CRC for the *PSI tables (PAT, PMT, CAT, NIT)* as well as for some *SI tables (EIT, BAT, SDT, TOT)*.

Decoding Time Stamp (DTS)
The DTS is a 33-bit value in the *PES header* and represents the decoding time of the corresponding PES packet. The value refers to the 33 most significant bits of the associated program clock reference. A DTS is only available if it differs from the *presentation time stamp (PTS)*. For video streams this is the case if differential pictures are transmitted and if the order of decoding does not correspond to that of output.

Digital Video Broadcasting (DVB)
The European DVB project stipulates methods and regulations for the digital transmission of TV signals. Abbreviations such as DVB-C (for transmission via cable), DVB-S (for transmission via satellite) and DVB-T (for terrestrial transmission) are used as well.

Elementary Stream (ES)
The elementary stream is a ‘continuous’ data stream for vision, sound or user-specific data. The data originating from the digitization of vision and sound are compressed by means of methods defined in *MPEG2*-Video and *MPEG2*-Audio.

Entitlement Control Messages (ECM)
ECM comprise information for the descrambler in the receiver of a CA system providing further details about the descrambling method.

Entitlement Management Messages (EMM)
EMM comprise information for the descrambler in the receiver of a CA system providing further details about the access rights of the customer to specific scrambled programs or broadcasts.

Event Information Table (EIT)
EIT is defined both as an SI table (*DVB*) and a *PSIP* table (*ATSC).* It provides information about program contents like a TV guide.

In *DVB* the EIT is transmitted in *TS packets* with *PID* 0x0012 and indicated by a *table_ID* from 0x4E to 0x6F. Depending on the *table_ID* different information is contained:

Table_ID 0x4E	actual TS / pesent+following	measured TS + present/following transmission
Table_ID 0x4E	actual TS / pesent+following	measured TS + present/following transmission
Table_ID 0x4F	other TS / present+following	further TS + present/following transmission
Table_ID 0x50...0x5F	actual TS / schedule	measured TS + subsequent transmissions
Table_ID 0x60...0x6F	other TS / schedule	further TS + subsequent transmissions

EIT-0 to EIT-127 are defined in *ATSC*. Each of the EIT-k comprises information on program contents of a three-hour section where EIT-0 is the current time window. EIT-4 to EIT-127 are optional. Each EIT can be transmitted in a *PID* defined by the *MGT* with *table_id* 0xCB

Extended Text Table (ETT)
ETT is a *PSIP* table (*ATSC*) and comprises information on a program (channel ETT) or on individual transmissions (ETT-0 to ETT-127) in the form of text. ETT-0 to ETT-127 are assigned to *ATSC* tables EIT-0 to EIT-127 and provide information on the program contents of a three-hour section. ETT-0 is with reference to the current time window, the other ETTs to later time sections. All ETTs are optional. Each ETT can be transmitted in a *PID* defined by *MGT* with *table_id* 0xCC.

Integrated Receiver Decoder (IRD)
The IRD is a receiver with integrated MPEG2 decoder. A more colloquial expression would be set-top box.

Channel Coding
The channel coding is performed prior to the modulation and transmission of a transport stream. The channel coding is mainly used for a forward error correction (FEC) allowing to correct bit errors occurring in the receiver during transmission.

Low Voltage Differential Signalling (LVDS)
LVDS is used for the parallel interface of the transport stream. It is a positive differential logic. The difference voltage is 330 mV into 100 Ω.

Master Guide Table (MGT)
MGT is a reference table for all other *PSIP* tables (*ATSC*). It lists the version number, the table length and the *PID* for each *PSIP* table with the exception of the *STT*. MGT is always transmitted with a *Section* in the *PID* 0x1FFB and indicated by the *table_ID* 0xC7.

MIP
Mega-frame Initialization Packet

MP@ML
MP@ML stands for Main Profile / Main Level and is a type of source coding for video signals. The profile determines the source coding methods that may be used while the level defines the picture resolution.

Moving Picture Experts Group (MPEG)
MPEG is an international standardization committee working on the coding, transmission and recording of (moving) pictures and sound.

MPEG2
MPEG2 is a standard consisting of three main parts and written by the *Moving Picture Experts Group* (ISO/IEC 13818). It describes the coding and compression of video (Part 2) and audio (Part 3) to obtain the corresponding *elementary stream* as well as the multiplexing of elementary streams to a *transport stream* (Part 1).

Network Information Table (NIT)
The NIT is a *PSI* table (*MPEG2/DVB*). It comprises technical data about the transmission network (eg orbit positions of satellites and transponder numbers). The NIT is transmitted in *TS packets* with *PID* 0x0010 and indicated by t*able_ID* 0x40 or 0x41.

Null Packet
Null packets are *TS packets* by which the *transport stream* can be filled to obtain a specific data rate. Null packets do not contain any useful data and have the *packet identity* 0x1FFF. The *continuity counter* is undefined.

Orthogonal Frequency Division Multiplex (OFDM)
The modulation method is used in *DVB* systems for emitting transport streams with terrestrial transmitters. It is a multicarrier method and is suitable for the operation of single-frequency networks.

Packet Identity (PID)
The PID is a 13-bit value in the *TS header*. It shows that a *TS packet* belongs to a substream of the *transport stream.* A substream may contain a *packetized elementary stream (PES)*, user-specific data, *program specific information (PSI)* or *service information (SI)*. For some PSI and SI tables the associated PID values are predefined (see 1.3.6.). All other PID values are defined in the PSI tables of the transport stream.

Packetized Elementary Stream (PES)
For transmission, the "continuous" *elementary stream* is subdivided into packets. In case of video streams one picture constitutes the PES, whereas with audio streams an audio frame which may represent an audio signal between 16 ms and 72 ms is the PES. Each *PES packet* is preceded by a *PES header.*

Payload
Payload signifies useful data in general. With reference to the t*ransport stream* all data except for the *TS header* and the adaptation field is payload. With reference to an *elementary stream (ES)* only the useful data of the corresponding ES without the *PES-header* is payload.

Payload Unit Start Indicator
The payload unit start indicator is a 1-bit flag in the second byte of a *TS header*. It indicates the beginning of a *PES packet* or of a section of *PSI* or *SI* tables in the corresponding TS packet.

PCMCIA (PC Card)

PCMCIA is a physical interface standardized by the Personal Computer Memory Card International Association for the data exchange between computers and peripherals. A model of this interface is used for the *common interface*.

PCR Jitter
The value of a *PCR* exactly refers to the beginning of a *TS packet* in which it is located. The reference to the 27-MHz system clock yields an accuracy of approx. ±20 ns. If the difference of the transferred values deviates from the actual difference of the beginning of the corresponding packets, this is called a PCR jitter. It can be caused, for example, by an inaccurate PCR calculation during *transport stream* multiplexing or by the subsequent integration of *null packets* on the transmission path without PCR correction.

PES Header
Each *PES packet* in the transport stream starts with a *PES header*. The *PES header* contains information for decoding the *elementary stream*. The *presentation time stamp (PTS)* and *decoding time stamp (DTS)* are of vital importance. The beginning of a PES header and thus also the beginning of a PES packet is indicated in the associated *TS packet* by means of the set *payload unit start indicator*. If the PES header is to be scrambled, it is scrambled at the transport-stream level. It is not affected by scrambling at the elementary-stream level (see *Conditional Access*).

PES Packet
The PES packet (not to be mixed up with *TS packet*) contains a *packetized elementary stream (PES)* as transmission unit. In a video stream, for example, this is a *source-coded* picture. The length of a PES packet is normally reduced to 64 Kbytes. Only if a video picture requires more capacity may a PES packet be longer than 64 Kbytes. A *PES header* precedes each PES packet.

Presentation Time Stamp (PTS)
The PTS is a 33-bit value in the *PES header* and represents the output time of the content of a PES packet. The value refers to the 33 most significant bits of the associated program clock reference. If the order of output does not correspond to the order of decoding, a decoding time stamp (DTS) is additionally transmitted. This is the case for video streams containing differential pictures.

Program and System Information Protocol (PSIP)
PSIP is the summary of tables defined by *ATSC* for the emission of transmission parameters, program descriptions etc. They contain the structure defined by *MPEG2* systems for 'private' *sections*. These tables are as follows:
Master Guide Table (MGT)
Terrestrial Virtual Channel Table (TVCT)
Cable Virtual Channel Table (CVCT)
Rating Region Table (RRT)
Event Information Table (EIT)
Extended Text Table (ETT)
System Time Table (STT)

Program Association Table (PAT)
The PAT is a *PSI T*able *(MPEG2)*. It lists all the programs contained in a *transport stream* and refers to the associated *PMT*s containing further information about the programs. The PAT is transmitted in *TS packets* with *PID* 0x0000 and indicated by t*able_ID* 0x00.

Program Clock Reference (PCR)
The PCR is a 42-bit value contained in an *adaptation field* and helps the decoder to synchronize its system clock (27 MHz) to the clock of the encoder or TS multiplexer by means of PLL. In this case, the 33 most significant bits refer to a 90-kHz clock while the 9 least significant bits count from 0 to 299 and thus represent a clock of 300 x 90 kHz (= 27 MHz). Each program of a transport stream relates to a PCR which is transmitted in the adaptation field by *TS packets* with a specific PID. The *presentation time stamps (PTS)* and *decoding time stamps (DTS)* of all the *elementary streams* of a program refer to the 33 most significant bits of the PCR. According to *MPEG2* PCRs have to be transmitted at intervals of max. 100 ms, according to the *DVB* regulations at intervals of max. 40 ms.

Program Map Table (PMT)
The PMT is a *PSI* table (*MPEG2*). The *elementary streams* (vision, sound, data) belonging to the individual programs are described in a PMT. A PMT consists of one or several *sections* each containing information about a program. The PMT is transmitted in *TS packets* with a *PID* from 0x0020 to 0x1FFE (referenced in the *PAT*) and indicated in *table_ID* 0x02.

Program Stream (PS)

Like the *transport stream,* the program stream is a multiplex stream but can only contain elementary streams for a program and is only suitable for the transmission in ‘undisturbed’ channels (eg recording in storage media).

Program Specific Information (PSI)
The four tables below defined by *MPEG2* are summed up as program specific information:
Program Association Table (PAT),
Program Map Table (PMT),
Conditional Access Table (CAT),
Network Information Table (NIT).

Quadrature Amplitude Modulation (QAM)
QAM is the modulation method used for transmitting a *transport stream* via cable. The *channel coding* is performed prior to QAM.

Quadrature Phase Shift Keying (QPSK)
QPSK is the modulation method used for transmitting a *transport stream* via satellite. The *channel coding* is performed prior to QPSK.

Source Coding
The aim of source coding is data reduction by eliminating redundancy to the greatest possible extent and by affecting the relevance in a video or audio signal as little as possible. The methods to be applied are defined in *MPEG2*. They are the precondition that the bandwidth required for the transmission of digital signals is narrower than that for the transmission of analog signals.

Rating Region Table (RRT)
The RRT is a *PSIP* table (*ATSC*). It comprises reference values for different geographical regions for the classification of transmissions (eg 'suitable for children older than X years'). RRT is transmitted with a se*ction* in the *PID* 0x1FFB and indicated by the *table_ID* 0xCA.

Running Status Table (RST)
The RST is an *SI table* (*DVB)* and contains status information about the individual broadcasts. It is transmitted in *TS packets* with *PID* 0x0013 and indicated by *table_ID* 0x71.

Section
Each table (*PSI* and *SI*) may comprise one or several sections. A section may have a length of up to 1 Kbyte (for *EIT* and *ST* up to 4 Kbytes). For most of the tables, 4 bytes at the end of each section are available for the *CRC*.

Service Description Table (SDT)
The SDT is an *SI* table (DVB) and contains the names of programs and broadcasters. It is transmitted in *TS packets* with *PID* 0x0011 and indicated by t*able_ID* 0x42 or 0x46.

Service Information (SI)
The following tables defined by *DVB* are called service information. They comprise the structure for 'private' sections defined by *MPEG2* systems:
Bouquet Association Table (BAT),
Service Description Table (SDT),
Event Information Table (EIT),
Running Status Table (RST),
Time and Date Table (TDT),
Time Offset Table (TOT).
Sometimes, the *program specific information (PSI)* is also included.

Stuffing Table (ST)
The ST is an *SI* table (*DVB)*. It has no relevant content and is obtained by overwriting tables that are no longer valid on the transmission path (eg at cable headends). It is transmitted in *TS packets* with a *PID of* 0x0010 to 0x0014 and indicated by *table_ID* 0x72.

Sync byte
The sync byte is the first byte in the *TS header* and thus also the first byte of each *TS packet*. Its value is 0x47.

System Target Decoder (STD)
The System Target Decoder describes the (theoretical) model for a decoder of *MPEG2* transport streams. A 'real' decoder has to fulfill all the conditions based on STD if it is to be guaranteed that the contents of all transport streams created to *MPEG2* are decoded error-free.

System Time Table (STT)
STT is a *PSIP* table (*ATSC*). It comprises date and time (UTC) as well as the local time difference. STT is transmitted in *TS packets* with the *PID* 0x1FFB and indicated by the *table_ID* 0xCD.

Table_ID
The table_identity defines the type of table (eg *PAT*, *NIT*, *SDT*, etc.) and is always located at the beginning of a s*ection* of the corresponding table. The table_ID is necessary especially because different tables can be transmitted with a *PID* (eg *BAT* and *SDT* with PID 0x0011, in one substream (see Table 1-3).

Terrestrial Virtual Channel Table (TVCT)
TVCT is a PSIP table (*ATSC*) comprising the characteristic data of a program (eg channel number, frequency, modulation method) for terrestrial emission (transmission in cable → CVCT). TVCT is transmitted in *TS packets* with the *PID* 0x1FFB and indicated by the *table_id* 0xC8.

Time and Date Table (TDT)
The TDT is an *SI* table (*DVB)* and contains date and time (UTC). It is transmitted in *TS packets* with *PID* 0x0014 and indicated by *table_ID* 0x70.

Time Offset Table (TOT)
The TOT is an *SI* table (*DVB)* and contains information about the local time offset in addition to date and time (UTC). It is transmitted in *TS packets* with *PID* 0x0014 and indicated by *table_ID* 0x73.

Transport Error Indicator
The transport error indicator is contained in the *TS header* and is the first bit after the s*ync byte* (MSB of the second byte). It is set during *channel decoding* if channel decoding could not correct all the bit errors generated in the corresponding TS packet on the transmission path. As it is basically not possible to find the incorrect bits (eg the PID could also be affected), the errored packet must not be processed any further. The occurrence frequency of a set transport error indicator is no measure for the bit error rate on the transmission path. The set transport error indicator shows that the quality of the transmission path is not sufficient for an error-free transmission despite error control coding. A slight drop in transmission quality will already quickly increase the occurrence frequency of a set transport error indicator and transmission will not take place.

Transport Stream (TS)
The transport stream is a multiplex data stream defined by *MPEG2* which may contain several programs that may consist of several e*lementary streams.* A program clock reference *(PCR)* is carried along for each program. Multiplexing is by forming *TS packets* for each elementary stream and by stringing together these TS packets originating from different elementary streams.

TS Header
The TS header is provided at the beginning of each *TS packet* and has a length of four bytes. The TS header always begins with the *sync byte* 0x47. Further important elements are the *PID* and the *continuity counter.* The TS header must never be scrambled when it is to be transmitted (see *Conditional Access*).

TS Packet
The *transport stream* is transmitted in packets of 188 bytes (after *channel coding* 204 bytes). The first four bytes form the *TS header* which is followed by the 184 useful bytes.

Vestigial Sideband Modulation (VSB)
The VSB (=vestigial sideband amplitude modulation) modulation method is used in *ATSC s*ystems. For terrestrial emission, 8-VSB with 8 amplitude levels is used while 16-VSB is mainly for cable transmission.

Annex C: Software MPEG2 REALTIME MONITOR

For your safety **0-2**
Notes on the handbook **0-2**

1. Chapter 1-1

How does the MPEG2 REALTIME MONITOR work **1-2**
Background knowledge and system requirements **1-2**

2. Chapter 2-1

System requirements **2-2**
Hardware 2-2
Software 2-2
Equipment supplied 2-2

3. Chapter 3-1

Preparation **3-2**
Connection to measuring instrument 3-2
Installation **3-3**
To install the MPEG2 REALTIME MONITOR software proceed as follows: 3-3
Program start **3-5**
Starting the software from the program group 3-5
Starting a program via the start menu 3-7
Deinstallation **3-9**

4. Chapter 4-1

User interface **4-2**
Main window 4-2
Menu bar **4-3**
File 4-3
Mode 4-3
View 4-4
Options 4-4
Help 4-11
Toolbar 4-12
Status bar 4-12
Structure window 4-13
Tree Navigator 4-13
List Navigator 4-14
Parameter window 4-14
Statistics / Report 4-14
Data rates 4-17

For your safety

Please observe the following instructions for your own safety and to ensure trouble-free operation of your new product and your computer system:

- Computer modules are sensitive to electrostatic discharges. Discharge any electrostatics from your body before handling components manually or with tools.
- Fit the dongle supplied with the computer switched off.

Notes on the handbook

This user handbook explains how to install and to use **MPEG2 REALTIME MONITOR** software.

Passages of special importance are marked with a light bulb.

Note: *Special features are highlighted by a note.*

The index finger indicates important details on figures.

① The numbering indicates instructions for handling specific situations step by step.

File Menus, commands, options, buttons, etc. to be selected by the user are represented in another colour and in italics.

1. Chapter

Overview

This section contains basic information on the MPEG2 REALTIME MONITOR software.

Topics covered by this chapter	Page

How does the MPEG2 REALTIME MONITOR work 1.2
Background knowledge and system requirements.................. 1.2

How does the MPEG2 REALTIME MONITOR work

The **MPEG2 REALTIME MONITOR** software serves as a user interface for the MPEG2 REALTIME MONITOR DVRM. It runs under **Windows 95/98 and Windows NT** on any PC or laptop connected to DVRM via a serial interface. Ease of operation and a clear presentation of measurement results in two windows of variable size ensure fast and effective working of the system.

The MPEG2 REALTIME MONITOR software allows the DVRM to be completely remote-controlled and displays measurement results of the instrument on a Windows interface. Test reports can be stored on the hard disk for an extended period of time.

MPEG2 REALTIME MONITOR software supports the DVB and ATSC standards. After connection to DVRM, the software automatically switches to the correct standard depending on the operating mode selected on DVRM.

The software supports connection to networked monitoring systems via the COM/DCOM interface. This makes MPEG2 REALTIME MONITOR a server application for the exchange of data with other software packages (client applications) and via network links.

Background knowledge and system requirements

- The easy-to-operate software and integrated help information make it possible for any user to work fast and successfully with the MPEG2 REALTIME MONITOR right from the start without the need for any specialized knowledge in MPEG2 or DVB (**d**igital **v**ideo **b**roadcasting).
- The **MPEG2 REALTIME MONITOR** software runs under ***Windows 95/98*** and ***Windows NT 4.0*** on any PC or laptop.

Note: *Information on system requirements is given in chapter 2.*

2. Chapter

Before You Start

This section describes the requirements to be met for running the MPEG2 REALTIME MONITOR software package.

Topics covered by this chapter **Page**

System requirements 2.2
Hardware 2.2
Software 2.2
Equipment supplied 2.2

System requirements

Before installing the MPEG2 REALTIME MONITOR make sure that the following system requirements are met:

Hardware

- IBM PC or laptop with PENTIUM II processor
  (recommended clock frequency min. 100 MHz),
  (recommended graphics resolution 1024 x 768 pixels),
- RAM
  - min. 16 Mbyte for Windows 95/98
  - min. 32 Mbyte for Windows NT,
- approx. 10 Mbyte free hard disk memory,
- CDROM drive,
- 1 free serial interface (RS-232, recommended data rate 115 kbit/s).

Software

- Windows 95/98 or Windows NT 4.0 operating system

Equipment supplied

Before you start installing the MPEG2 REALTIME MONITOR make sure that the following components are available:

- CDROM with MPEG2 REALTIME MONITOR software,
- cable for connecting the PC/laptop to MPEG2 REALTIME MONITOR DVRM.

3. Chapter

Start up

This chapter describes installation, program start and deinstallation of the **MPEG2 REALTIME MONITOR** software.

Topics covered by this chapter **Page**

Preparation 3.2
Connection to measuring instrument 3.2
Installation 3.3
To install the MPEG2 REALTIME MONITOR software proceed as follows: 3.3
Program start 3.5
Starting the software from the program group 3.5
Starting a program via the start menu 3.7
Deinstallation 3.9

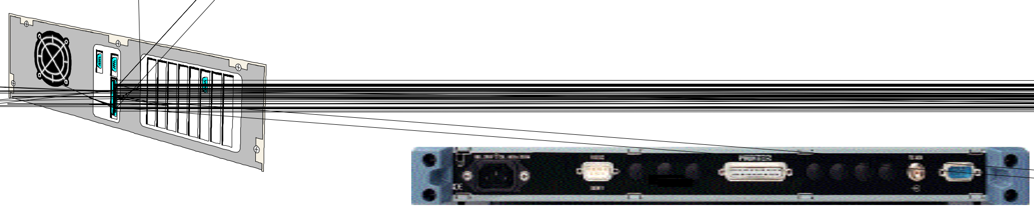
Preparation

Connection to measuring instrument

The PC is connected to DVRM via the serial interface by means of the supplied serial cable. The connection can be made to COM1 or COM2 of the PC. A data transfer rate of 115 kbit/s is ideal.

Note: The MPEG2 REALTIME MONITOR software allows the respective parameters (baud rate and COM interface) to be adapted to the system in use.

Note: *A data transfer rate of 115 kbit/s is not supported by every PC / laptop!* Therefore, in case of communication problems with the 115 kbit/s setting, use a lower bit rate.

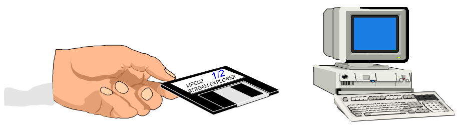


Installation

To install the MPEG2 REALTIME MONITOR software proceed as follows:

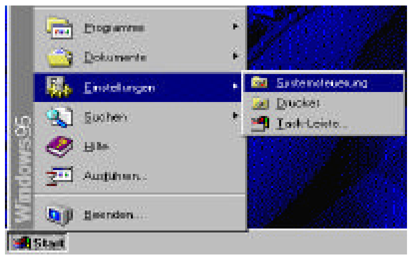
① Start Windows if it is not active.

② Insert floppies 1 to 3 (or CD) containing **MPEG2 REALTIME MONITOR** software into the disk drive of your computer.



③ Select from the task bar:

Start
⇩
Settings
⇩
Control Panel.

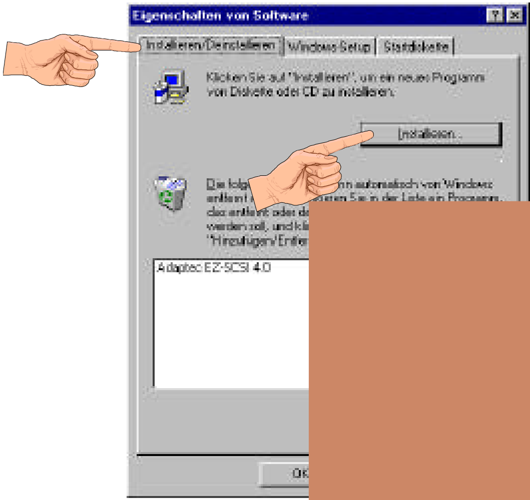


④ Double-click the **Software** icon in the **Control Panel** window.

⑤ Select the **Install / Uninstall** tab in the **Add/Remove Programs Properties** window.

⑥ Click ***Install...*** and follow the instructions displayed on the screen.

.



Note: *The Setup.exe installation program of the floppy is found and started automatically on the floppy.*

Note: *When the installation is completed, the program group "Rohde & Schwarz / MPEG2 REALTIME MONITOR" is displayed on the desktop together with the icons of the installed programs.*

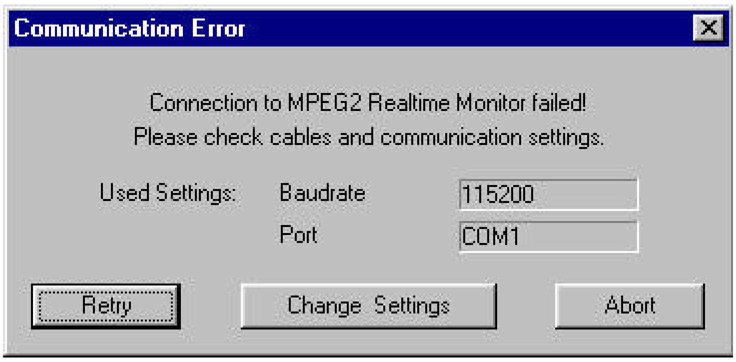
Program start

To start the MPEG2 REALTIME MONITOR software proceed as follows:

Starting the software from the program group

① Start the program by double-clicking the **MPEG2 REALTIME MONITOR** icon in the **ROHDE & SCHWARZ / MPEG2 REALTIME MONITOR** program group.

In the case of communication problems between the MPEG2 Realtime Monitoring software and the MPEG2 Realtime Monitor DVRM, the program issues a message in the "Communication Error" window.



Usual causes of error:

- ☐ **COM interface of PC not correctly set.**
- ☐ **Baud rate set on the measurement decoder is not supported by the PC.**

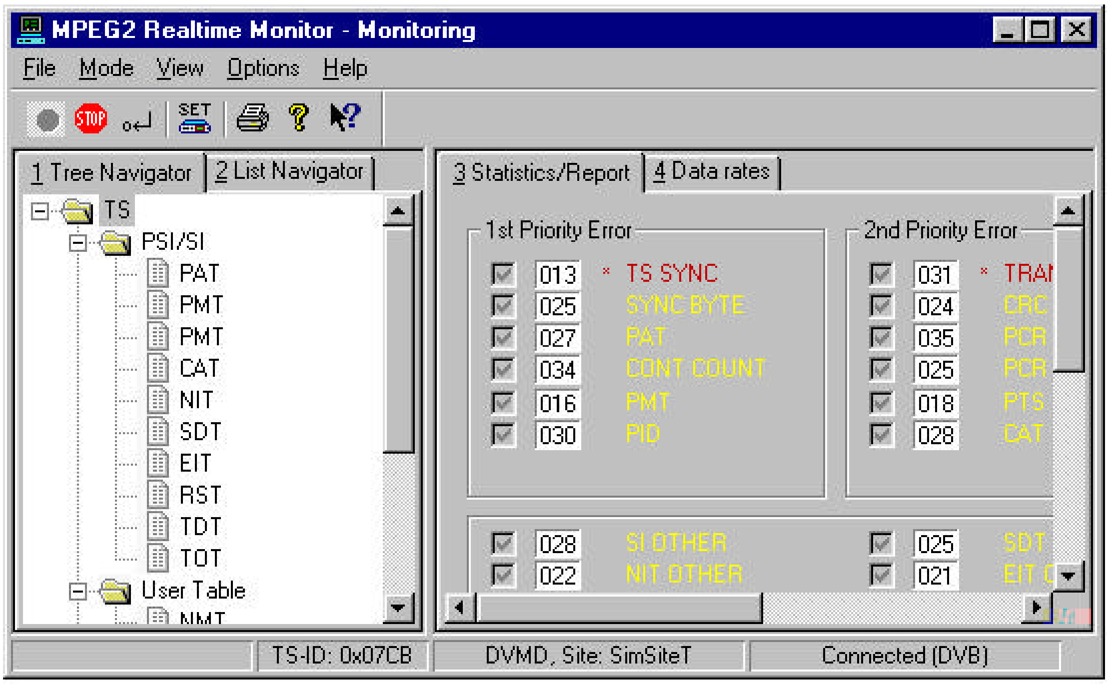
The required COM interface must always be activated manually using the Change Settings button and the selected interface must be shown in the Port field. If the Retry button is pressed, the MPEG2 Realtime Monitor software tries to establish a data link to the DVRM by automatically varying the PC baud rate. The program then sets the PC and MPEG2 Realtime Monitor DVRM synchronously to the required baud rate. Therefore, changing the baud rate on DVRM is only necessary if the set baud rate of DVRM is not supported by your PC / laptop.

Remedy

Activate the change of the baud rate and the COM interface selection using the Change Settings button in the message window.

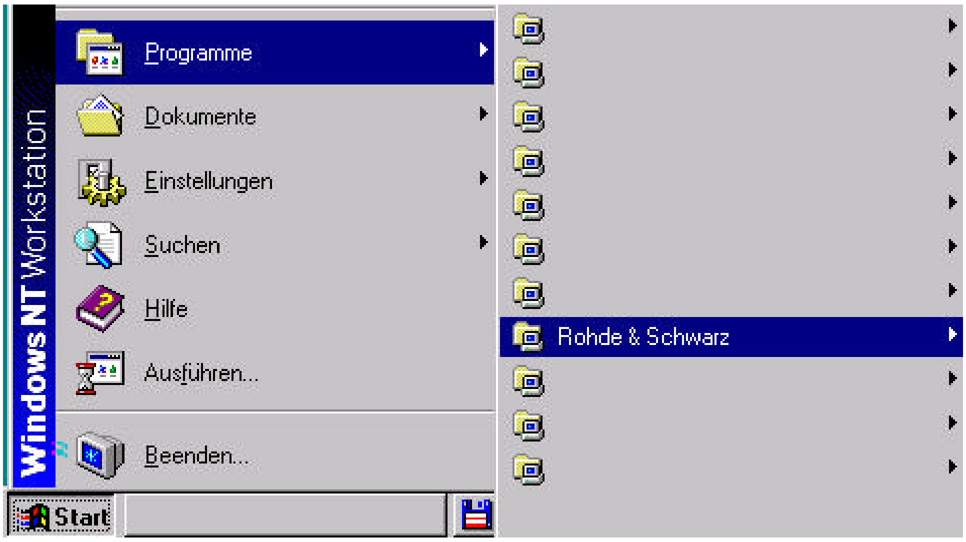
- ☐ Select the COM interface (COM1 to COM4) of the PC.
- ☐ Set a baud rate of 115 kbit/s or lower (depending on PC / laptop).
- ☐ Set the same baud rate on DVRM.

② The user interface of the MPEG2 Realtime Monitor program is displayed.

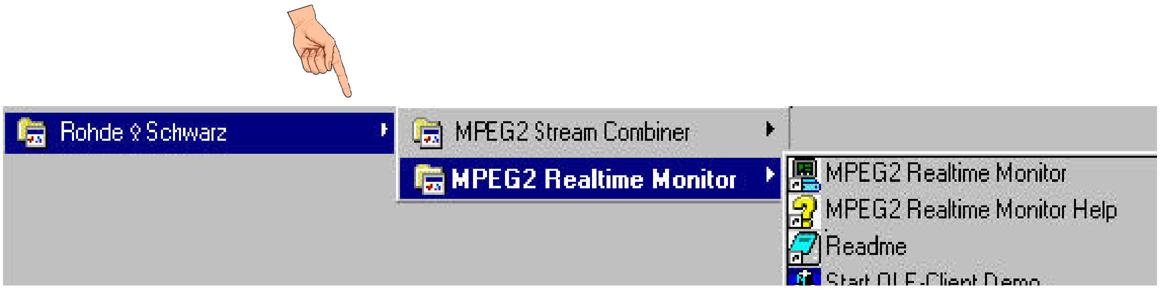


Starting a program via the start menu

① Open the Programs submenu in the Start menu.



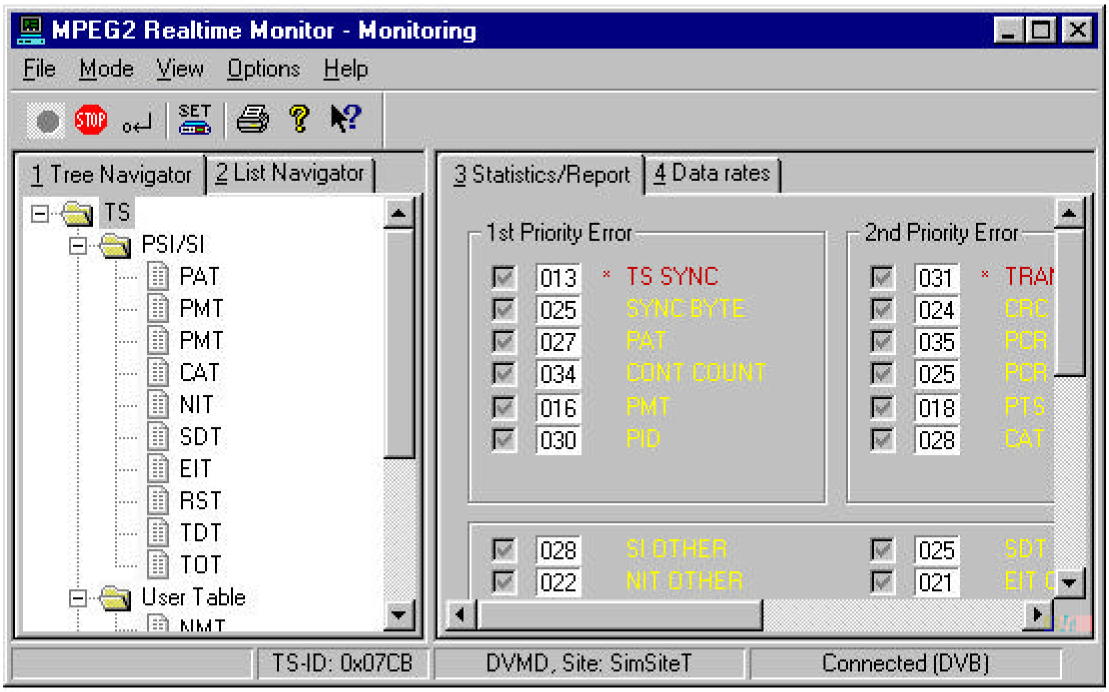
② Select 'Rohde & Schwarz' and then the MPEG2 Realtime Monitor.



③ Select the MPEG2 Realtime Monitor program.

The user interface of the MPEG2 Realtime Monitor program is displayed.

④

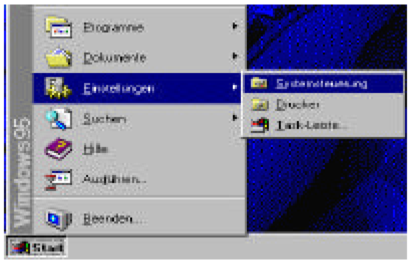


Deinstallation

To deinstall the MPEG2 Realtime Monitor software proceed as follows:

① Start Windows if it is not active.

② Select in the task bar:
Start
⇩
Settings
⇩
Control Panel.

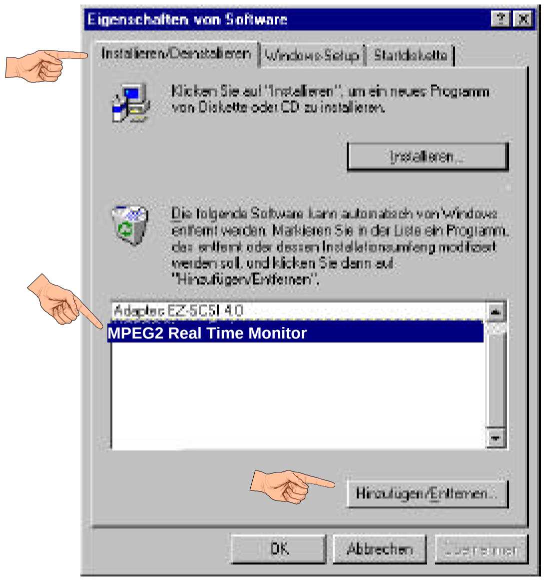


③ Double-click the **Software** icon in the **Control Panel** window.

④ Select the **Install / Uninstall** tab in the **Add / Remove Program Properties** window

⑤ Select the **MPEG2 Realtime Monitor** program package for deinstallation in the lower section of the **Add/Remove Programs Properties** window.

⑥ Click ***Add / Remove*** and follow the instructions displayed on the screen.



The hard disk memory occupied by the MPEG2 Realtime Monitor program is available again. Entries in the initialization files of Windows are cleared.

4. Chapter

Introduction to MPEG2 REALTIME MONITOR

This chapter will help you to get acquainted with the user interface of the MPEG2 Realtime Monitor. It describes the individual interface components and their use.

Topics covered by this chapter	**Page**

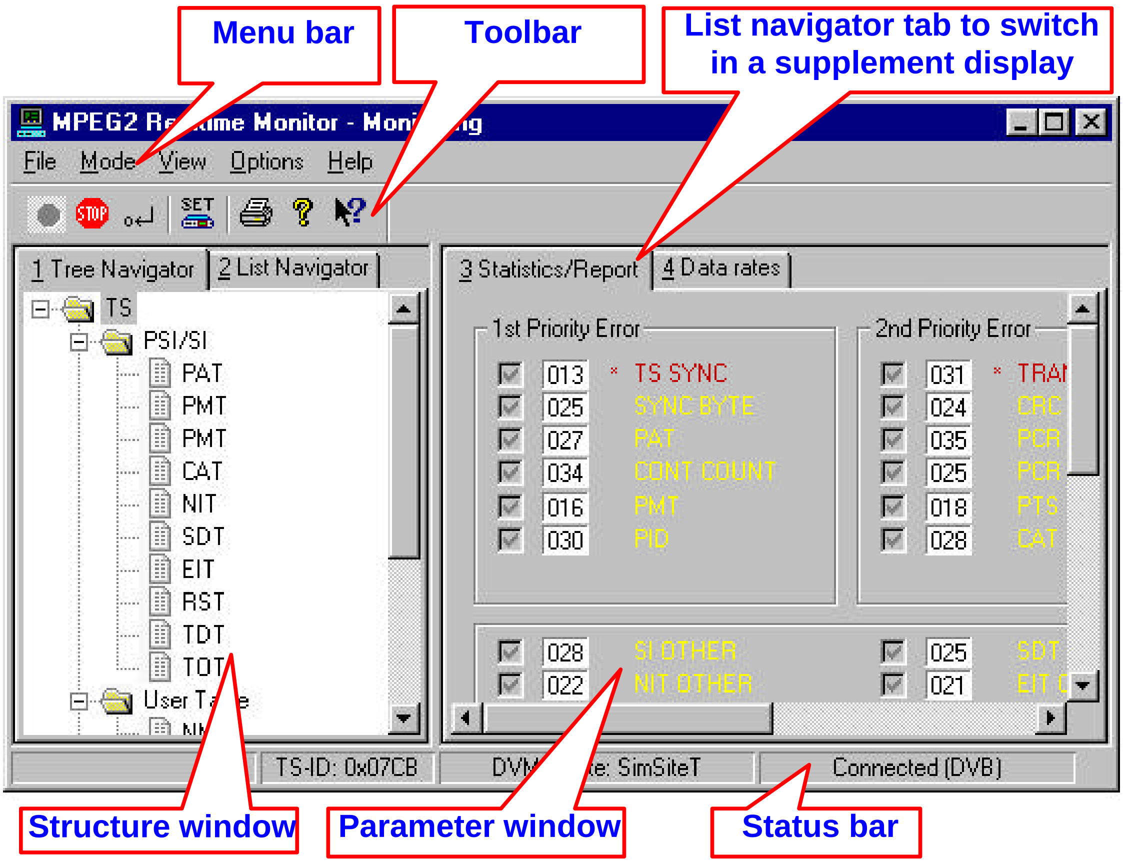
User interface **4.2**
Main window.. 4.2
Menu bar **4.3**
File 4.3
Mode 4.3
View 4.4
Options.. 4.4
Help.. 4.11
Toolbar.. 4.12
Status bar.. 4.12
Structure window.. 4.13
Tree Navigator.. 4.13
List Navigator 4.14
Parameter window.. 4.14
Statistics / Report 4.14
Data rates.. 4.17

User interface

Note: The user interface of the MPEG2 Realtime Monitor is in compliance with the Windows conventions.

Main window

The main window is made up of the following areas:



Menu bar

Note: *Menus can be called by a mouse click or by the shortcut ALT + <first letter>.*

Menu items and submenus can be called by a mouse click or by the shortcut Ctrl + <underlined letter> when such a key combination is specified.

File

This menu contains file management functions. Click **File** to display the menu items.

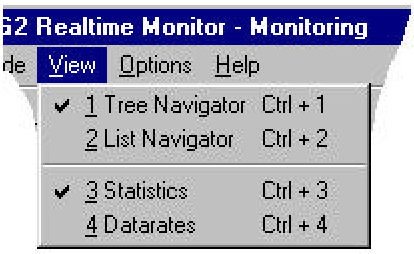
- Store Error Report — Stores the error report.
- Print — Prints current window.
- Print Setup... — Changes printer and printer options.
- Exit: — Terminates MPEG2 Realtime Monitor.

Mode

This menu serves for starting and stopping the realtime measurement on MPEG2 Realtime Monitor DVRM. The current operating mode of DVRM is maintained when the MPEG2 Realtime Monitor software establishes a connection to DVRM.

View

This menu contains functions for selecting result display modes. Click on **View** to make the settings.



The user interface of MPEG2 Realtime Monitor software uses two windows for results display.

In the window on the left, the structure of the transport stream can be displayed in the form of a tree (Tree Navigator) or as a table (List Navigator).

In the window on the right, error statistics/report can be displayed or the data rate of sum and single data streams is shown in graphical form.

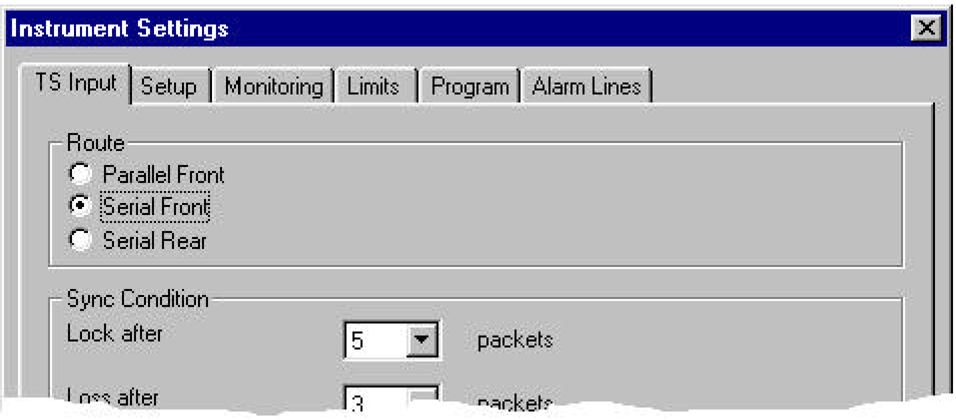
Switchover is also possible by means of the tabs at the top edge of the window.

Options

This menu contains submenus for making settings in the remote-controlled DVRM as well as data transmission settings.
Click **Options** to display the submenus.

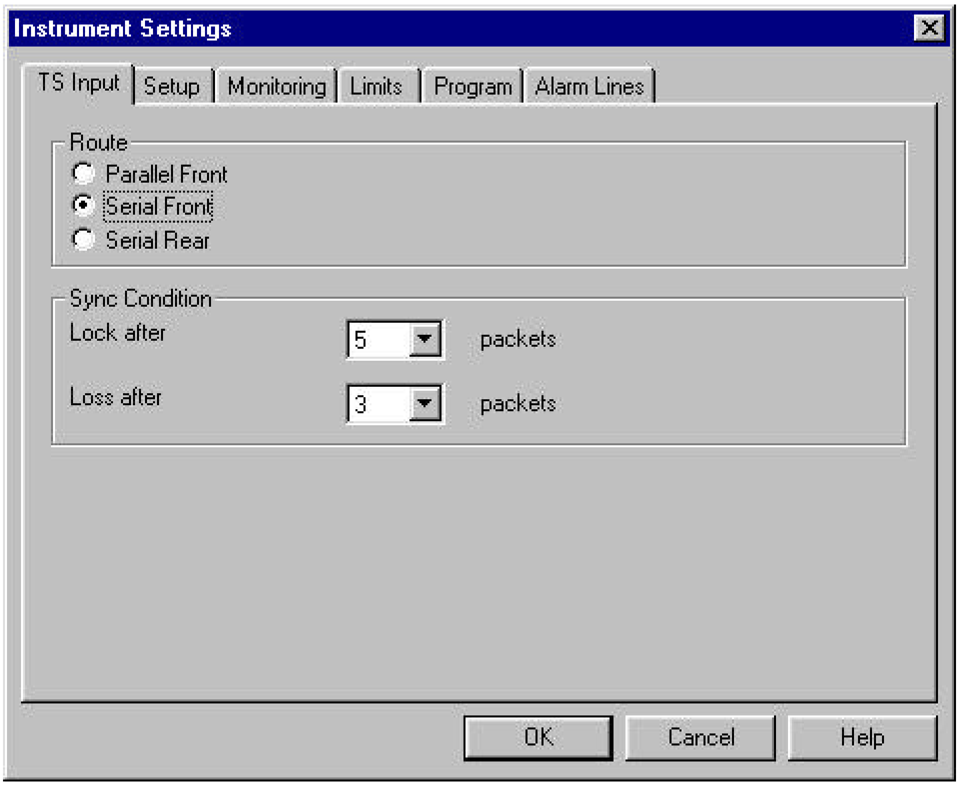
Instrument Settings...	Opens a dialog window for settings of MPEG2 Realtime Monitor DVRM.
Instrument Settings Store...	Stores or loads settings of MPEG2 Realtime Monitor DVRM.
Communication Settings...	Settings for data transmission

Click **Instrument Settings...** to open the dialog window:



The dialog window contains 6 tabs for all settings of MPEG2 Realtime Monitor DVRM. Select the desired setting fields by a click on the respective tab.

On the **TS Input** tab, one of three transport stream inputs (Route) is selected and the hysteresis parameters for transport stream synchronization are set (Sync Condition).



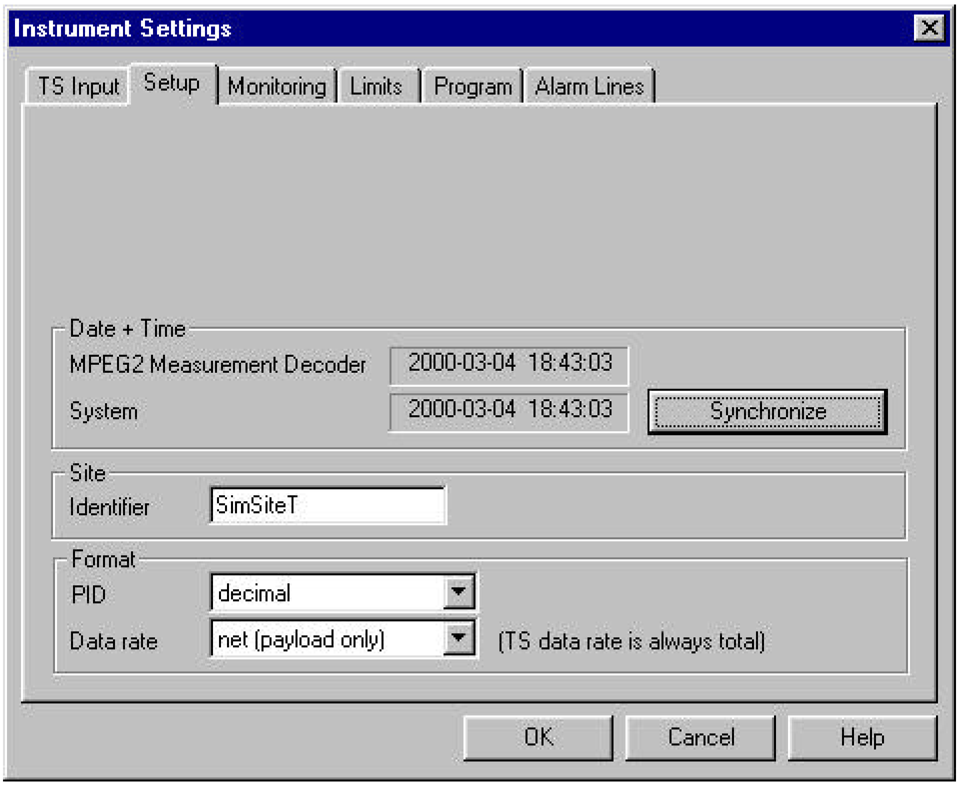
On the **Setup** tab, the following settings can be made:

Date + Time: synchronization of realtime clock of remote-controlled instrument to system time of the PC.

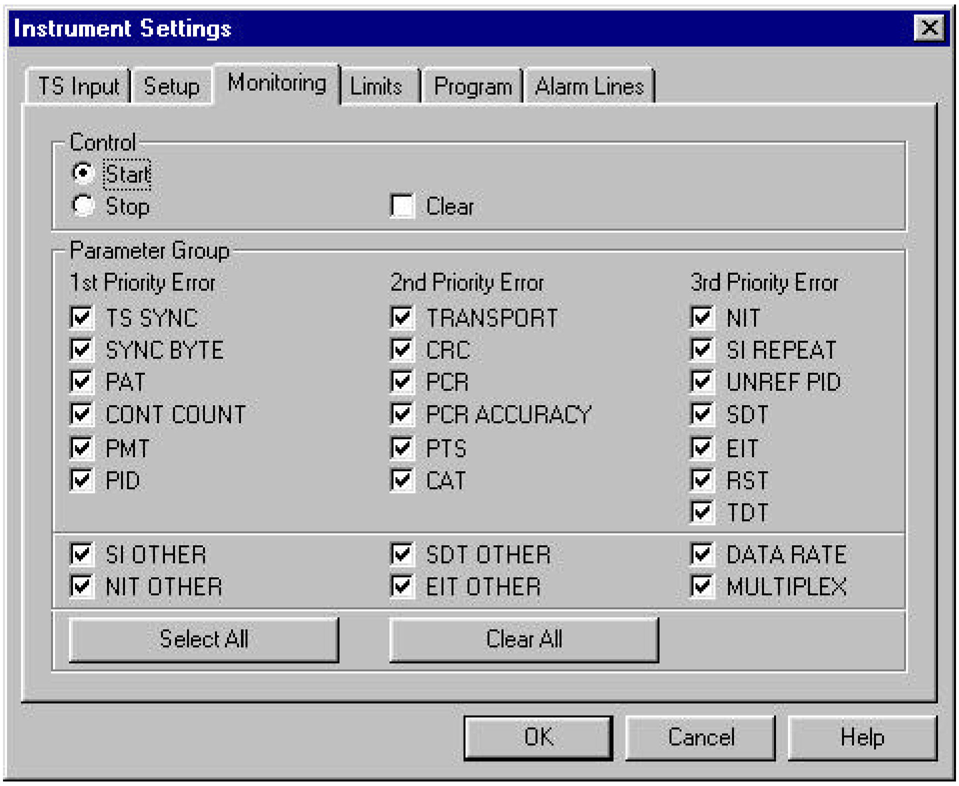
Site: assigning a code word (max. 8 characters) to the remote-controlled DVRM (eg position identification code).

Format PID: Decimal or hexadecimal display of PIDs for print function of DVRM

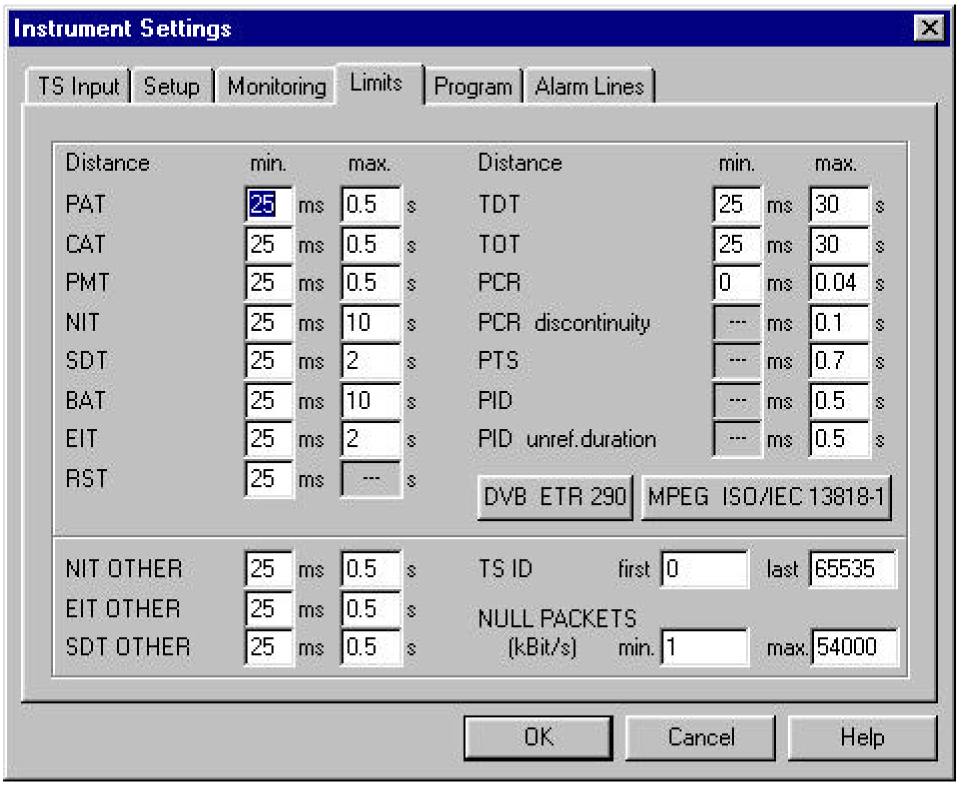
Format Data rate: measurement of total data rate (all data of a PID) or net data rate (payload only without packet header). For the TS data rate always the total data rate is measured.



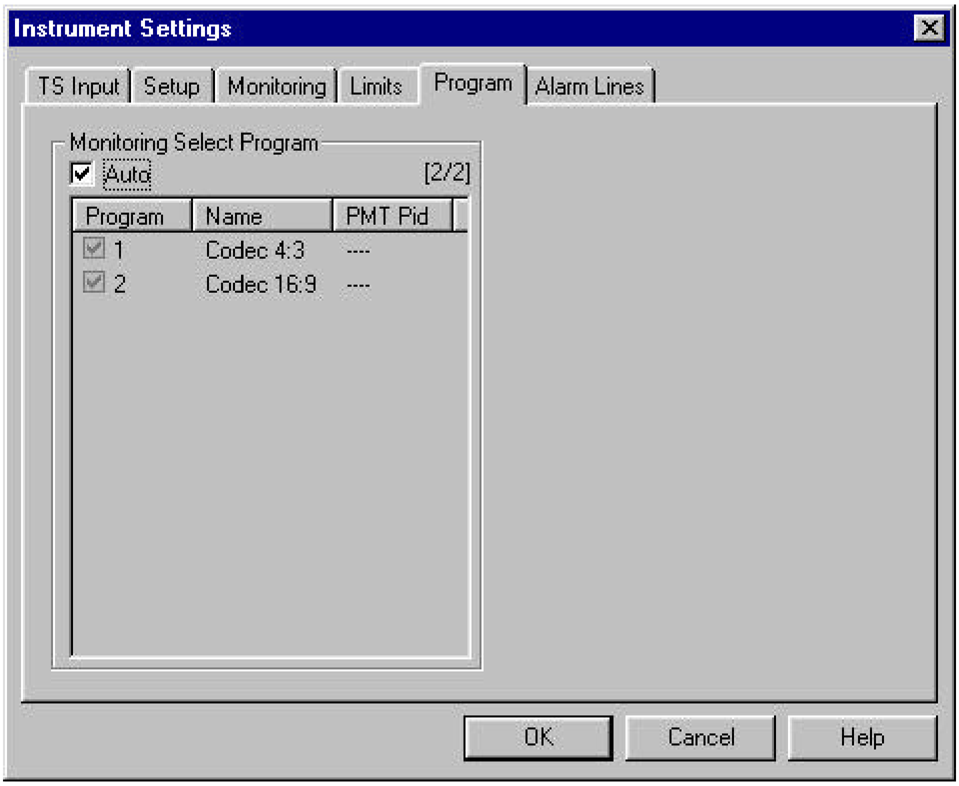
On the **Monitoring** tab, the parameter to be measured is selected (see chapter 2).



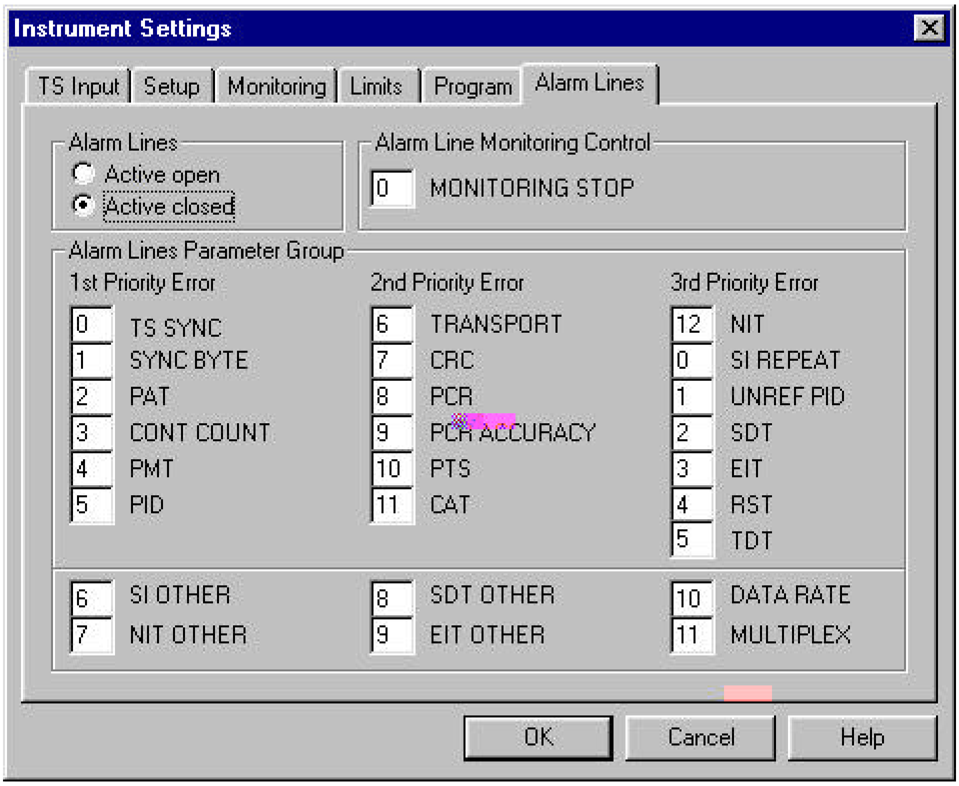
On the **Limits** tab, all limit values required for parameter monitoring are set (see chapter 2).



On the **Program** tab, the programs to be monitored are selected. This concerns the parameters PMT, PID, CRC, PCR, PCR_ACCURACY, PTS, SI_REPEAT and the measurement of data rates of all individual streams pertaining to the respective program. If not all programs are monitored, the UNREF_PID parameter cannot be evaluated.



On the **Alarm Lines** tab, one of twelve alarm contacts is assigned to the individual parameters (15-contact male connector at the rear of DVRM). Active alarm contacts can be opened or closed. Signalling of a realtime monitoring fault (MONITORING STOP) can also be assigned to an alarm contact.



Click **Instrument Settings Store...** to store or load settings.

Button / Field	Function
Close	Closes the dialog window.
Store	Stores the settings made under **Instrument Settings** in the desired memory.
Recall	Loads settings from the selected memory.
Preset	Sets the factory-set default parameters.
Select Memory (1: Preset_ … 9: Preset_)	List of available memories.
Name (Preset_0)	Indication of selected memory (name can be edited).

Click **Communication Settings...** to make data transmission settings.

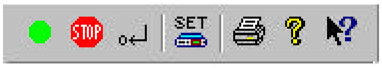
Button / Field	Function
OK	Effects the settings and closes dialog window.
Cancel	Closes the dialog window without making new settings.
Connect	Checks data transmission to **DVRM** (result displayed in **Status** field).
Help	Opens the topic-oriented online help.
Port: COM1	Selection of COM interface of your computer to which **DVRM** is connected.
Baud Rate: 115200	Setting the data transmission rate.
Parity: None	Parity setting (cannot be changed).
Handshake: RTS/CTS	Handshake setting (cannot be changed).
Status: Communication Ok	Status of data transmission to **DVRM.**

Help

The online help can be called with this menu. Click **Help** to display the submenus.

Contents	Opens the topic-oriented online help.
About Stream Explorer...	Opens a window with information on the software.

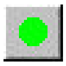
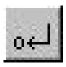
Toolbar



The toolbar offers a number of command buttons for carrying out frequently needed menu functions.

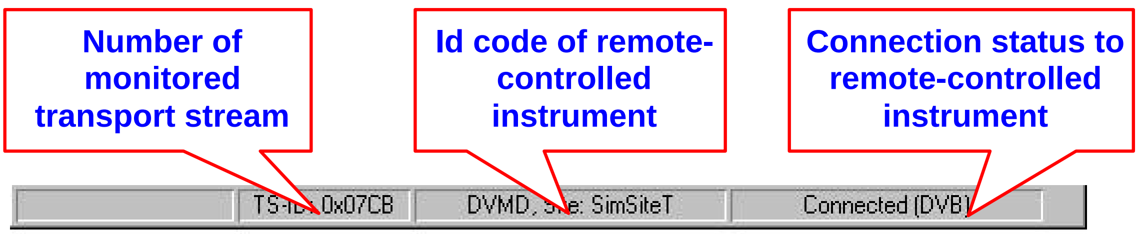
The availability of some buttons depends on the selected operating mode.

Note: *If the mouse cursor points to an active command icon, a short function description of this icon is displayed next to the mouse cursor.*

Button	Function
	Starts the activated operating mode.
	Stops the activated operating mode.
	Clears the entered values (reset) (only available in the measurement and monitoring mode).
	Opens the dialog window with settings of MPEG2 Realtime Monitor DVRM.
	Prints the contents of the currently displayed window.
	Opens a window giving information on the software.
	Opens the context-oriented online help.

Status bar

The status bar serves for displaying status information such as:

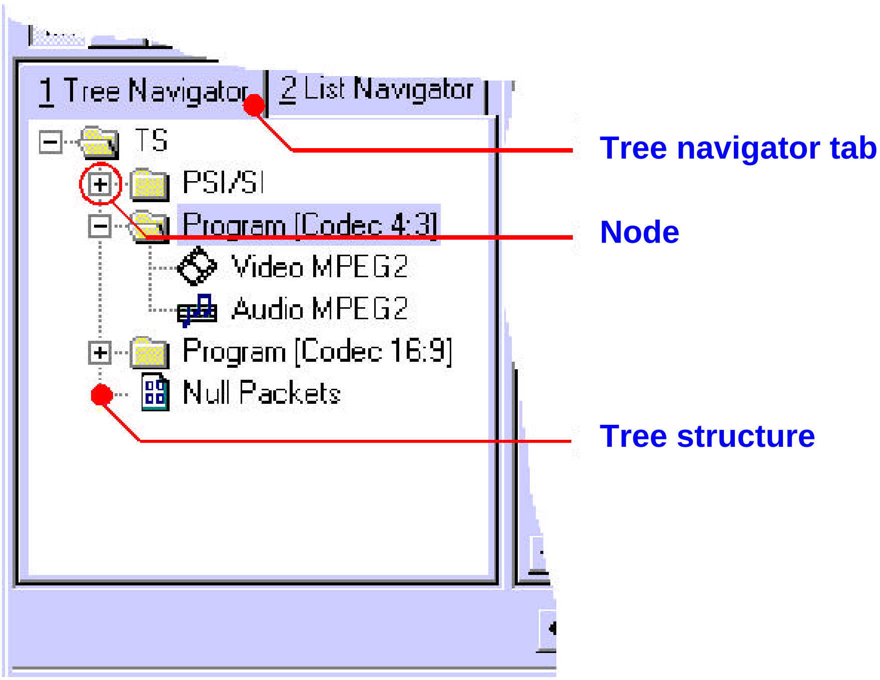


Structure window

In the structure window, the individual elements of the loaded transport stream are represented in the form of a tree (Tree Navigator) or as a table (List Navigator).

Tree Navigator

The tree structure display is selected with the Tree Navigator tab.

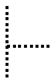


The tree structure elements are linked via nodes (+)/(-).

By **clicking the nodes**, the tree structure display can be expanded or reduced:

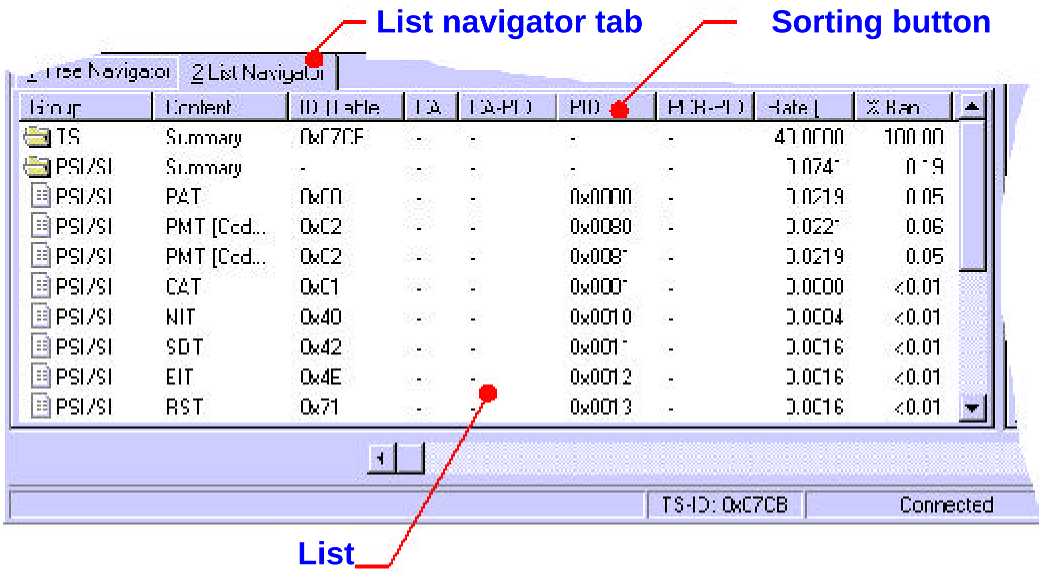
 more details

 less details

 no further details available

List Navigator

Tabular display is selected with the Tree Navigator tab.



By clicking the sorting button, the list is sorted in ascending or descending order according to the selected sorting criterion.

Parameter window

In the parameter window, measurement results are displayed. Statistics/Report or Data Rate display can be selected.

Statistics / Report

Statistics / report display is switched on with the Statistics / Report tab.

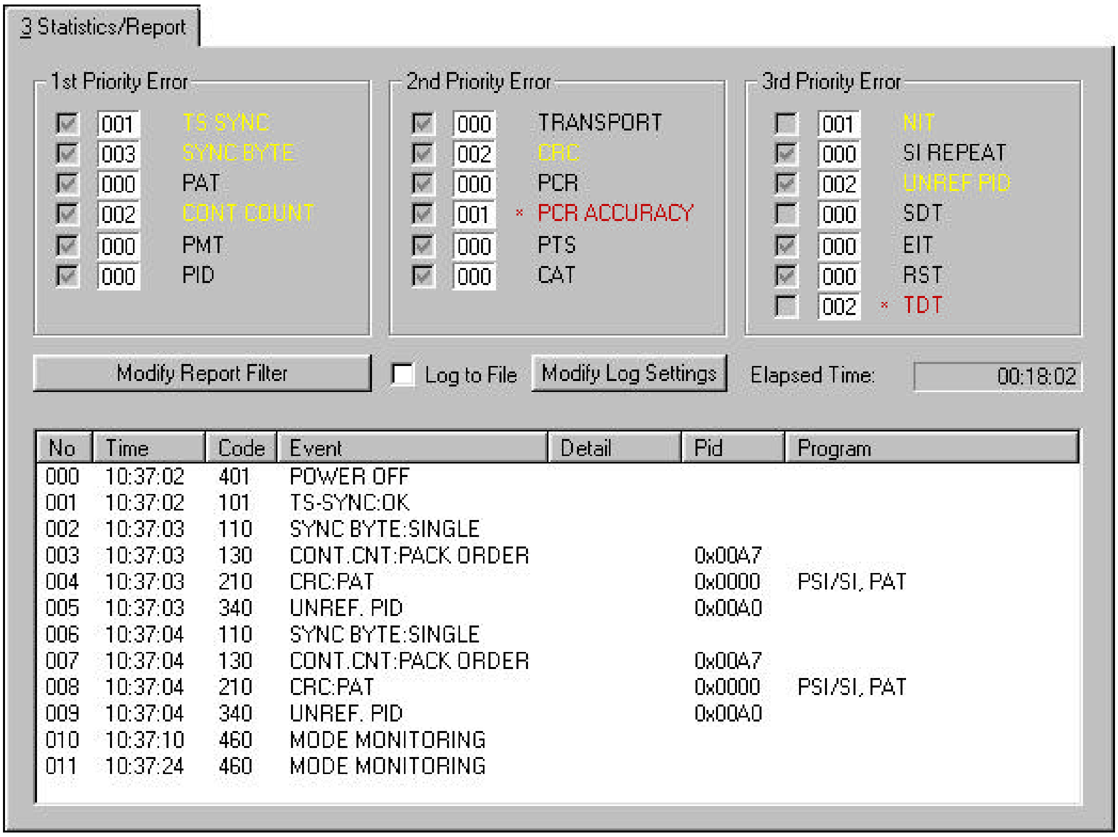
The monitoring results of the connected MPEG2 Realtime Monitor DVRM are displayed in this window made up of the **statistics** and **report** sections.

Statistics

In the Statistics section, the error counts of the individual parameters are displayed. If an error occurs within a test period (approx. 1 s), the error count is incremented by one and the parameter name for this test period is marked in red. A parameter marked yellow indicates that one error has occurred since the measurement was started.

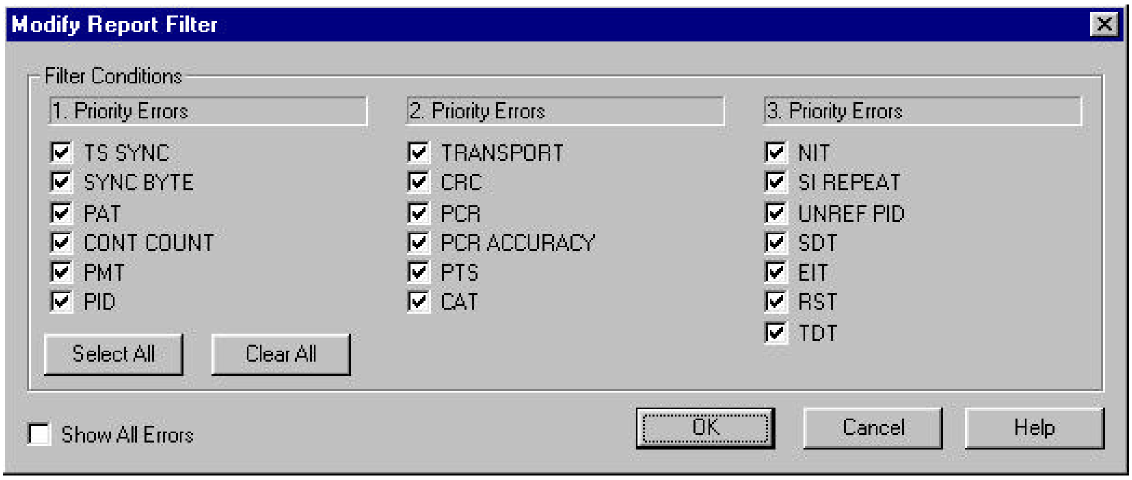
Report

The monitoring report is displayed in the lower half of the window. **DVRM** is able to record errors within a transport stream. A nonvolatile memory for up to 1000 entries is available for this purpose.



Display filter for monitoring report

A click on the **Modify Report Filter** button in the Statistics / Report parameter window opens a selection menu. This selection concerns only the display in the parameter window but not the settings made in the MPEG2 Realtime Monitor DVRM (see also Options / Instrument Settings... / Monitoring menu).



Storing monitoring results in a file

Monitoring results can be stored in a file in two ways:

1.) After analysis, the report data (max. 1000 entries from MPEG2 Realtime Monitor DVRM) are stored in a .cvs file (see File / Store Error Report). The Store Error Report function opens a dialog window where the file name (.cvs) can be selected. The default file name is RepLog.cvs.

2.) A long-term analysis with automatic continual storage of report data is performed. To this end the **Log to File** check box in the Statistics / Report parameter window is activated. Pressing the Modify Log Settings button opens a dialog box where the file name can be specified (.cvs). The default file name is RepLog.cvs.

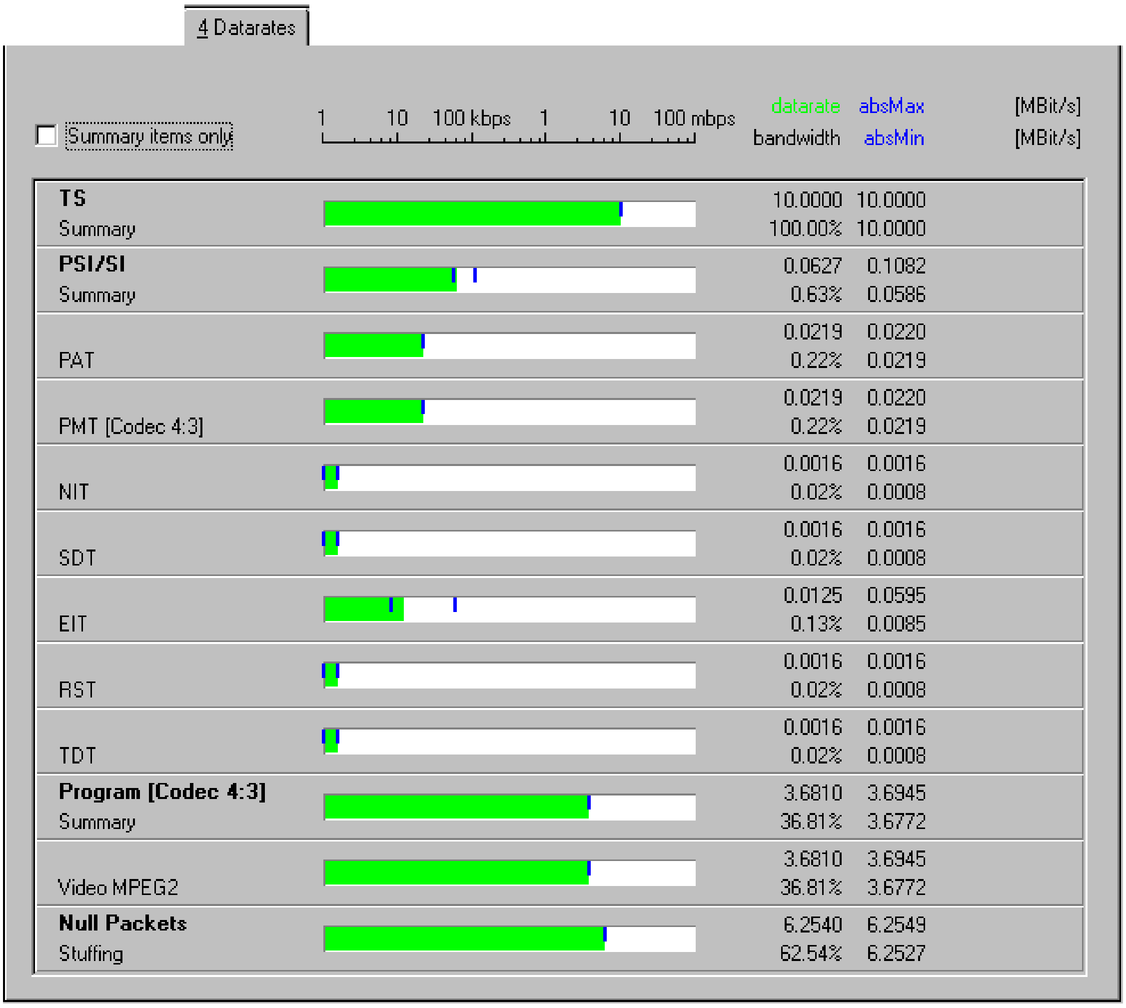
Data rates

Data rate display is switched on with the **Datarates** tab.

Bargraph The data rate of each element from the Tree Navigator or the List Navigator (TS, sum stream, single stream) is displayed in a bargraph. Because of the wide range of data rates (1 kbit/s to 100 Mbit/s) logarithmic scaling applies to the bargraphs.

Datarate / bandwidth In the first column following the bargraph, the current data rate (top) and the ratio of the element to the total transport stream (bandwidth, below) is displayed.

Min / Max data rate These values correspond to the maximum and minimum data rate since start of the measurement. The two values are shown as blue markers in the bargraph and displayed numerically in the second column at the right.


4 Datarates
Summary items only
1 10 100 kbps 1 10 100 mbps
datarate absMax [MBit/s]
bandwidth absMin [MBit/s]
TS Summary 10.0000 10.0000 100.00% 10.0000
PSI/SI Summary 0.0627 0.1082 0.63% 0.0586
PAT 0.0219 0.0220 0.22% 0.0219
PMT [Codec 4:3] 0.0219 0.0220 0.22% 0.0219
NIT 0.0016 0.0016 0.02% 0.0008
SDT 0.0016 0.0016 0.02% 0.0008
EIT 0.0125 0.0595 0.13% 0.0085
RST 0.0016 0.0016 0.02% 0.0008
TDT 0.0016 0.0016 0.02% 0.0008
Program [Codec 4:3] Summary 3.6810 3.6945 36.81% 3.6772
Video MPEG2 3.6810 3.6945 36.81% 3.6772
Null Packets Stuffing 6.2540 6.2549 62.54% 6.2527