# MPEGscope User's Guide

Application Version: CD-ROM A.06

Manual Part Number: E6277-92021 Ed. 5

©Copyright Hewlett-Packard Company 1997 – 2000

All rights reserved

**Copyright** © Hewlett-Packard Company 1997 - 2000
All rights reserved.

**Notice**

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

**Warranty**

A copy of the specific warranty terms applicable to your product and replacement parts can be obtained from your local Sales and Service Office.

**MPEGscope**

contains technology licensed from David Sarnoff Research Center, Subsidiary of SRI International. MPEGscope is manufactured under license from Dolby Laboratories. "Dolby" and the double-D symbol are trademarks of Dolby Laboratories. Confidential Unpublished Works. © 1992-1997 Dolby Laboratories Inc. All rights reserved.

**Microsoft®**

is a U.S. registered trademark of Microsoft Corporation

**Windows® and MS Windows®**

are U.S. registered trademarks of Microsoft Corporation

**Pentium®**

is a U.S. registered trademark of Intel Corporation

**Acrobat®**

is a trademark of Adobe Systems Incorporated

**Printing history**

New editions of this guide are issued to reflect extensive changes made to the software. Revisions may be issued between editions to correct errors in the manual. A new edition may not be issued with every application release. This guide is released on CD-ROM along with the application software as part of the MPEGscope online help system.

Manual Name: MPEGscope User's Guide
Product Number: E6277A/B, E6300A, E6301A, E6302A

| Release Date | Manual Part Number | Application Version |
|---|---|---|
| May, 1998 | E6277-92004 | A.03 CD-ROM |
| August, 1998 | E6277-92009 | A.04 CD-ROM |
| January, 1999 | E6277-92021 Ed1 | A.04.02 CD-ROM |
| August, 1999 | E6277-92021 Ed2 | A.05 CD-ROM |
| December, 1999 | E6277-92021 Ed3 | A.05.02 CD-ROM |
| February, 2000 | E6277-92021 Ed4 | A.05.03 CD-ROM |
| June, 2000 | E6277-92021 Ed5 | A.06 CD-ROM |

**Product support**

Contact your local Agilent Technologies representative, or contact us at

Agilent Technologies
Advanced Networks Division
PO Box 221
Blackburn, 3130
Victoria, Australia

Phone: +61-3-8877-8633
Fax: +61-3-8877-5550
Email: y900_support@agilent.com or dv-support@agilent.com
Web: http://advanced.comms.agilent.com/mpegscope

Printed in Canada

# 1

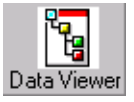
# Introduction


MPEGscope (E6277A/B/C, E6300A, E6301A, E6302A) is a digital video test system for verifying and debugging video encoders, multiplexers, and decoders. With MPEGscope you can monitor, record, compose, capture, store, play, trigger on errors, impair, and analyze data in a variety of ways. This manual shows how to


- monitor data with the Real-time Analyzer


- record and play data with the Recorder/Player


- view MPEG, DVB, or ATSC table decodes as they update in real time




- analyze transport streams and define private tables with the Protocol Data Viewer


- multiplex elementary streams and tables to multi–program MPEG-2 transport streams with the Composer


- generate ISDB-S single and multiple transport stream files with the ISDB MultiTS Generator


- impair transport streams with the Impairments Generator


- verify compliance to MPEG–2, DVB, and ATSC standards with the Compliance Verifier


- analyze video elementary streams with the Video ES Analyzer


- analyze audio elementary streams with the Audio ES Analyzer

**File types**

MPEGscope can monitor, record, transmit, and analyze any valid transport stream file. MPEGscope can also extract elementary streams from a transport stream. For example, you can record a transport stream with the Recorder/Player, extract and save an elementary stream, then analyze the elementary stream with the Video ES Analyzer.

When you use the Recorder/Player, MPEGscope creates two special types of files—transport stream plus files (*.tsp) and auxiliary files (*.aux).

## Transport stream plus files

During recording, MPEGscope adds a 16–byte trailer to each packet. This trailer contains a timestamp (the time at which MPEGscope received the last byte of the packet) and interface status information. MPEGscope stores the recorded data in a transport stream plus file in the following format:

| transport stream packet<br>188 bytes | timestamp & status<br>16 bytes |
|---|---|

If the incoming data contains 204– or 208–byte packets, MPEGscope first removes the 16– or 20–byte trailer, then replaces it with the transport stream plus trailer.

## Auxiliary files

During recording MPEGscope also creates an auxiliary file to accompany the transport stream plus file. Auxiliary files contain statistical information about the recorded data and help speed retransmission and analysis of the file. You can view the contents of the auxiliary file from the Recorder/Player's **File/Properties** menu, but you cannot transmit or analyze an auxiliary file.

**WARNING** Do not delete an auxiliary file if you want to keep its associated transport stream plus file. Similarly, if you move a transport stream plus file to a different location, also move the corresponding auxiliary file. If you rename a file from the Recorder/Player's File/Rename menu, MPEGscope changes both the transport stream plus and auxiliary file names. If you rename a file from Windows NT® Explorer, you must change the auxiliary file name yourself.

**File names**

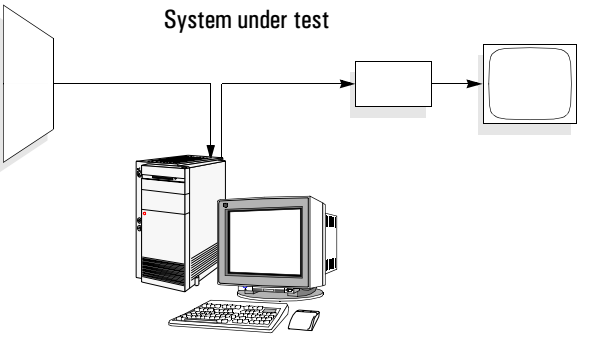
When MPEGscope creates transport stream plus and auxiliary files during recording, it assigns a file name based on the current date and time (YYMMDD_HHMMSS format). Transport stream plus files have a .tsp extension, and auxiliary files have an .aux extension.

**File location**

MPEGscope stores all recorded files on E: drive, a specialized, high–speed hard disk. When you open a transport stream file with the Recorder/Player from a location other than E drive, MPEGscope first copies the file to E: drive. If the file is not a transport stream plus file, MPEGscope converts the file to transport stream plus format. If the file has no auxiliary file, MPEGscope creates one.


For more information on transport stream plus and auxiliary files, click the **Tips** button to see MPEGscope's online help. To locate the topics, type **tsp** or **aux** at the **Index** dialog of the Help Topics Browser.

Before you can monitor (with the Real-time Monitor or Real-time Table Analyzer), record, or play, you must first connect MPEGscope to a system under test.




For information on connecting to MPEGscope interfaces, refer to the MPEGscope Startup Guide.

**CAUTION**

Hewlett–Packard does not guarantee that MPEGscope is compatible with other Windows® applications or hardware, and will not support problems caused by altering system configuration and initialization files.

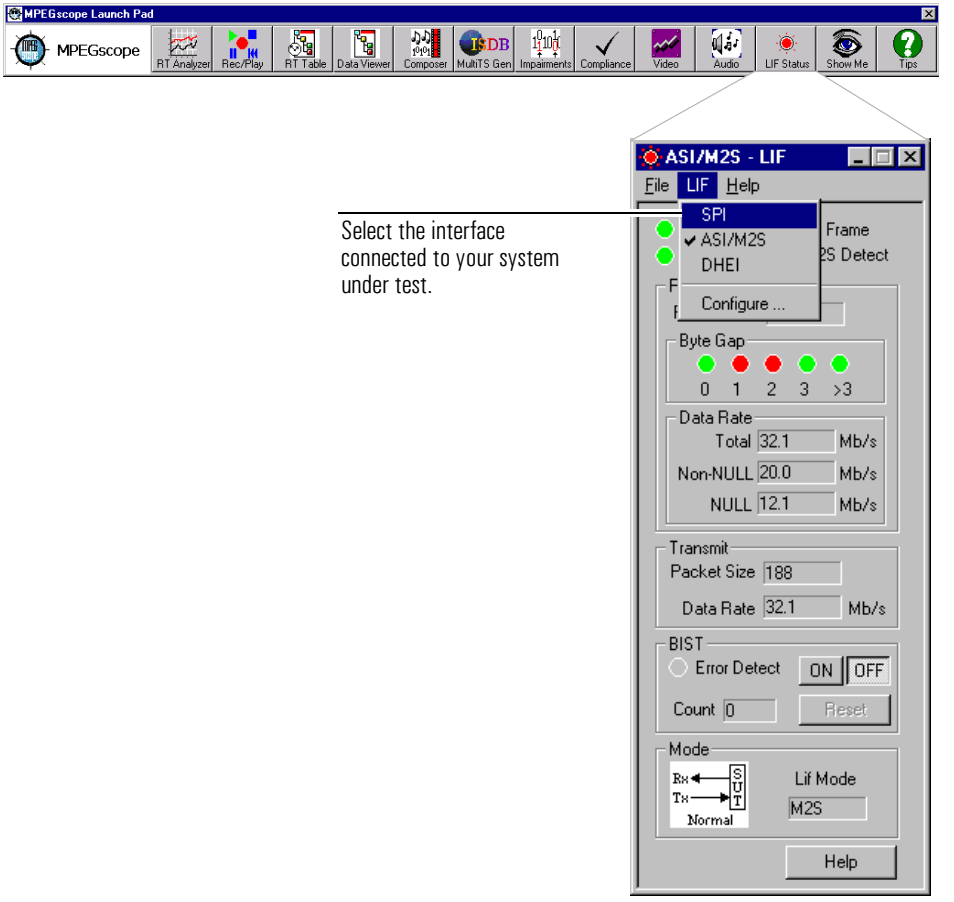
# 2

# Using the Real-time Monitor

# Monitoring a transport stream

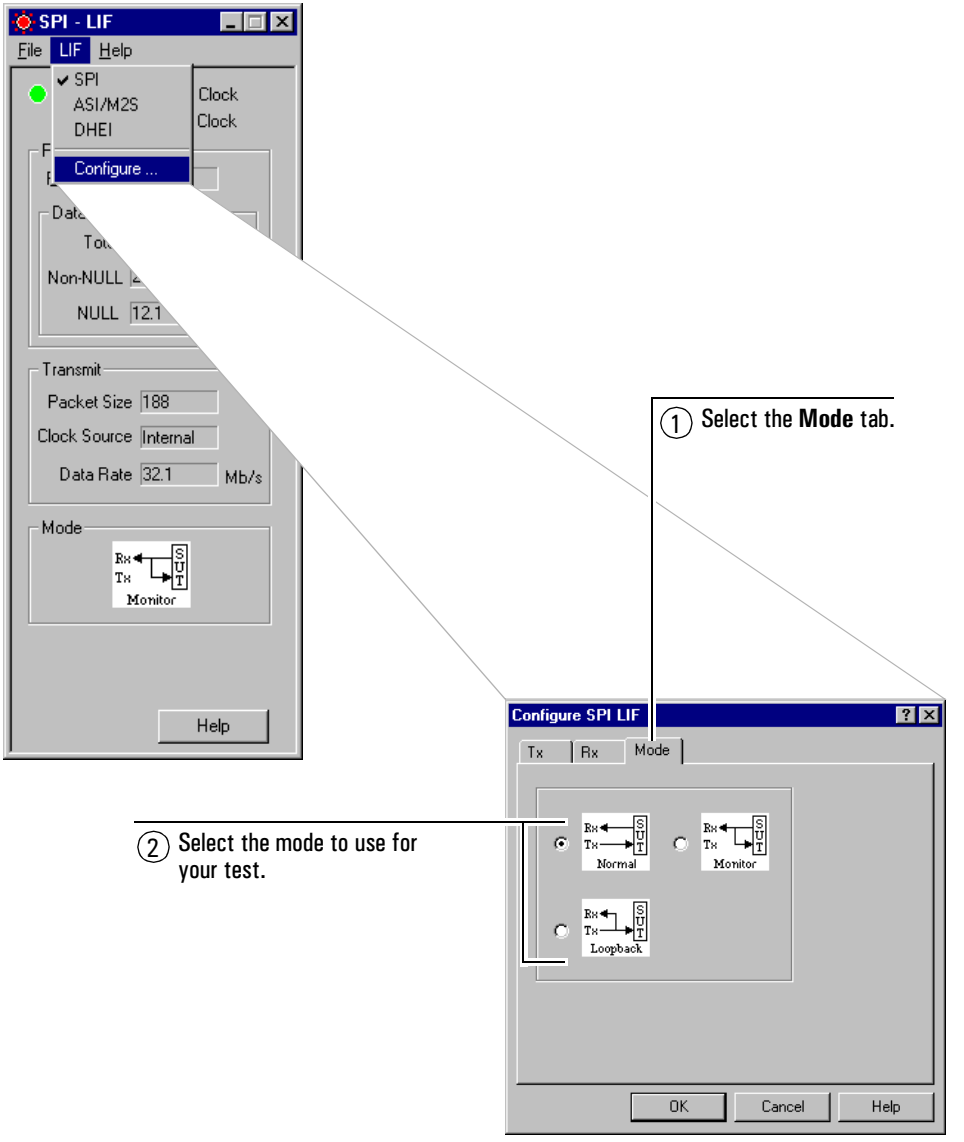

This example shows how to use the Real–time Analyzer to monitor transport stream errors (as defined in ETR 290), and make PID, PCR, and buffer measurements.

## Step 1: Select the interface

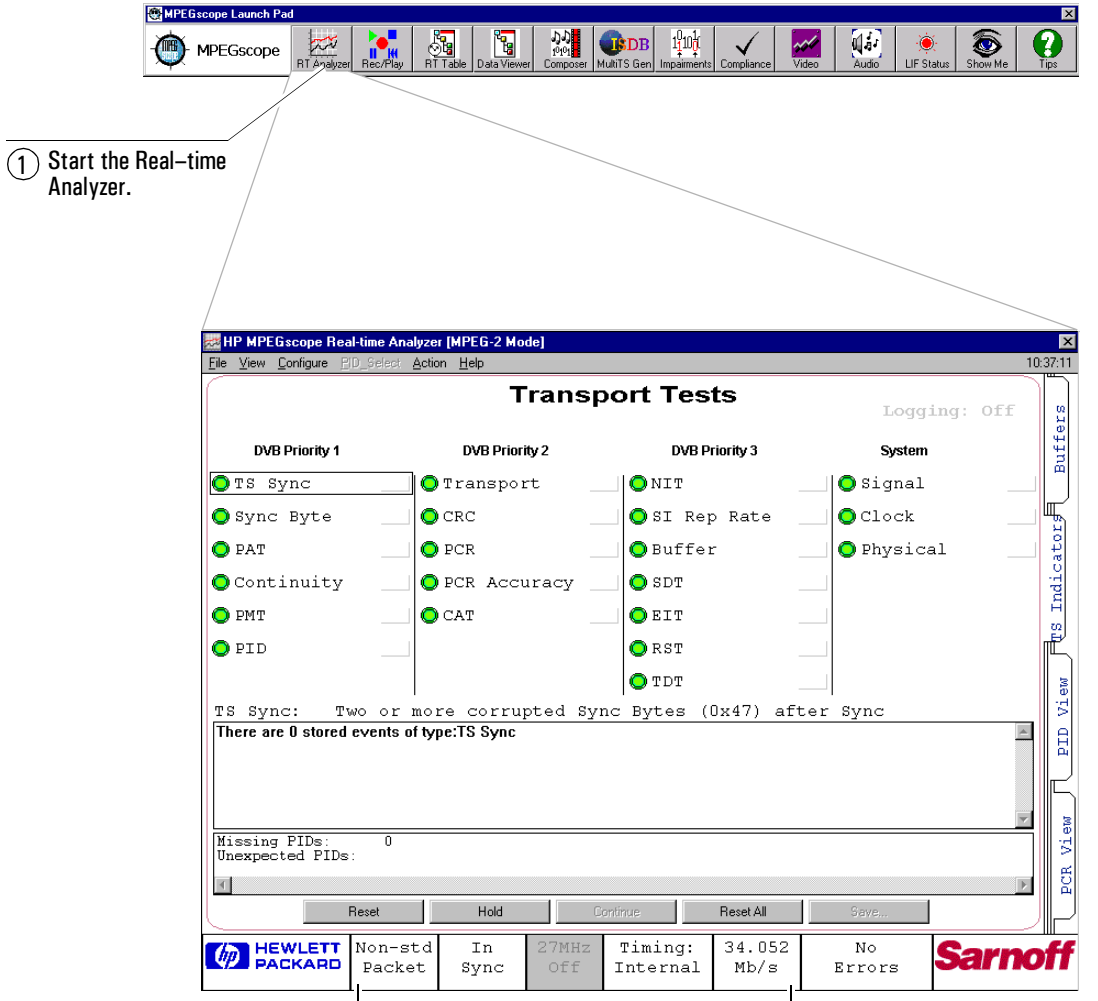


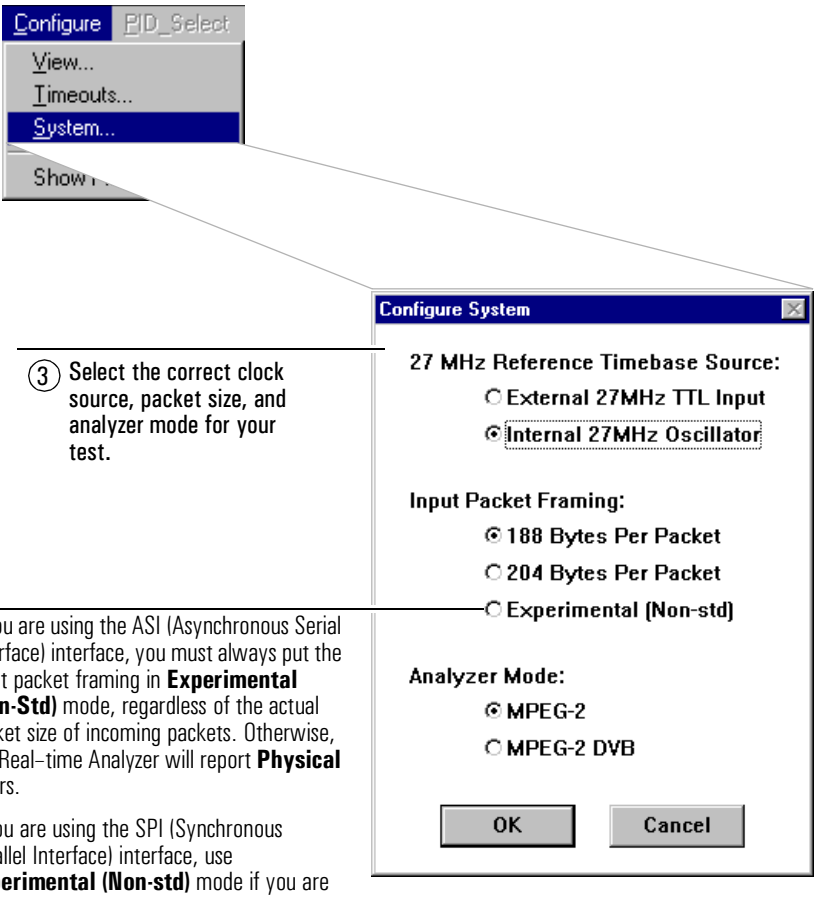

Select the interface connected to your system under test.

## Step 2: Configure the interface mode



## Step 3: Configure the Real–time Analyzer





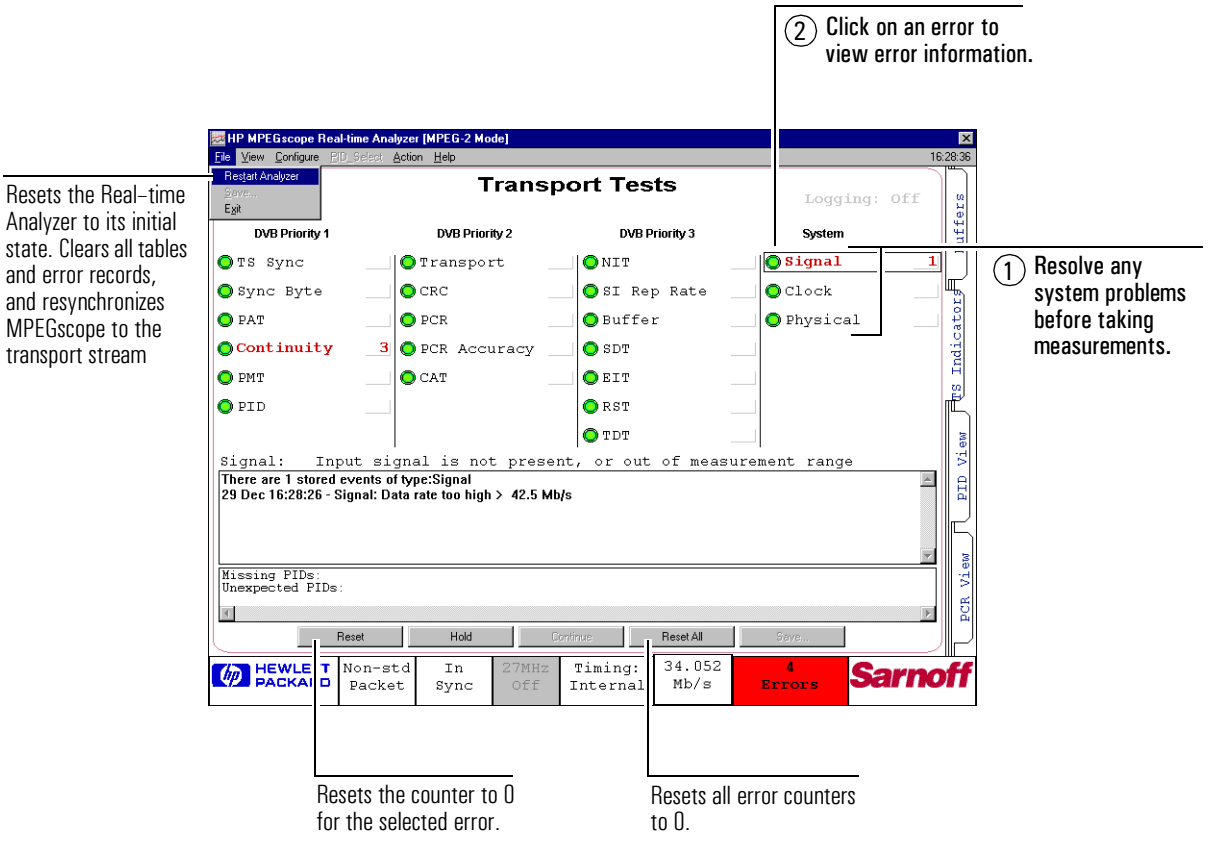

③ Select the correct clock source, packet size, and analyzer mode for your test.

If you are using the ASI (Asynchronous Serial Interface) interface, you must always put the input packet framing in **Experimental (Non-Std)** mode, regardless of the actual packet size of incoming packets. Otherwise, the Real–time Analyzer will report **Physical** errors.

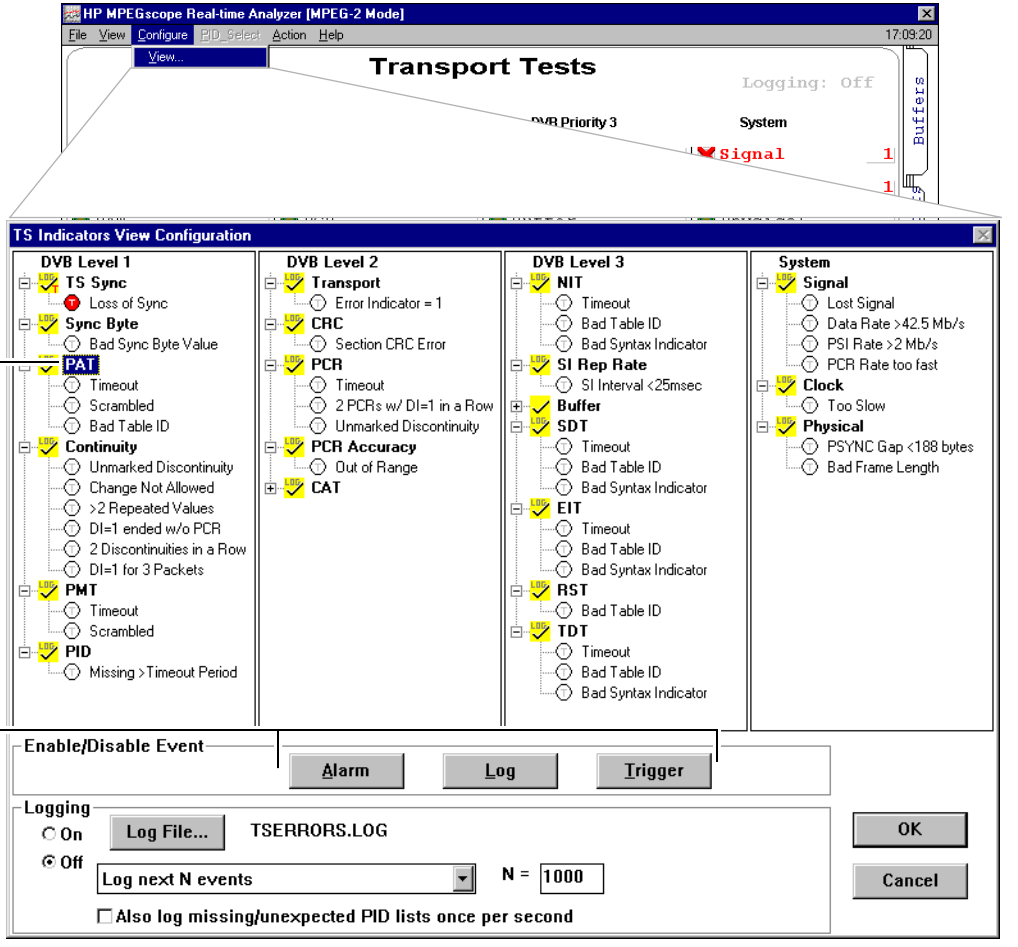
If you are using the SPI (Synchronous Parallel Interface) interface, use **Experimental (Non-std)** mode if you are receiving 208-byte packets.

# Step 4: Check System and TS Indicator LEDs

The **TS Indicators View** shows the status of the transport stream, including transport stream errors (as defined in ETR 290).



# Step 5: Configure the TS Indicators View



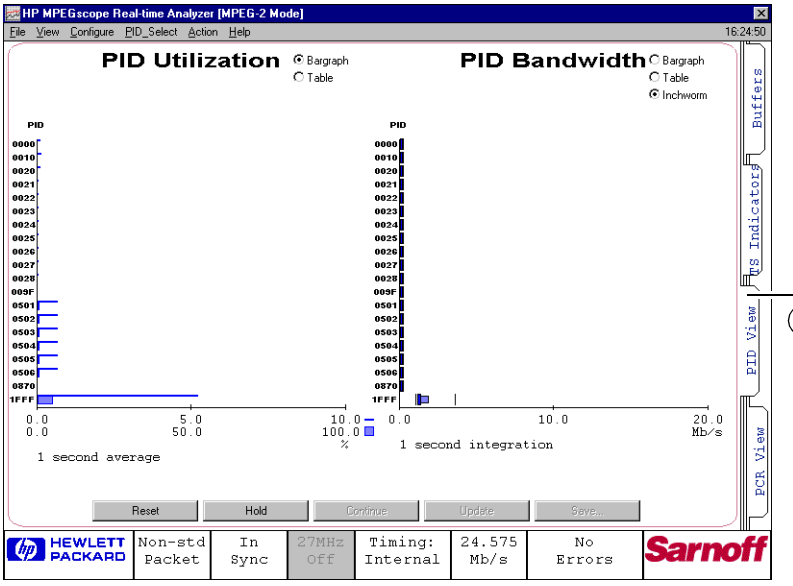

① Select a TS Indicator.

② Click on an **Event** toggle button to enable or disable an event.

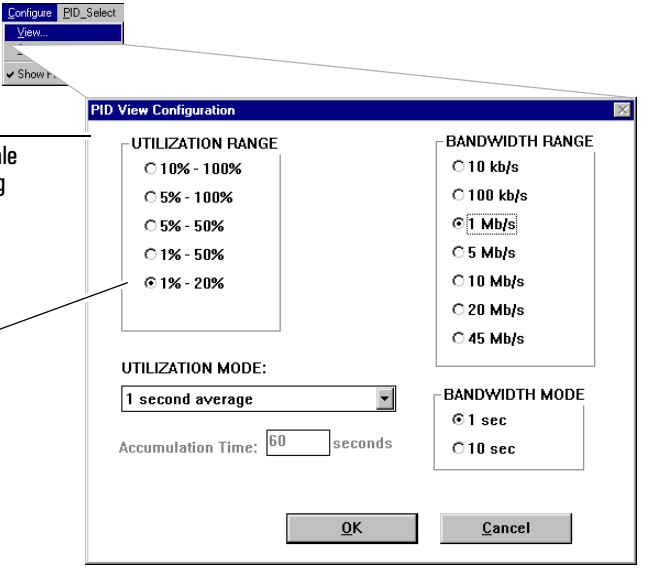
You can also enable or disable an event from the shortcut menu. Select the icon next to the TS Indicator LED with the right mouse button to open this menu.

# Step 6: Measure PID utilization and bandwidth

The **PID view** shows how much of the transport stream each PID uses.



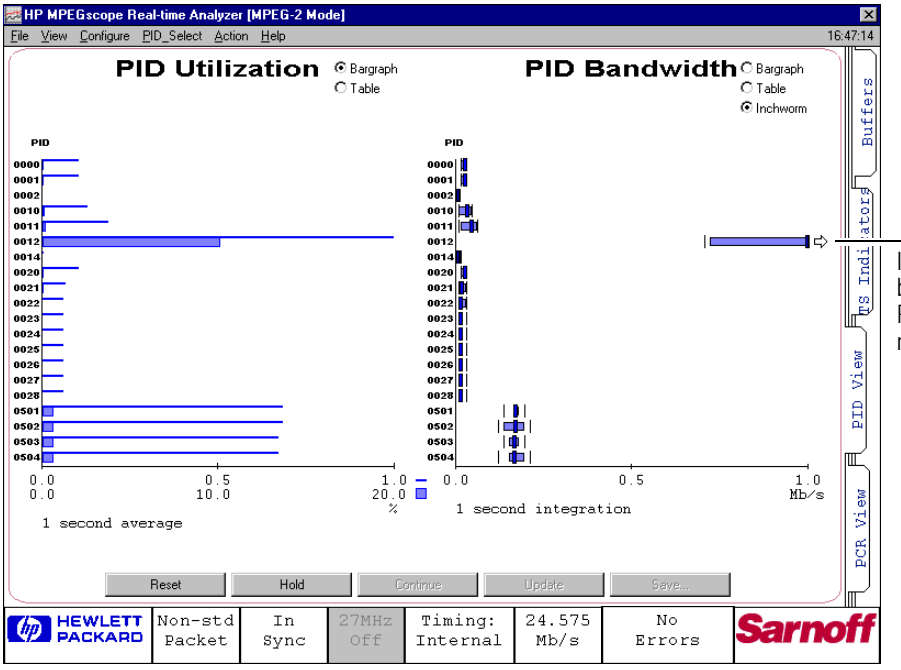

① Select **PID View.**




② If necessary, rescale graphs by changing ranges or modes.

The low end of this range is displayed as the upper scale on the **PID Utilization** graph (the thin blue line). The high end is displayed as the lower scale (the thick blue line).

## Rescaled Graphs



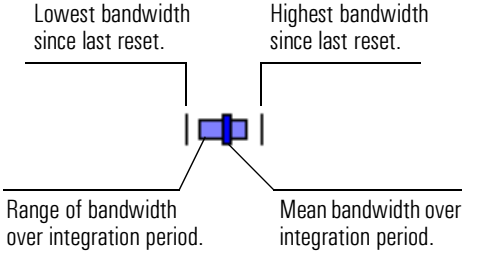

Indicates that the bandwidth for this PID exceeds the range.

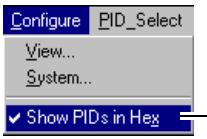
### PID Utilization Graph

The thin horizontal line displays PID utilization relative to the upper scale on the horizontal axis. It is most useful for low utilization PIDs, such as audio, PSI, and data PIDs.

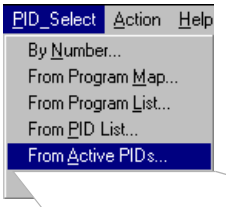
The thick horizontal line displays PID utilization relative to the lower scale on the horizontal axis. It is most useful for high utilization PIDs, such as video PIDs, which may exceed the range of the upper scale.

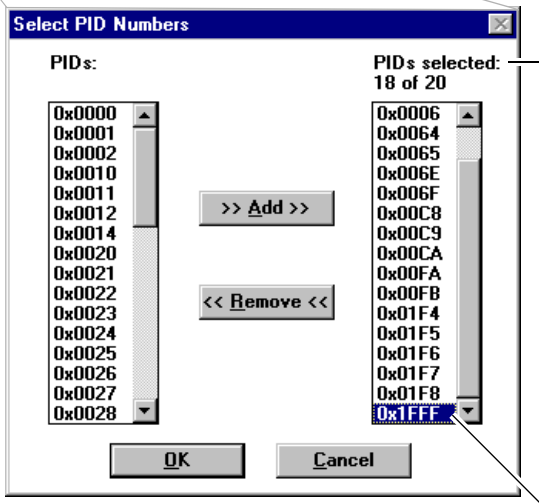
### PID Bandwidth Graph






③ If desired, display PID numbers in hexadecimal format.

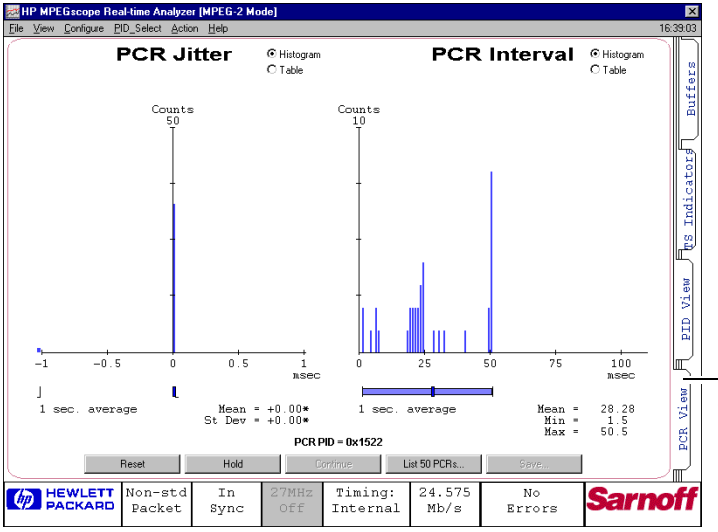


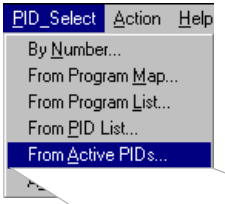

④ Change PID display by adding or removing from selection.

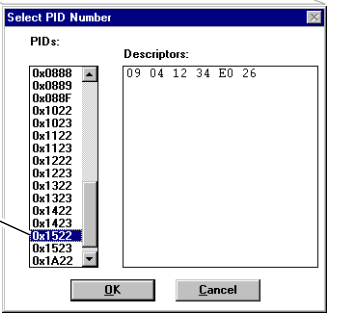
If the transport stream contains a large number of null packets, remove the null PIDs and rescale graphs.

# Step 7: Measure PCR jitter and interval



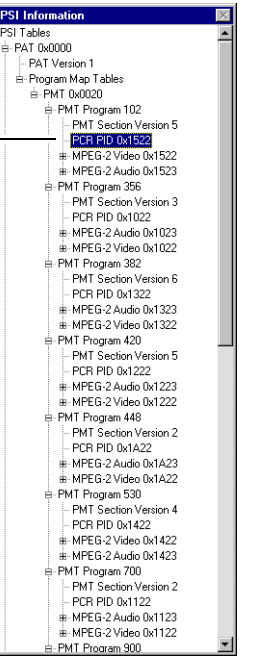

① Select **PCR View.**

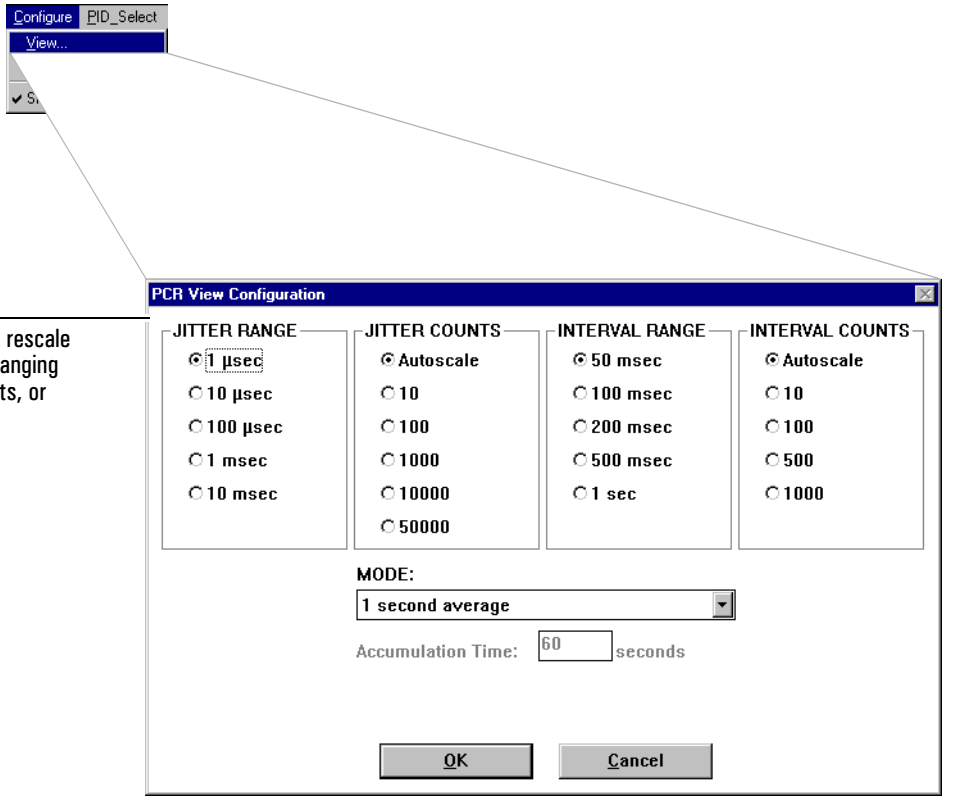



② Select the PCR PID to display.

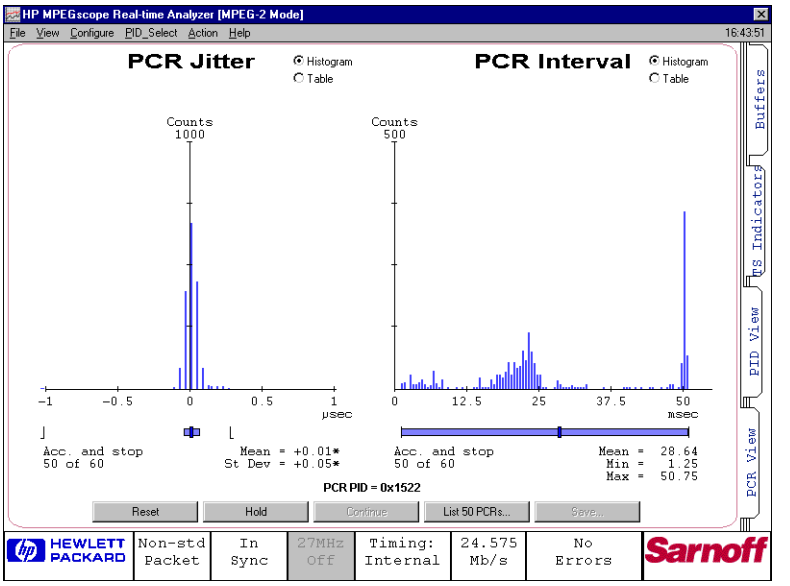
Check the PSI table display to see which PCR PIDs are on the stream.



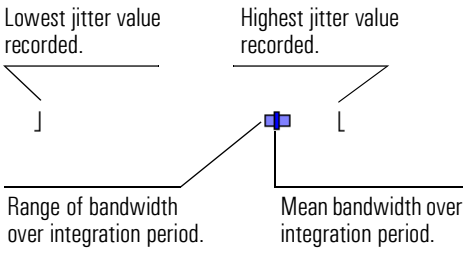



3 If necessary, rescale graphs by changing ranges, counts, or modes.

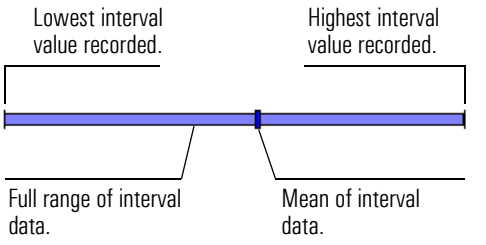
Rescaled Graphs



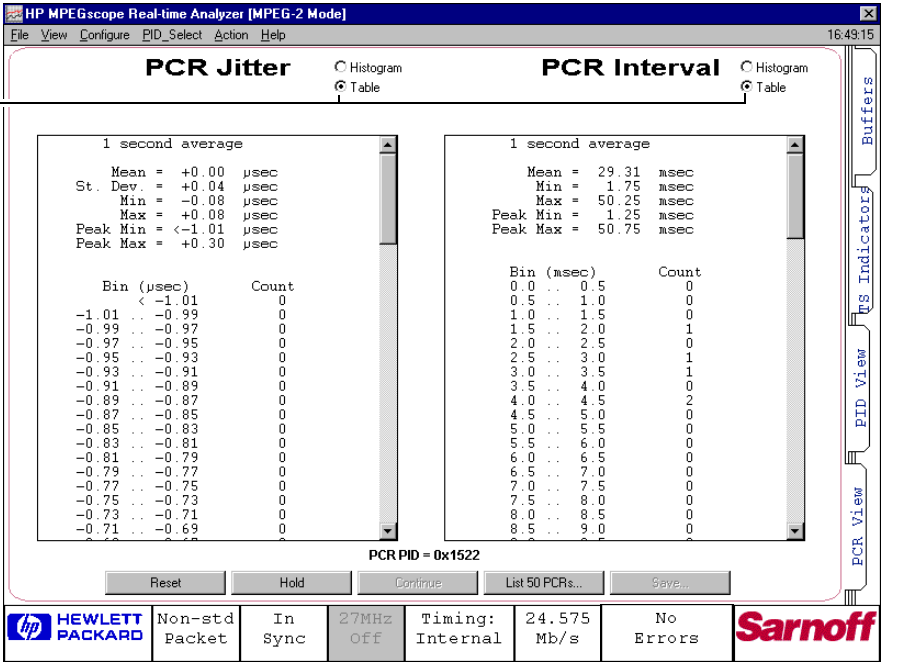

PCR Jitter Graph



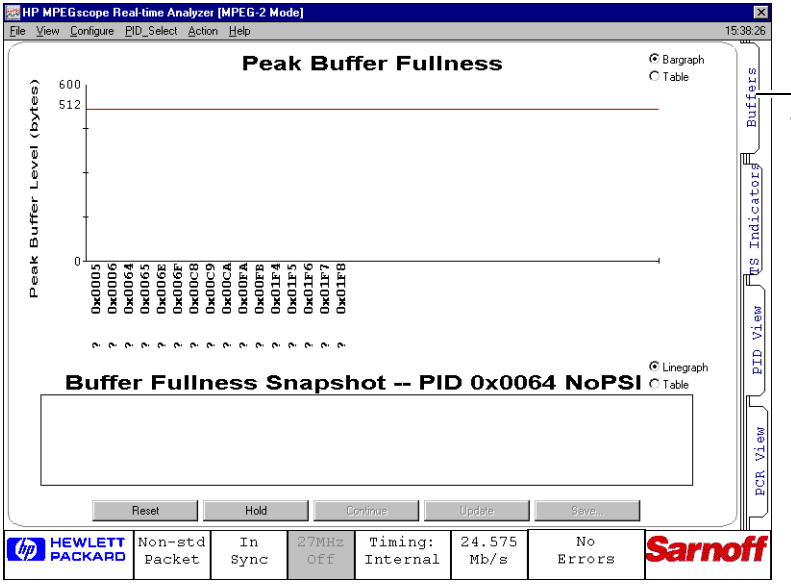

PCR Interval Graph



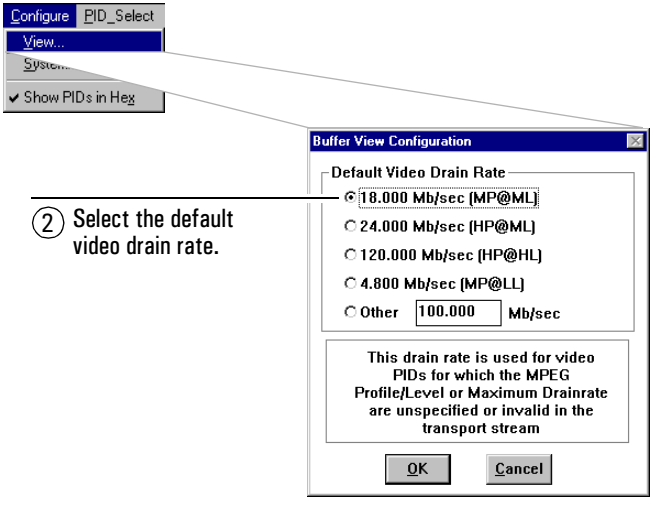
You can also view data in table format.



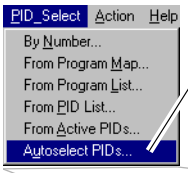
## Step 8: Monitor buffers




1 Select Buffer view.

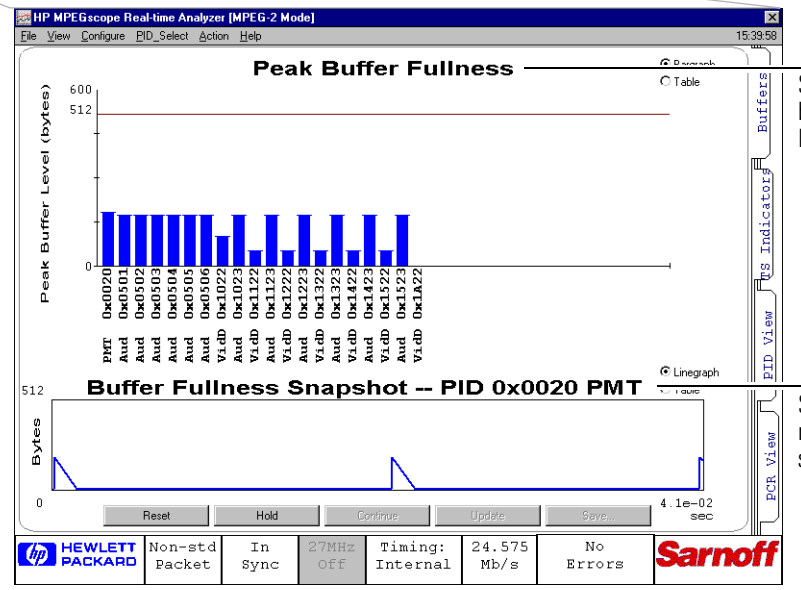



2 Select the default video drain rate.



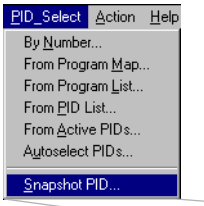

3 **Select which PID buffers to display.**

You can select PIDs individually from one of the **PID Select** lists or let the Real-time Analyzer autoselect PIDs for which buffers exist.

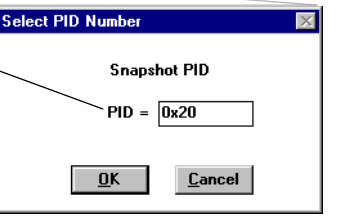



Shows the peak buffer level of each PID selected.

Shows the most recent trend of a selected buffer.




Enter a PID to change the PID buffer to display in the **Buffer Fullness Snapshot** graph.



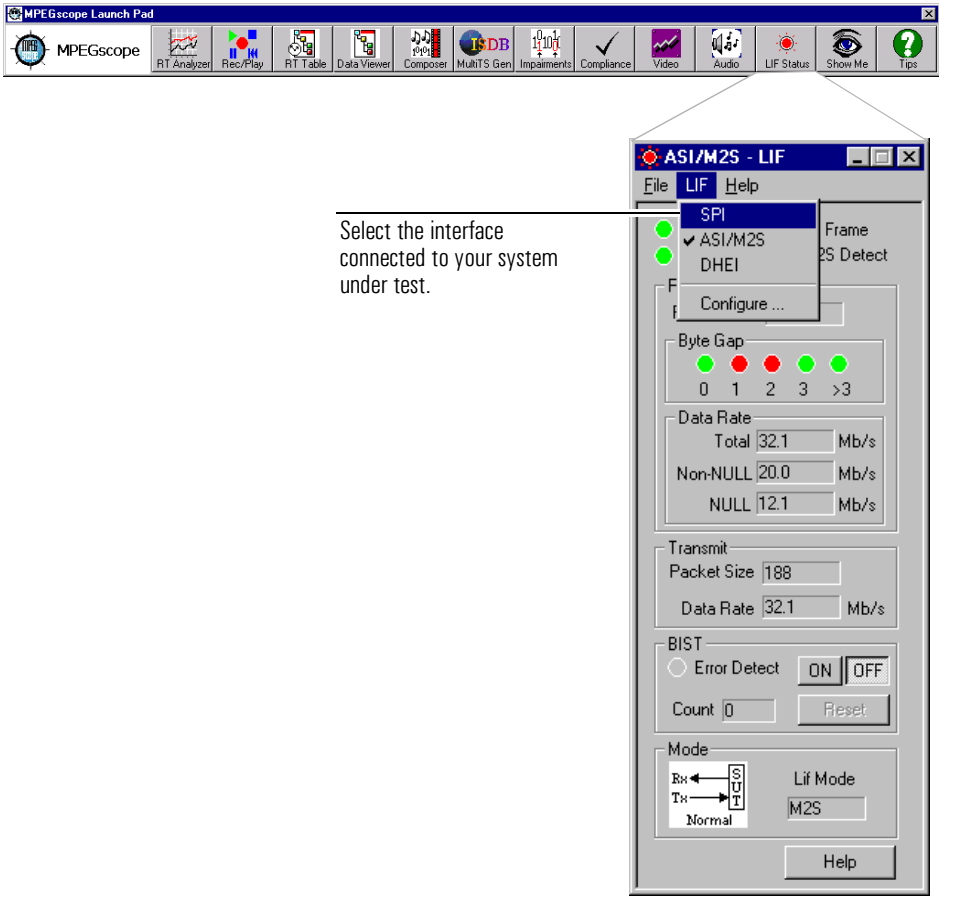
# 3

# Using the Recorder

# Recording a transport stream

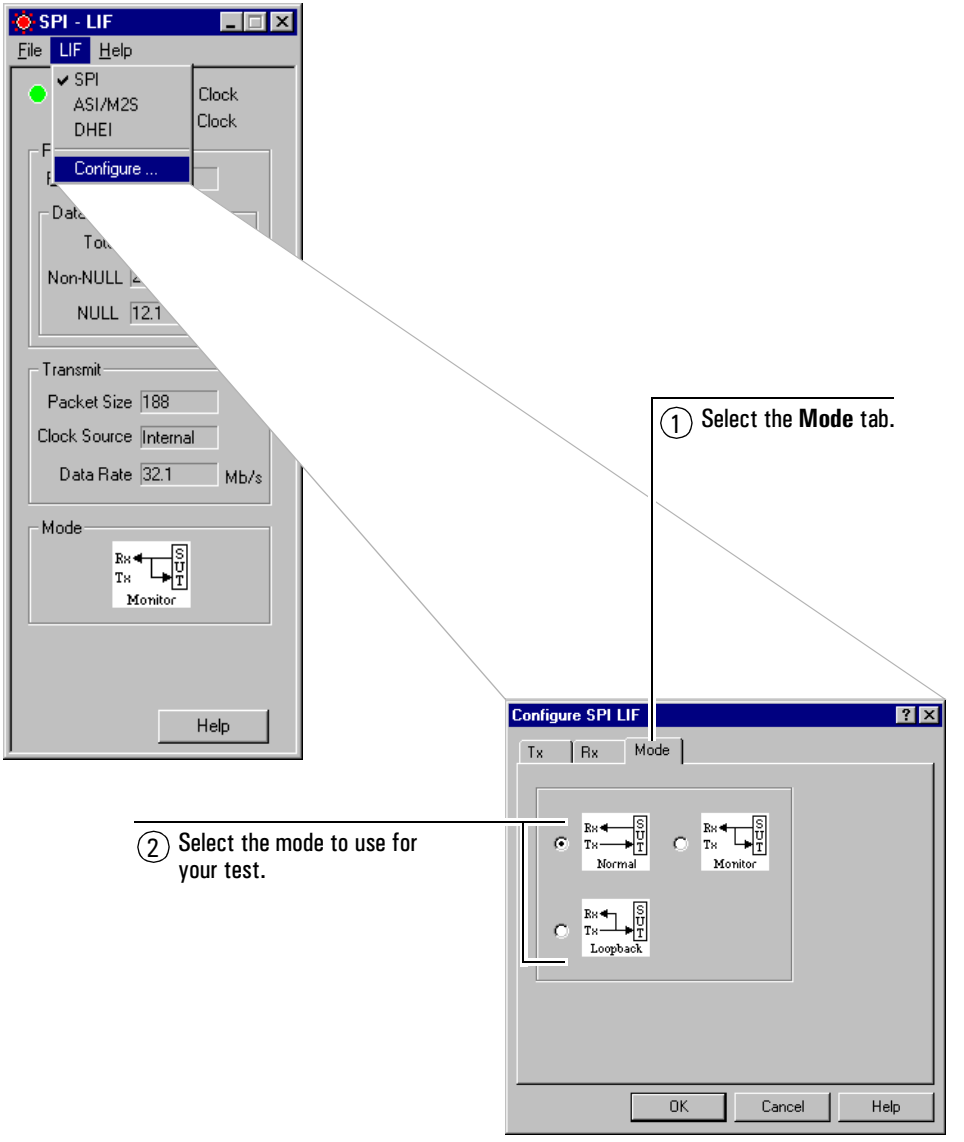

This example illustrates how to configure MPEGscope, set triggers, and record a transport stream.

## Step 1: Select the interface



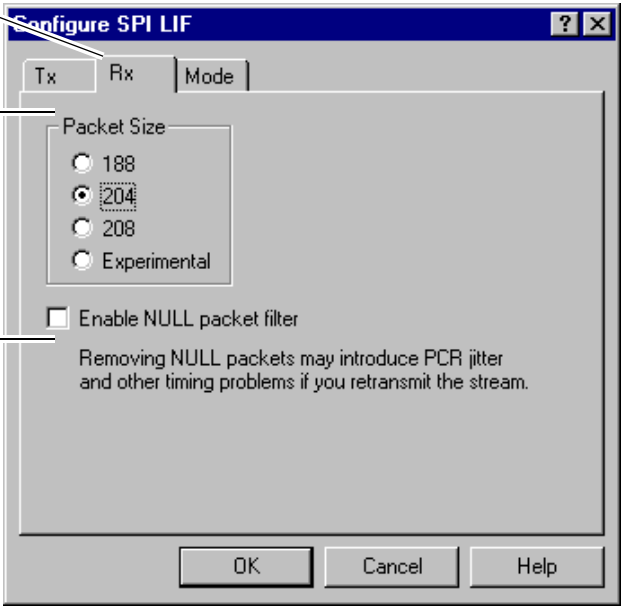

Select the interface connected to your system under test.

## Step 2: Configure the interface mode



## Step 3: Configure the receiver

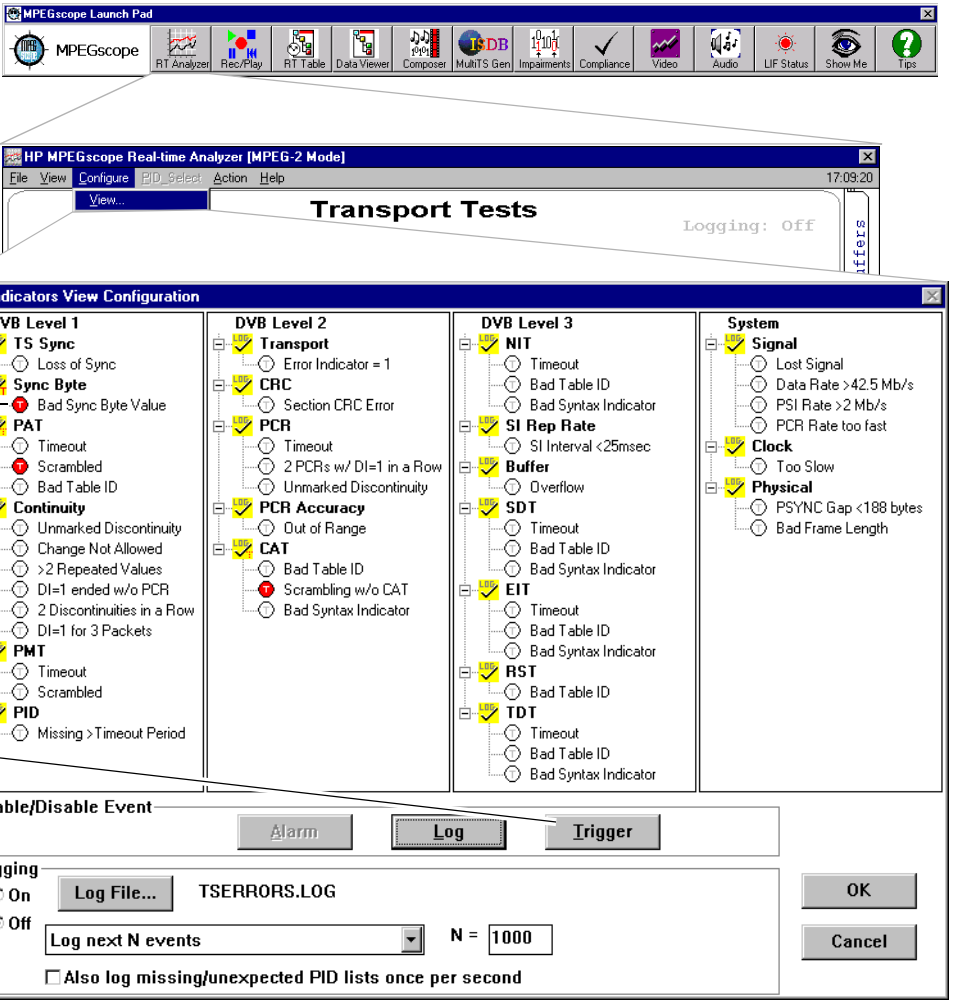
1. Select the **Rx** tab.
2. **Select receiver options.**
   Options differ depending on the interface you are using.




For information about each option, click on a field using the right mouse button.

What's This?

# Step 4: Set triggers




1. Select a TS indicator to trigger on.

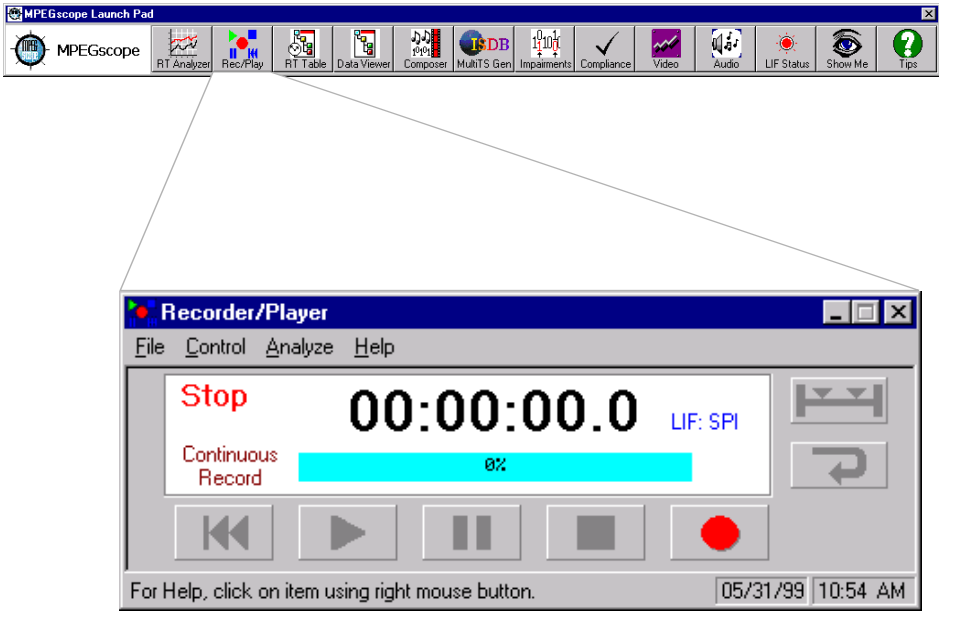
   A red **T** icon indicates a trigger has been set.

2. Press the **Trigger** toggle button to set the trigger.

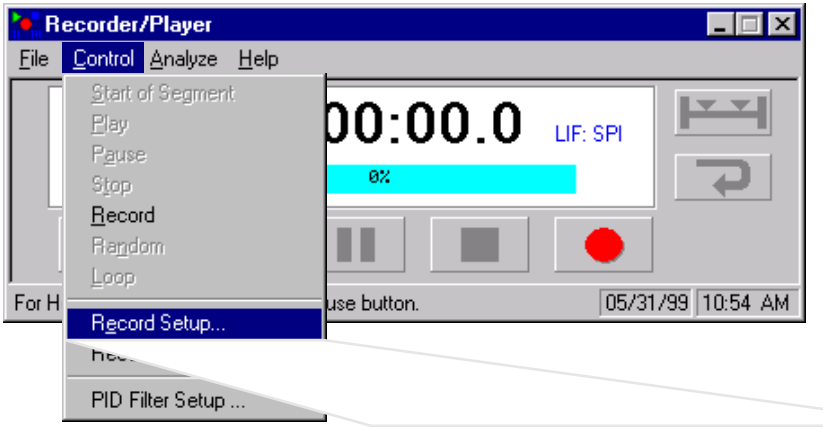
3. Repeat Steps 1 and 2 to set more triggers.

You can also enable or disable a trigger from the shortcut menu. Select the TS Indicator icon with the right mouse button to open this menu.

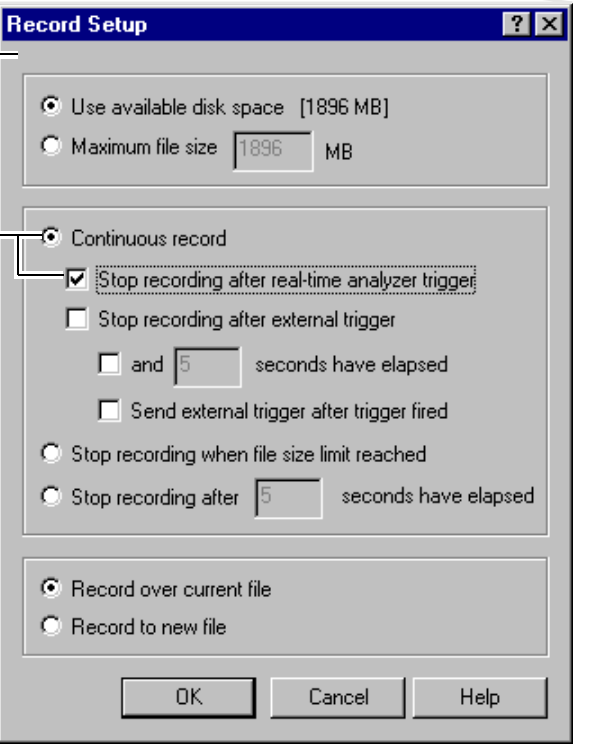
## Step 5: Start the Recorder/Player



## Step 6: Set up the recorder




Select the desired record options.

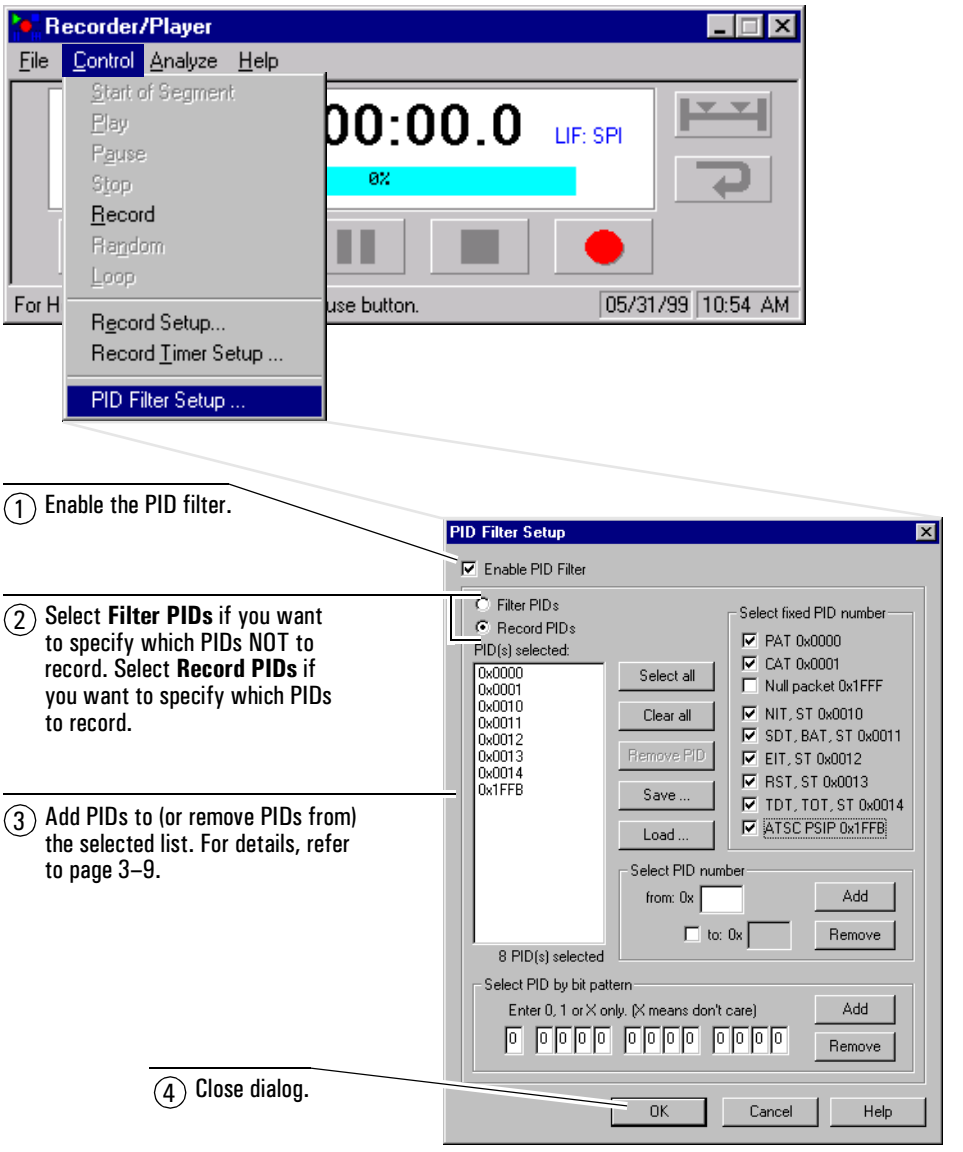



Select these options if you have set triggers from the Real-time Analyzer.

For information about each option, click on a field using the right mouse button.

## Step 7: Set PID filters

If you want to record only certain PIDs, MPEGscope allows you to specify which PIDs to record or which PIDs *not* to record.

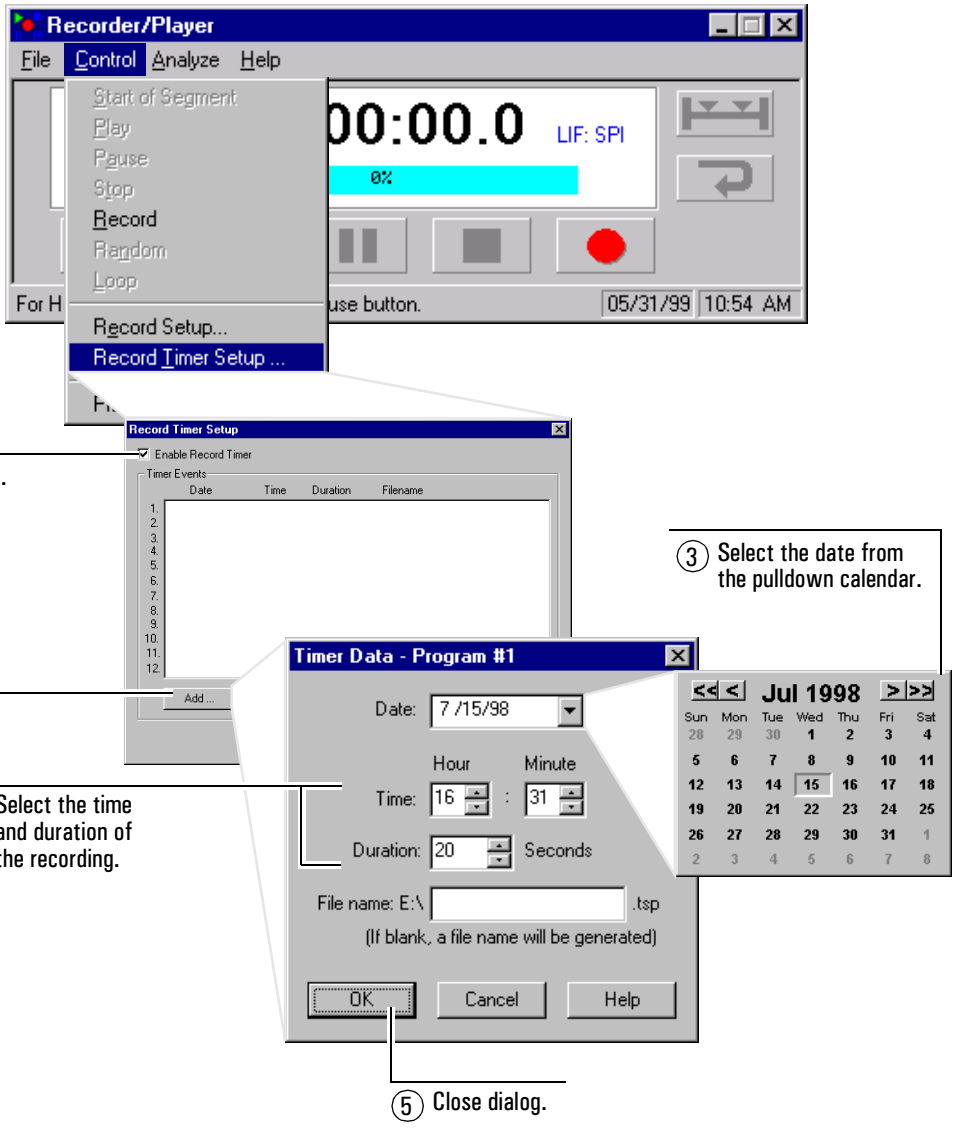


There are a number of ways to add or remove PIDs from the **PID(s) selected** list.

| | |
|---|---|
| **Buttons to the right of the selected list** | Add all PIDs, remove all PIDs, or highlight a PID from the list and remove only that PID. Once you have a PID list defined, you can save it as a special PID Filter Setup (**.pfs**) file, or you can load a previously-defined **.pfs** file. |
| **Select fixed PID number** | Add or remove specific tables or null packets from the list. |
| **Select PID number** | Specify a contiguous range of PIDs to add or remove. |
| **Select PID by bit pattern** | Add or remove a group of PIDs by defining a mask to match a specific bit pattern. Enter an 'X' to indicate that the bit may be any value. For example, the mask '1XX' will match '100', '101', '110', and '111'. |

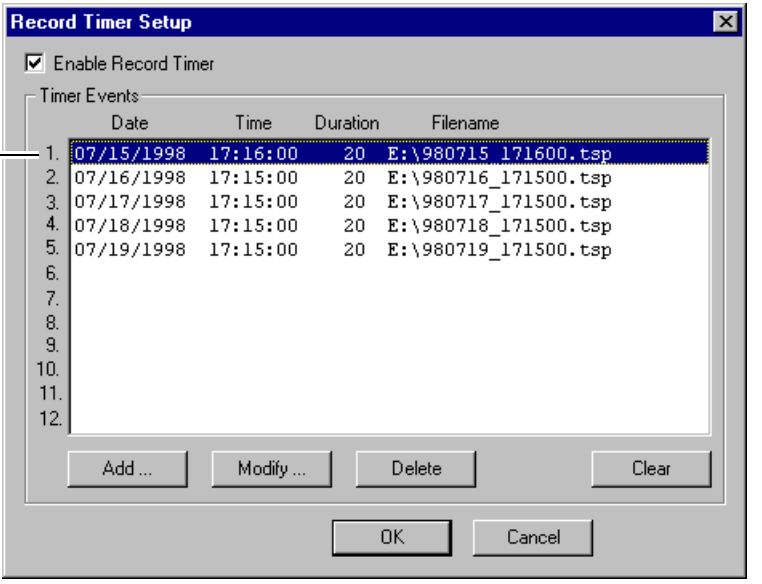
# Step 8: Set the record timer

You can configure MPEGscope to record at specific dates and times in the future.



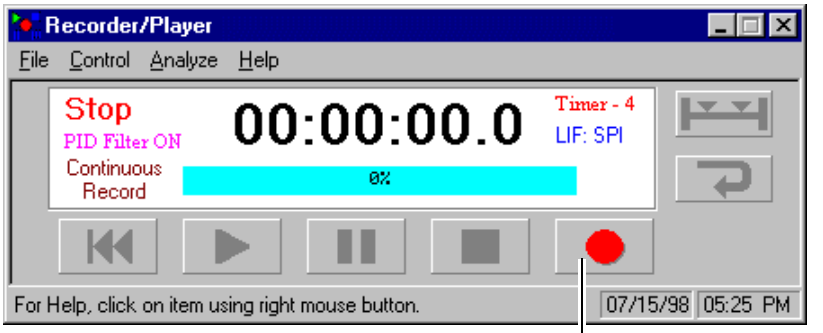

**For each recording:**

1. Enable the record timer.
2. Add a timer event.
3. Select the date from the pulldown calendar.
4. Select the time and duration of the recording.
5. Close dialog.




You can also modify or delete any selected timer event.

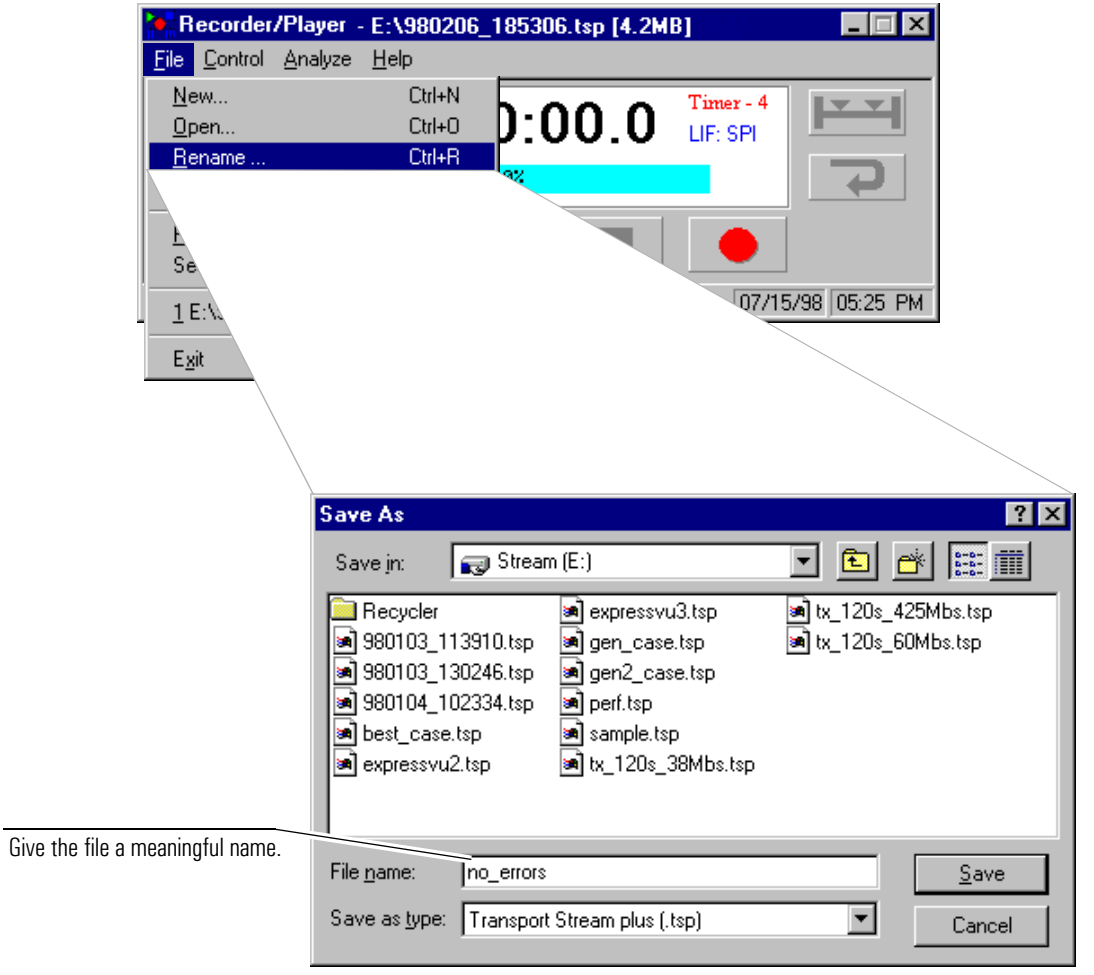
## Step 9: Begin recording




Press the **Record** button.

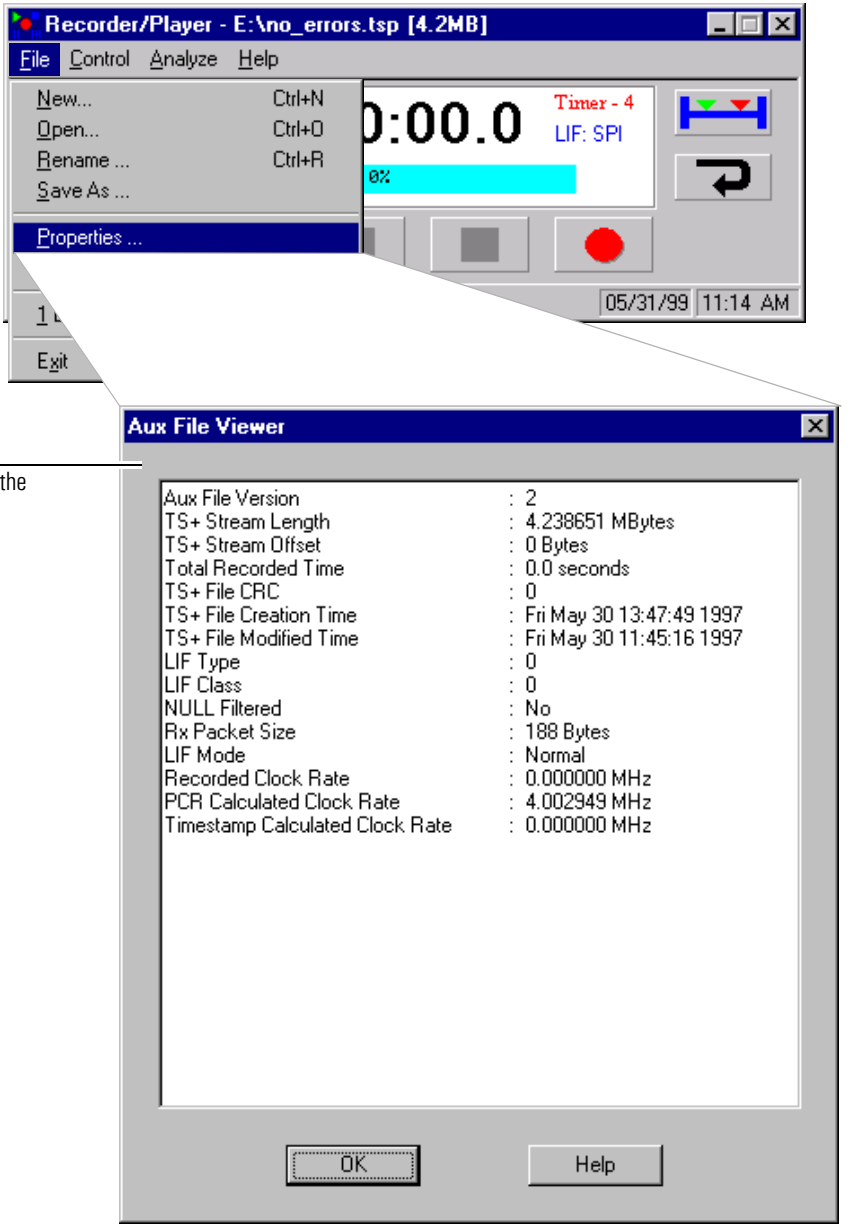
The selected record options determine when recording stops. You can also manually stop recording at any time by pressing the **Stop** button.

## Step 10: Rename the recorded file




Give the file a meaningful name.

## Step 11: View information about the recorded data




View information stored in the auxiliary file.

## Aux File Viewer Fields

| | |
|---|---|
| Aux File Version | Current version of the auxiliary file format. |
| TS+ Stream Length | Length of the recorded transport stream in megabytes. |
| TS+ Stream Offset | Starting offset in bytes of the transport stream file.<br>Note: The starting offset is important when you want to retransmit a file that has been recorded in **Continuous Record** mode. In this mode, MPEGscope overwrites the file when the maximum file size is reached. In order to retransmit the file, you need to know the offset, i.e., where in the file the recorded stream begins. For example, if the offset is 204 bytes, the beginning of the stream occurs on the 205th byte. |
| Total Recorded Time | Number of seconds of recorded time. |
| TS+ File CRC | Cyclical Redundancy Check carried out by the operating system when the file was created. |
| TS+ File Creation Time | Time the file was created. |
| TS+ File Modified Time | Time the file was modified. |
| LIF Type | Indicates the type of interface, as follows:<br>SPI Synchronous Parallel Interface<br>ASI Asynchronous Serial Interface<br>M2S MPEG-2 Serial Interface<br>DHEI Digicable Headend Expansion Interface |
| LIF Class | Indicates the class of interface, as follows:<br>Async Asynchronous<br>Sync Synchronous |
| NULL Filtered | **YES** indicates NULL packets were filtered out during recording.<br>**NO** indicates NULL packets were not filtered out during recording. |
| Rx Packet Size | Size of recorded transport stream packets (188, 204, or 208 bytes). |
| LIF Mode | Indicates whether the stream was recorded with the interface in **Normal**, **Monitor**, or **Loopback** mode. |
| Recorded Clock Rate | Actual clock rate of recorded stream. If the file was not recorded with MPEGscope or if the original auxiliary file was deleted, this number will be 0. |
| PCR Calculated Clock Rate | Clock rate of the recorded stream derived from packet PCR values. |
| Timestamp Calculated Clock Rate | Clock rate of the recorded stream derived from the timestamps that MPEGscope appends to each packet during recording. |

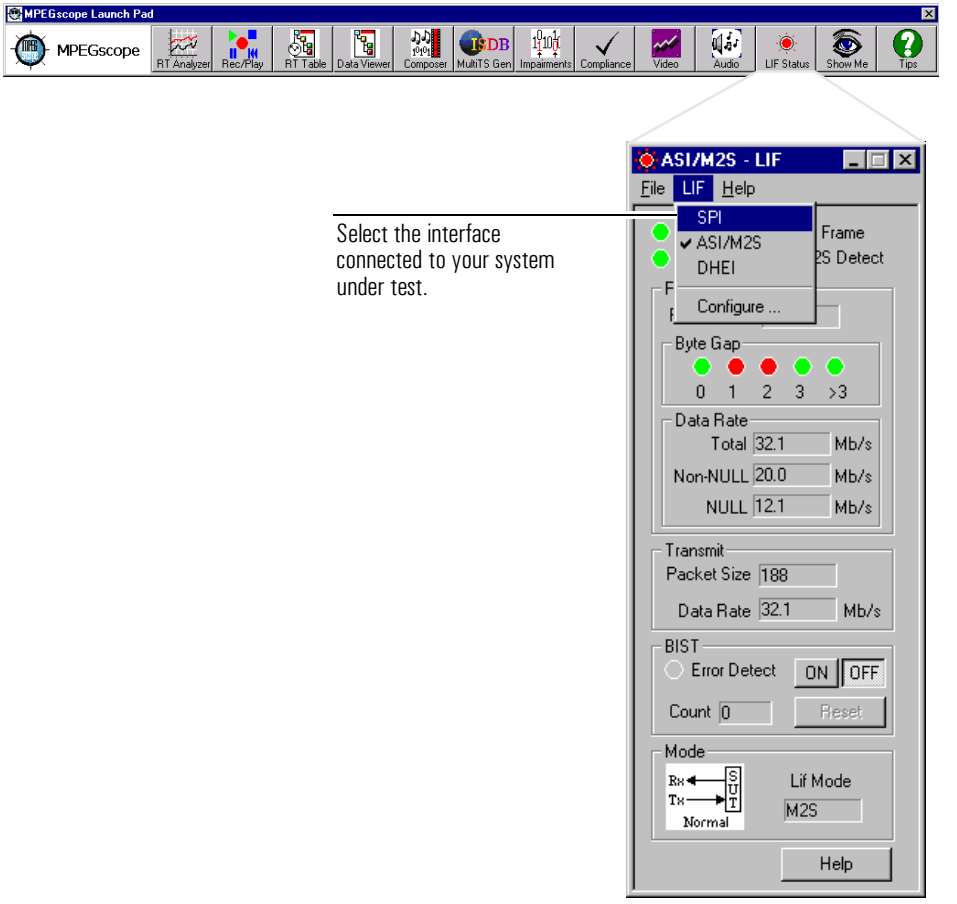
# 4

# Using the Player

# Playing a transport stream

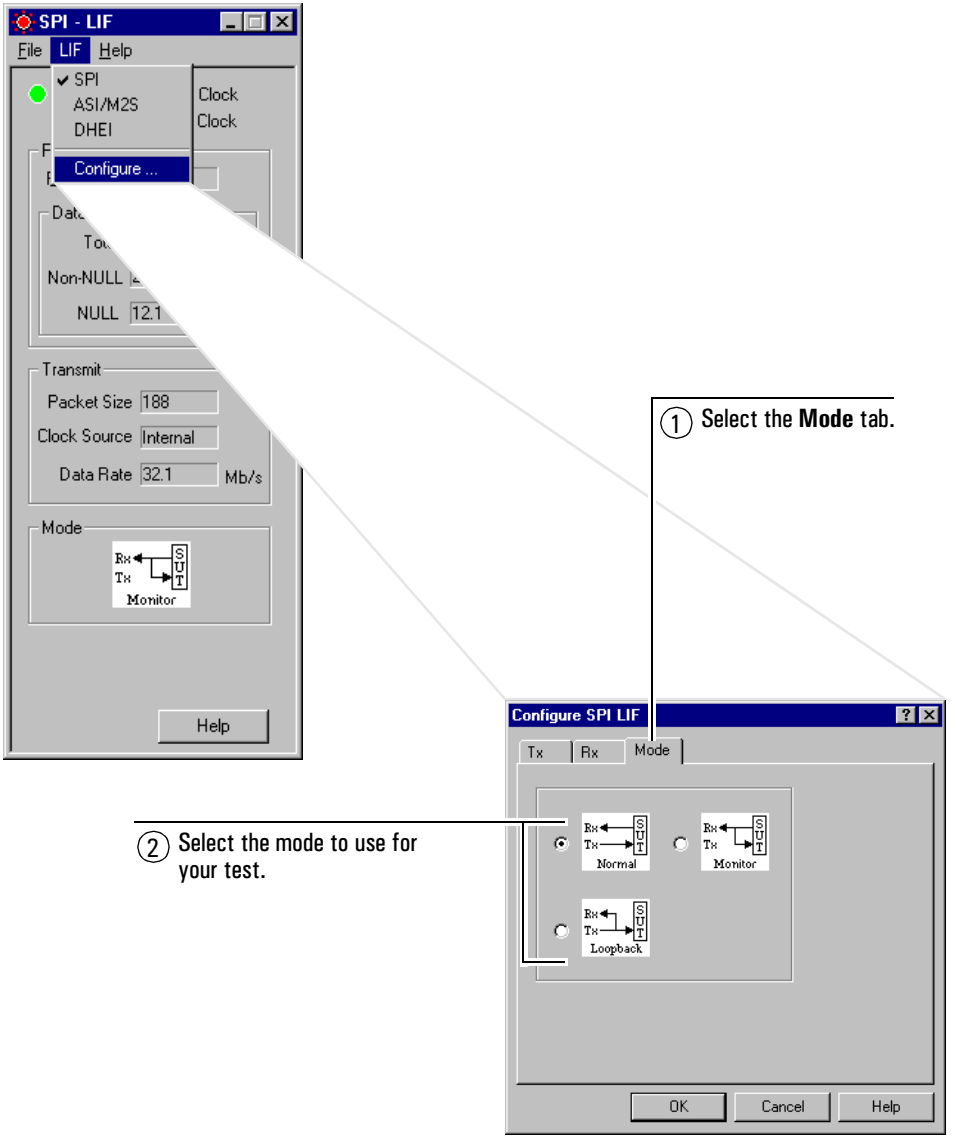

This example illustrates how to open a transport stream file and play a segment repeatedly.

## Step 1: Select the interface




Select the interface connected to your system under test.

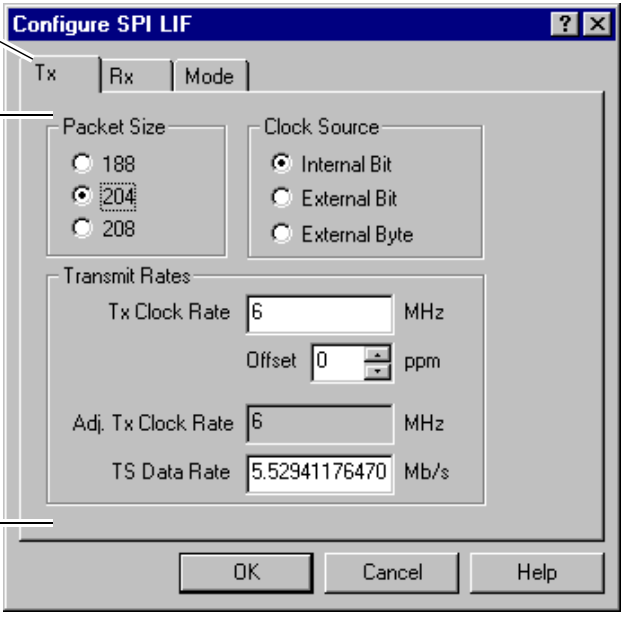
## Step 2: Configure the interface mode



## Step 3: Configure the transmitter

If you open a transport stream file after setting transmitter options, the default packet size and transmit rate will change to the packet size and transmit rate of the data in the loaded file.

1. Select the **Tx** tab.
2. **Select transmitter options.**
   Options differ depending on the interface you are using.



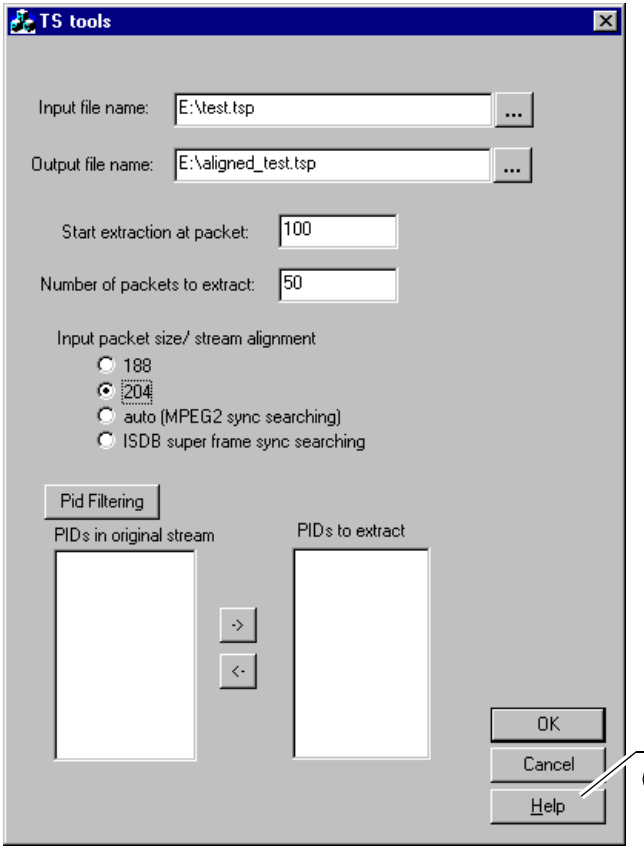

For information about each option, click on a field using the right mouse button.

What's This?

## Step 4: Align the transport stream file

Before opening the file you want to transmit from the Recorder/Player, you can align it with a packet-aligning utility called TS Tools that discards partial packets at the beginning and end of a transport stream file so the file starts and finishes at a packet boundary. TS Tools also allows you to truncate a transport stream file to the segment you are interested in, or to extract only the PIDs you want to analyze.

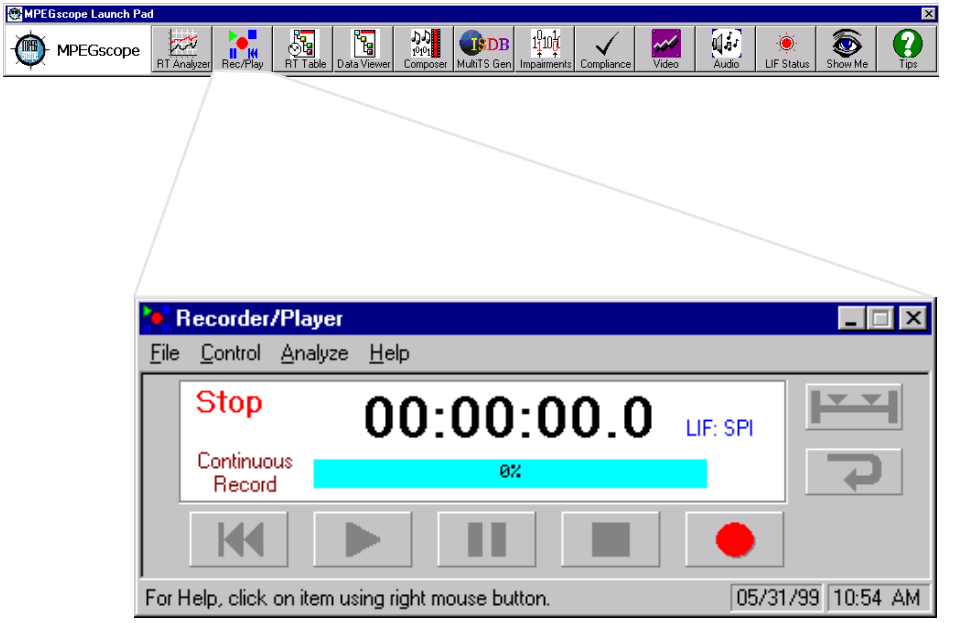
① Launch TS Tools by double clicking on tsutil.exe in the C:\HP-Apps\Resources\bin directory from Windows® NT Explorer.




② Press the Help button for complete instructions on using the TS Tools utility.

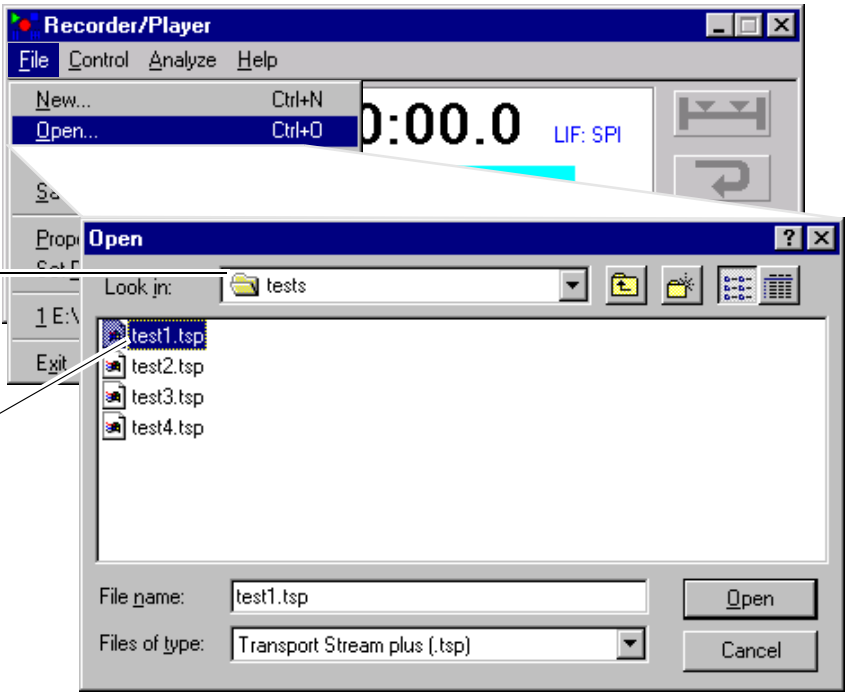
PID filtering is useful when you want to analyze only certain PIDs in the stream, for example, only the PATs or PMTs. However, because the TS Tools utility does not restamp PCR values, PID-filtered streams are not suitable for playing out.

## Step 5: Start the Recorder/Player



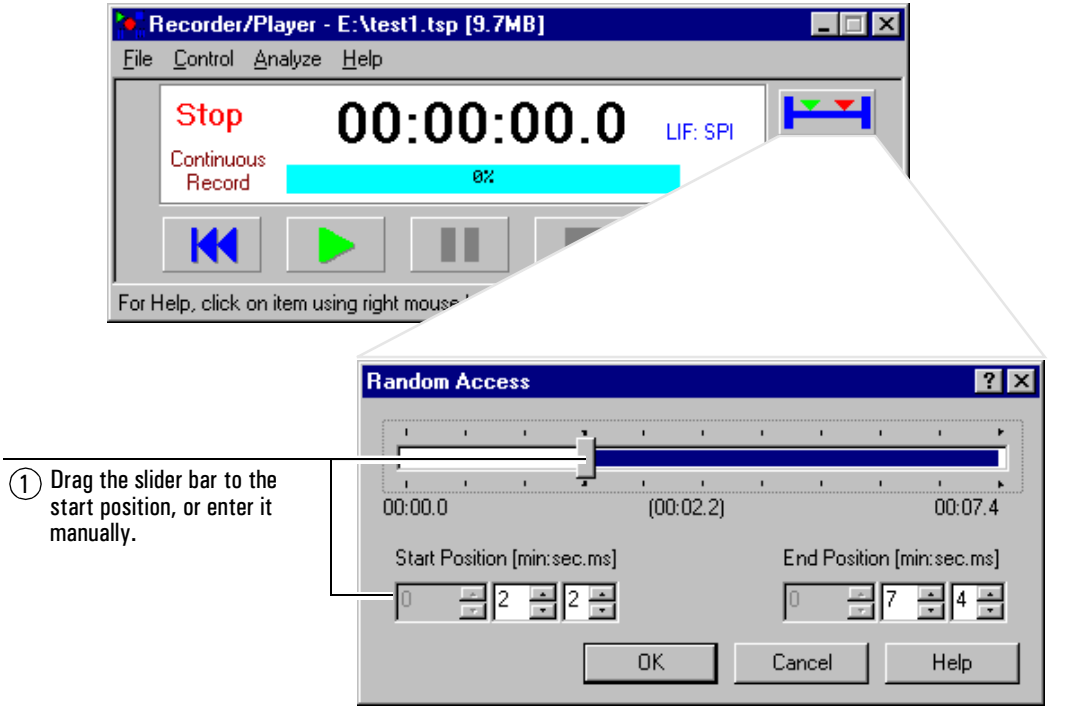
## Step 6: Open (or record) a transport stream file

You can open a file then play it, or you can play a file you have just recorded. This step shows how to open a file. Refer to “Recording a transport stream”, page 3–2 for instructions on how to record a transport stream.

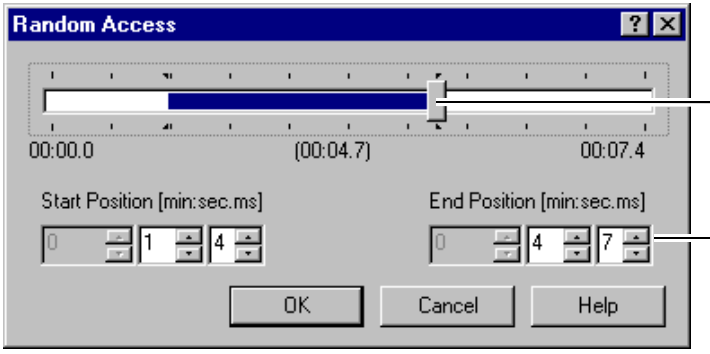



If you open a file from a location other than E: drive, MPEGscope first copies the file to E: drive.

## Step 7: Select a segment to play




① Drag the slider bar to the start position, or enter it manually.

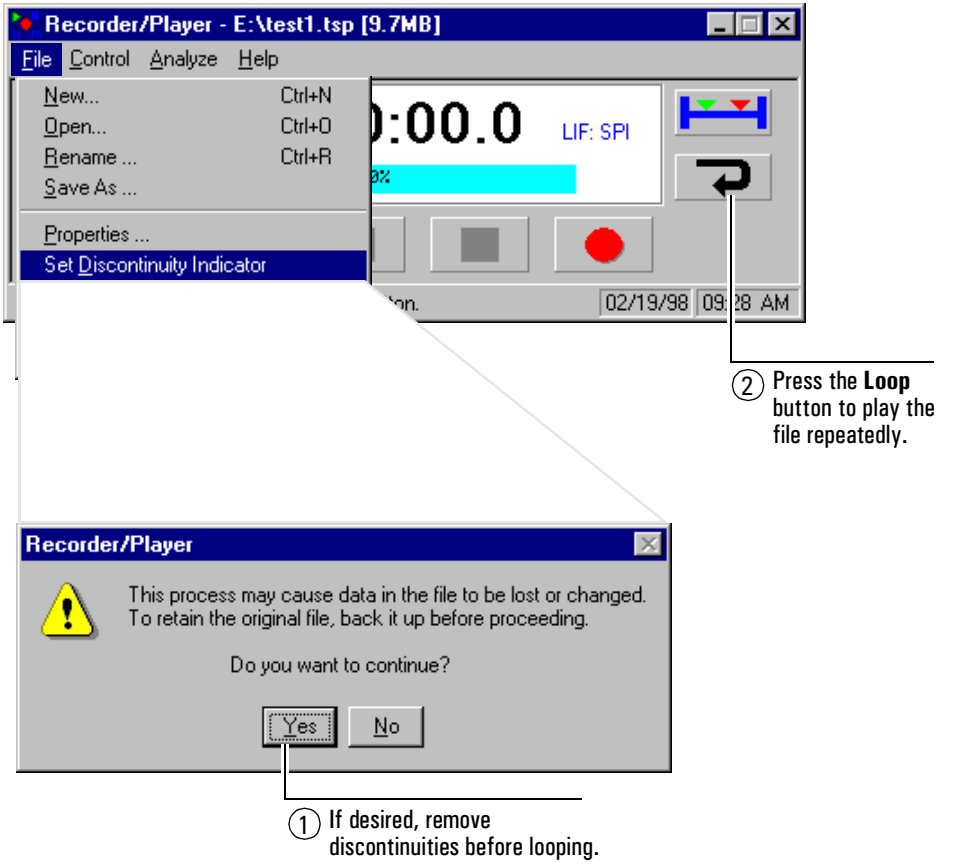



② Holding down the **Shift** key, drag the slider bar to the end position, or enter it manually.

MPEGscope cannot remove discontinuities from a file segment when transmitting the file in **Loop Play** mode.

If you do not select a segment, MPEGscope will play the entire file.

## Step 8: Loop the data

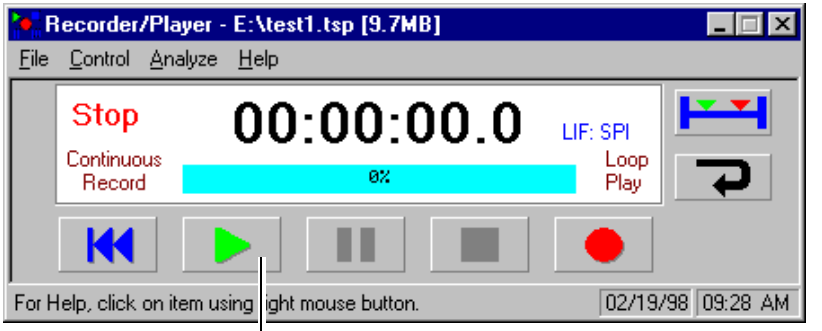



If you remove discontinuities, MPEGscope will ensure that the last packet on each PID has a **continuity_counter** value of one less than the first packet. To achieve this, the file may be permanently changed, as packets may be lost, changed, or replaced with null packets. If you want to keep the original file, you should first back it up. You can only remove discontinuities on a file of 2 GB or less.

If you play a stream in **Loop Play** mode without removing discontinuities and also monitor it from the Real-time Analyzer, the Real–time Analyzer will report a **continuity_counter** error for each PID analyzed.

Note: MPEGscope cannot remove discontinuities from a file segment.

## Step 9: Begin playing




Press the **Play** button.

You can manually stop playing at any time by pressing the **Stop** button.

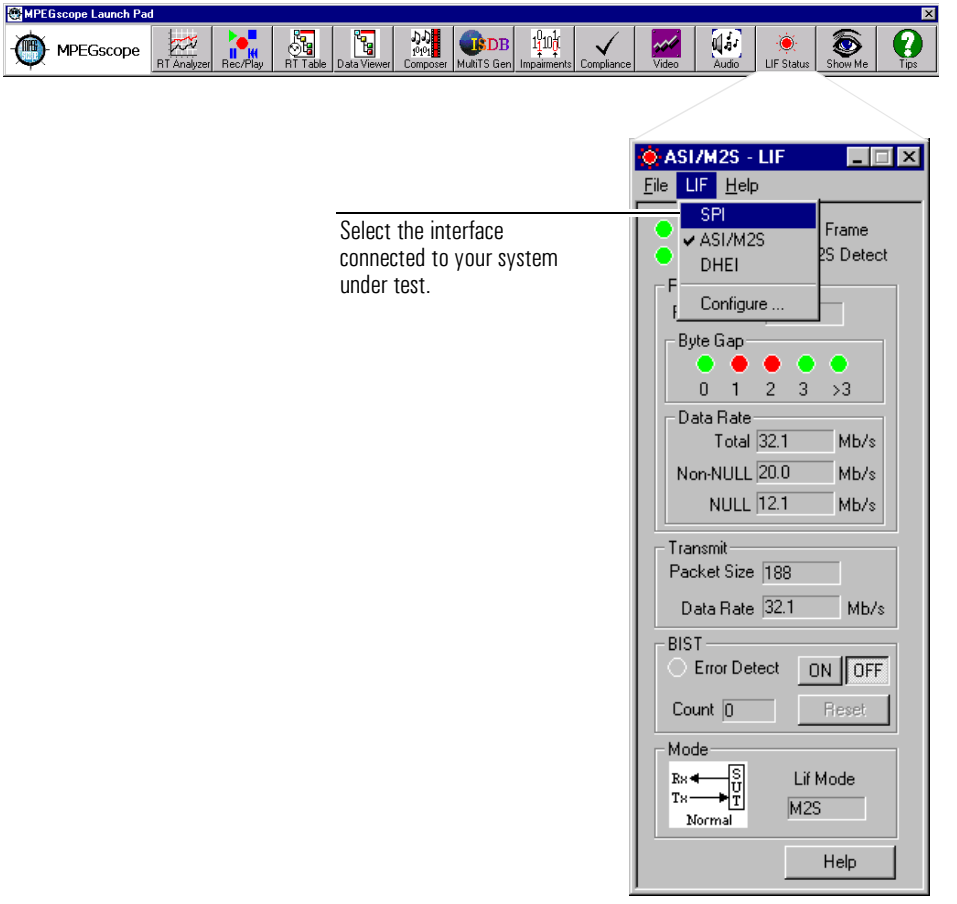
# 5

# Using the Real-time Table Analyzer

# Viewing real-time table decodes

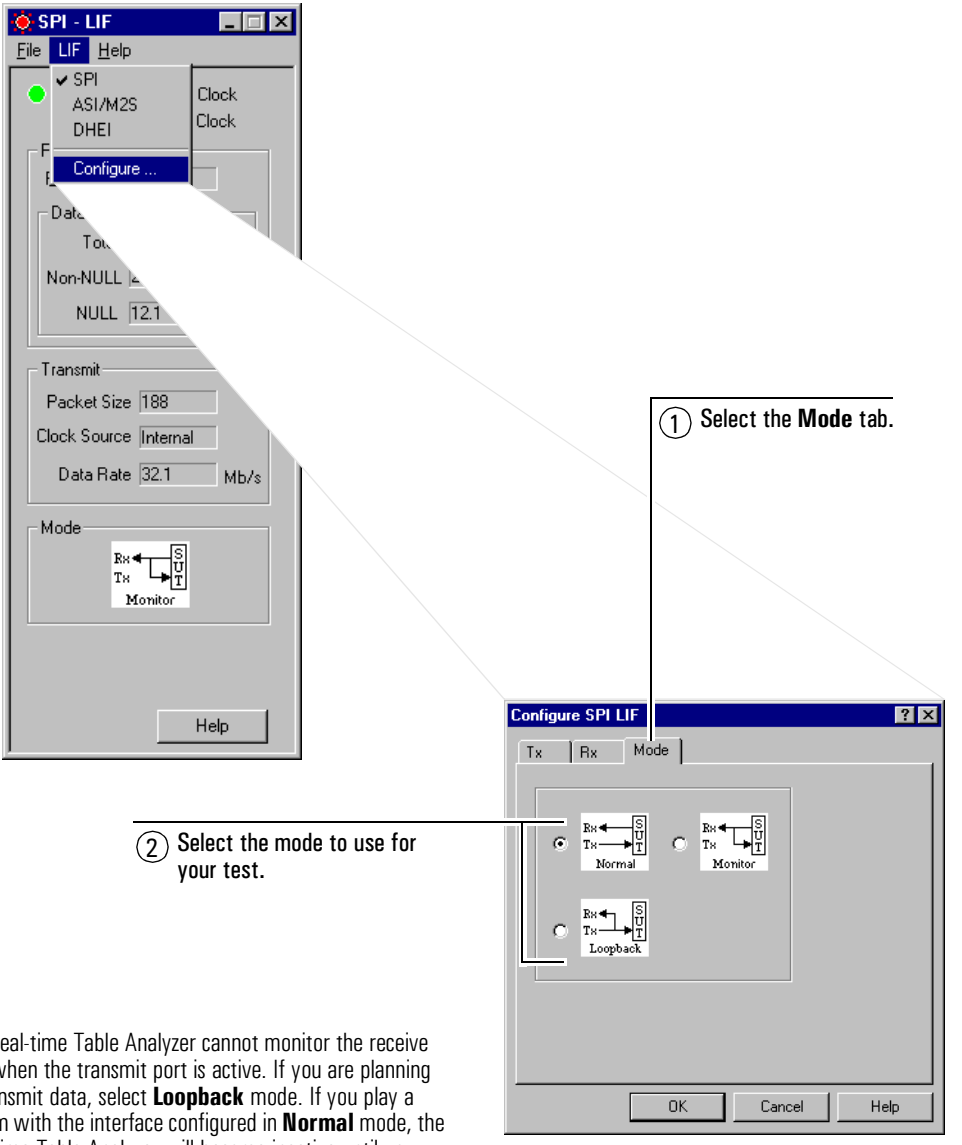

This example illustrates how to view incoming MPEG, DVB, or ATSC table decodes as they update in real time.

## Step 1: Select the interface




Select the interface connected to your system under test.

## Step 2: Configure the interface mode

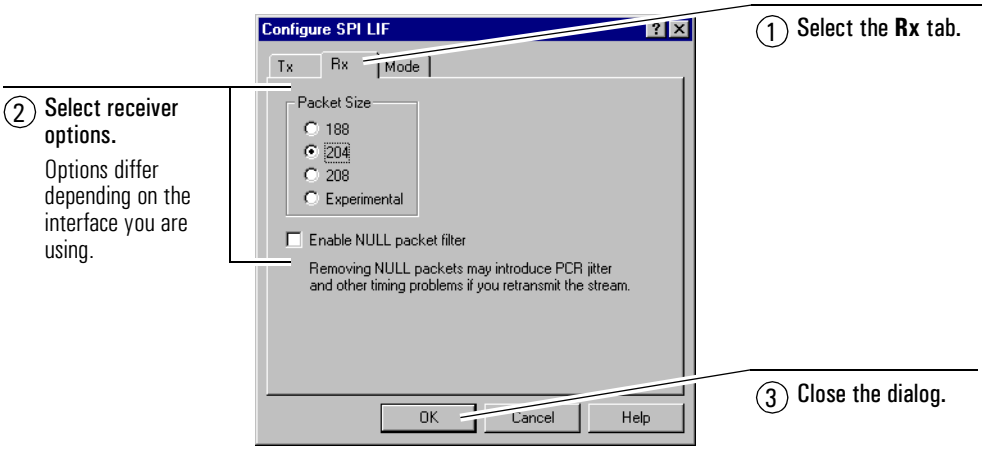



The Real-time Table Analyzer cannot monitor the receive port when the transmit port is active. If you are planning to transmit data, select **Loopback** mode. If you play a stream with the interface configured in **Normal** mode, the Real-time Table Analyzer will become inactive until you stop transmitting.

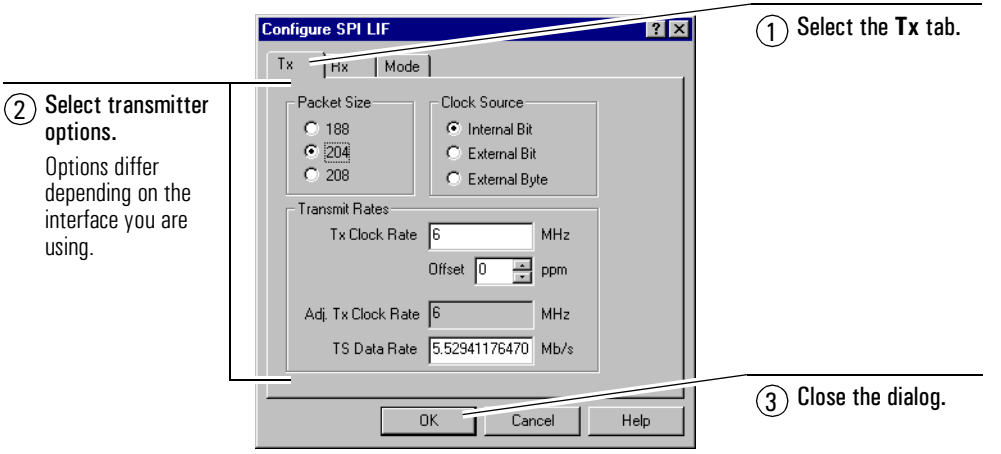
## Step 3: Configure the receiver or transmitter

The Real-time Table Analyzer can either monitor the incoming stream you are recording or the outgoing stream you are transmitting (providing the interface is in **Loopback** mode).

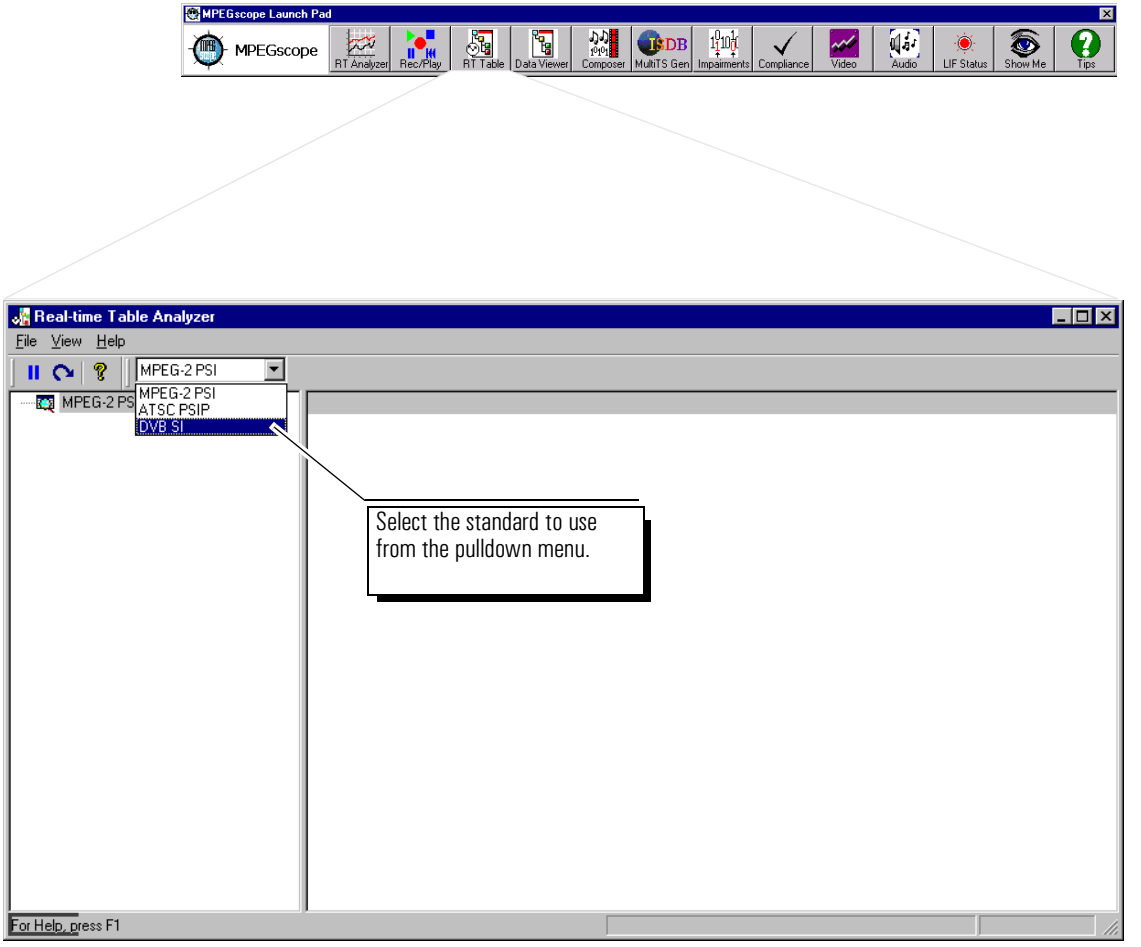
**If you are recording...**



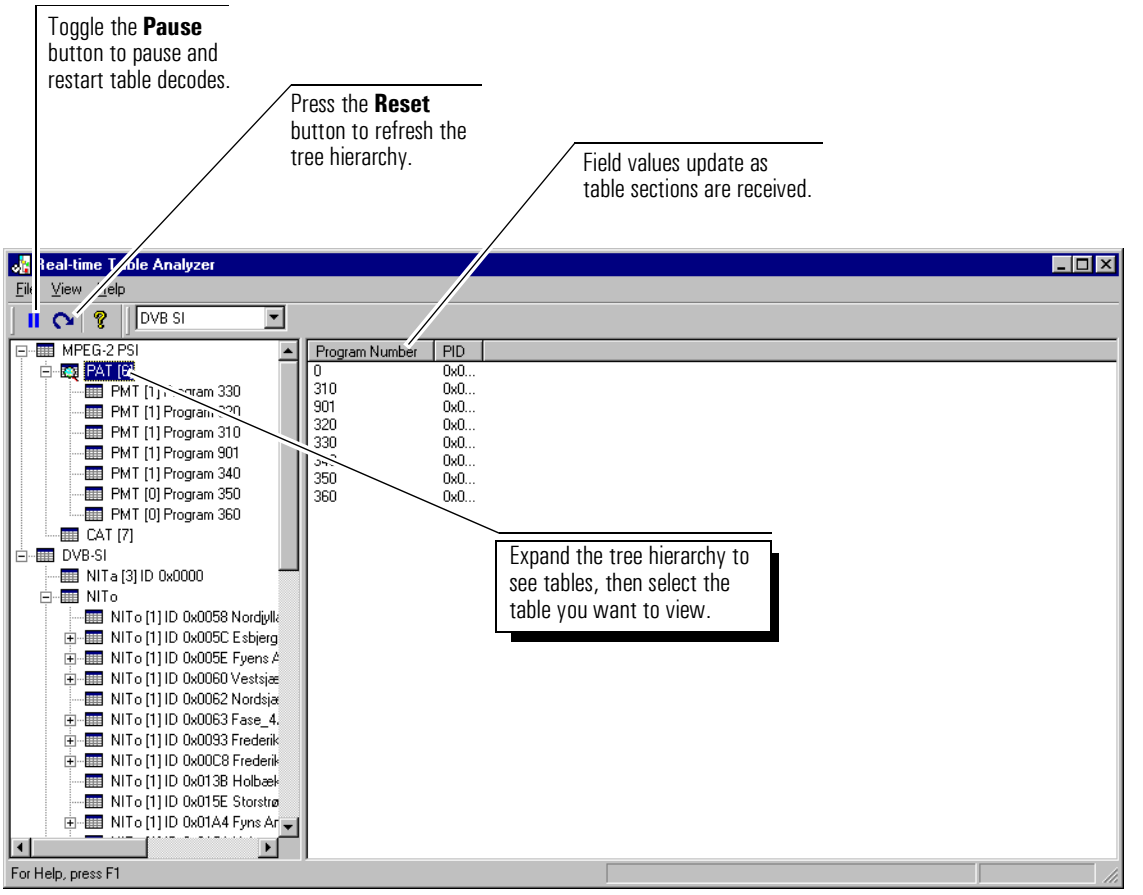

**If you are transmitting...**



## Step 4: Start the Real-time Table Analyzer

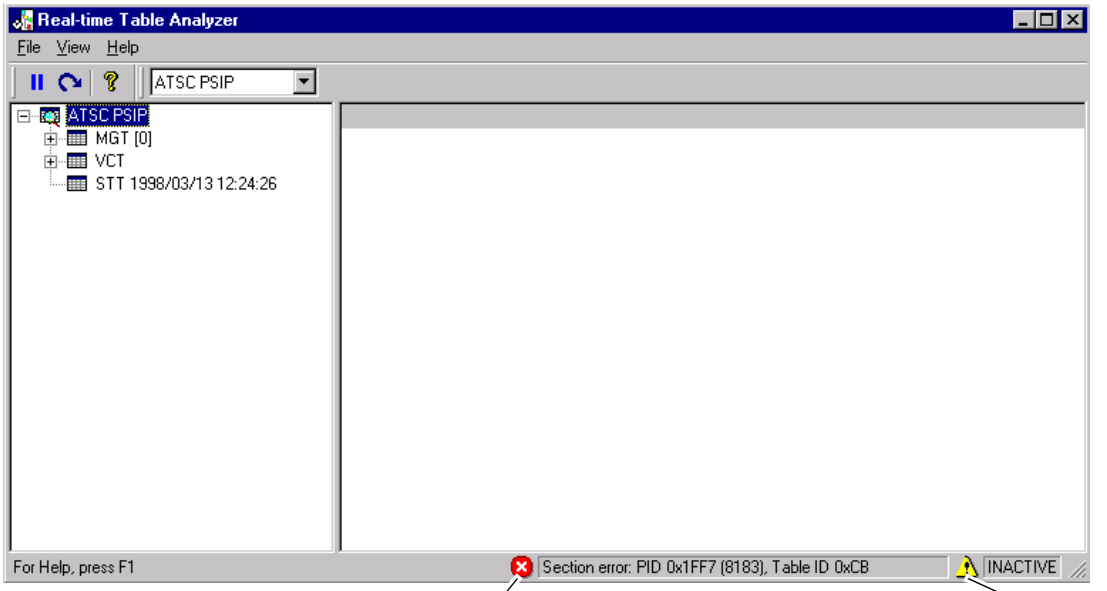


## Step 5: View table decodes



## Troubleshooting problems

The status bar at the bottom of the dialog can indicate two types of problems. An error icon in the first status area means that a serious error has prevented the Real-time Analyzer from decoding a table section. A description of the error location displays immediately to the right. A warning icon in the second status area alerts you whenever the Real-time Analyzer is not actively monitoring the receive port.



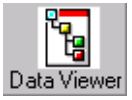

When the error icon displays, the Real-time Analyzer has encountered an error that prevents decoding of a table section.

When the warning icon displays, the Real-time Analyzer is unable to monitor the receive port. This occurs if you transmit data while the interface is configured in **Normal** mode. To avoid this problem, configure the interface in **Loopback** mode before transmitting.

# 6

# Using the Protocol Data Viewer

# Analyzing a transport stream

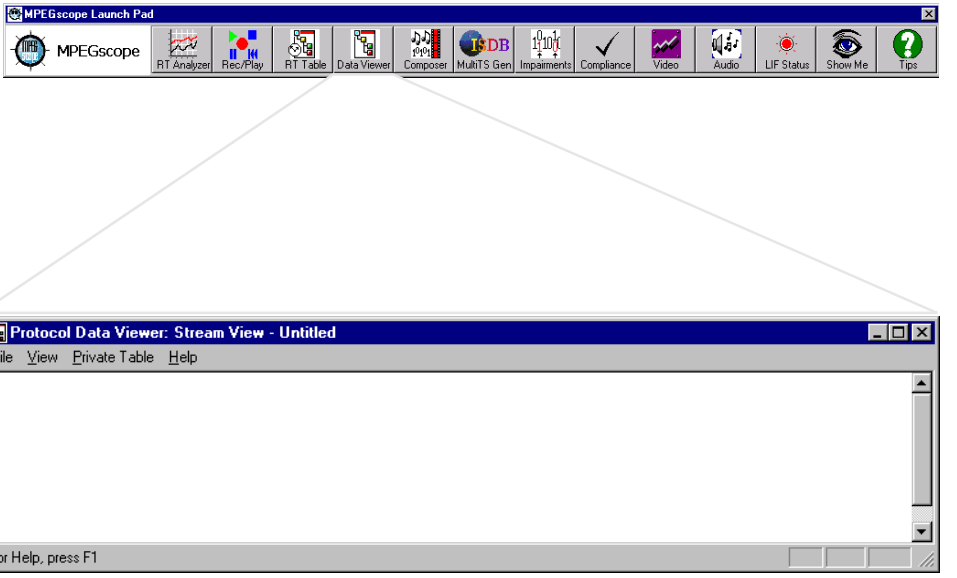



This example illustrates how to analyze a transport stream with the Protocol Data Viewer.

## Step 1: Start the Protocol Data Viewer

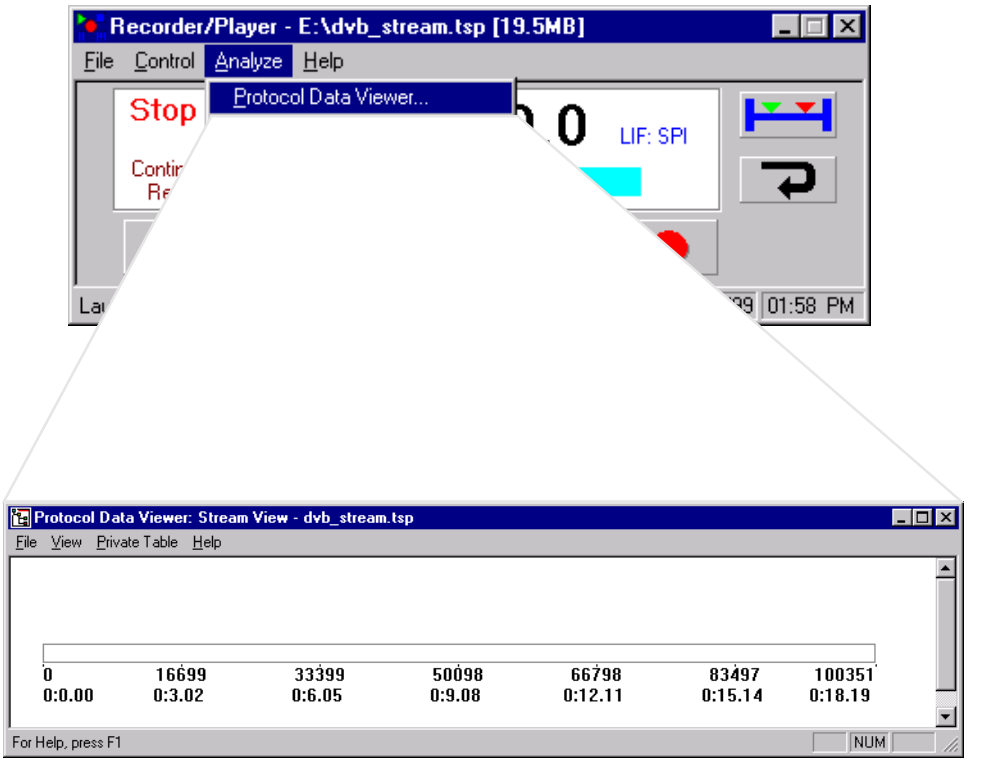
There are two ways to start the Protocol Data Viewer.

From the Launch Pad:




Use this method when you want to open and analyze a file you have saved to disk.

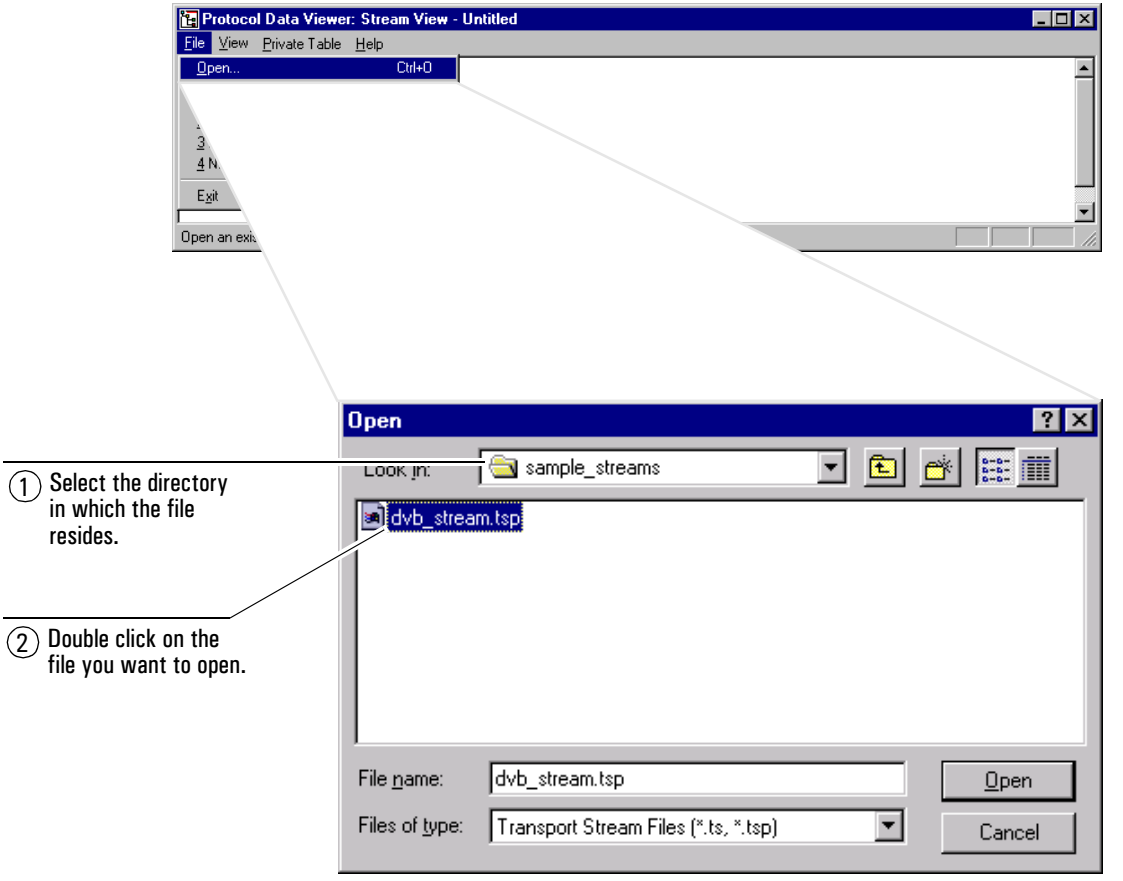
From the Recorder/Player:



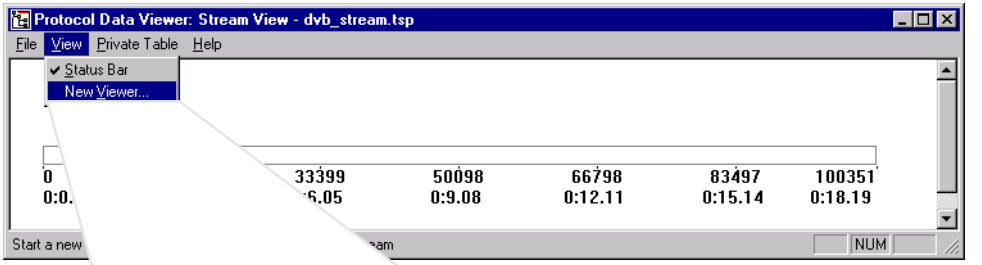

Use this method when you want to analyze a file you have opened or recorded from the Recorder/Player.

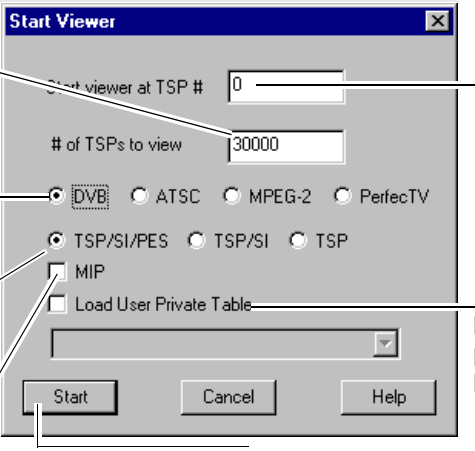
## Step 2: Open a transport stream file

Complete this step if you have opened the Protocol Data Viewer from the Launch Pad.



## Step 3: Open a substream view




(2) Enter the length of the segment you want to view.

(1) Enter the packet number at which to start the segment.

(3) Select the type of transport stream.

(4) Select the level of analysis you want.

For instructions on loading user private tables, refer to page 6-47.

(5) If the DVB stream is carrying MIP (Mega-frame Initialization Packet) packets for Single Frequency Network synchronization, select MIP.

If this option is selected, the **TS Hierarchy** view will display a section called **MIP Packets** that groups all packets on PID 0x15.

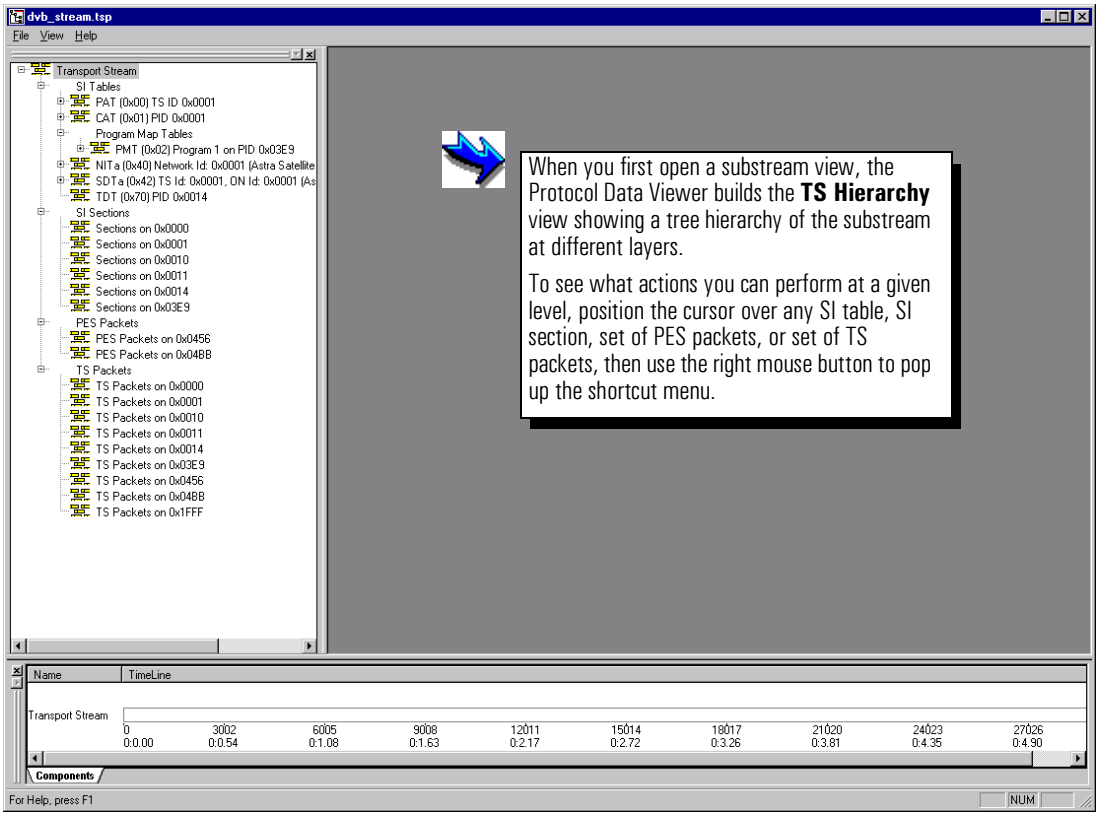
(6) Start the Substream Viewer.

You can also right click on a starting point from the **Stream View** time line, then select **Start Viewer** from the shortcut dialog. MPEGscope automatically enters the starting point you have selected.

## Step 4: Check the **TS Hierarchy** view

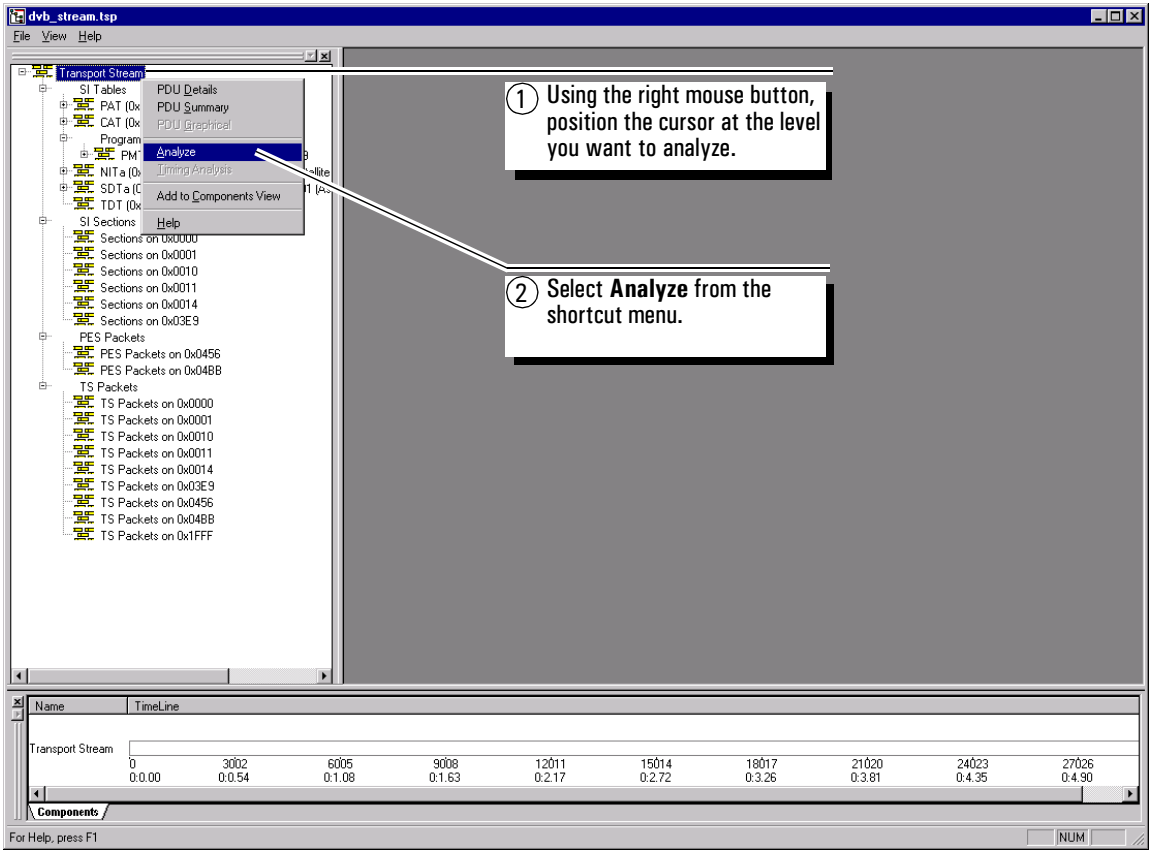
The **TS Hierarchy** view displays the PDUs (Protocol Data Units) in the substream at different layers. A PDU is a unit of data that is meaningful in the protocol, such as SI tables, SI sections, PES packets, and TS packets. You can expand or collapse the levels as you would folders in the Windows NT® Explorer.

From the **TS Hierarchy** view, you can check the stream for errors or select more detailed views of tables, sections, PES packets, or TS packets. You can also launch the timing analysis application from this view.



## Step 5: Check for errors

The Protocol Data Viewer analyzes all protocol fields in the stream for violations of ISO, DVB, and ATSC standards, depending on the options you selected from the **Start Viewer** dialog on page 6–5. After you analyze the substream for errors, the icons next to the levels change. An icon with a red background indicates errors were found at that level. Refer to page 6–8 for a complete description of icons. Errors are displayed in the **Components** view as vertical red lines.




You can analyze the entire substream by starting analysis at the **TS Stream** level, or by selecting **Analyze** at the **File** menu.

Icons at each level show the status of each item in the hierarchy, as follows:

(white background) A stream of PDUs that has not been analyzed.

(green background)) A stream of PDUs that has been analyzed and has no errors

(red background) A stream of PDUs that has been analyzed and has errors.

(blue background)) A scrambled stream of packets that has not been analyzed.

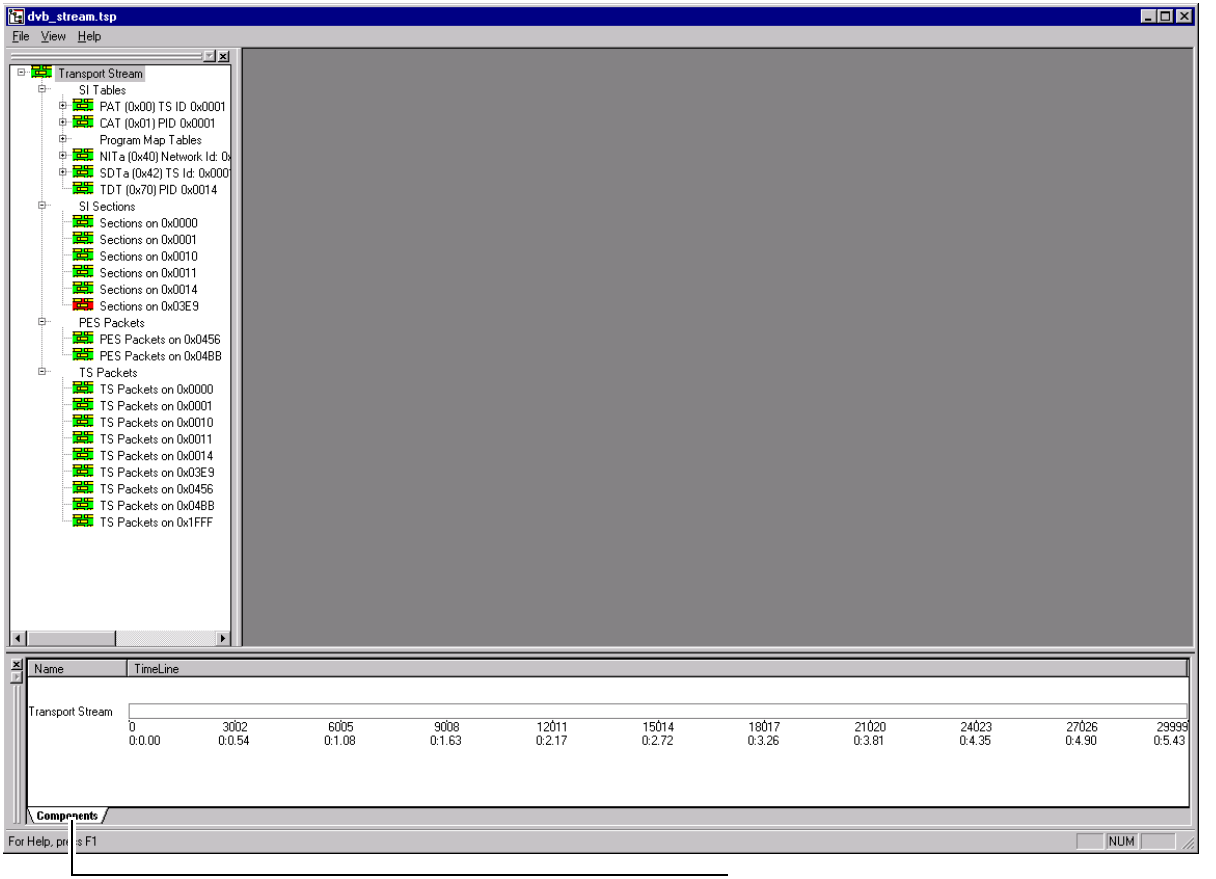
(green background) A scrambled stream of packets that has been analyzed and has no errors. (Scrambled packets are analyzed for header errors only.)

(red background) A scrambled stream of packets that has been analyzed and has errors.

(gray background) PDUs that were expected but not found in this stream.

## Step 6: Check the **Components** view

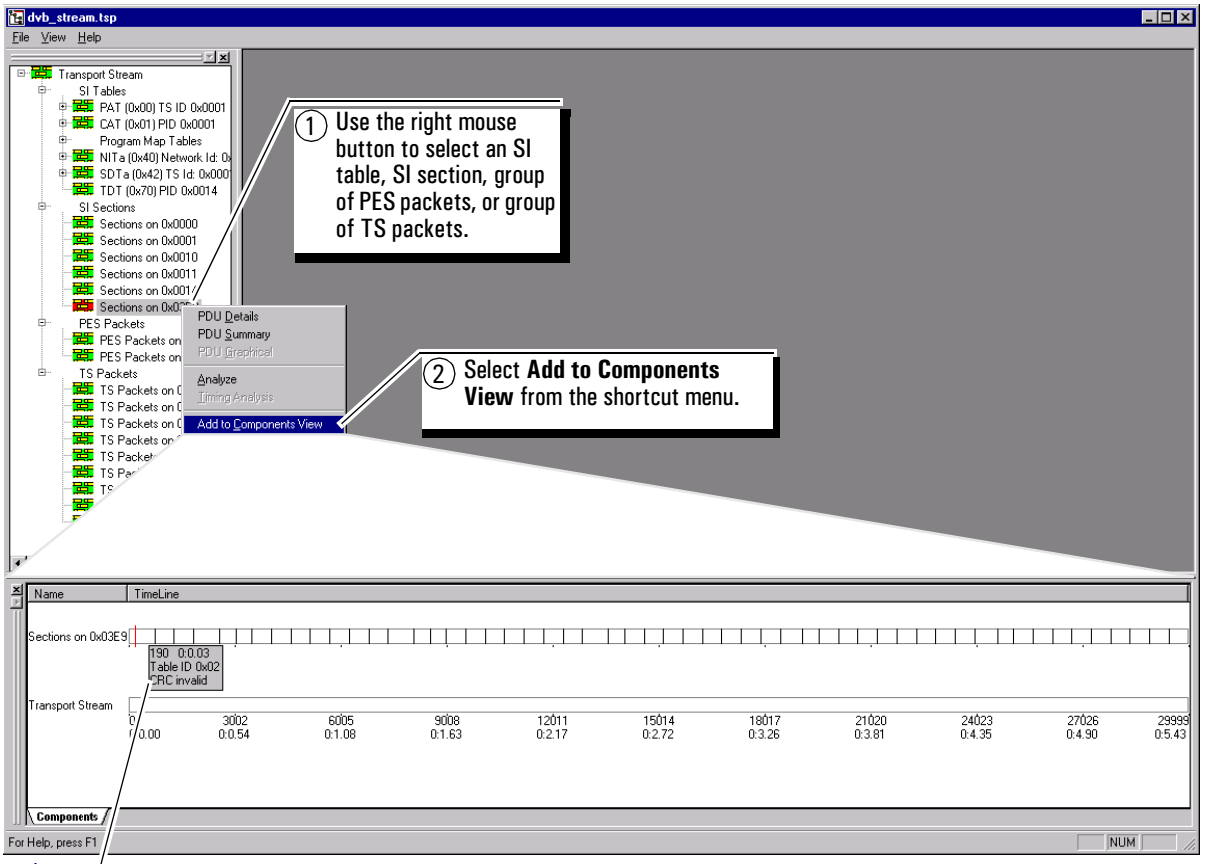
The **Components** view presents time lines of the transport stream. You can add new time lines for SI tables, SI sections, PES packets, or TS packets to the **Components** view to show the positions of the PDUs you select relative to the time line of the entire transport stream segment.




The **Components** view shows a time line for the transport stream segment with numbers and timestamps of the packets in the stream.

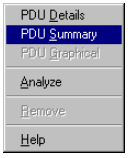
To see what actions you can perform from the **Components** view, position the cursor on the time line, then use the right mouse button to pop up the shortcut menu.

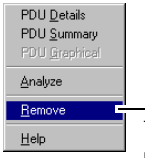
## Step 7: Create a new time line




If the PDU has been analyzed, the locations of known errors are displayed as red lines. You can click on a red error line to view the packet number of the errored packet and a brief description of the error.

You can also select an error line with the right mouse button and bring up the **PDU Summary** view. The errored packet will be within the first few packets. For more information, refer to page 6–28.

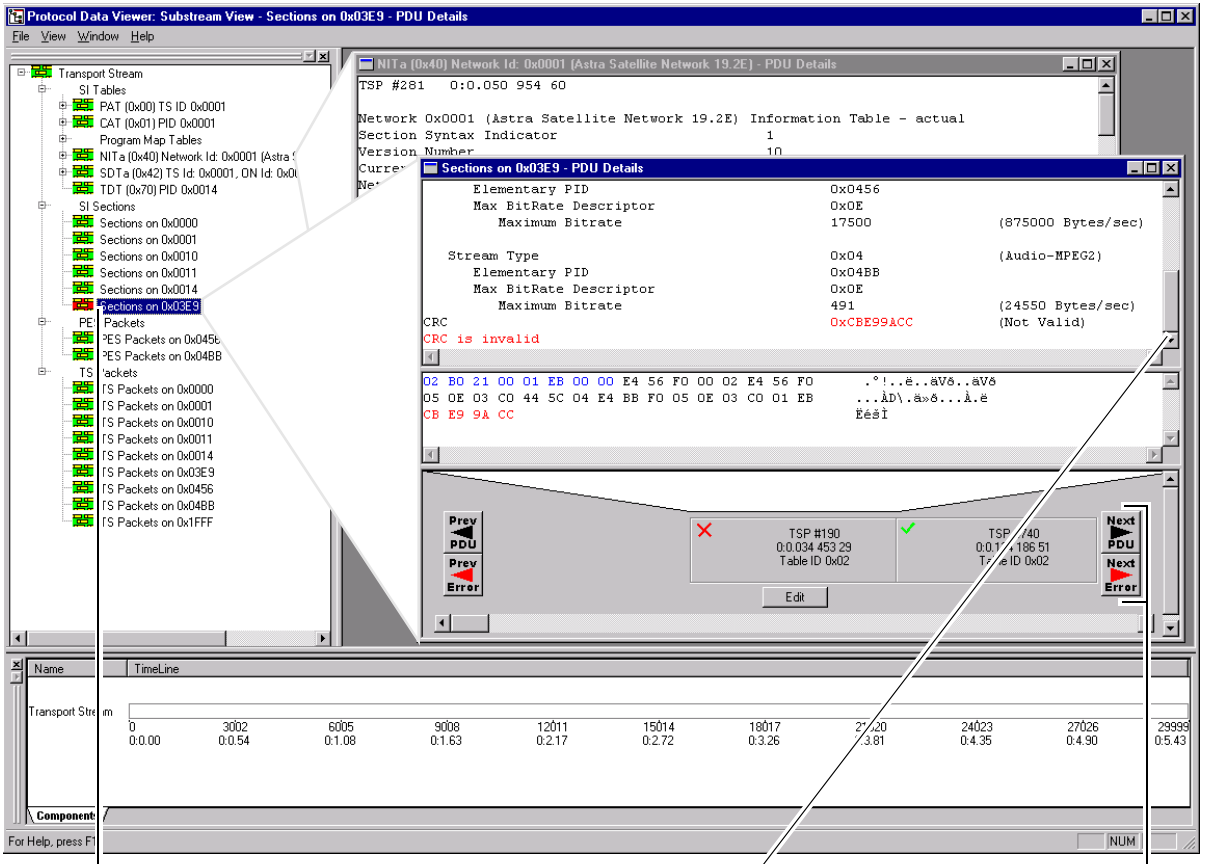



To remove a time line, use the right mouse button to select the time line, then select **Remove** from the shortcut menu.

## Step 8: Check the **PDU Details** view

The **PDU Details** view provides decodes and detailed error information for the selected PDU. You can navigate between consecutive PDUs or between errored PDUs. You can also edit table sections from this view.




① Double click on the PDU to open the **PDU Details** view.

You can open multiple **PDU Details** windows.

② Scroll down to find details about errors.

Error information is displayed in red text.

③ Use the arrow buttons to navigate between PDUs. The red arrow buttons jump from errored PDU to errored PDU.

The **PDU Details** view contains a top pane, middle pane, and bottom pane.

**Top pane**

The top pane displays the following decoded information:

| If you selected... | The decode pane displays the... |
|---|---|
| an SI table | contents of the first complete, valid table in the substream—with the first section of the table starting in the TS packet number displayed in the bottom pane |
| an SI section | contents of the table section starting in the TS packet number displayed in the bottom pane |
| a PES packet on a PID | PES header starting in the TS packet number displayed in the bottom pane |
| TS packets on a PID | 4-byte TS packet header of the TS packet number displayed in the bottom pane |

If the PDU has been analyzed and has errors, error messages appear at the bottom of the decode pane. Reserved values are displayed in blue, and forbidden values in red.

**Middle pane**

The middle pane displays the entire PDU, byte-by-byte, in hexadecimal format. Header bytes are blue, payload bytes are black, and trailer bytes (where present) are green.

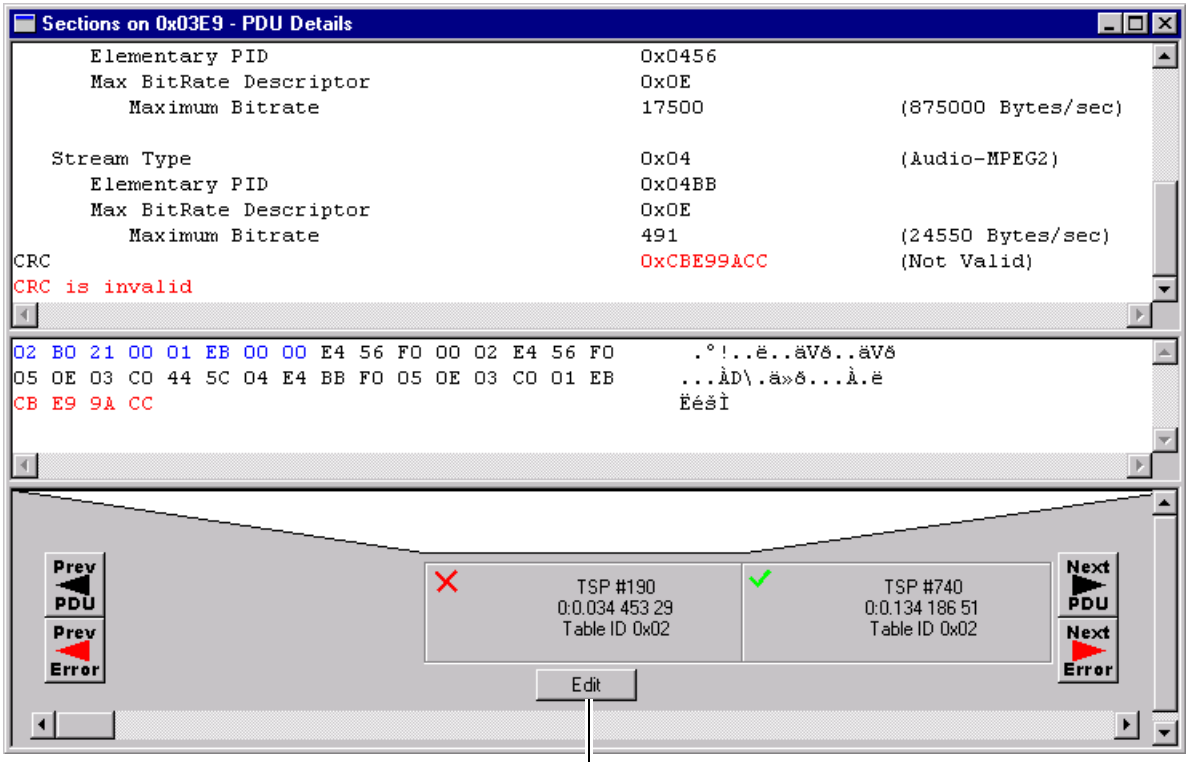
PDU contents are also displayed in ASCII format to the right of the hexadecimal display.

**Bottom pane**

The bottom control pane allows you to navigate between PDUs using the arrow buttons or slider bar.

The control pane displays the following information about the decoded PDU you are viewing:

- # of the TS packet containing the start of the PDU
- packet timestamp in minutes:seconds.decimal-seconds format (if you are analyzing a .tsp file)
- Table type and version number for tables, Table ID for sections, Stream ID and length for PES packets, and PID for TS packets

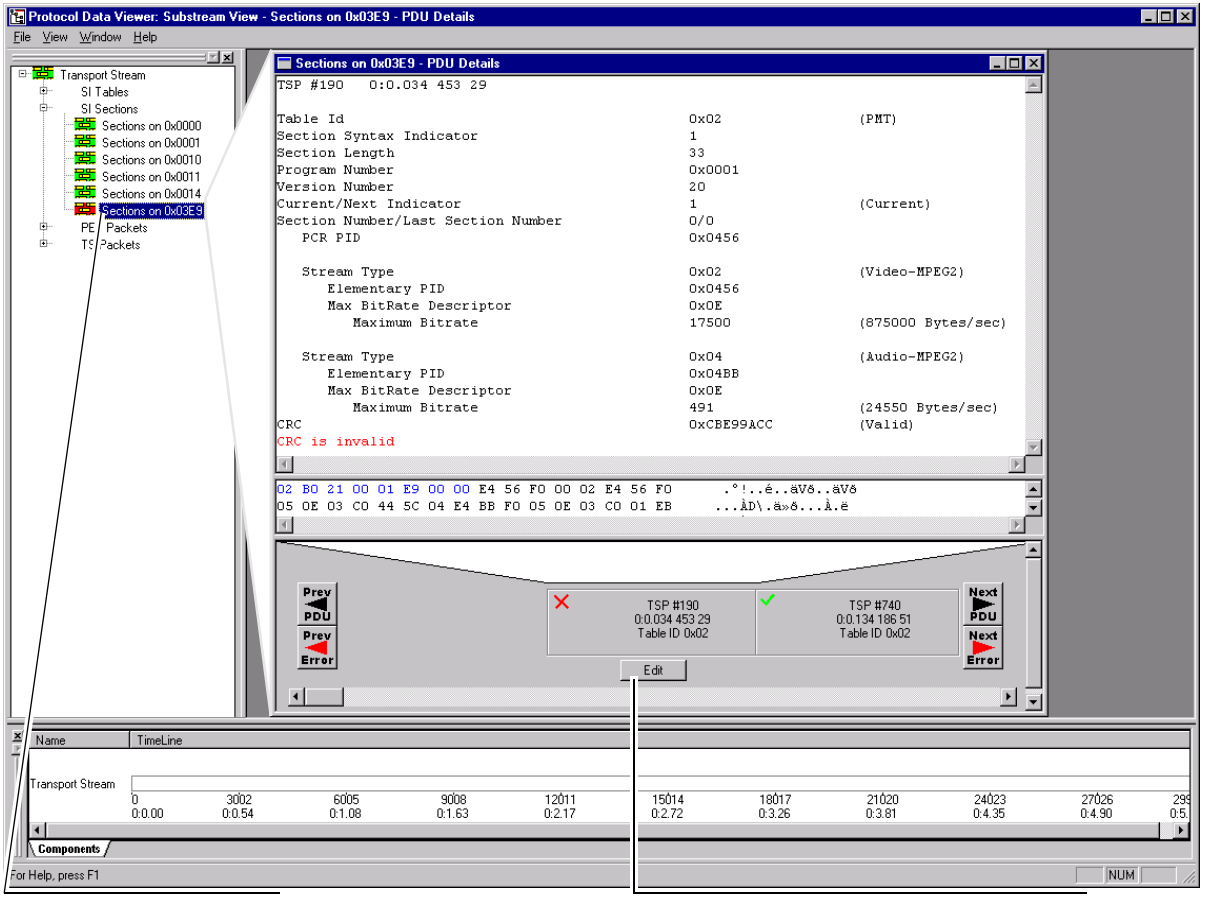



If the selected PDU is a table section, the **Edit** button allows you to edit the contents of table and descriptor fields. For details, refer to "Edit a table section", page 6–14.

You can print the decoded PDU displayed in the top pane by selecting the desired **PDU Details** window, then selecting **Print** from the **File** menu.

## Step 9: Edit a table section

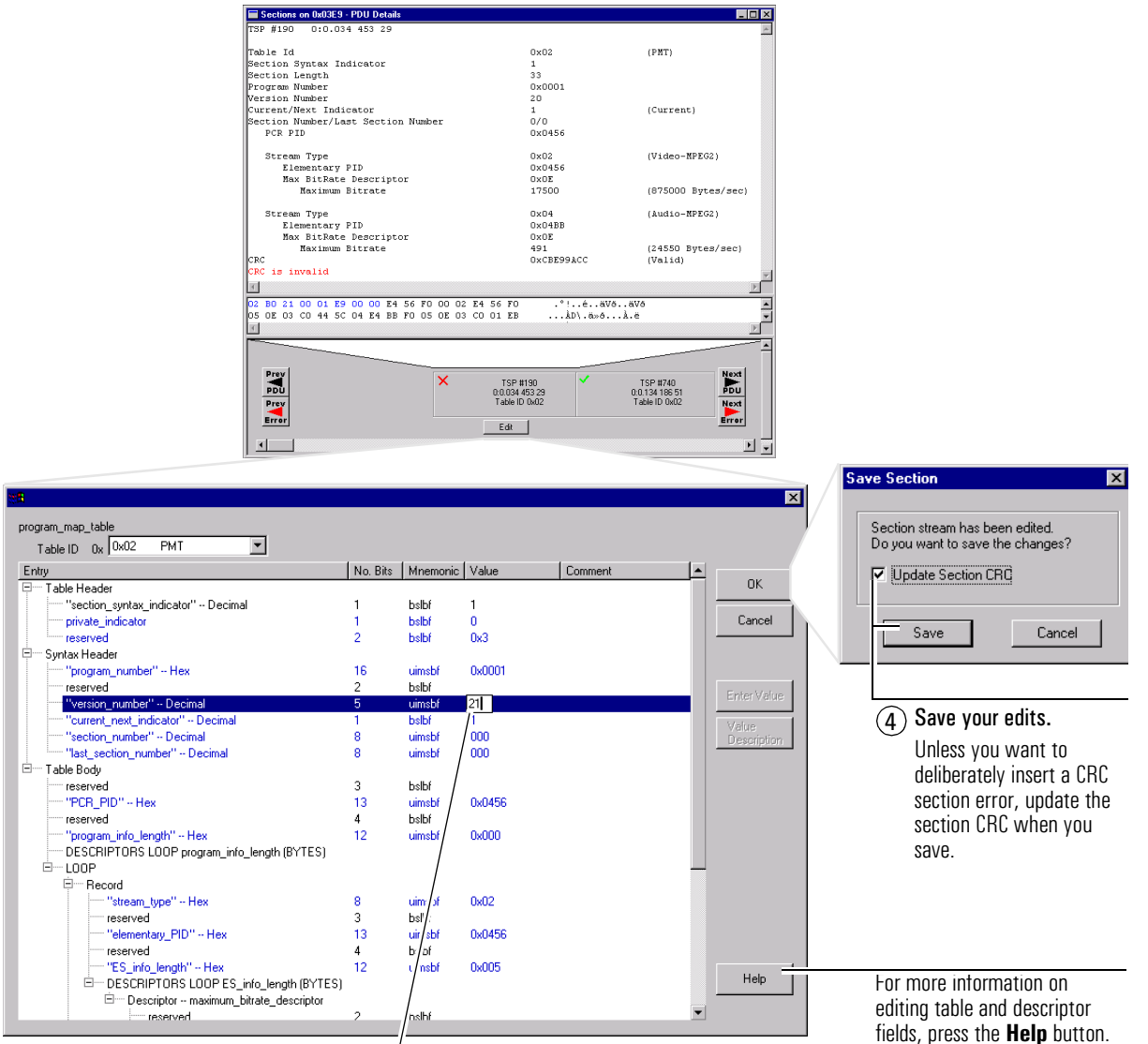
The Protocol Data Viewer allows you to edit individual table sections. For example, you can change a field's value, correct an error, enable or disable versions of a table, or deliberately inject an error into a section.




1. From the SI Sections layer, double click on the SI Sections PID containing the section(s) you want to edit.

2. Use the navigation buttons to find the SI section you want to edit, then press **Edit**.

The **Edit** button does not appear if the stream you are analyzing is read-only. To remove the read-only attribute, use the right mouse button to select the stream from Windows® NT Explorer, then select **Properties**. From the **Properties** dialog, deselect **Read-only**. You will also have to close then restart the Substream Viewer.



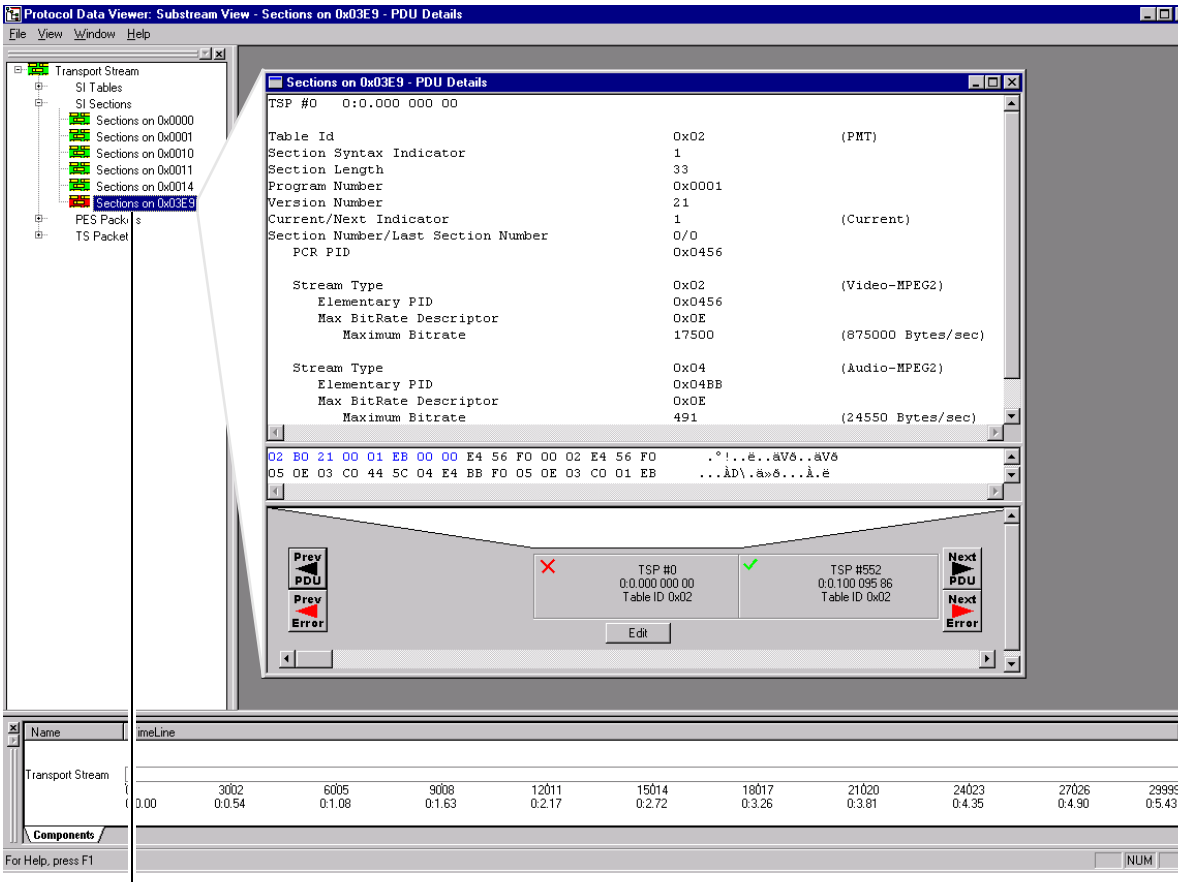

**4 Save your edits.**

Unless you want to deliberately insert a CRC section error, update the section CRC when you save.

For more information on editing table and descriptor fields, press the **Help** button.

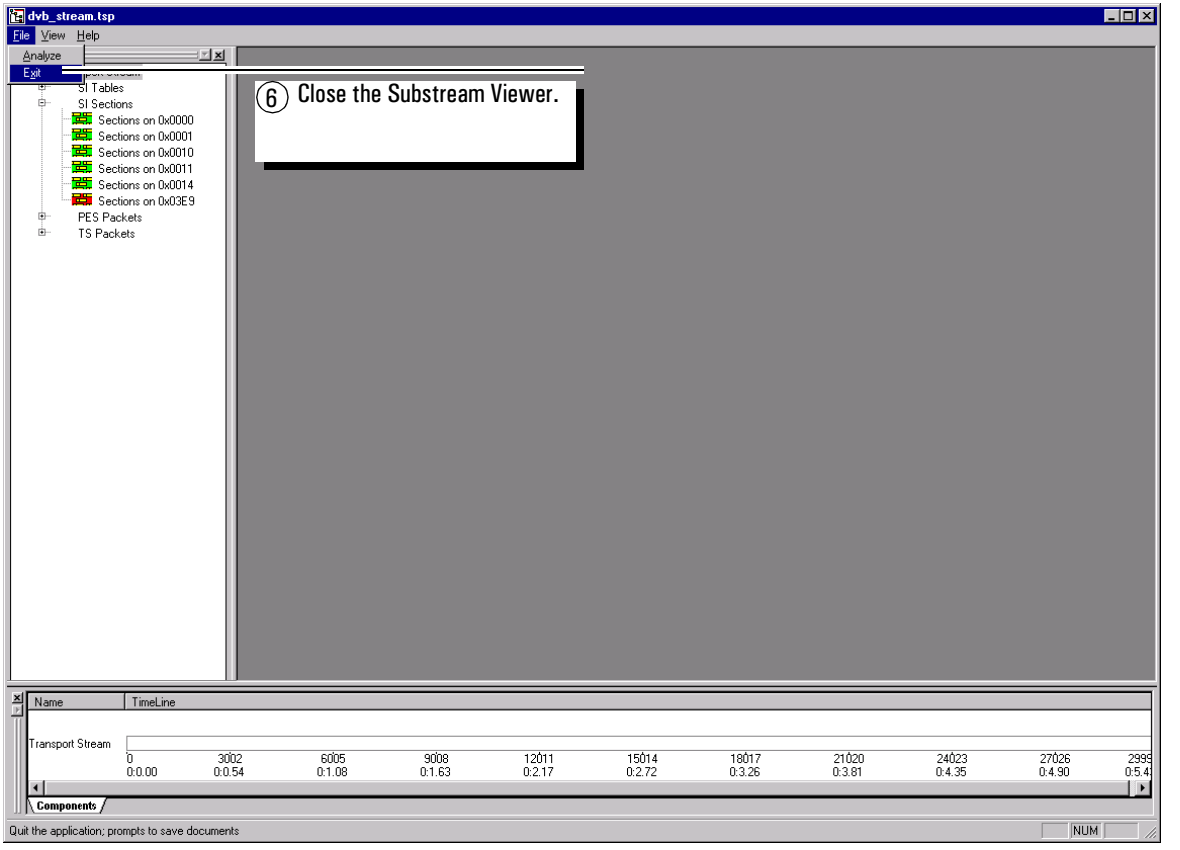
**3 Click in the Value column of any blue table or descriptor field to edit the contents of the field. Only fields highlighted in blue can be edited.**

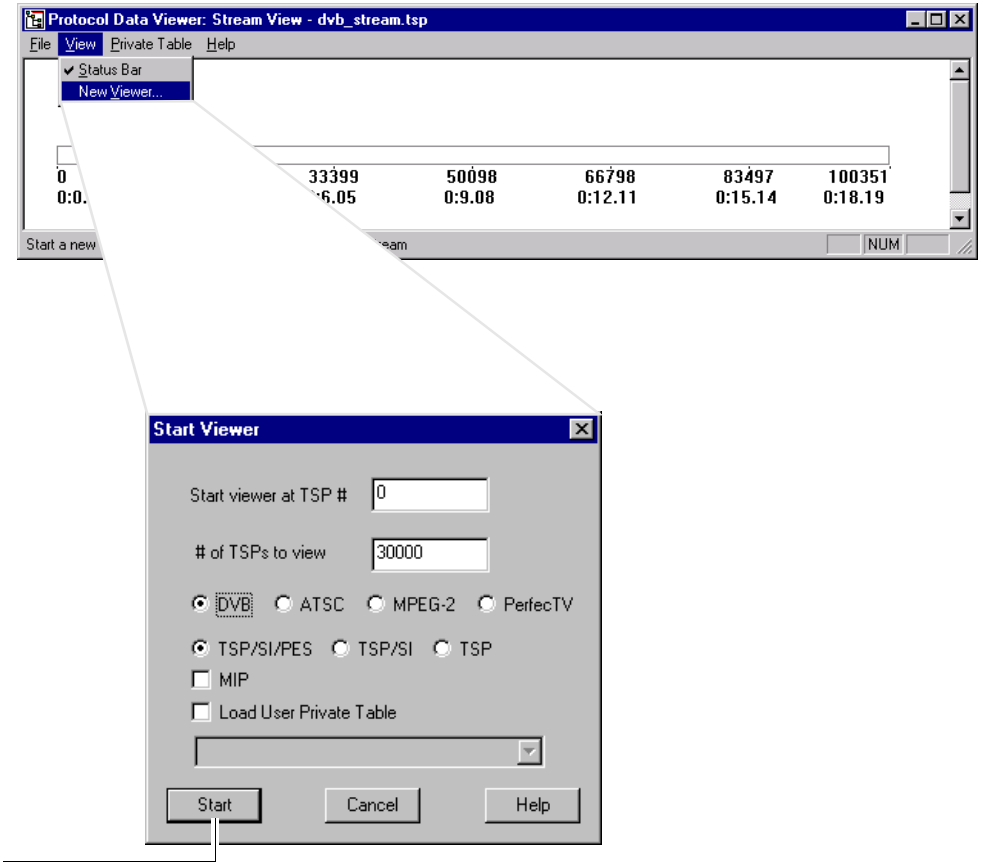
You can only edit individual table sections. Edits apply exclusively to the table section you are editing and are not propagated to other table sections on the PID.



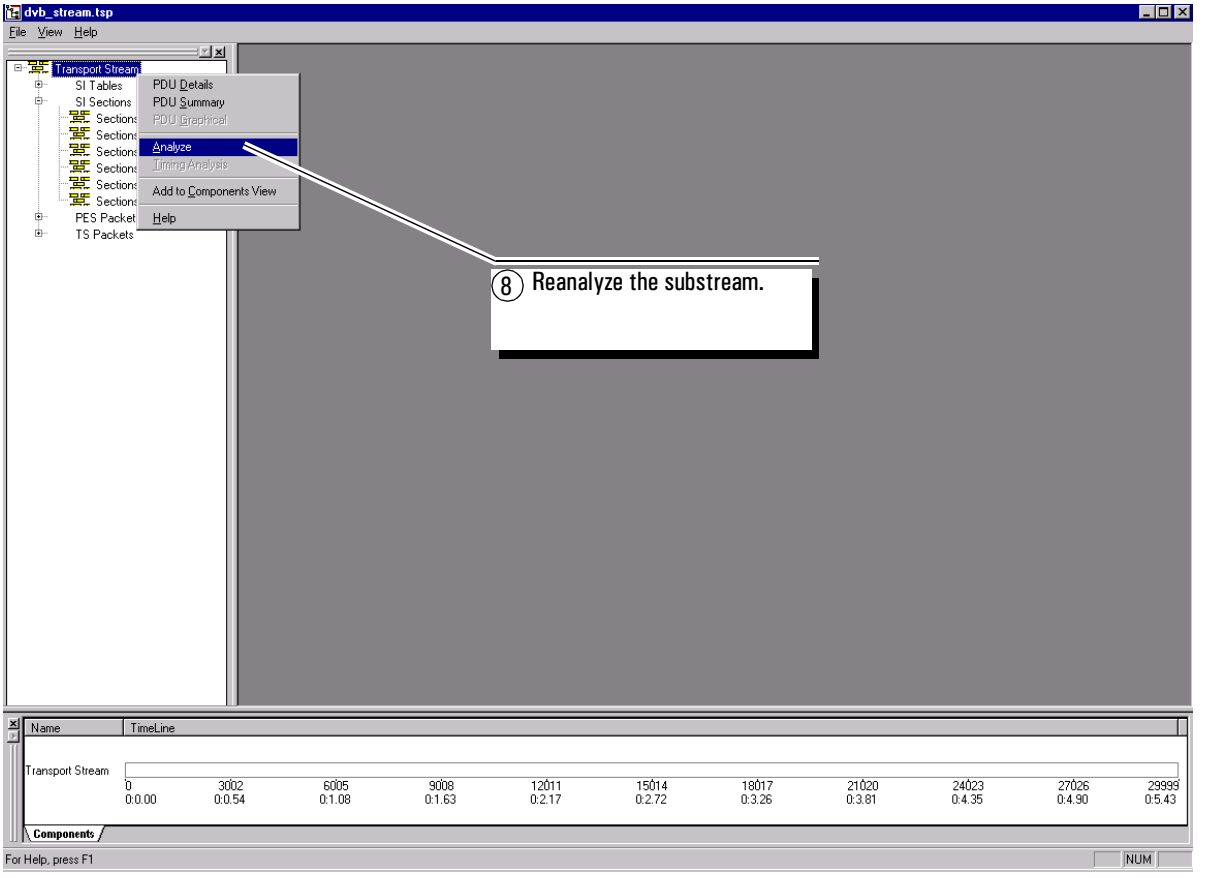

(5) To see your changes after editing, close then reopen the **PDU Details** window.

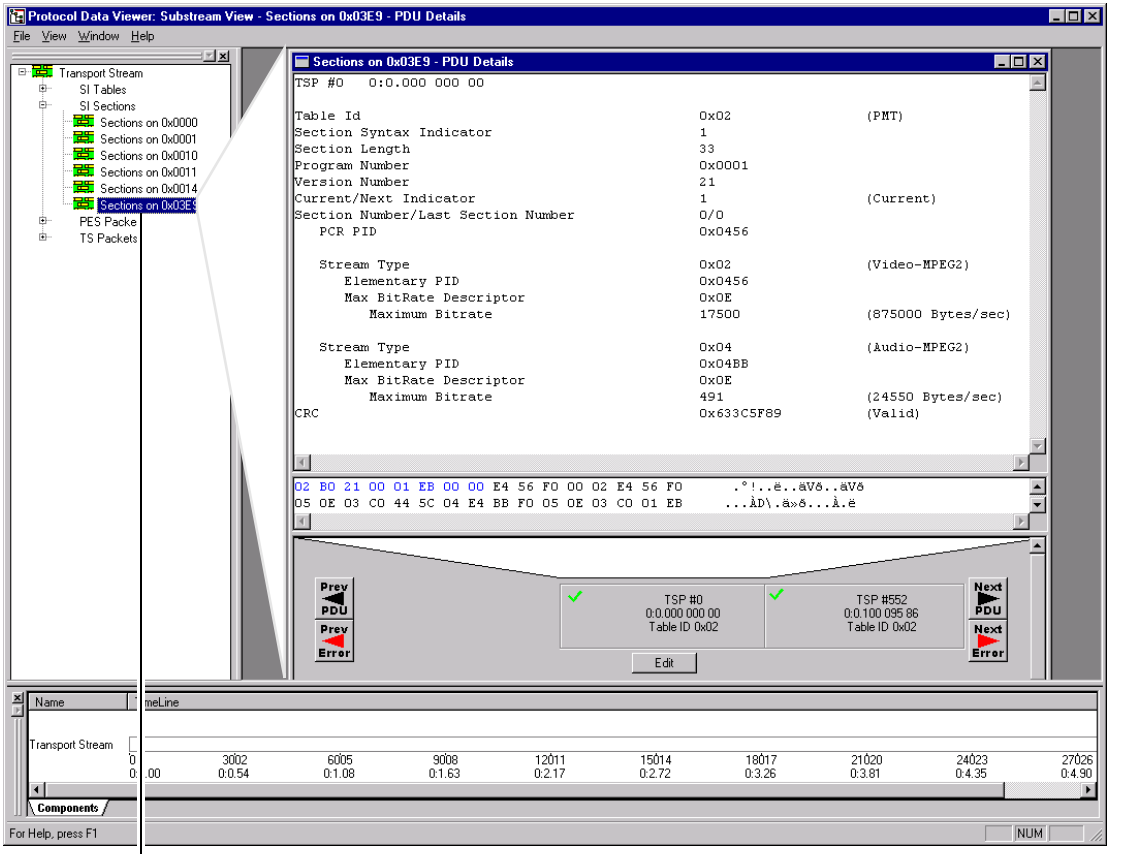
Editing SI section header field values, such as Version Number, Current/Next Indicator, or Section Number/Last Section Number may alter the SI Tables PDU stream. It is therefore important to restart the Substream Viewer and reanalyze the stream after editing.






⑦ Reopen the Substream Viewer.

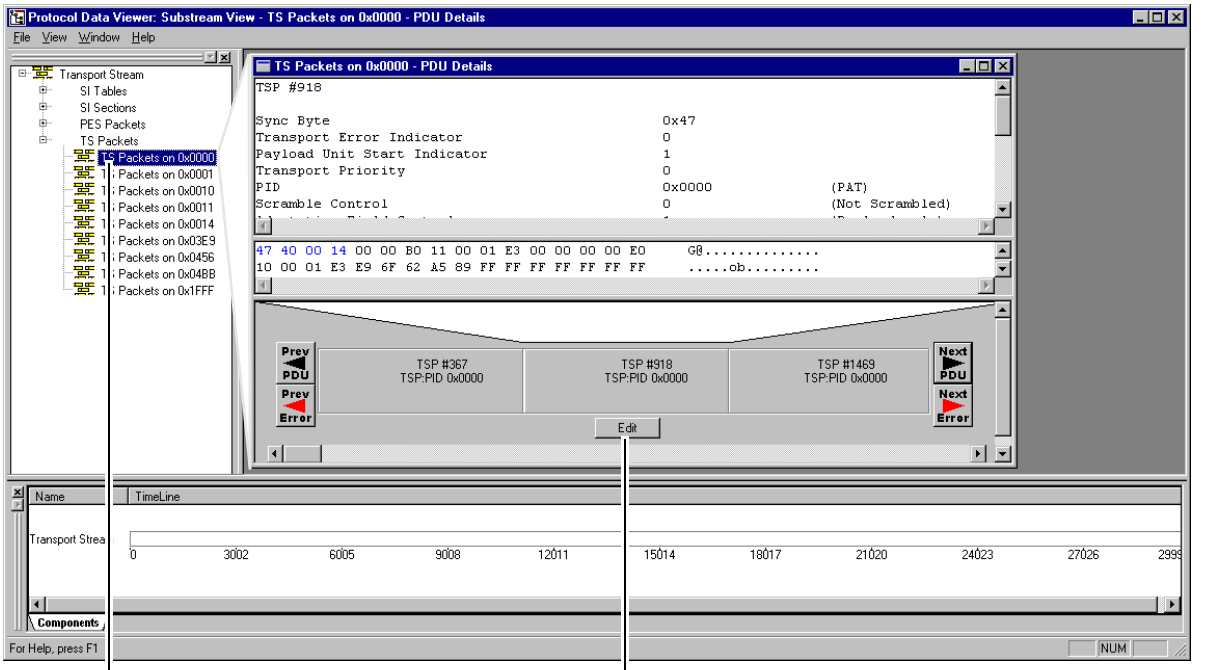

dvb_stream.tsp
File View Help
Transport Stream
SI Tables
SI Sections
Sections
Sections
Sections
Sections
Sections
Sections
PES Packet
TS Packets
PDU Details
PDU Summary
PDU Graphical
Analyze
Timing Analysis
Add to Components View
Help
8 Reanalyze the substream.
Name
TimeLine
Transport Stream
0 3002 6005 9008 12011 15014 18017 21020 24023 27026 29999
0:0.00 0:0.54 0:1.08 0:1.63 0:2.17 0:2.72 0:3.26 0:3.81 0:4.35 0:4.90 0:5.43
Components
For Help, press F1
NUM




(9) View the edited SI section.

# Step 10: Edit a transport stream

As of version A.05.03, the Protocol Data Viewer includes a transport stream editor that allows you edit a transport stream packet in hexadecimal. You can edit any part of the packet—for example, a TS header field or a PES header field in the packet payload. In this example, a packet's sync byte is errored, invalidating the packet.

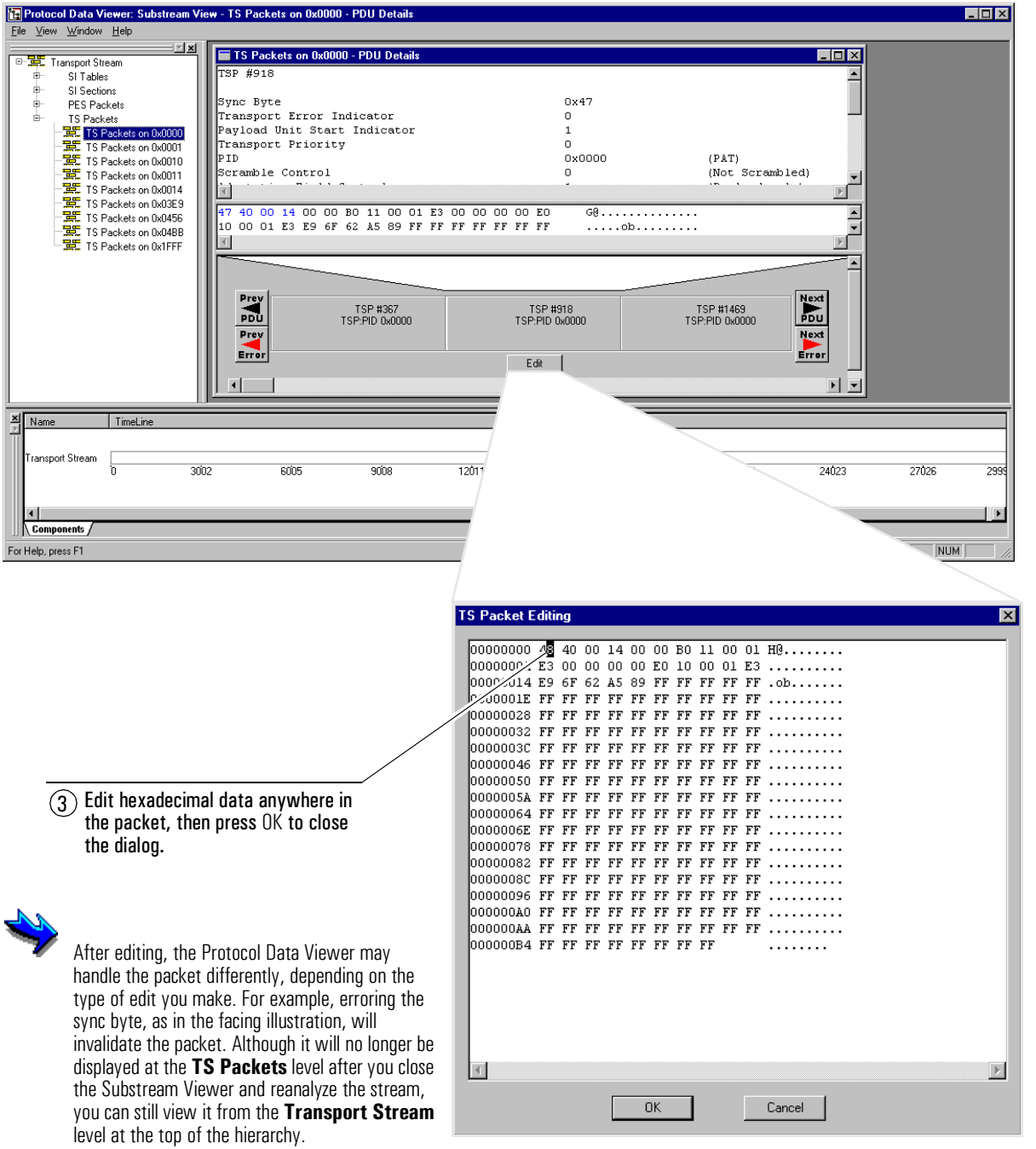



1. **From the** TS Packets **level, double click on the PID containing the packet(s) you want to edit.**

   You can also use the transport stream packet editor from the **Transport Stream** level.

2. **Use the navigation buttons to find the packet you want to edit, then press Edit.**

The **Edit** button does not appear if the stream you are analyzing is read-only. To remove the read-only attribute, use the right mouse button to select the stream from Windows® NT Explorer, then select **Properties**. From the **Properties** dialog, deselect **Read-only**. You will also have to close then restart the Substream Viewer.

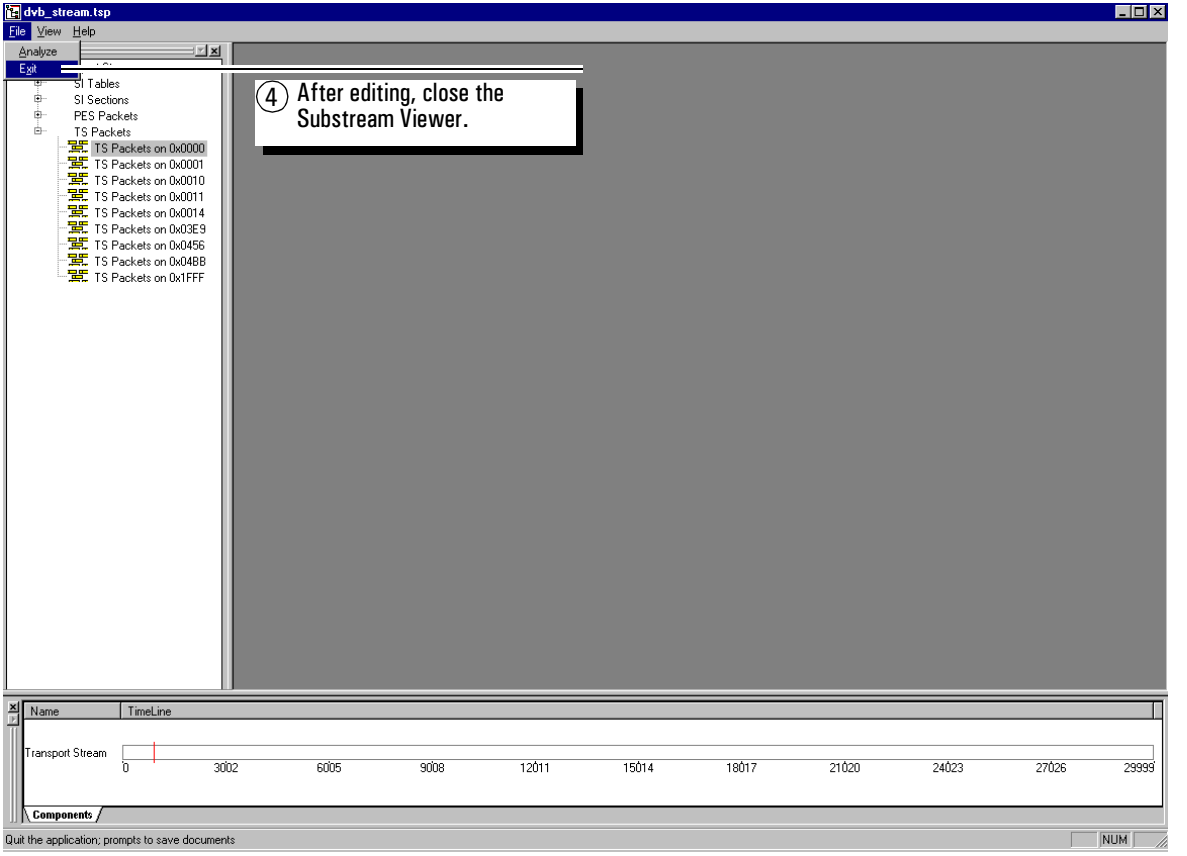




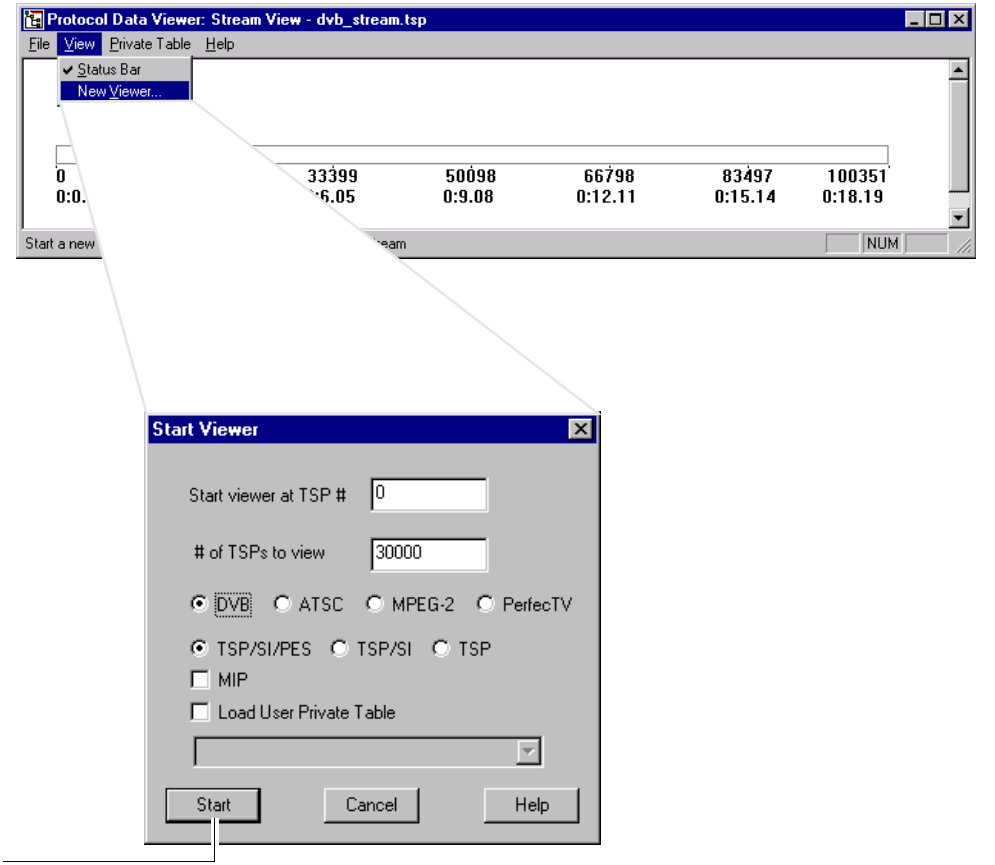


③ **Edit hexadecimal data anywhere in the packet, then press** OK **to close the dialog.**

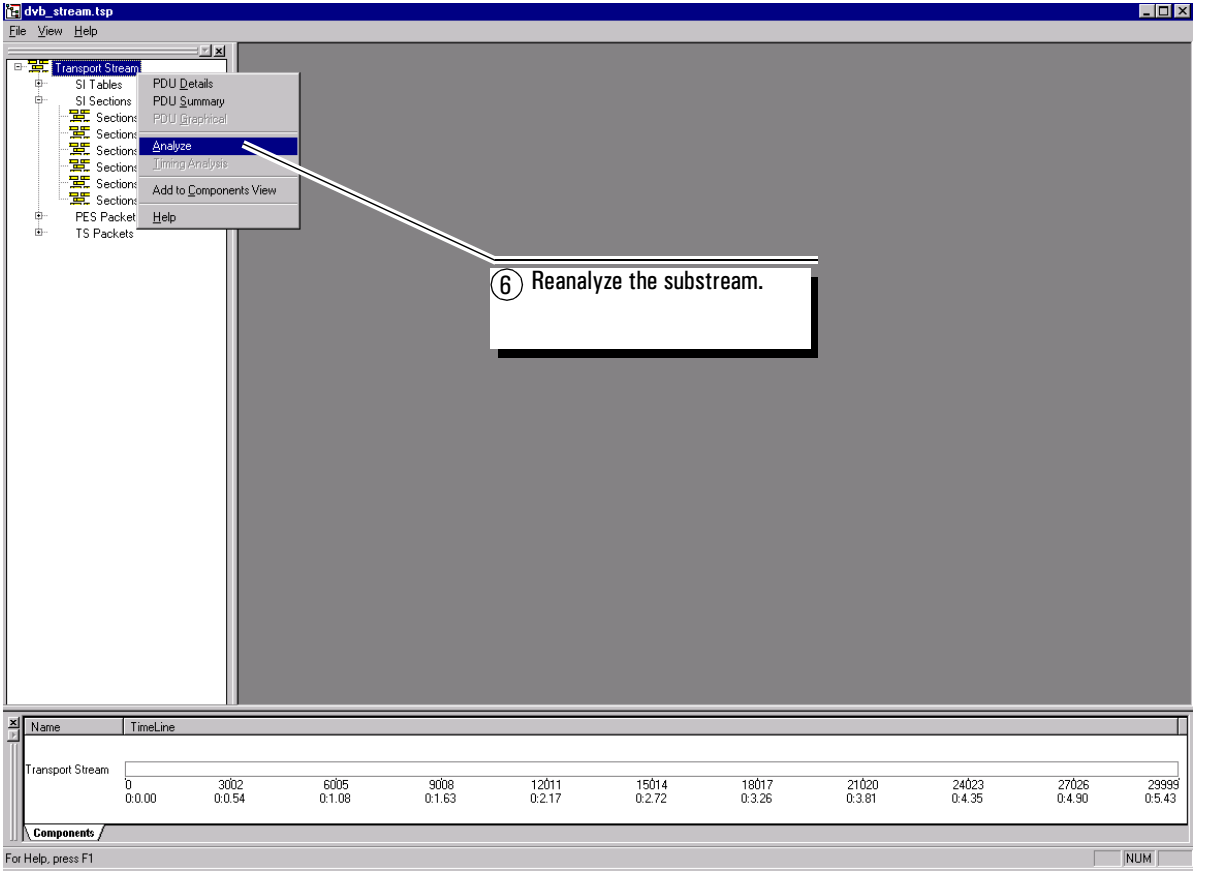
After editing, the Protocol Data Viewer may handle the packet differently, depending on the type of edit you make. For example, erroring the sync byte, as in the facing illustration, will invalidate the packet. Although it will no longer be displayed at the **TS Packets** level after you close the Substream Viewer and reanalyze the stream, you can still view it from the **Transport Stream** level at the top of the hierarchy.




It is important to close the Substream Viewer so the Protocol Data Viewer can rebuild the **TS Hierarchy** view.

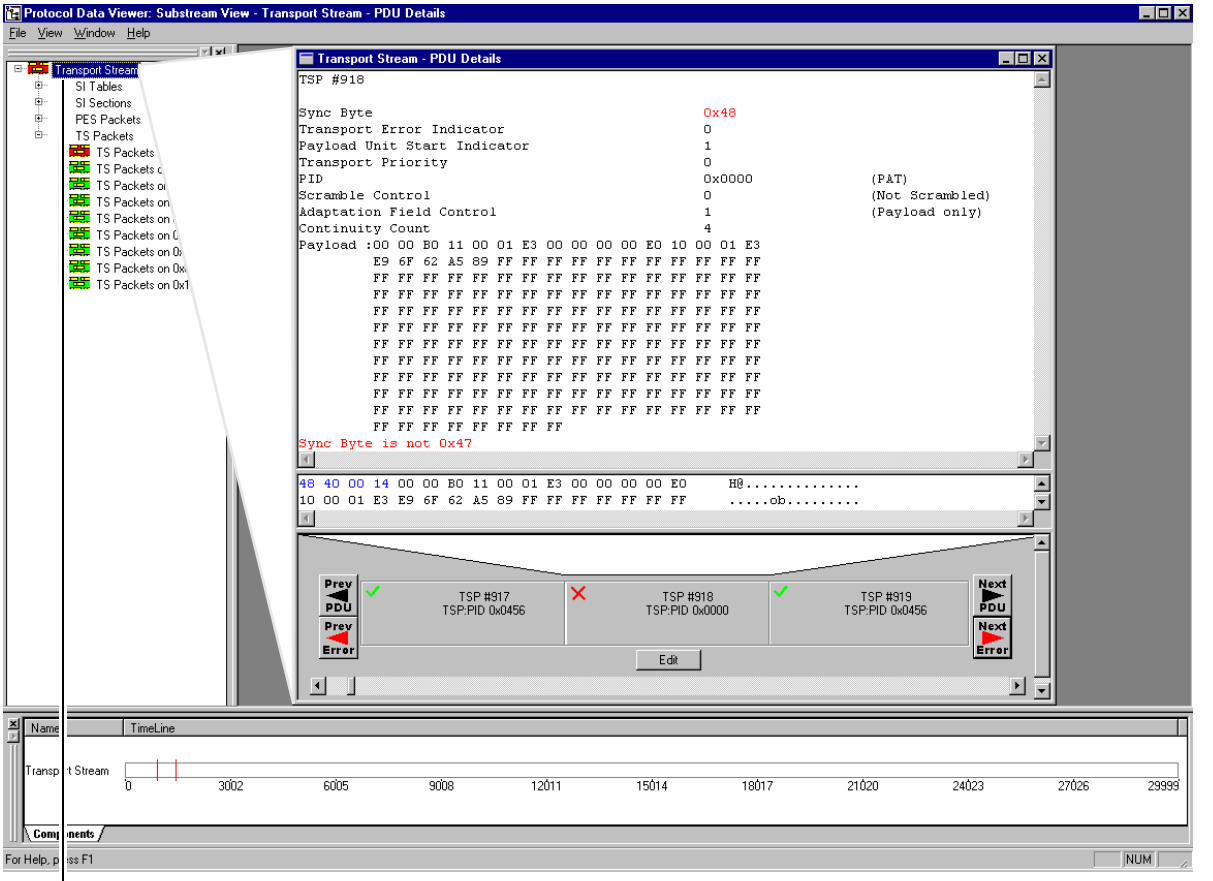



(5) Reopen the Substream Viewer.


dvb_stream.tsp
File View Help
Transport Stream
SI Tables
SI Sections
PES Packet
TS Packets
PDU Details
PDU Summary
PDU Graphical
Analyze
Timing Analysis
Add to Components View
Help
6 Reanalyze the substream.
Name
TimeLine
Transport Stream
0 3002 6005 9008 12011 15014 18017 21020 24023 27026 29999
0:0.00 0:0.54 0:1.08 0:1.63 0:2.17 0:2.72 0:3.26 0:3.81 0:4.35 0:4.90 0:5.43
Components
For Help, press F1
NUM

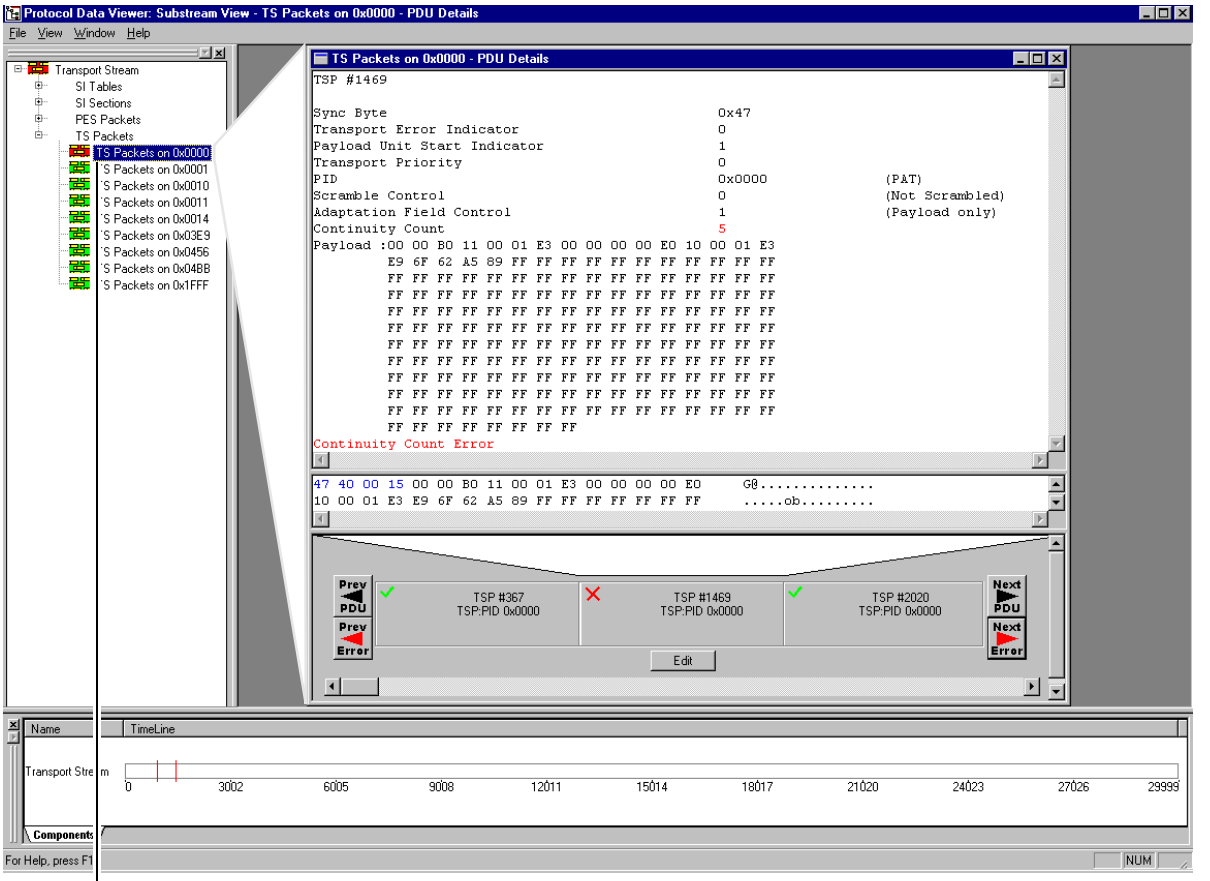
As indicated by the red lines on the time line, the stream now contains *two* errors. The first error is in TS packet 918 on PID 0x0000, the errored sync byte packet. Because it is now invalid, you can only see it from the **Transport Stream** level, where all TS packets on the stream are displayed.

The second error is a continuity count error in packet 1469, which follows packet 918 on PID 0x0000. It exists because packet 918 is now invalid. This error displays at both the **TS Packets** and the **Transport Stream** levels.




⑦ **Double click on the** Transport Stream **level, then use the** Next Error **button to locate the errored packet.**

Because the packet is invalid, it only displays at the **Transport Stream** level.

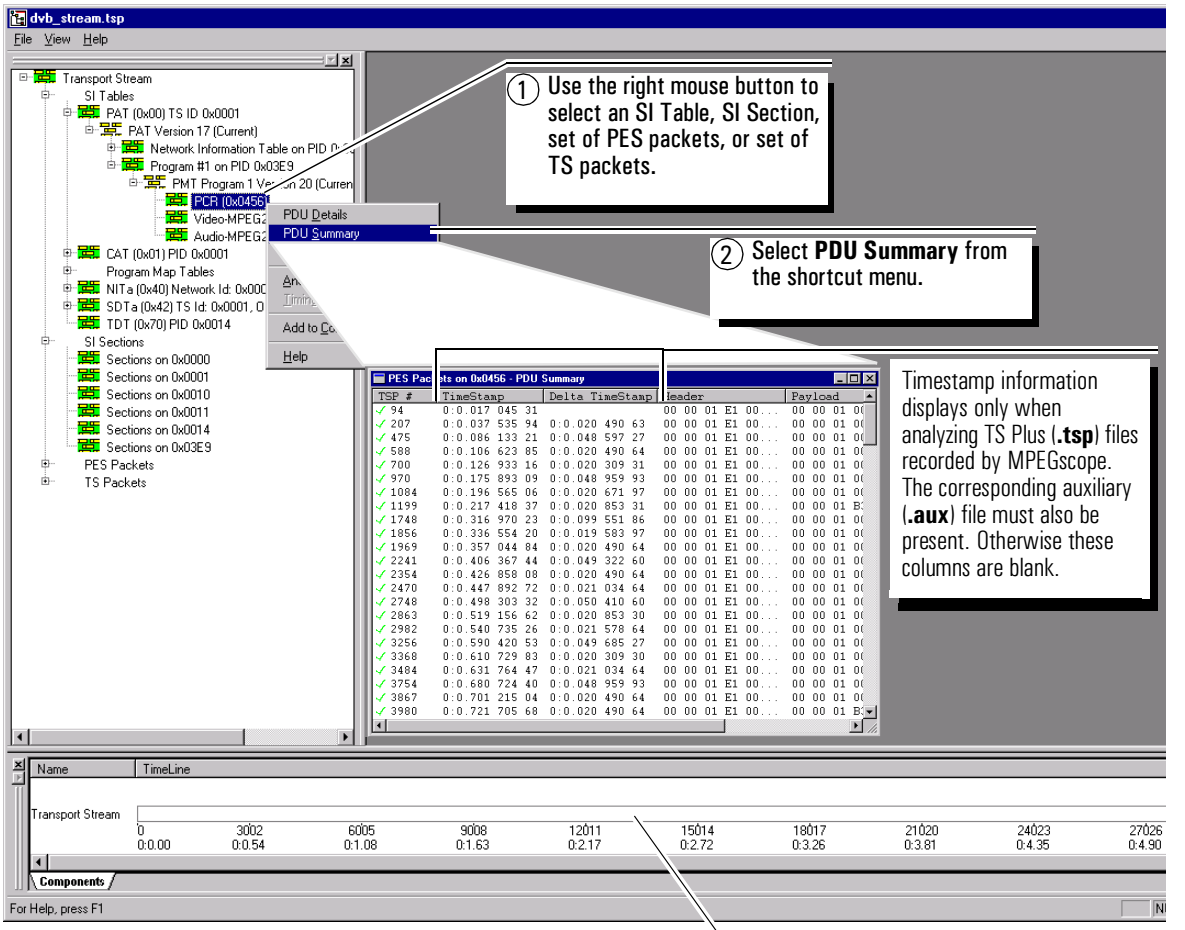



8 **Double click on** TS Packets on 0x0000 **element then use the** Next Error **button to locate the errored packet.**

Because packet 918 has been invalidated, the next packet on the same PID now has a continuity count error.

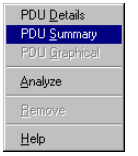
## Step 11: Check the **PDU Summary** view

The **PDU Summary** view presents a summary of the PDU you have selected. For each PDU it displays the packet number containing the start of the PDU, packet timestamp, timestamp increment, and the PDU header, payload, and trailer.




Indicates the packet does not contain errors.

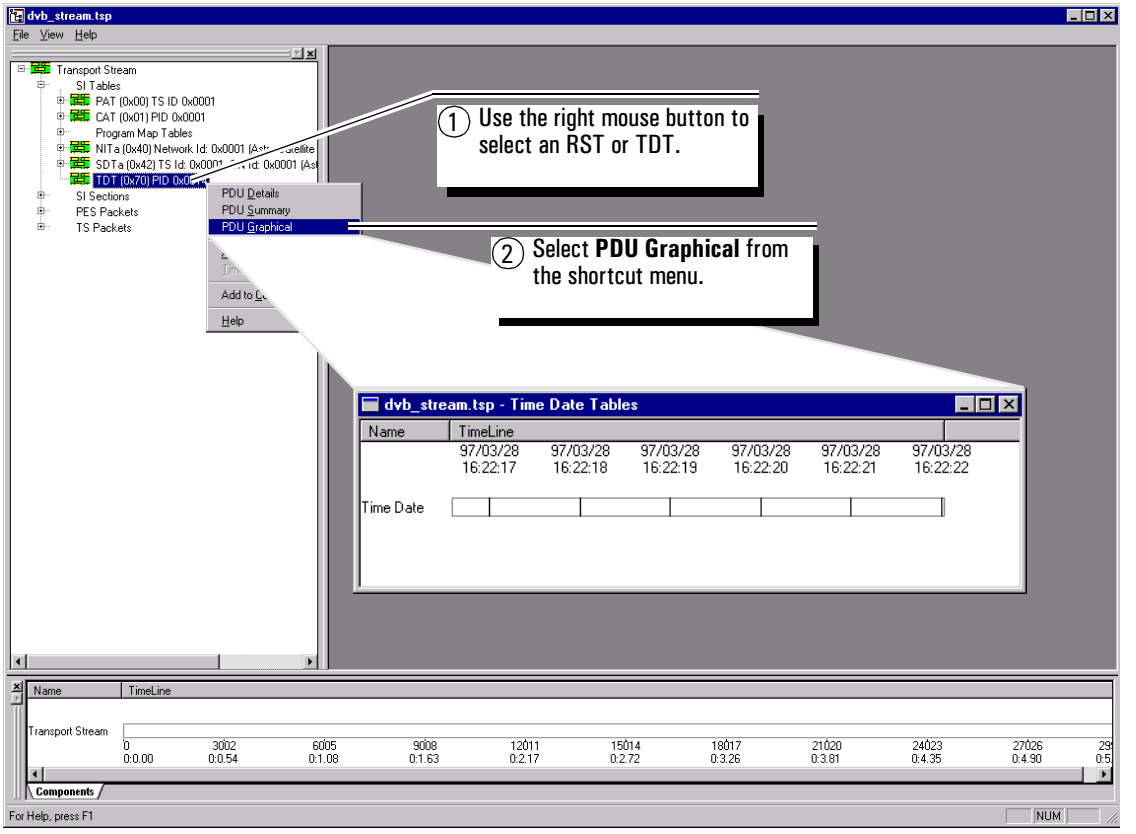
Indicates the packet contains one or more errors.

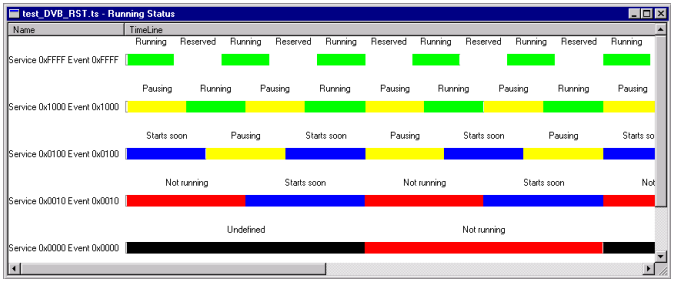



You can also open the **PDU Summary** view from the **Components** view. Use the right mouse button to select a position on the time line, then select **PDU Summary** from the shortcut menu.

## Step 12: Check the **PDU Graphical** view

The **PDU Graphical** view applies only to the DVB Running Status Table (RST) and Time and Date Table (TDT). It shows the position of RST or TDT sections on a time line of the substream as well as other information relevant to the table.

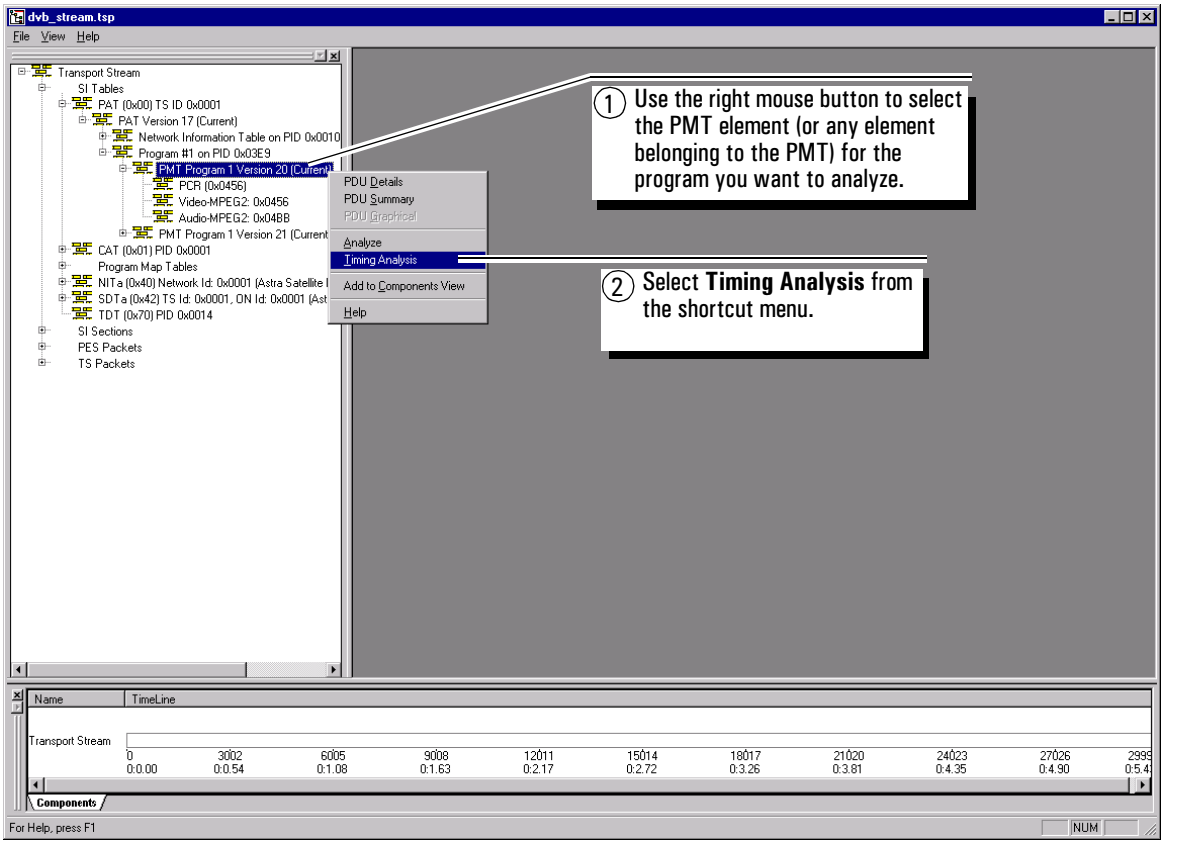



Example of the **PDU Graphical** view for an RST showing a color-coded time line of running states for each service and event ID combination in the table.

## Step 13: Check timing information

Timing analysis allows you to check important timing information for each program, such as PCR values, jitter, PCR drift, and correct encoding of the PTS and DTS. You can only perform timing analysis on streams captured and timestamped by MPEGscope.




This feature is only available when you analyze an MPEGscope **.tsp** (Transport Stream Plus) file. If **Timing Analysis** is grayed out at the shortcut menu, you may be analyzing a **.ts** file (transport stream) instead of a **.tsp** file.

To convert a transport stream file to a **.tsp** file, close the Protocol Data Viewer application, then open the **.ts** file from the Recorder/Player. From the Recorder/Player's **Analyze** menu, select **Protocol Data Viewer**. Open a substream view and follow the steps above to access the shortcut menu at the **PMT** level. The **Timing Analysis** feature should now be available.

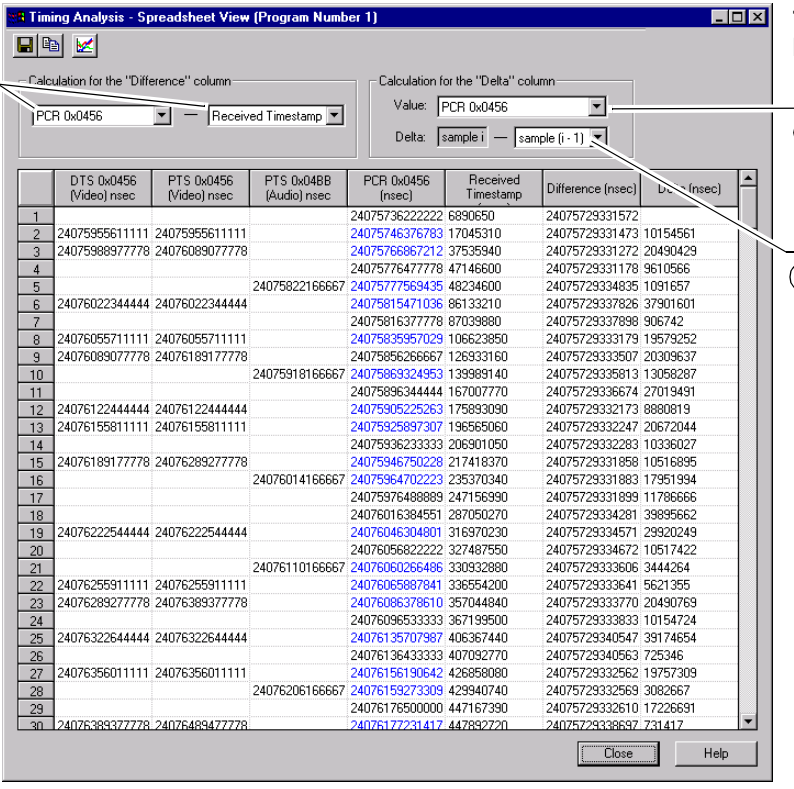
The **Spreadsheet View** displays

- PTS and DTS values for the video and audio PIDs in the selected program.
- PCR values for each packet in the program. Values displayed in blue are interpolated.
- MPEGscope timestamp for each packet in the program, added when the interface receives the last byte of the packet. Timestamps have an accuracy of +/- 5 ppm.
- Difference in timestamp values for two timestamps you select.
- Increment (delta) between consecutive samples of a timestamp you select.

Packets are displayed in the order they were received at the MPEGscope interface. Video streams are therefore in decode order.

**To calculate the** Difference **column...**

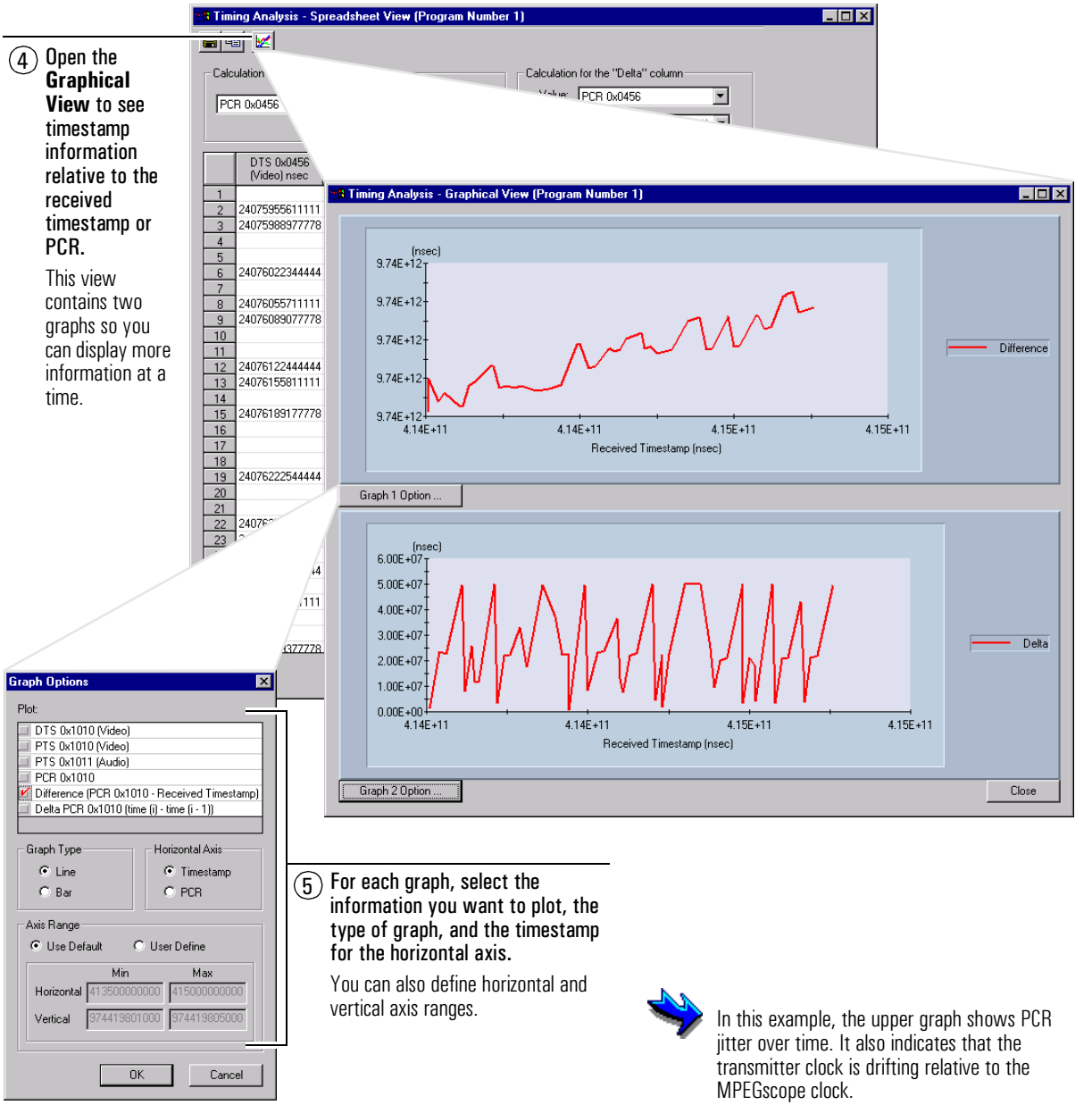
1. Select the timestamps you want to compare from the pulldown lists.




**To calculate the** Delta **column...**

2. Select the timestamp.
3. Select the timestamp increment.

The PCR value minus the received timestamp value shows the amount of jitter on the stream. Another useful comparison is the DTS or PTS minus the received timestamp. This calculation indicates the actual time the decoder has to decode or present each frame, taking PCR jitter into consideration.




④ **Open the Graphical View to see timestamp information relative to the received timestamp or PCR.**

This view contains two graphs so you can display more information at a time.

⑤ **For each graph, select the information you want to plot, the type of graph, and the timestamp for the horizontal axis.**

You can also define horizontal and vertical axis ranges.

In this example, the upper graph shows PCR jitter over time. It also indicates that the transmitter clock is drifting relative to the MPEGscope clock.

The lower graph shows PCR timestamp increments over time.

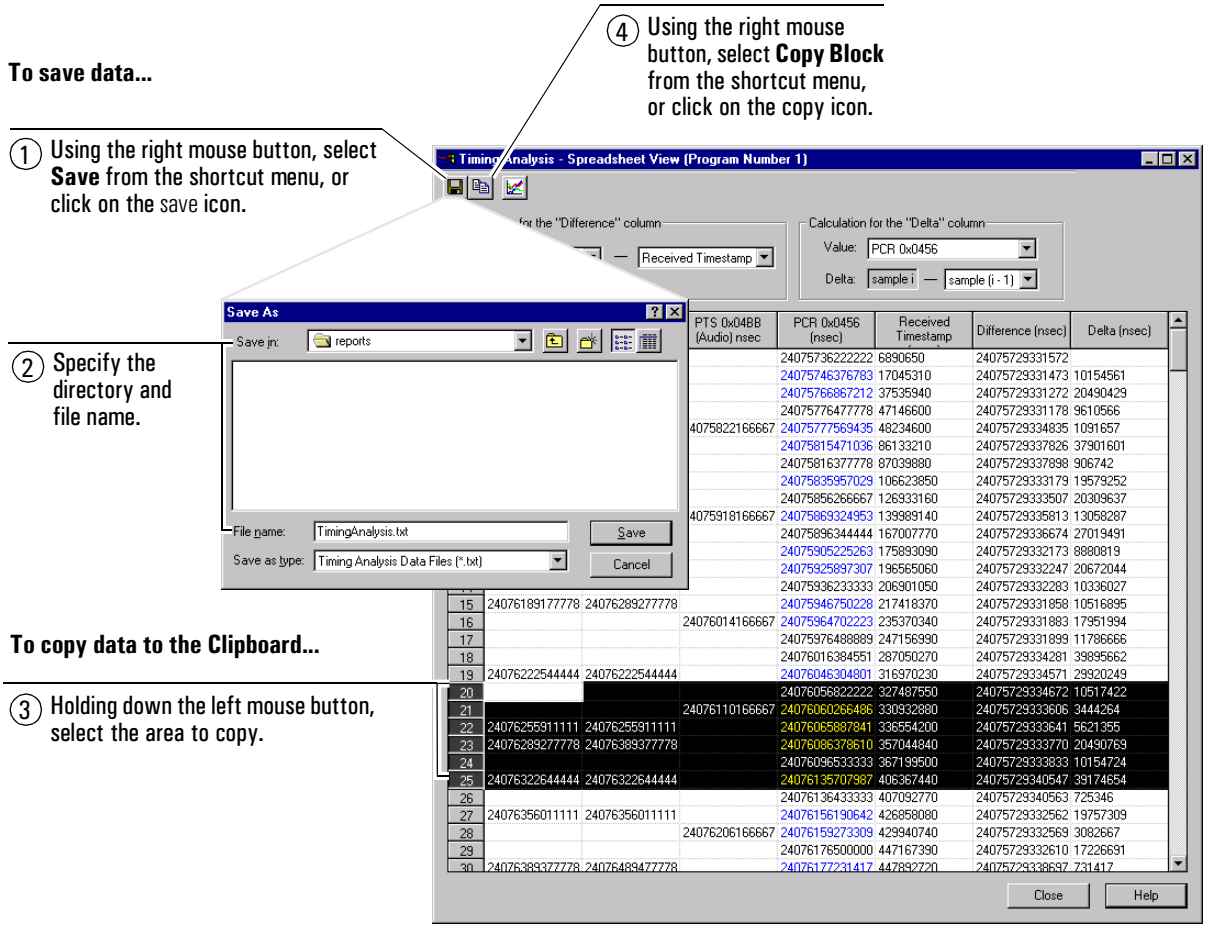
You can save **Spreadsheet View** data to a text file or copy selected values to the Windows® Clipboard to use in other applications.

**To save data...**

1. Using the right mouse button, select **Save** from the shortcut menu, or click on the save icon.
2. Specify the directory and file name.

**To copy data to the Clipboard...**

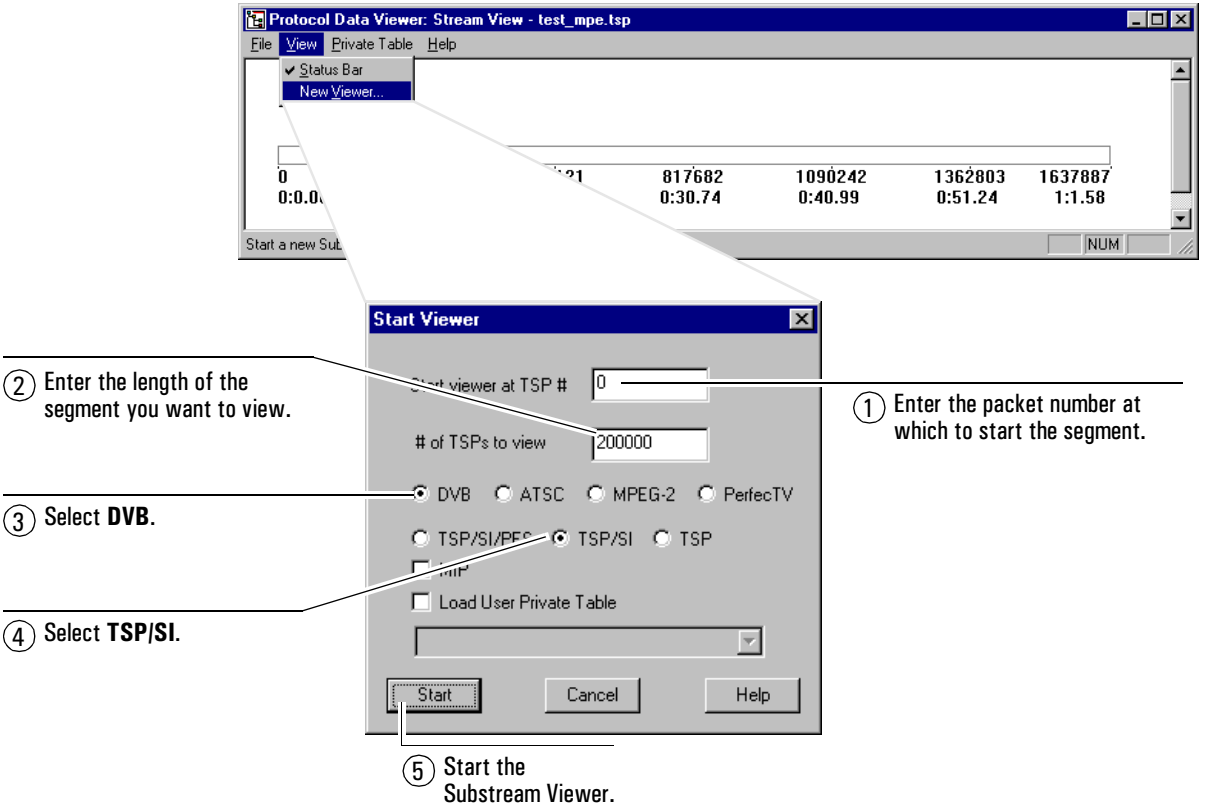
3. Holding down the left mouse button, select the area to copy.
4. Using the right mouse button, select **Copy Block** from the shortcut menu, or click on the copy icon.



# Analyzing an MPE transport stream

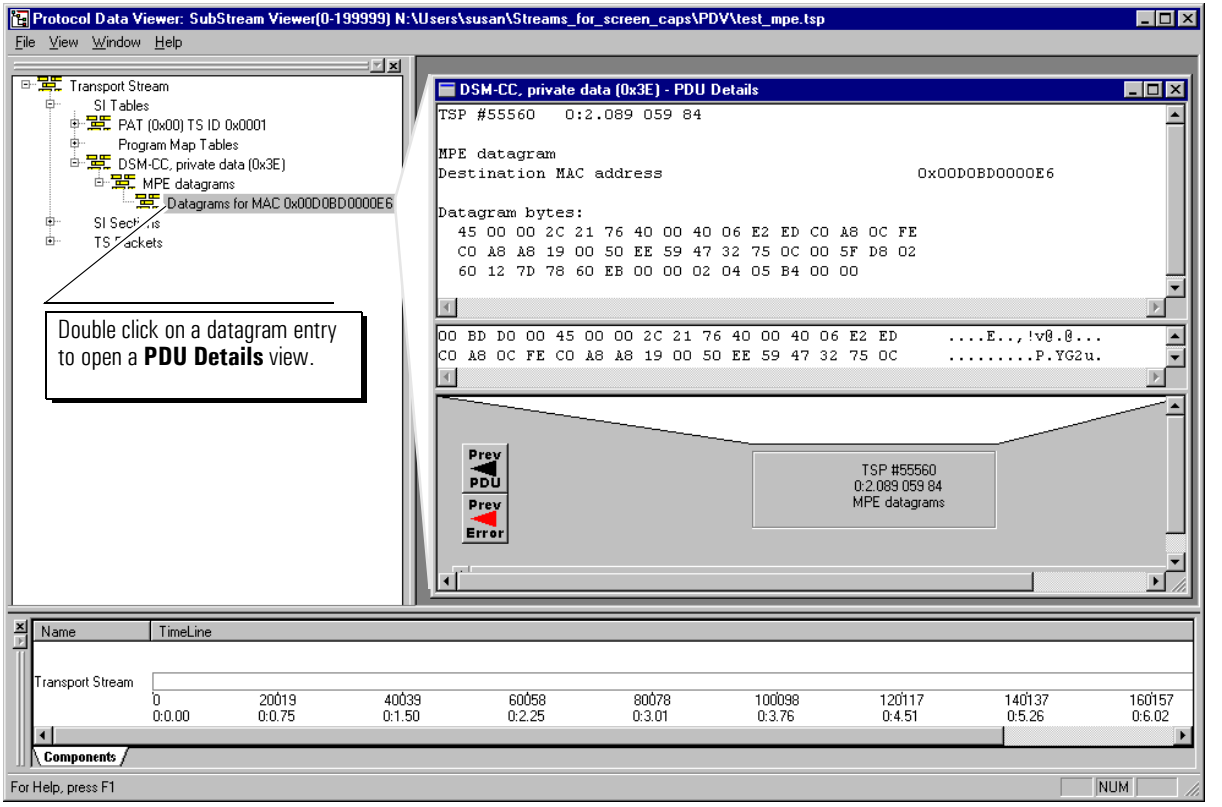
The Protocol Data Viewer can decode and analyze transport streams containing encapsulated datagrams carried in DSM-CC sections (multiprotocol encapsulation, or MPE) as specified in the DSM-CC section format for private data. This section illustrates the **DSM-CC, private view** stream element and shows how to extract and save the datagram to a file.

## Step 1: Open a substream view

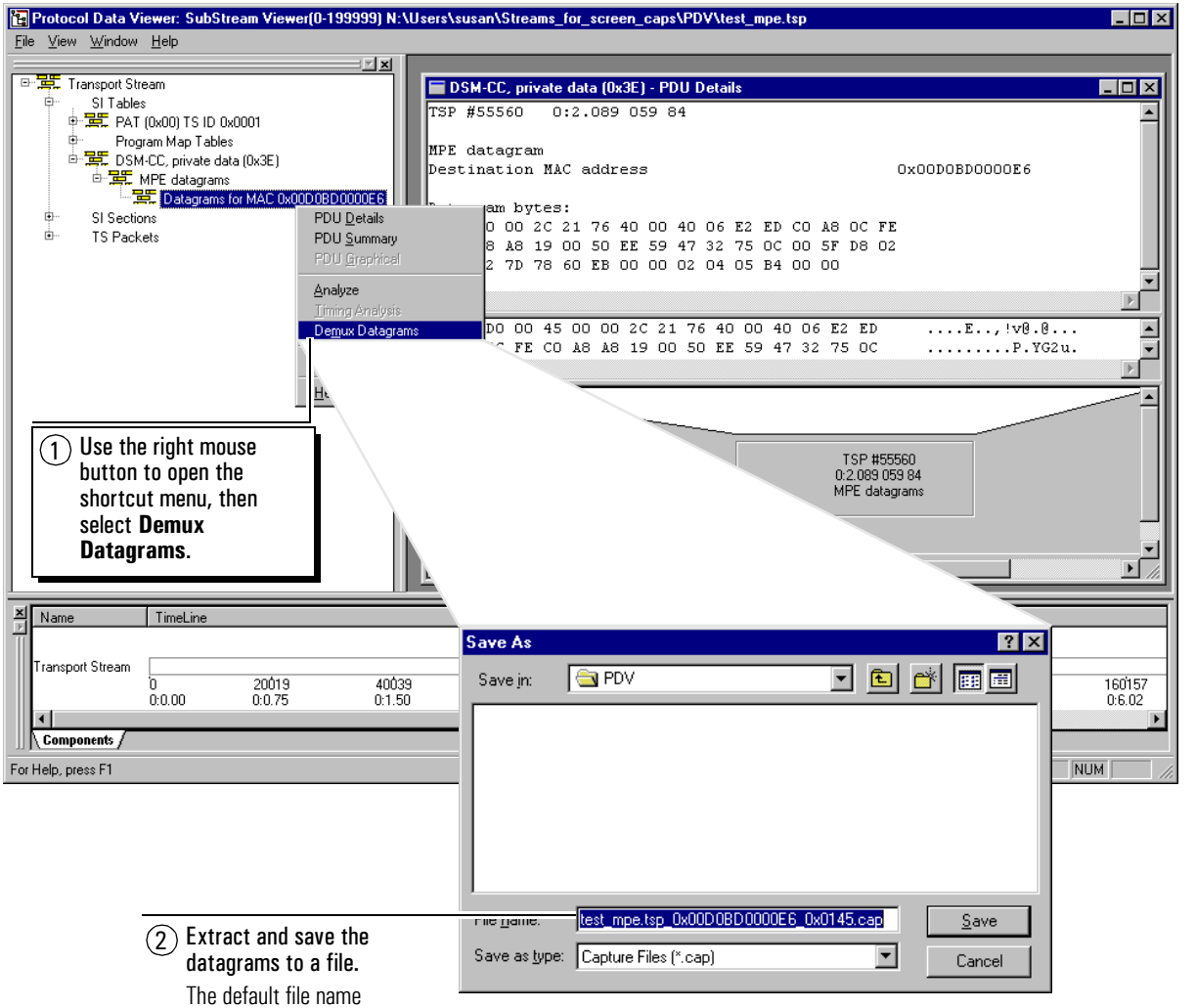


## Step 2: View datagrams by MAC address

From the **SI Tables** level in the **TS Hierarchy** view, you can see the contents of the DSM-CC private data tables. Datagrams are listed by MAC address in the tree hierarchy. From this level the **PDU Details** view provides the decoded address and a hexadecimal display of the datagram.



## Step 3: Save the datagrams




① Use the right mouse button to open the shortcut menu, then select **Demux Datagrams.**

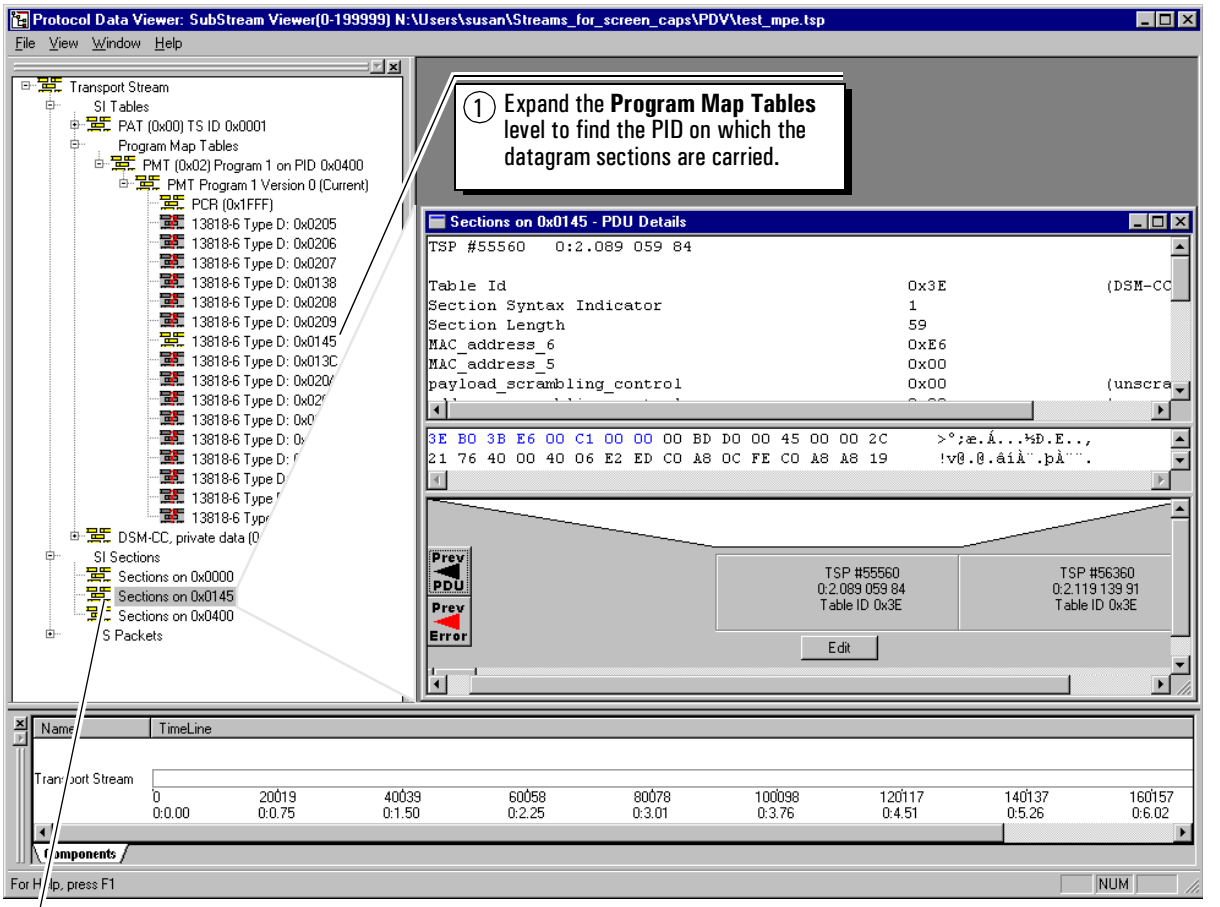
② **Extract and save the datagrams to a file.**

The default file name contains the MAC address and PID for the datagrams.

The Protocol Data Viewer saves the datagrams to a **capture** (**.cap**) file, which is a file format used by the Agilent LAN Analyzer (J1950A). You can then load and analyze the file from the LAN Analyzer.

## Step 4: View datagram sections

You can also view MPE datagram sections from the **SI Sections** level.

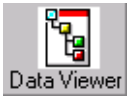



2 **Double click on the section entry for the PID you want to anlayze.**

This view shows the decoded datagram sections on the specified PID.

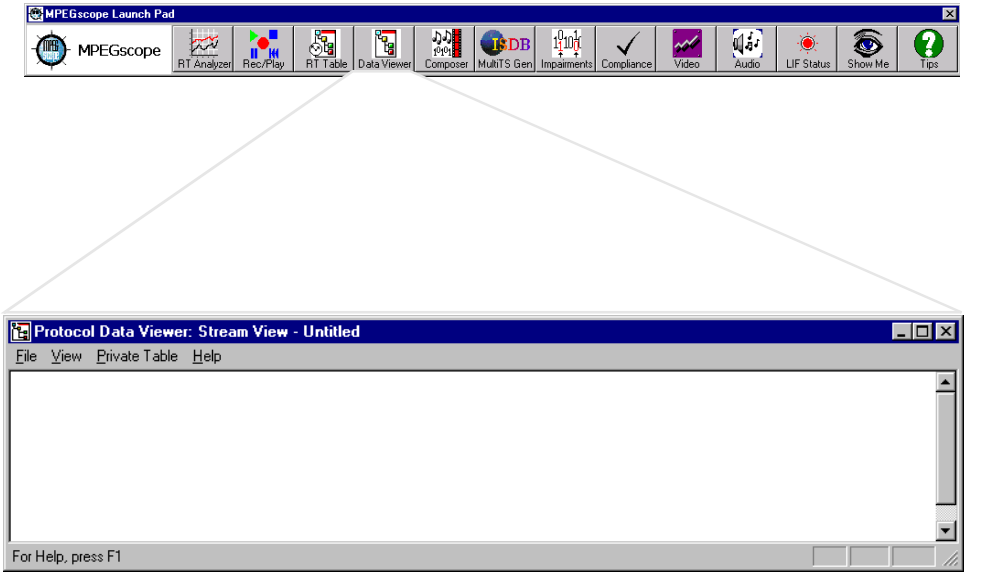
See previous sections in this chapter for information on analyzing the stream and viewing the data through other Protocol Data Viewer views.

# Analyzing a private table

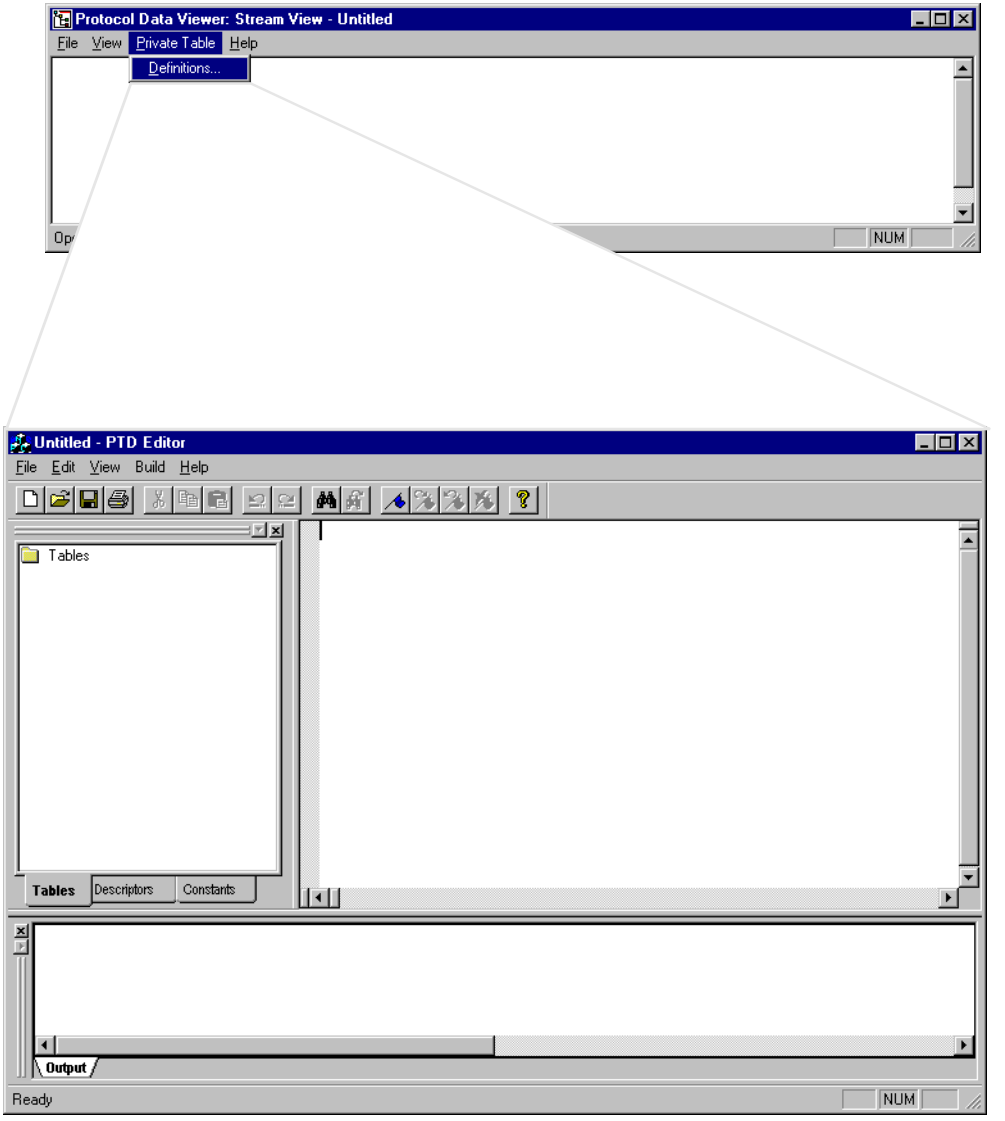



This example illustrates how to compile a private table definition, then analyze a private table from the Protocol Data Viewer. In the current release, you must first manually define the structure of the private table you want to analyze. You can then compile this definition file from the Protocol Data Viewer and use it as a template for decoding the private table.

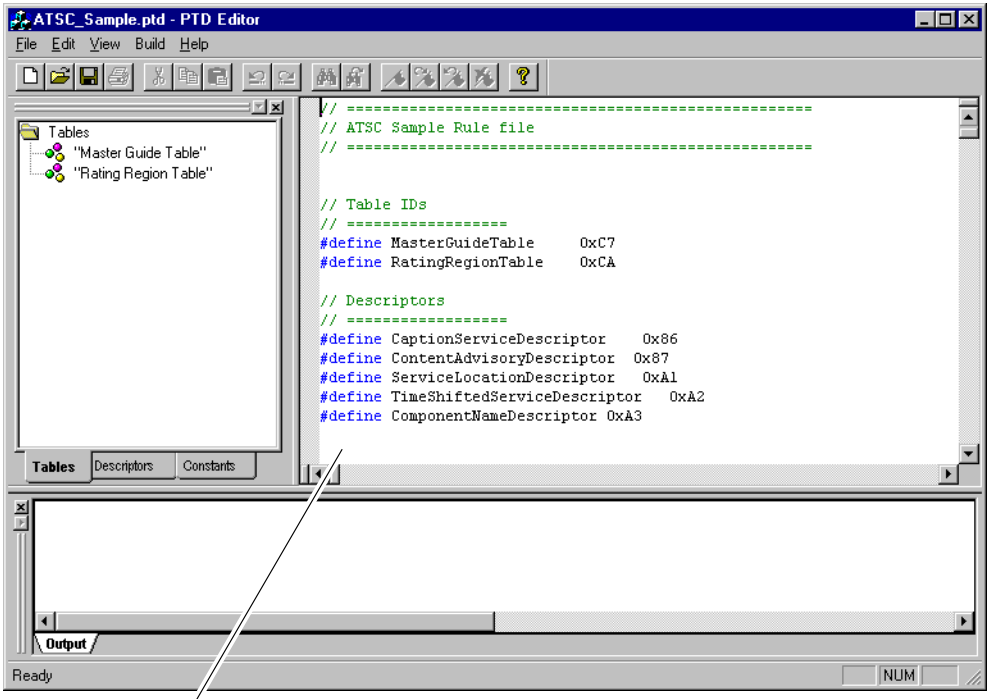
## Step 1: Start the Protocol Data Viewer



## Step 2: Start the PTD Editor



## Step 3: Create a private table definition (PTD) source file




Using the Protocol Data Viewer's PTD script language, define the structure of the private tables and descriptors from the PTD Editor.

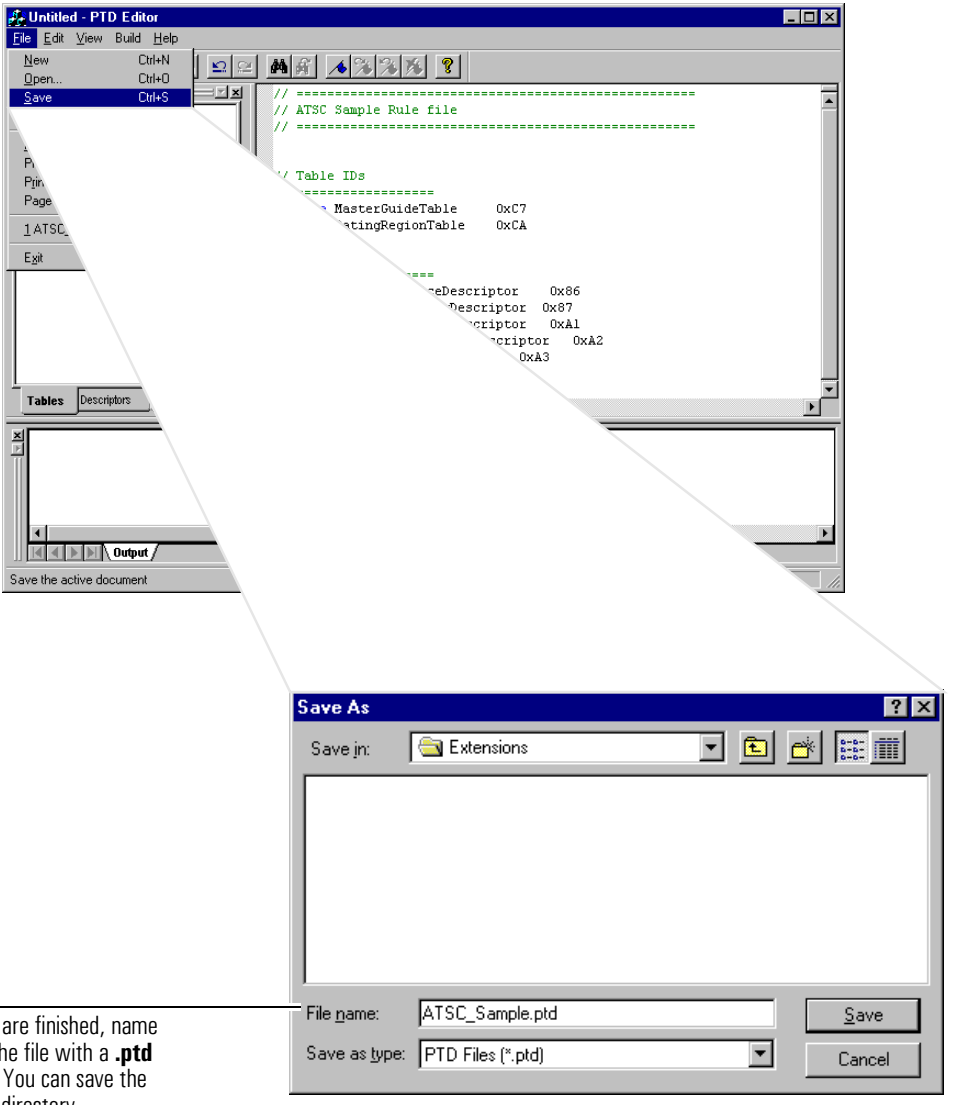
For help with the PTD script language, refer to the "Private table definition script reference", page 6–51.

This example shows the sample ATSC definition file included with the Protocol Data Viewer software. You can define any private table in your **.ptd** file.

The PTD Editor automatically color-codes syntactic elements as follows:

| | |
|---|---|
| green | comments |
| blue | keywords |
| red | quoted strings—this stext will display in the Protocol Data Viewer's **PDU Details** view |
| black | all other elements |

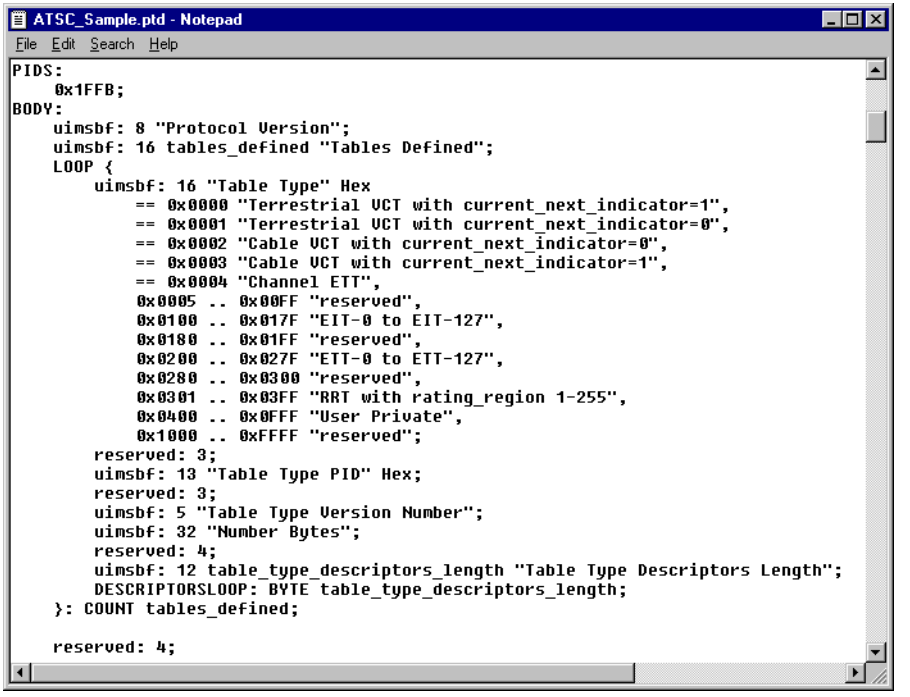
## Step 4: Save the file




When you are finished, name and save the file with a **.ptd** extension. You can save the file to any directory.

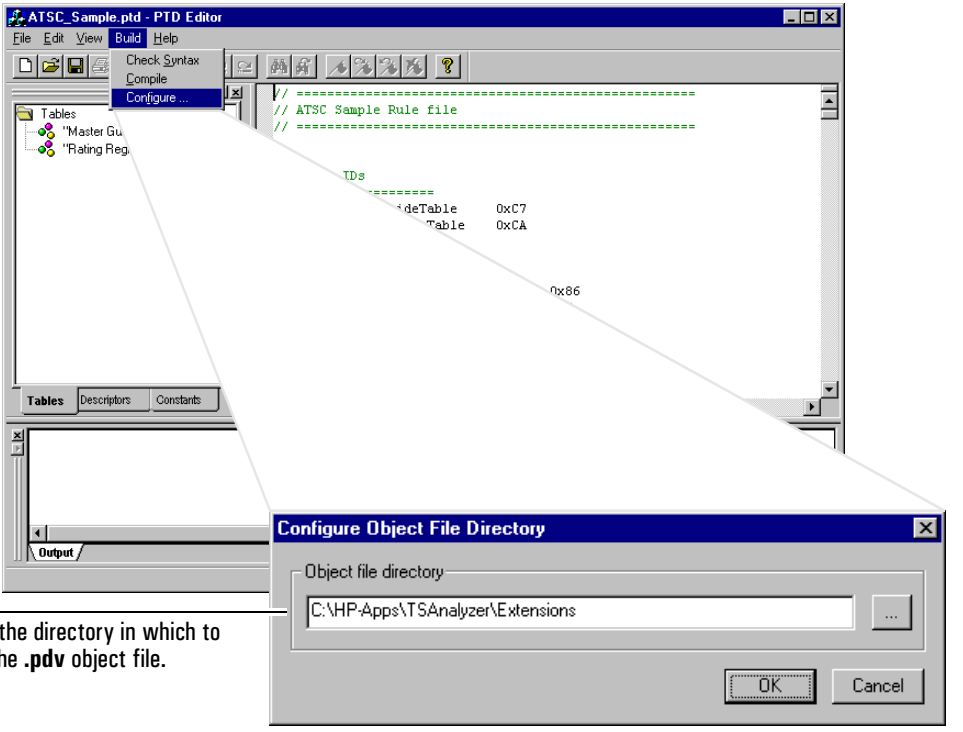
You can also use Notepad or another Windows-based text editor to create the private table definition file.

Save all private table definition files with a **.ptd** extension.



## Step 5: Compile the private table definition file

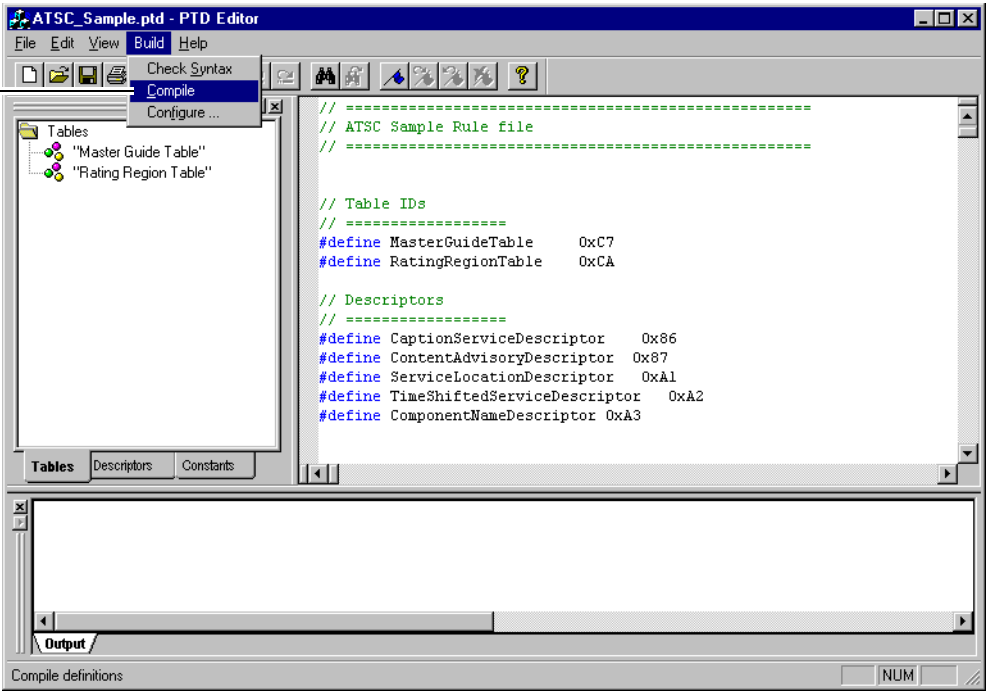
When you compile the **.ptd** source file, the PTD Compiler creates a **.pdv** object file that extends the Protocol Data Viewer's decoding capability.




1 Select the directory in which to place the **.pdv** object file.

Although you can store **.pdv** files in any directory, the Protocol Data Viewer can only use them to extend decoding if they are stored in the **C:\HP-Apps\TSAnalyzer\Extensions** directory.

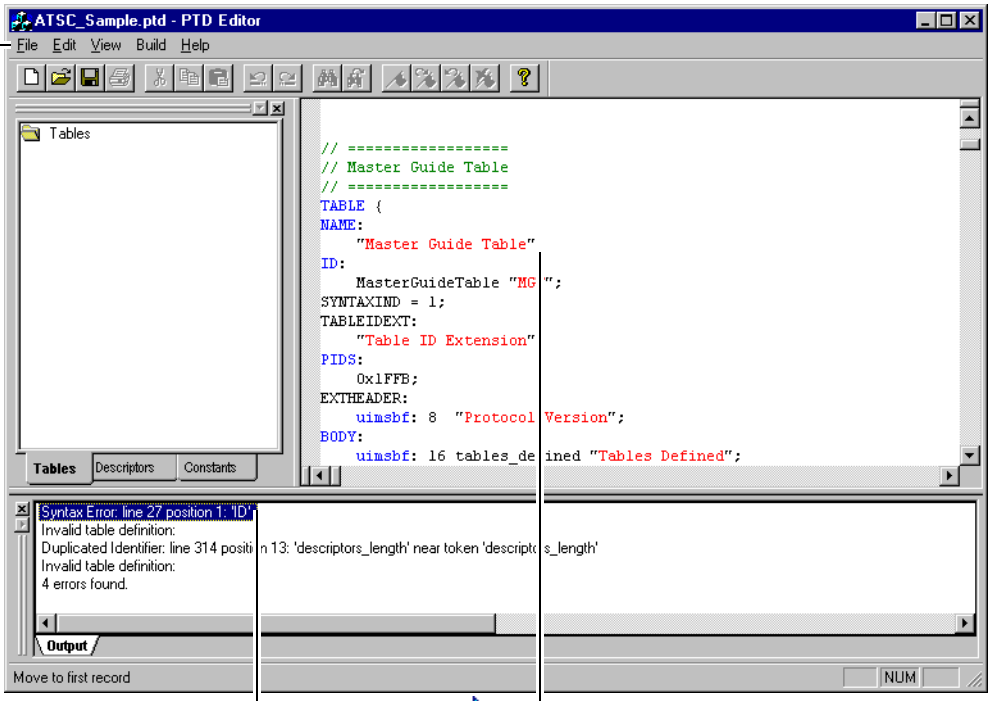
② Check the syntax and/or compile the file.




You can either compile the file directly or check the syntax first. Checking the syntax does not create a **.pdv** object file.

If you modify the source **.ptd** file, the PTD Editor will prompt you to save the file each time you compile or check the syntax.

④ When you have successfully compiled, select **Exit** from the **File** menu to close the PTD Editor.

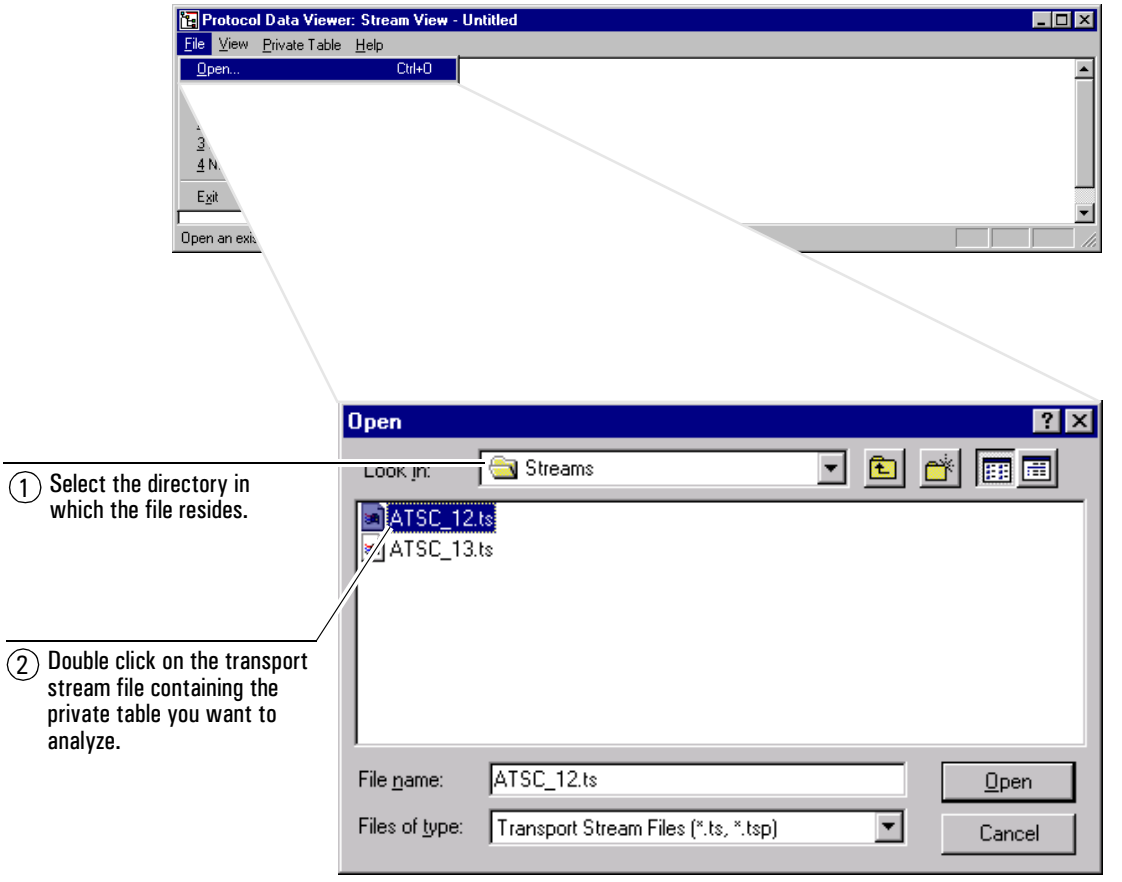



③ **If errors occur, double click on the first error to find its location in the file. The error often occurs directly before the cited position.**

Save and recompile the file after you correct each error until the compiler reports "No Errors Found".

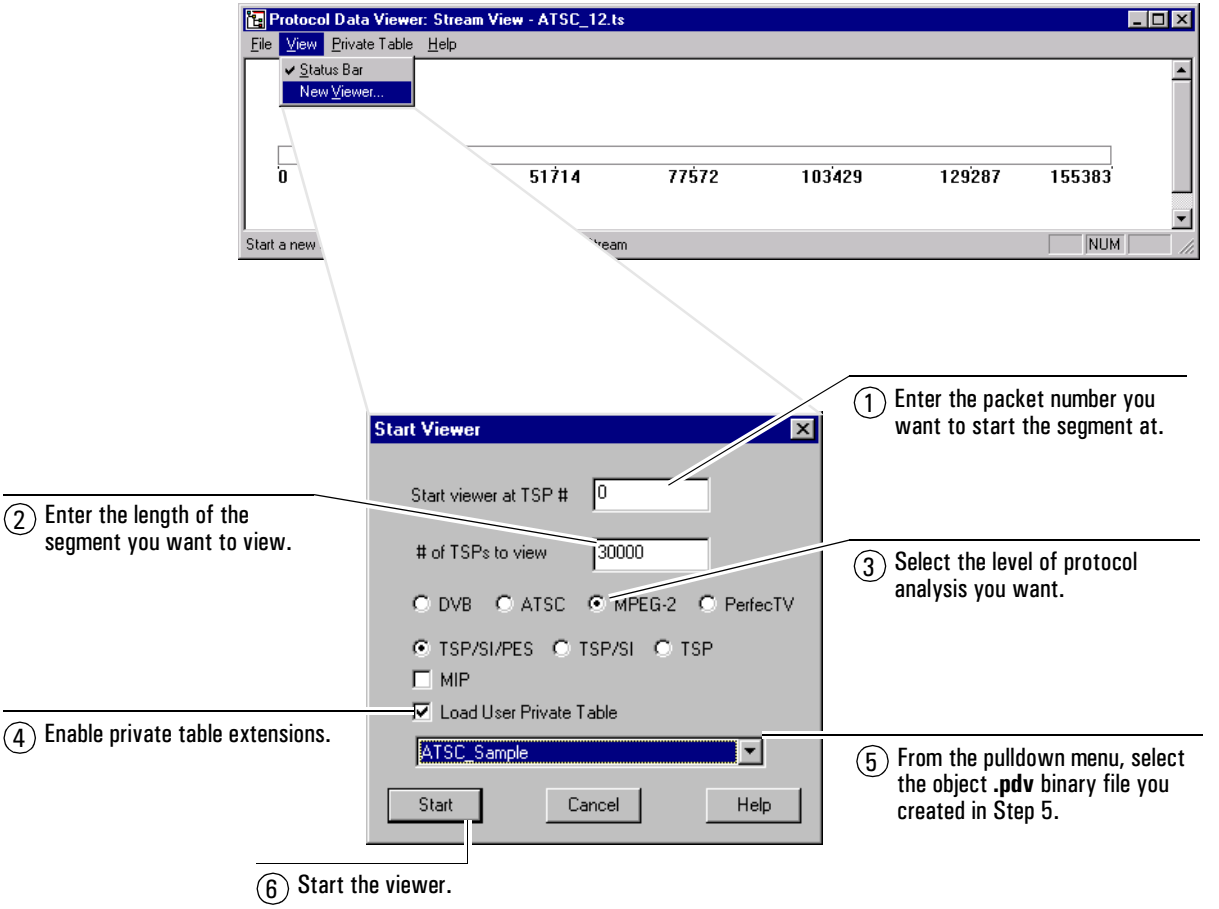
Error: Missing semi-colon.The PTD compiler reports the error on line 27, position 1 ("ID"), but the error actually occurs on the last position of line 26.

## Step 6: Open a transport stream file



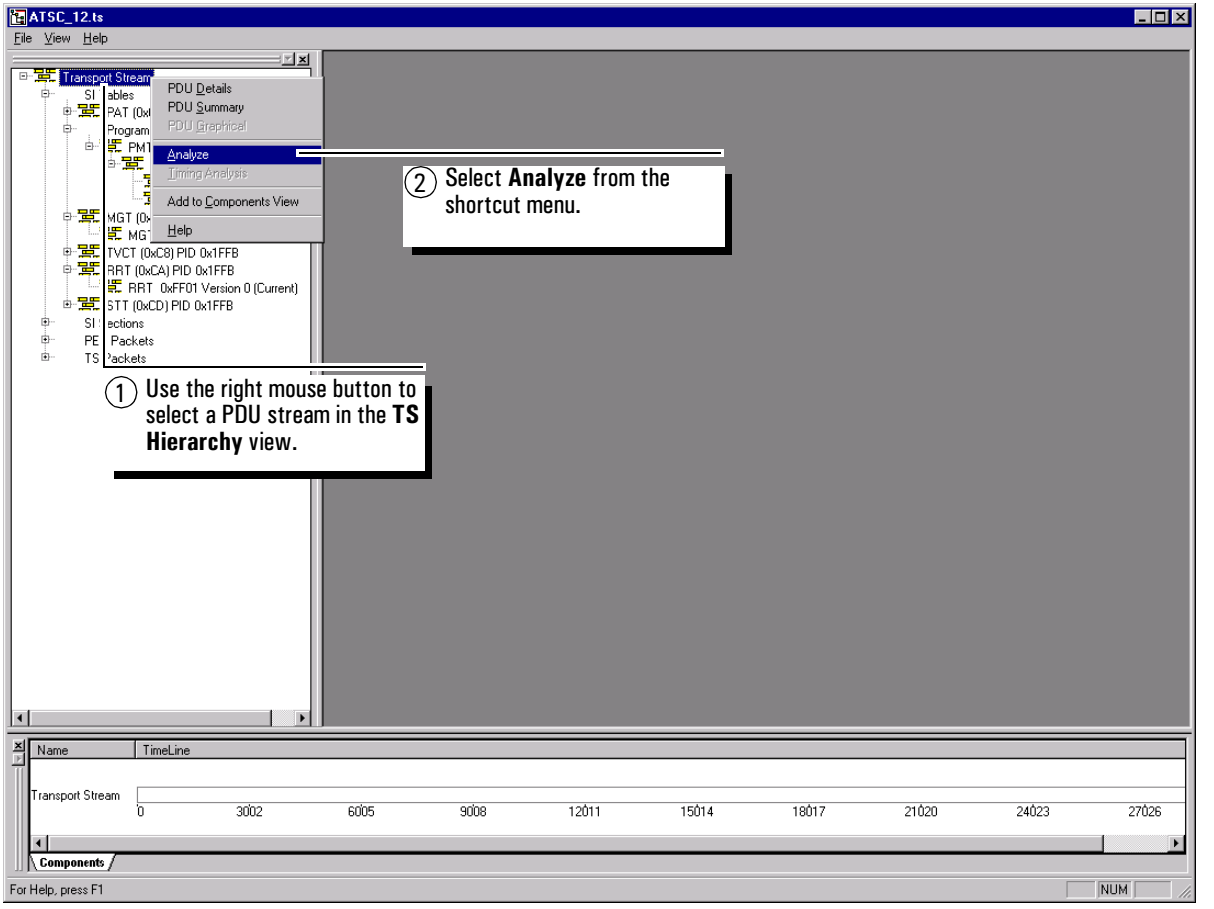

1. Select the directory in which the file resides.
2. Double click on the transport stream file containing the private table you want to analyze.

## Step 7: Create a substream using the new extension



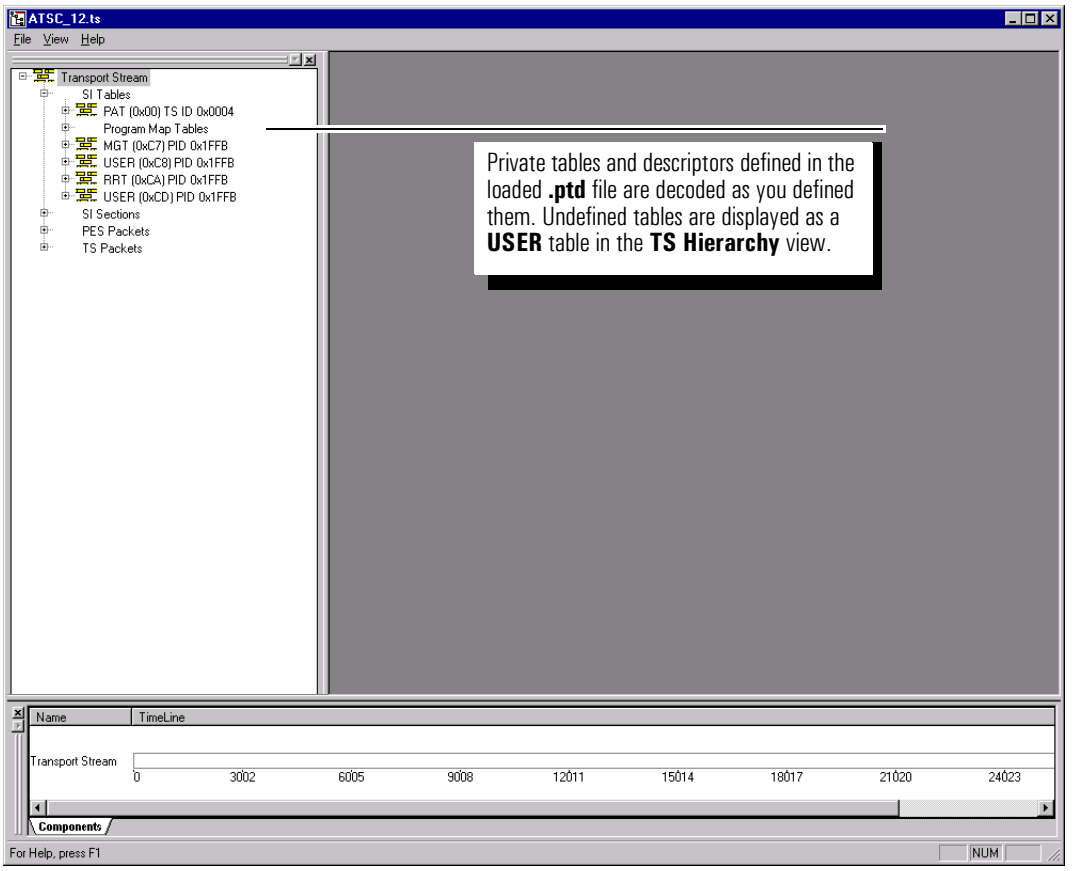

1. Enter the packet number you want to start the segment at.
2. Enter the length of the segment you want to view.
3. Select the level of protocol analysis you want.
4. Enable private table extensions.
5. From the pulldown menu, select the object **.pdv** binary file you created in Step 5.
6. Start the viewer.

## Step 8: Analyze data in the substream



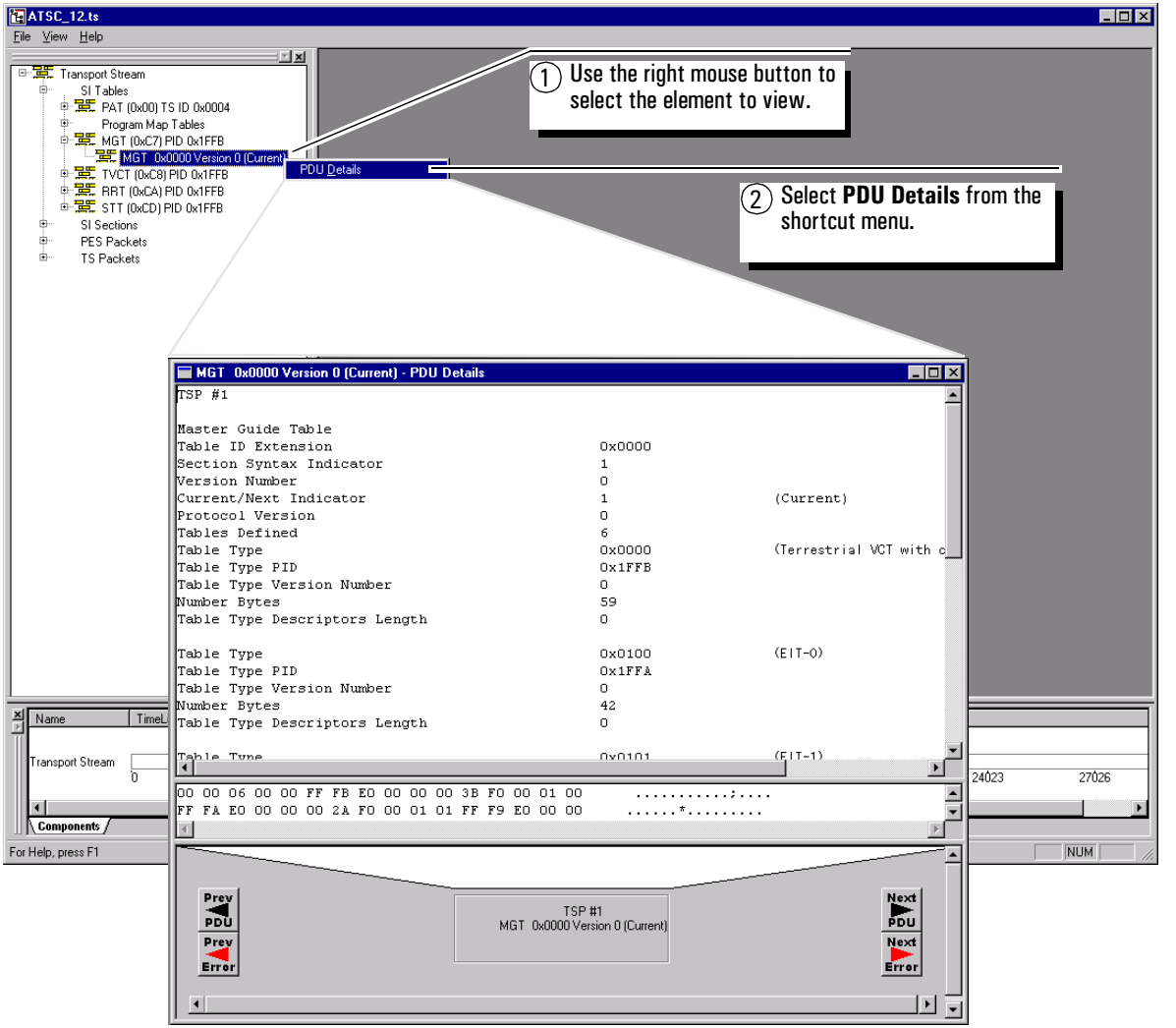

After analyzing, icons may change color. For an explanation of the color-coded icons in the **TS Hierarchy** view, refer to page 6–8.

For more information on analyzing files with the Protocol Data Viewer, refer to "Analyzing a transport stream" pages 6–2 to 6–33.


ATSC_12.ts
File View Help
Transport Stream
SI Tables
PAT (0x00) TS ID 0x0004
Program Map Tables
MGT (0xC7) PID 0x1FFB
USER (0xC8) PID 0x1FFB
RRT (0xCA) PID 0x1FFB
USER (0xCD) PID 0x1FFB
SI Sections
PES Packets
TS Packets
Private tables and descriptors defined in the loaded **.ptd** file are decoded as you defined them. Undefined tables are displayed as a **USER** table in the **TS Hierarchy** view.
Name
TimeLine
Transport Stream
0
3002
6005
9008
12011
15014
18017
21020
24023
Components
For Help, press F1
NUM

## Step 9: Check the **PDU Details** view

Table data will be decoded into table fields and descriptors as defined in the private table definition.



# Private table definition script reference

This reference describes the private table definition (PTD) script language used in the Protocol Data Viewer. The purpose of this language is to define the structures and characteristics of user private tables and descriptors which can exist in MPEG transport streams.

The PTD script language allows you to define table names, table IDs, section syntax indicator, associated PIDs, and each of the table fields. All other table elements are automatically defined and decoded by the Protocol Data Viewer. The PTD script language also allows you to extend the table header to any (byte-aligned) length, to rename the table_id_extension field or restructure the 16 bits into subfields, and to define user private descriptors.

This reference contains the following sections:

- "Structure of a PTD file"
- "Language constructs"
- "Syntax summary"
- "Sample private table definition (PTD) file"

## Structure of a PTD file

The basic structure of a private table definition (.ptd) file is simple:

1 Define symbolic constants (optional)

2 Define tables

3 Define descriptors

This can be further broken down as follows:

*define symbolic constants*

*define tables*
- *define table header*
  - *define table name*
  - *define table ID*
  - *define section syntax indicator*
  - *define table ID extension*
  - *define table PIDs*
- *define table body*
  - *define table statements*

*define descriptors*
- *define descriptor header*
  - *define descriptor tag*
  - *define descriptor name*
- *define descriptor body*
  - *define descriptor statements*

Depending on the table or descriptor, statements can consist of complex fields, if statements, or loops. The basic building blocks of the PTD script language are explained in “Language constructs” on pages 6–53 to 6–58, and a quick reference for the syntactic elements of the language is provided in “Syntax summary” on pages 6–59 to 6–64. You can also study the sample PTD file on pages 6–65 to 6–77 for an example of complete table and descriptor definitions.

## Language constructs

This section describes the main elements of the PTD script language and provides examples for many of the constructs. For an example of a complete table or descriptor definition, refer to the sample PTD file on pages 6–65 to 6–77.

**table**

Defines the structure of a user private table as specified in ISO/IEC 13818-1, section 2.4.4.10.

Within the table header you can define the **table_id**, and **section_syntax_indicator**. You can also rename the **table_id_extension** field or restructure the 16 bits into subfields. Other table header elements are automatically defined by the Protocol Data Viewer.

| | **No. of bits** | **User-definable** |
|---|---|---|
| private_section header | | |
| **table_id** | **8** | ✔ |
| **section_syntax_indicator** | **1** | ✔ |
| private_indicator | 1 | |
| reserved | 2 | |
| private_section_length | 12 | |
| **table_id_extension** | **16** | ✔ |
| reserved | 2 | |
| version_number | 5 | |
| current_next_indicator | 1 | |
| section_number | 8 | |
| last_section_number | 8 | |

Within the entire table you can define the table name, table ID, section syntax indicator, table ID extension, associated PIDs (if any), and each of the fields.

**Example:**

```
TABLE {
NAME:        "Master Guide Table";
ID:          MasterGuideTable "MGT";
SYNTAXIND    = 1;
TABLEIDEXT:  "Table ID Extension";
PIDS:        0x1FFB;
BODY:        uimsbf : 8 "Protocol Version";
             uimsbf : 16 tables_defined "Tables Defined";
             ... etc. }
```

**descriptor** Defines the structure of any user private descriptors as specified in ISO/IEC 13818-1.

**Example:**
```
DESCRIPTOR CaptionServiceDescriptor {
NAME:   "Caption Service Descriptor";
BODY:   reserved: 3;
        uimsbf: 5 number_of_services "Number of
                Services";
        ... etc. }
```

**name-declarator** Defines the name of a private table or descriptor that will display in the PDV's **PDU Details** view.

**Example:**
```
NAME:   "Time Shifted Service Descriptor";
```

**table-id** Defines the associated table IDs for the private table. In the example below, "MasterGuideTable" is the symbolic constant that defines the table ID, and "MGT" is the text that will display in the PDV's **TS Hierarchy** view.

**Example:**
```
ID:     MasterGuideTable "MGT";
```

**section-syntax-indicator** Defines the default value for the **section_syntax_indicator** flag. If you do not define a specific value, the Protocol Data Viewer will check the value of this flag and decode the table according to section 2.4.4.10 (Syntax of the Private section) of ISO/IEC 13818-1.

If you define the default value as 0, the Protocol Data Viewer expects the private data to start immediately after the **private_section_length** field.

If you define the default value as 1, the Protocol Data Viewer expects five more bytes of header information after **private_section_length** comprising the following fields: **table_id_extension** (16 bits), **reserved** (2 bits), **version_number** (5 bits), **current_next_indicator** (1 bit), **section_number** (8 bits), and **last_section_number** (8 bits).

**Example:**
```
SYNTAXIND  = 0;
```

**table-pids** Defines the allowable PIDs for the private table.

**Example:**
```
PIDS:   0x1FFB;
```

**table-id-extension** Allows you to override the default name or decoded structure of the **table_id_extension** field found in the section header (ISO/IEC 13818-1, section 2.4.4.10). You can use this construct to override the text for this field displayed in the PDV's **PDU Details** view, or you can redefine the 16 bits in the subfields. The default decode type of this field is a 16-bit unsigned integer.

**Example:**
```
TABLEIDEXT:  "Table ID Extension";
```

or, to redefine the subfields:

```
TABLEIDEXT:
        reserved: 8;
        uimsbf: 8 "Rating Region";
```

**table-extended-header** Allows you to extend the default header from 3 or 8 bytes (depending on the value of the **section_syntax_indicator**) to any length, providing the total header section is byte-aligned (i.e., a multiple of 8 bits).

**Example:**
```
EXTHEADER:
        uimsbf: 16 "Original Network ID";
        reserved: 8;
```

**table-statement** Defines the data content of a private table. A table statement can be a complex field, an if statement, a loop, or a descriptors loop. For examples of table statements, refer to the table body statements of the Master Guide Table on page 6–66.

**descriptor-statement** Defines the data content of a descriptor. Similar to a table statement, except a descriptor statement cannot be a descriptors loop since a descriptor cannot be nested inside another descriptor. For examples of descriptor statements, refer to the descriptor definitions on pages 6–73 to 6–77.

**complex-field** Can be a (normal) field, a field-array, or an aggregated-field (see below).

**field-array** A convenient way to define a loop of a single field. For example, instead of

```
for (i=0; i<N; i++)
    uimsbf : 8 "data";
```

you can define the loop as

```
array : count N uimsbf : 8 "data";
```

**field** Defines any single decoded data item occurring in a private table or descriptor. The simplest form of a field is

```
type-specifier : N;
```

where **type-specifier** can be **uimsbf**, **bslbf**, or **tcimsbf**, and **N** is the number of bits associated with the item.

Other optional parameters are as follows:

**identifier** Associates a unique identifier to a field to be used as a control variable for an if or loop statement.

**string** A text string to be displayed as the field name in a decoded **PDU Details** window.

**display-type** The type of formatting when the value is displayed in a decoded **PDU Details** window:

- **hide** Do not display.
- **decimal** Display as a decimal integer. (If the field is longer than 32 bits, it will display in hexadecimal even if defined as a decimal field.)
- **hex** Display in hexadecimal.
- **bcd** Display as a binary-coded integer.

**bcd : N** Display as a binary-coded integer with a decimal point in the $N^{th}$ position from the left. For example 0x1234 formatted as bcd:2 will be displayed as 12.34.

**mjdutc** Displayed in Modified Julian Date and Coordinated Universal Time.

**bcdtime** Displayed as hh:mm:ss.

**ascii** Display an 8-bit field as a character. This display type is usually used in an array, as in the following example:

**Example:** ARRAY: COUNT 7 uimsbf : 8 "Short Name" ascii;

**iso639lang** Display a 24-bit field as one hexadecimal number comprising three 8-bit hexadecimal numbers. This display type is used specifically in text strings.

**Example:** uimsbf: 24 "ISO 639 Language Code" ISO639LANG;

Alternatively, you can use the **hex** display type and code the statement as follows:

**Example:** ARRAY: COUNT 3 uimsbf: 8 "ISO 639 Language Code" Hex;

**field-semantics** Associates a user-defined field with a possible value or range of values.

**Example:**
```
uimsf:  3 “logical_cell_presentation_info”
        = = 0x00 “undefined”,
        = = 0x1   “Video”,
        = = 0x2   “Still picture”,
        = = 0x3   “Graphics/Text”,
        0x04 .. 0x07 “reserved for future use”;
```

A field can also be defined for reserved data items as shown below. Note that reserved fields are not displayed in the **PDU Details** view.

**Example:** reserved : N;

**aggregated-field** Associates the values of two or more consecutive fields as in the example below where values are associated with both the **stream_content** *and* the **component_type** fields.

**Example:**
```
aggregated {
        uimsbf: 4 “stream_content”;
        uimsbf: 8 “component_type”;
}       0x000 .. 0x100 “reserved for future use”,
        = =0x101 “video, 4 : 3 aspect ratio”,
        = =0x102 “video, 16 : 9 aspect ratio with pan vectors”;
```

**if-statement** A control structure used to express the conditional execution of one or more statements. An if statement evaluates a condition to determine the decoding path. The controlling data item must be within the scope of the current table or descriptor. (Scoping rules are similar to variable scoping rules in the C language.) It must also be defined before it is used as a control element in the if statement.

**Example:** In this example the identifier **cc_type** is used as a control for the if statement.

```
bslbf: 1 cc_type "CC Type" Hex;

if (cc_type = = 0)
{
    reserved: 5;
    bslbf: 1 "Line21 field" Hex;
}
else
{
    uimsbf: 6 "Caption Service Number " Hex;
}
```

**loop-statement** An iterative control structure used to execute a sequence of statements repeatedly. Loop statements are specified frequently in ISO/IEC 13838-1 as

```
for (i = 0; i < N; i + + )
... etc.
```

When defining tables or descriptors, you can use the loop statement to associate a control decoded data item with a series of possible values or ranges of values. If the loop is not associated with the control data item, the decoding process will continue until the section is exhausted. As with an if statement, the controlling data item must be within the scope of the current table or descriptor and must be defined before it is used as a control in the loop statement.

You can specify two types of loop size controls:

**byte** The value of the associated data item will be interpreted as the total number of bytes available within the loop.

**count** The value of the associated data item will be interpreted as the total number of counts, or times, the decoder needs to process the loop.

**Example:** In this example, the identifier **tables_defined** is used as a control for the loop.

```
LOOP {
    uimsbf: 16 "Table Type" Hex
        = = 0x0000 "Terrestrial VCT with
                    current_next_indicator = 1",
        = = 0x0001 "Terrestrial VCT with
                    current_next_indicator = 0",
        = = 0x0002 "Cable VCT with
                    current_next_indicator = 0",
        = = 0x0003 "Cable VCT with
                    current_next_indicator = 1",
        = = 0x0004 "Channel ETT",
        0x0005 .. 0x00FF "reserved",
        0x0100 .. 0x017F "EIT-0 to EIT-127",
        0x0180 .. 0x01FF "reserved",
        0x0200 .. 0x027F "ETT-0 to ETT-127",
        0x0280 .. 0x0300 "reserved",
        0x0301 .. 0x03FF "RTT with rating_region 1-255",
        0x0400 .. 0x0FFF "User Private",
        0x1000 .. 0xFFFF "reserved";
    reserved: 3;
    uimsbf: 13 "Table Type PID" Hex;
    reserved: 3;
    uimsbf: 5 "Table Type Version Number";
    reserved: 4;
    uimsbf: 12 table_type_descriptors_length "Table Type
            Descriptors Length";
    DESCRIPTORSLOOP: BYTE table_type_descriptors_length;
} : COUNT tables_defined;
```

## Syntax summary

In the reference below, syntactic categories are in *italic* type. Literal words and characters are in **bold** type. Alternative elements are listed on separate lines. For example, in the entry for **constant**, **number** and **defined-constant** are listed on separate lines. This means that a constant can take the form of a number *or* a defined constant. A recursive element is indicated when the element itself is listed as an alternative. For example, a **table-statement-list** can be a **table-statement** plus another **table-statement-list**, effectively allowing you to have any number of table statements.

The following are reserved keywords and may not be used in any other context. All keywords are case-insensitive.

| | | |
|---|---|---|
| **aggregated** | **descriptor** | **mjdutc** |
| **array** | **descriptorsloop** | **name** |
| **ascii** | **else** | **pids** |
| **bcd** | **extheader** | **reserved** |
| **bcdtime** | **hex** | **syntaxind** |
| **body** | **hide** | **table** |
| **bslbf** | **id** | **tableidext** |
| **byte** | **if** | **tcimsbf** |
| **count** | **iso639lang** | **uimsbf** |
| **decimal** | **loop** | |

| | |
|---|---|
| **Constant** | *number*<br>*defined-constant* |
| **Defined-constant** | **# define** *identifier number* |
| **Number** | *integer-constant*<br>*negative-integer-constant*<br>*hex-constant*<br>*octal-constant* |
| **Table** | **table {** *table-declarator* **body :** *table-statement* **}** |
| **Table-declarator** | *name-declarator table-id*<br>*name-declarator table-id table-declarator-option* |
| **Name-declarator** | **name :** *string* **;** |
| **Table-id** | **id :** *table-id-list* **;** |
| **Table-id-list** | *constant string*<br>*constant string* **,** *table-id-list*<br>*range string*<br>*range string* **,** *table-id-list* |
| **Section-syntax-indicator** | **syntaxind = 1;**<br>**syntaxind = 0;** |
| **Table-declarator-option** | *table-pids*<br>*table-pids table-id-extension*<br>*table-pids table-id-extension table-extended-header*<br>*table-id-extension*<br>*table-id-extension table-extended-header*<br>*table-extended-header* |

| | |
|---|---|
| **Table-pids** | **pids :** *table-pids-list* **;** |
| **Table-pids-list** | *number*<br>*number* **,** *table-pids-list* |
| **Table-id-extension** | **tableidext :** *string* **;**<br>**tableidext :** *header-ext-field-list* **;** |
| **Table-extended-header** | **extheader :** *header-ext-field-list* **;** |
| **Header-ext-field-list** | *header-ext-field* **;**<br>*header-ext-field header-ext-field-list* |
| **Header-ext-field** | *type-specifier* **:** *number string* **;**<br>**reserved :** *number* **;** |
| **Descriptor** | **descriptor** *constant* **{** *name-declarator* **body :** *descriptor-statement* **}** |
| **Table-statement** | *complex-field-list*<br>*table-statement-list*<br>**if (** *expression* **) {** *table-statement* **}**<br>**if (** *expression* **) {** *table-statement* **} else {** *table-statement* **}**<br>**loop {** *table-statement* **};**<br>**loop {** *table-statement* **} :** *loop-size-specifier constant* **;**<br>**loop {** *table-statement* **} :** *loop-size-specifier identifier* **;**<br>*descriptors-loop* **;** |
| **Table-statement-list** | *table-statement*<br>*table-statement table-statement-list* |

**Descriptor-statement**

*complex-field-list*
*descriptor-statement-list*
**if (** *expression*) **{** *descriptor-statement* **}**
**if (** *expression* ) **{** *descriptor-statement* **} else {** *descriptor-statement* **}**
**loop {** *descriptor-statement* **} ;**
**loop {** *descriptor-statement* **} :** *loop-size-specifier constant* **;**
**loop {** *descriptor-statement* **} :** *loop-size-specifier identifier* **;**

**Descriptor-statement-list**

*descriptor-statement*
*descriptor-statement descriptor-statement-list*

**Descriptors-loop**

**descriptorsloop**
**descriptorsloop :** *loop-size-specifier constant*

**Complex-field-list**

*complex-field*
*complex-field complex-field-list*

**Complex-field**

*field*
*field-array*
*aggregated-field*

**Type-specifier**

| | |
|---|---|
| **uimsbf** | (unsigned integer, most significant bit first) |
| **bslbf** | (bit string, left bit first) |
| **tcimsbf** | (two's complement integer, most significant (sign) bit first) |

**Aggregated-field**

**aggregated {** *field-list* **} ;**
**aggregated {** *field-list* **}** *field-semantics-list* **;**

**Field-list**

*field*
*field field-list*

**Field-array**

**array** *field*
**array :** *loop-size-specifier constant field*
**array :** *loop-size-specifier identifier field*

**Loop-size-specifier**

**byte**
**count**

**Field**

*type-specifier* **:** *number identifier string* **;**
*type-specifier* **:** *number identifier string field-modifier* **;**
*type-specifier* **:** *number identifier* **;**
*type-specifier* **:** *number identifier field-modifier* **;**
*type-specifier* **:** *number string* **;**
*type-specifier* **:** *number string field-modifier* **;**
*type-specifier* **:** *number* **;**
*type-specifier* **:** *number field-modifier* **;**
**reserved :** *number* **;**

**Field-modifier**

*display-type*
*display-type field-semantics-list*
*field-semantics-list*

**Display-type**

| | |
|---|---|
| **hide** | (not displayed) |
| **decimal** | (displayed in decimal) |
| **hex** | (displayed in hexadecimal) |
| **bcd** | (displayed in binary-coded decimal) |
| **bcd : number** | (displayed in binary-coded decimal, with decimal in the $N^{th}$ position from left) |
| **mjdutc** | (displayed in Modified Julian Date and Coordinated Universal Time) |
| **bcdtime** | (displayed in binary-coded decimal time as hh:mm:ss) |
| **ascii** | (displayed as an 8-bit ASCII character) |
| **iso639lang** | (displayed as a 24-bit hexadecimal number) |

**Field-semantics-list**

*field-semantics*
*field semantics , field-semantics-list*

| | |
|---|---|
| **Field-semantics** | *operator number string*<br>*range string* |
| **Range** | *constant* **..** *constant* |
| **Expressions** | *identifier operator identifier*<br>*identifier operator constant* |
| **Operators** | <<br>><br><=<br>>=<br>==<br>!= |

# Sample private table definition (PTD) file

This section shows a sample ATSC (Advanced Television Systems Committee) private table definition file that defines a number of tables and descriptors. You can use this as an example when building your own PTD file.

**Conventions**

As in the C programming language, the PTD script language is case insensitive. You can use upper, lower, or mixed case. White space—blanks, tabs, newlines, formfeeds, and comments—is ignored by the compiler. The block capitals, mixed case, and indents in the sample below exist only for ease of reading.

To write a comment line, use two forward slashes **//** at the beginning of each line. For multi-line comments, you can also use the characters **/*** to introduce the comment and the characters ***/** at the end of the comment. Punctuation is used as follows:

| | |
|---|---|
| **;** (semi-colon) | Indicates the end of a statement. |
| **:** (colon) | Separates a keyword from the statements that follow. |
| **,** (comma) | Indicates a repeating element (e.g., as in a list of fields). |

## Sample PTD file

```
// ========================================================================
// ATSC Sample Rule file
// ========================================================================


// Table IDs
// ==================
#define MasterGuideTable     0xC7
#define RatingRegionTable    0xCA

// Descriptors
// ==================
#define CaptionServiceDescriptor    0x86
#define ContentAdvisoryDescriptor  0x87
#define ServiceLocationDescriptor   0xA1
#define TimeShiftedServiceDescriptor   0xA2
#define ComponentNameDescriptor 0xA3
```

```
// ==================
// Master Guide Table
// ==================
TABLE {
NAME:
   "Master Guide Table";
ID:
   MasterGuideTable "MGT";
SYNTAXIND = 1;
TABLEIDEXT:
   "Table ID Extension";
PIDS:
   0x1FFB;
EXTHEADER:
   uimsbf: 8  "Protocol Version";
BODY:
   uimsbf: 16 tables_defined "Tables Defined";
   LOOP {
      uimsbf: 16 "Table Type" Hex
         == 0x0000 "Terrestrial VCT with current_next_indicator=1",
         == 0x0001 "Terrestrial VCT with current_next_indicator=0",
         == 0x0002 "Cable VCT with current_next_indicator=0",
         == 0x0003 "Cable VCT with current_next_indicator=1",
         == 0x0004 "Channel ETT",
         0x0005 .. 0x00FF "reserved",
         0x0100 .. 0x017F "EIT-0 to EIT-127",
         0x0180 .. 0x01FF "reserved",
         0x0200 .. 0x027F "ETT-0 to ETT-127",
         0x0280 .. 0x0300 "reserved",
         0x0301 .. 0x03FF "RRT with rating_region 1-255",
         0x0400 .. 0x0FFF "User Private",
         0x1000 .. 0xFFFF "reserved";
      reserved: 3;
      uimsbf: 13 "Table Type PID" Hex;
      reserved: 3;
      uimsbf: 5 "Table Type Version Number";
      uimsbf: 32 "Number Bytes";
      reserved: 4;
      uimsbf: 12 table_type_descriptors_length "Table Type Descriptors Length";
      DESCRIPTORSLOOP: BYTE table_type_descriptors_length;
   }: COUNT tables_defined;

   reserved: 4;
   uimsbf: 12 descriptors_length "Descriptors Length";
```

```
   DESCRIPTORSLOOP: BYTE descriptors_length;
}

// ==================
// Rating Region Table
// ==================
TABLE {
NAME:
   "Rating Region Table";
ID:
   RatingRegionTable "RRT";
SYNTAXIND = 1;
TABLEIDEXT:
   reserved: 8;
   uimsbf: 8 "Rating Region";
PIDS:
   0x1FFB;
BODY:
   uimsbf: 8 "Protocol Version";

   uimsbf: 8 rating_region_name_length "Rating Region Name Length";
   //rating_region_name_text()var
   // A/65 Table 6.24: Multiple String Structure on p.42
   uimsbf: 8 number_strings "Number of Strings";
   LOOP {
                  uimsbf: 24 "ISO 639 Language Code" ISO639LANG;
                  //obsolete
      //ARRAY: COUNT 3 uimsbf: 8 "ISO 639 Language Code" Hex;
      uimsbf: 8 number_segments "Number Segments";
      LOOP {
         uimsbf: 8 "Compression Type" Hex
            == 0x00 "No Compression",
            == 0x01 "Huffman Coding using English-language Program Title Encode/Decode Table",
            == 0x02 "Huffman Coding using English-language Program Description Encode/Decode Table",
            0x03 .. 0xAF "reserved",
            0xB0 .. 0xFF "User Private";
         uimsbf: 8 "Mode" Hex
            == 0x00 "Select ISO/IEC 10646-1 page 0x00, ASCII, ISO Latin-1 (Roman)",
            == 0x01 "Select ISO/IEC 10646-1 page 0x01, European Latin (many)",
            == 0x02 "Select ISO/IEC 10646-1 page 0x02, Standard Phonetic",
            == 0x03 "Select ISO/IEC 10646-1 page 0x03, Greek",
            == 0x04 "Select ISO/IEC 10646-1 page 0x04, Russian, Slavic",
            == 0x05 "Select ISO/IEC 10646-1 page 0x05, Armenian, Hebrew",
            == 0x06 "Select ISO/IEC 10646-1 page 0x06, Arabic",
```

```
            0x07 .. 0x08 "reserved",
            == 0x09 "Select ISO/IEC 10646-1 page 0x09, Devanagari, Bengali",
            == 0x0A "Select ISO/IEC 10646-1 page 0x0A, Punjabi, Gujarati",
            == 0x0B "Select ISO/IEC 10646-1 page 0x0B, Oriya, Tamil",
            == 0x0C "Select ISO/IEC 10646-1 page 0x0C, Telugu, Kannada",
            == 0x0D "Select ISO/IEC 10646-1 page 0x0D, Malayalam",
            == 0x0E "Select ISO/IEC 10646-1 page 0x0E, Thai, Lao",
            == 0x0F "reserved",
            == 0x10 "Select ISO/IEC 10646-1 page 0x10, Tibetan, Georgian",
            0x11 .. 0x1F "reserved",
            == 0x20 "Select ISO/IEC 10646-1 page 0x20, Miscellaneous",
            == 0x21 "Select ISO/IEC 10646-1 page 0x21, Misc. Symbols, Arrows",
            == 0x22 "Select ISO/IEC 10646-1 page 0x22, Mathematical operators",
            == 0x23 "Select ISO/IEC 10646-1 page 0x23, Misc. Technical",
            == 0x24 "Select ISO/IEC 10646-1 page 0x24, OCR, Enclosed Alpha-num",
            == 0x25 "Select ISO/IEC 10646-1 page 0x25, Form and Chart Components",
            == 0x26 "Select ISO/IEC 10646-1 page 0x26, Miscellaneous Dingbats",
            == 0x27 "Select ISO/IEC 10646-1 page 0x27, Zapf Dingbats",
            0x28 .. 0x2F "reserved",
            == 0x30 "Select ISO/IEC 10646-1 page 0x30, Hiragana, Katakana",
            == 0x31 "Select ISO/IEC 10646-1 page 0x31, Bopomopho, Hangul Elem.",
            == 0x32 "Select ISO/IEC 10646-1 page 0x32, Enclosed CJK Letters, Ideo.",
            == 0x33 "Select ISO/IEC 10646-1 page 0x33, Enclosed CJK Letters, Ideo.",
            0x34 .. 0x3E "reserved",
            == 0x3F "Select 16-bit ISO/IEC 10646-1 mode, all",
            0x40 .. 0xDF "reserved",
            0xE0 .. 0xFE "User Private",
            == 0xFF "Not Applicable";
        uimsbf: 8 number_bytes "Number of Bytes";
        ARRAY: COUNT number_bytes bslbf: 8 "Compression String Byte" Hex;
    }: COUNT number_segments;
}: COUNT number_strings;

uimsbf: 8 dimensions_defined "Dimension Defined";
LOOP {
    uimsbf: 8 dimension_name_length "Dimension Name Length";
    //dimension_name_text()var
    // A/65 Table 6.24: Multiple String Structure on p.42
    uimsbf: 8 dim_number_strings "Number of Strings";
    LOOP {
                uimsbf: 24 "ISO 639 Language Code" ISO639LANG;
                //obsolete
        //ARRAY: COUNT 3 uimsbf: 8 "ISO 639 Language Code" Hex;
        uimsbf: 8 dim_number_segments "Number Segments";
```

```
LOOP {
   uimsbf: 8 "Compression Type" Hex
      == 0x00 "No Compression",
      == 0x01 "Huffman Coding using English-language Program Title Encode/Decode Table",
      == 0x02 "Huffman Coding using English-language Program Description Encode/Decode Table",
      0x03 .. 0xAF "reserved",
      0xB0 .. 0xFF "User Private";
   uimsbf: 8 "Mode" Hex
      == 0x00 "Select ISO/IEC 10646-1 page 0x00, ASCII, ISO Latin-1 (Roman)",
      == 0x01 "Select ISO/IEC 10646-1 page 0x01, European Latin (many)",
      == 0x02 "Select ISO/IEC 10646-1 page 0x02, Standard Phonetic",
      == 0x03 "Select ISO/IEC 10646-1 page 0x03, Greek",
      == 0x04 "Select ISO/IEC 10646-1 page 0x04, Russian, Slavic",
      == 0x05 "Select ISO/IEC 10646-1 page 0x05, Armenian, Hebrew",
      == 0x06 "Select ISO/IEC 10646-1 page 0x06, Arabic",
      0x07 .. 0x08 "reserved",
      == 0x09 "Select ISO/IEC 10646-1 page 0x09, Devanagari, Bengali",
      == 0x0A "Select ISO/IEC 10646-1 page 0x0A, Punjabi, Gujarati",
      == 0x0B "Select ISO/IEC 10646-1 page 0x0B, Oriya, Tamil",
      == 0x0C "Select ISO/IEC 10646-1 page 0x0C, Telugu, Kannada",
      == 0x0D "Select ISO/IEC 10646-1 page 0x0D, Malayalam",
      == 0x0E "Select ISO/IEC 10646-1 page 0x0E, Thai, Lao",
      == 0x0F "reserved",
      == 0x10 "Select ISO/IEC 10646-1 page 0x10, Tibetan, Georgian",
      0x11 .. 0x1F "reserved",
      == 0x20 "Select ISO/IEC 10646-1 page 0x20, Miscellaneous",
      == 0x21 "Select ISO/IEC 10646-1 page 0x21, Misc. Symbols, Arrows",
      == 0x22 "Select ISO/IEC 10646-1 page 0x22, Mathematical operators",
      == 0x23 "Select ISO/IEC 10646-1 page 0x23, Misc. Technical",
      == 0x24 "Select ISO/IEC 10646-1 page 0x24, OCR, Enclosed Alpha-num",
      == 0x25 "Select ISO/IEC 10646-1 page 0x25, Form and Chart Components",
      == 0x26 "Select ISO/IEC 10646-1 page 0x26, Miscellaneous Dingbats",
      == 0x27 "Select ISO/IEC 10646-1 page 0x27, Zapf Dingbats",
      0x28 .. 0x2F "reserved",
      == 0x30 "Select ISO/IEC 10646-1 page 0x30, Hiragana, Katakana",
      == 0x31 "Select ISO/IEC 10646-1 page 0x31, Bopomopho, Hangul Elem.",
      == 0x32 "Select ISO/IEC 10646-1 page 0x32, Enclosed CJK Letters, Ideo.",
      == 0x33 "Select ISO/IEC 10646-1 page 0x33, Enclosed CJK Letters, Ideo.",
      0x34 .. 0x3E "reserved",
      == 0x3F "Select 16-bit ISO/IEC 10646-1 mode, all",
      0x40 .. 0xDF "reserved",
      0xE0 .. 0xFE "User Private",
      == 0xFF "Not Applicable";
   uimsbf: 8 dim_number_bytes "Number of Bytes";
```

```
        ARRAY: COUNT dim_number_bytes bslbf: 8 "Compression String Byte" Hex;
    }: COUNT dim_number_segments;
}: COUNT dim_number_strings;

reserved: 3;
bslbf: 1 "Graduated Scale";
uimsbf: 4 values_defined "Values Defined";
LOOP {
    uimsbf: 8 "Abbrev Rating Value Length";
    //abbrev_rating_value_ text()var
    // A/65 Table 6.24: Multiple String Structure on p.42
    uimsbf: 8 abbrev_number_strings "Number of Strings";
    LOOP {
            uimsbf: 24 "ISO 639 Language Code" ISO639LANG;
            //obsolete
        //ARRAY: COUNT 3 uimsbf: 8 "ISO 639 Language Code" Hex;
        uimsbf: 8 abbrev_number_segments "Number Segments";
        LOOP {
            uimsbf: 8 "Compression Type" Hex
                == 0x00 "No Compression",
                == 0x01 "Huffman Coding using English-language Program Title Encode/Decode Table",
                == 0x02 "Huffman Coding using English-language Program Description Encode/Decode Table",
                0x03 .. 0xAF "reserved",
                0xB0 .. 0xFF "User Private";
            uimsbf: 8 "Mode" Hex
                == 0x00 "Select ISO/IEC 10646-1 page 0x00, ASCII, ISO Latin-1 (Roman)",
                == 0x01 "Select ISO/IEC 10646-1 page 0x01, European Latin (many)",
                == 0x02 "Select ISO/IEC 10646-1 page 0x02, Standard Phonetic",
                == 0x03 "Select ISO/IEC 10646-1 page 0x03, Greek",
                == 0x04 "Select ISO/IEC 10646-1 page 0x04, Russian, Slavic",
                == 0x05 "Select ISO/IEC 10646-1 page 0x05, Armenian, Hebrew",
                == 0x06 "Select ISO/IEC 10646-1 page 0x06, Arabic",
                0x07 .. 0x08 "reserved",
                == 0x09 "Select ISO/IEC 10646-1 page 0x09, Devanagari, Bengali",
                == 0x0A "Select ISO/IEC 10646-1 page 0x0A, Punjabi, Gujarati",
                == 0x0B "Select ISO/IEC 10646-1 page 0x0B, Oriya, Tamil",
                == 0x0C "Select ISO/IEC 10646-1 page 0x0C, Telugu, Kannada",
                == 0x0D "Select ISO/IEC 10646-1 page 0x0D, Malayalam",
                == 0x0E "Select ISO/IEC 10646-1 page 0x0E, Thai, Lao",
                == 0x0F "reserved",
                == 0x10 "Select ISO/IEC 10646-1 page 0x10, Tibetan, Georgian",
                0x11 .. 0x1F "reserved",
                == 0x20 "Select ISO/IEC 10646-1 page 0x20, Miscellaneous",
                == 0x21 "Select ISO/IEC 10646-1 page 0x21, Misc. Symbols, Arrows",
```

```
            == 0x22 "Select ISO/IEC 10646-1 page 0x22, Mathematical operators",
            == 0x23 "Select ISO/IEC 10646-1 page 0x23, Misc. Technical",
            == 0x24 "Select ISO/IEC 10646-1 page 0x24, OCR, Enclosed Alpha-num",
            == 0x25 "Select ISO/IEC 10646-1 page 0x25, Form and Chart Components",
            == 0x26 "Select ISO/IEC 10646-1 page 0x26, Miscellaneous Dingbats",
            == 0x27 "Select ISO/IEC 10646-1 page 0x27, Zapf Dingbats",
            0x28 .. 0x2F "reserved",
            == 0x30 "Select ISO/IEC 10646-1 page 0x30, Hiragana, Katakana",
            == 0x31 "Select ISO/IEC 10646-1 page 0x31, Bopomopho, Hangul Elem.",
            == 0x32 "Select ISO/IEC 10646-1 page 0x32, Enclosed CJK Letters, Ideo.",
            == 0x33 "Select ISO/IEC 10646-1 page 0x33, Enclosed CJK Letters, Ideo.",
            0x34 .. 0x3E "reserved",
            == 0x3F "Select 16-bit ISO/IEC 10646-1 mode, all",
            0x40 .. 0xDF "reserved",
            0xE0 .. 0xFE "User Private",
            == 0xFF "Not Applicable";
         uimsbf: 8 abbrev_number_bytes "Number of Bytes";
         ARRAY: COUNT abbrev_number_bytes bslbf: 8 "Compression String Byte" Hex;
      }: COUNT abbrev_number_segments;
   }: COUNT abbrev_number_strings;

   uimsbf: 8 "Rating Value Length";
   //rating_value_ text()var
   // A/65 Table 6.24: Multiple String Structure on p.42
   uimsbf: 8 rating_number_strings "Number of Strings";
   LOOP {
            uimsbf: 24 "ISO 639 Language Code" ISO639LANG;
            //obsolete
      //ARRAY: COUNT 3 uimsbf: 8 "ISO 639 Language Code" Hex;
      uimsbf: 8 rating_number_segments "Number Segments";
      LOOP {
         uimsbf: 8 "Compression Type" Hex
            == 0x00 "No Compression",
            == 0x01 "Huffman Coding using English-language Program Title Encode/Decode Table",
            == 0x02 "Huffman Coding using English-language Program Description Encode/Decode Table",
            0x03 .. 0xAF "reserved",
            0xB0 .. 0xFF "User Private";
         uimsbf: 8 "Mode" Hex
            == 0x00 "Select ISO/IEC 10646-1 page 0x00, ASCII, ISO Latin-1 (Roman)",
            == 0x01 "Select ISO/IEC 10646-1 page 0x01, European Latin (many)",
            == 0x02 "Select ISO/IEC 10646-1 page 0x02, Standard Phonetic",
            == 0x03 "Select ISO/IEC 10646-1 page 0x03, Greek",
            == 0x04 "Select ISO/IEC 10646-1 page 0x04, Russian, Slavic",
            == 0x05 "Select ISO/IEC 10646-1 page 0x05, Armenian, Hebrew",
```

```
                == 0x06 "Select ISO/IEC 10646-1 page 0x06, Arabic",
                0x07 .. 0x08 "reserved",
                == 0x09 "Select ISO/IEC 10646-1 page 0x09, Devanagari, Bengali",
                == 0x0A "Select ISO/IEC 10646-1 page 0x0A, Punjabi, Gujarati",
                == 0x0B "Select ISO/IEC 10646-1 page 0x0B, Oriya, Tamil",
                == 0x0C "Select ISO/IEC 10646-1 page 0x0C, Telugu, Kannada",
                == 0x0D "Select ISO/IEC 10646-1 page 0x0D, Malayalam",
                == 0x0E "Select ISO/IEC 10646-1 page 0x0E, Thai, Lao",
                == 0x0F "reserved",
                == 0x10 "Select ISO/IEC 10646-1 page 0x10, Tibetan, Georgian",
                0x11 .. 0x1F "reserved",
                == 0x20 "Select ISO/IEC 10646-1 page 0x20, Miscellaneous",
                == 0x21 "Select ISO/IEC 10646-1 page 0x21, Misc. Symbols, Arrows",
                == 0x22 "Select ISO/IEC 10646-1 page 0x22, Mathematical operators",
                == 0x23 "Select ISO/IEC 10646-1 page 0x23, Misc. Technical",
                == 0x24 "Select ISO/IEC 10646-1 page 0x24, OCR, Enclosed Alpha-num",
                == 0x25 "Select ISO/IEC 10646-1 page 0x25, Form and Chart Components",
                == 0x26 "Select ISO/IEC 10646-1 page 0x26, Miscellaneous Dingbats",
                == 0x27 "Select ISO/IEC 10646-1 page 0x27, Zapf Dingbats",
                0x28 .. 0x2F "reserved",
                == 0x30 "Select ISO/IEC 10646-1 page 0x30, Hiragana, Katakana",
                == 0x31 "Select ISO/IEC 10646-1 page 0x31, Bopomopho, Hangul Elem.",
                == 0x32 "Select ISO/IEC 10646-1 page 0x32, Enclosed CJK Letters, Ideo.",
                == 0x33 "Select ISO/IEC 10646-1 page 0x33, Enclosed CJK Letters, Ideo.",
                0x34 .. 0x3E "reserved",
                == 0x3F "Select 16-bit ISO/IEC 10646-1 mode, all",
                0x40 .. 0xDF "reserved",
                0xE0 .. 0xFE "User Private",
                == 0xFF "Not Applicable";
            uimsbf: 8 rating_number_bytes "Number of Bytes";
            ARRAY: COUNT rating_number_bytes bslbf: 8 "Compression String Byte" Hex;
        }: COUNT rating_number_segments;
    }: COUNT rating_number_strings;
  }: COUNT values_defined;
}: COUNT dimensions_defined;

                reserved: 6;
                uimsbf: 10 descriptors_length "Descriptors Length";
DESCRIPTORSLOOP: BYTE descriptors_length;
}
```

```
// ==================
// Caption Service Descriptor
// ==================
DESCRIPTOR CaptionServiceDescriptor {
NAME:
   "Caption Service Descriptor";
BODY:
   //descriptor_tag8                0x86
   //descriptor_length8             uimsbf
   reserved: 3;
   uimsbf: 5 number_of_services "Number of Services";
   LOOP {
      ARRAY: COUNT 8 uimsbf: 3 "Language" Hex;
      bslbf: 1 cc_type "CC Type" Hex;
      reserved: 1;
      if (cc_type== 0)    // cc_type = 0 if line21
      {
         reserved: 5;
         bslbf: 1 "Line21 Field" Hex;
      }
      else
      {
         uimsbf: 6 "Caption Service Number" Hex;
      }
      bslbf: 1 "Easy Reader";
      bslbf: 1 "Wide Aspect Ratio";
      reserved: 14;
   }: COUNT number_of_services;
}


// ==================
// Content Advisory Descriptor
// ==================
DESCRIPTOR ContentAdvisoryDescriptor {
NAME:
   "Content Advisory Descriptor";
BODY:
   //descriptor_tag8                0x87
   //descriptor_length8             uimsbf
   reserved: 2;
   uimsbf: 6 rating_region_count "Rating Region Count";
   LOOP {
      uimsbf: 8 "Rating Region" Hex;
```

```
uimsbf: 8 rated_dimensions "Rated Dimensions";
LOOP {
   uimsbf: 8 "Rating Dimension";
   reserved: 4;
   uimsbf: 4 "Rating Value";
}: COUNT rated_dimensions;

uimsbf: 8 "Rating Description Length";
//rating_description_text()var
// A/65 Table 6.24: Multiple String Structure on p.42
uimsbf: 8 dim_number_strings "Number of Strings";
LOOP {
            uimsbf: 24 "ISO 639 Language Code" ISO639LANG;
            //obsolete
   //ARRAY: COUNT 3 uimsbf: 8 "ISO 639 Language Code" Hex;
   uimsbf: 8 dim_number_segments "Number Segments";
   LOOP {
      uimsbf: 8 "Compression Type" Hex
         == 0x00 "No Compression",
         == 0x01 "Huffman Coding using English-language Program Title Encode/Decode Table",
         == 0x02 "Huffman Coding using English-language Program Description Encode/Decode Table",
         0x03 .. 0xAF "reserved",
         0xB0 .. 0xFF "User Private";
      uimsbf: 8 "Mode" Hex
         == 0x00 "Select ISO/IEC 10646-1 page 0x00, ASCII, ISO Latin-1 (Roman)",
         == 0x01 "Select ISO/IEC 10646-1 page 0x01, European Latin (many)",
         == 0x02 "Select ISO/IEC 10646-1 page 0x02, Standard Phonetic",
         == 0x03 "Select ISO/IEC 10646-1 page 0x03, Greek",
         == 0x04 "Select ISO/IEC 10646-1 page 0x04, Russian, Slavic",
         == 0x05 "Select ISO/IEC 10646-1 page 0x05, Armenian, Hebrew",
         == 0x06 "Select ISO/IEC 10646-1 page 0x06, Arabic",
         0x07 .. 0x08 "reserved",
         == 0x09 "Select ISO/IEC 10646-1 page 0x09, Devanagari, Bengali",
         == 0x0A "Select ISO/IEC 10646-1 page 0x0A, Punjabi, Gujarati",
         == 0x0B "Select ISO/IEC 10646-1 page 0x0B, Oriya, Tamil",
         == 0x0C "Select ISO/IEC 10646-1 page 0x0C, Telugu, Kannada",
         == 0x0D "Select ISO/IEC 10646-1 page 0x0D, Malayalam",
         == 0x0E "Select ISO/IEC 10646-1 page 0x0E, Thai, Lao",
         == 0x0F "reserved",
         == 0x10 "Select ISO/IEC 10646-1 page 0x10, Tibetan, Georgian",
         0x11 .. 0x1F "reserved",
         == 0x20 "Select ISO/IEC 10646-1 page 0x20, Miscellaneous",
         == 0x21 "Select ISO/IEC 10646-1 page 0x21, Misc. Symbols, Arrows",
         == 0x22 "Select ISO/IEC 10646-1 page 0x22, Mathematical operators",
```

```
                == 0x23 "Select ISO/IEC 10646-1 page 0x23, Misc. Technical",
                == 0x24 "Select ISO/IEC 10646-1 page 0x24, OCR, Enclosed Alpha-num",
                == 0x25 "Select ISO/IEC 10646-1 page 0x25, Form and Chart Components",
                == 0x26 "Select ISO/IEC 10646-1 page 0x26, Miscellaneous Dingbats",
                == 0x27 "Select ISO/IEC 10646-1 page 0x27, Zapf Dingbats",
                0x28 .. 0x2F "reserved",
                == 0x30 "Select ISO/IEC 10646-1 page 0x30, Hiragana, Katakana",
                == 0x31 "Select ISO/IEC 10646-1 page 0x31, Bopomopho, Hangul Elem.",
                == 0x32 "Select ISO/IEC 10646-1 page 0x32, Enclosed CJK Letters, Ideo.",
                == 0x33 "Select ISO/IEC 10646-1 page 0x33, Enclosed CJK Letters, Ideo.",
                0x34 .. 0x3E "reserved",
                == 0x3F "Select 16-bit ISO/IEC 10646-1 mode, all",
                0x40 .. 0xDF "reserved",
                0xE0 .. 0xFE "User Private",
                == 0xFF "Not Applicable";
            uimsbf: 8 dim_number_bytes "Number of Bytes";
            ARRAY: COUNT dim_number_bytes bslbf: 8 "Compression String Byte" Hex;
        }: COUNT dim_number_segments;
    }: COUNT dim_number_strings;

  }: COUNT rating_region_count;
}

// ==================
// Service Location Descriptor
// ==================
DESCRIPTOR ServiceLocationDescriptor {
NAME:
  "Service Location Descriptor";
BODY:
                //descriptor_tag      8           0xA1
                //descriptor_length   8           uimsbf
                reserved: 3;
                uimsbf: 13 "PCR PID";
                uimsbf: 8 number_elements "Number Elements";
  LOOP {
                uimsbf: 8 "Stream Type" Hex;
                reserved: 3;
                uimsbf: 13 "Elementary PID" Hex;
                uimsbf: 24 "ISO 639 Language Code" ISO639LANG;
                //obsolete
    //ARRAY: COUNT 3 uimsbf: 8 "ISO 639 Language Code" Hex;
                }: COUNT number_elements;
}
```

```
// ==================
// Time Shifted Service Descriptor
// ==================
DESCRIPTOR TimeShiftedServiceDescriptor {
NAME:
   "Time Shifted Service Descriptor";
BODY:
                  //descriptor_tag      8            0xA2
                  //descriptor_length   8            uimsbf
                  reserved: 3;
                  uimsbf: 5 number_of_services "Number of Services";
   LOOP {
                        reserved: 6;
                        uimsbf: 10 "Time Shift";
                        reserved: 4;
                        uimsbf: 10 "Major Channel Number" Hex;
                        uimsbf: 10 "Minor Channel Number" Hex;
                  }: COUNT number_of_services;
}


// ==================
// Component Name Descriptor
// ==================
DESCRIPTOR ComponentNameDescriptor {
NAME:
   "Component Name Descriptor";
BODY:
                  //descriptor_tag      8            0xA3
                  //descriptor_length   8            uimsbf

                  //component_name_string()var
   // A/65 Table 6.24: Multiple String Structure on p.42
   uimsbf: 8 dim_number_strings "Number of Strings";
   LOOP {
                  uimsbf: 24 "ISO 639 Language Code" ISO639LANG;
                  //obsolete
      //ARRAY: COUNT 3 uimsbf: 8 "ISO 639 Language Code" Hex;
      uimsbf: 8 dim_number_segments "Number Segments";
      LOOP {
         uimsbf: 8 "Compression Type" Hex
            == 0x00 "No Compression",
            == 0x01 "Huffman Coding using English-language Program Title Encode/Decode Table",
            == 0x02 "Huffman Coding using English-language Program Description Encode/Decode Table",
```

```
            0x03 .. 0xAF "reserved",
            0xB0 .. 0xFF "User Private";
         uimsbf: 8 "Mode" Hex
            == 0x00 "Select ISO/IEC 10646-1 page 0x00, ASCII, ISO Latin-1 (Roman)",
            == 0x01 "Select ISO/IEC 10646-1 page 0x01, European Latin (many)",
            == 0x02 "Select ISO/IEC 10646-1 page 0x02, Standard Phonetic",
            == 0x03 "Select ISO/IEC 10646-1 page 0x03, Greek",
            == 0x04 "Select ISO/IEC 10646-1 page 0x04, Russian, Slavic",
            == 0x05 "Select ISO/IEC 10646-1 page 0x05, Armenian, Hebrew",
            == 0x06 "Select ISO/IEC 10646-1 page 0x06, Arabic",
            0x07 .. 0x08 "reserved",
            == 0x09 "Select ISO/IEC 10646-1 page 0x09, Devanagari, Bengali",
            == 0x0A "Select ISO/IEC 10646-1 page 0x0A, Punjabi, Gujarati",
            == 0x0B "Select ISO/IEC 10646-1 page 0x0B, Oriya, Tamil",
            == 0x0C "Select ISO/IEC 10646-1 page 0x0C, Telugu, Kannada",
            == 0x0D "Select ISO/IEC 10646-1 page 0x0D, Malayalam",
            == 0x0E "Select ISO/IEC 10646-1 page 0x0E, Thai, Lao",
            == 0x0F "reserved",
            == 0x10 "Select ISO/IEC 10646-1 page 0x10, Tibetan, Georgian",
            0x11 .. 0x1F "reserved",
            == 0x20 "Select ISO/IEC 10646-1 page 0x20, Miscellaneous",
            == 0x21 "Select ISO/IEC 10646-1 page 0x21, Misc. Symbols, Arrows",
            == 0x22 "Select ISO/IEC 10646-1 page 0x22, Mathematical operators",
            == 0x23 "Select ISO/IEC 10646-1 page 0x23, Misc. Technical",
            == 0x24 "Select ISO/IEC 10646-1 page 0x24, OCR, Enclosed Alpha-num",
            == 0x25 "Select ISO/IEC 10646-1 page 0x25, Form and Chart Components",
            == 0x26 "Select ISO/IEC 10646-1 page 0x26, Miscellaneous Dingbats",
            == 0x27 "Select ISO/IEC 10646-1 page 0x27, Zapf Dingbats",
            0x28 .. 0x2F "reserved",
            == 0x30 "Select ISO/IEC 10646-1 page 0x30, Hiragana, Katakana",
            == 0x31 "Select ISO/IEC 10646-1 page 0x31, Bopomopho, Hangul Elem.",
            == 0x32 "Select ISO/IEC 10646-1 page 0x32, Enclosed CJK Letters, Ideo.",
            == 0x33 "Select ISO/IEC 10646-1 page 0x33, Enclosed CJK Letters, Ideo.",
            0x34 .. 0x3E "reserved",
            == 0x3F "Select 16-bit ISO/IEC 10646-1 mode, all",
            0x40 .. 0xDF "reserved",
            0xE0 .. 0xFE "User Private",
            == 0xFF "Not Applicable";
         uimsbf: 8 dim_number_bytes "Number of Bytes";
         ARRAY: COUNT dim_number_bytes bslbf: 8 "Compression String Byte" Hex;
      }: COUNT dim_number_segments;
   }: COUNT dim_number_strings;
}
// EOF
```

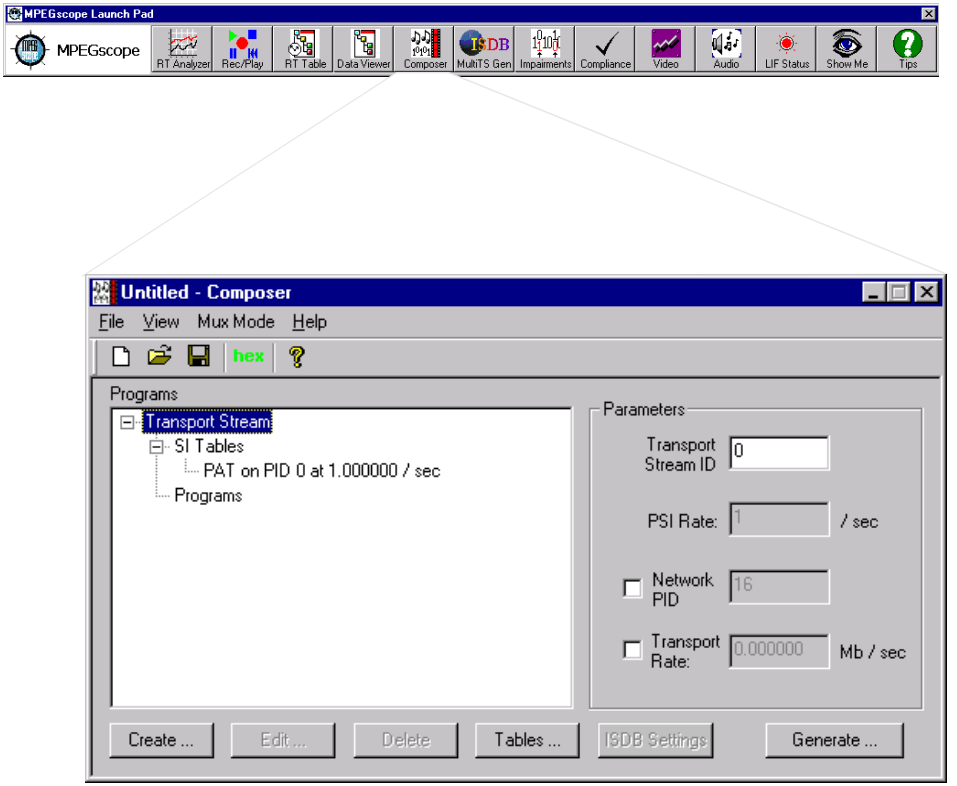
# 7

# Using the Composer

# Composing a transport stream

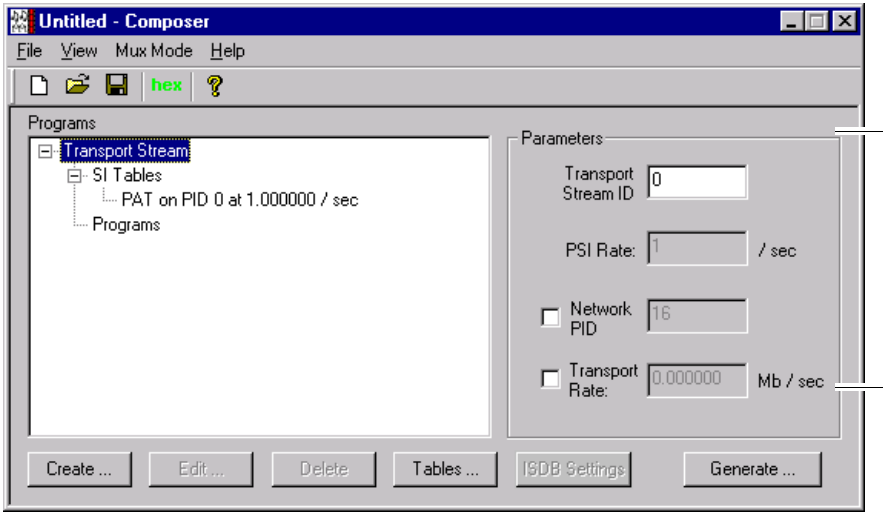

The Composer enables you to define and multiplex MPEG and ISDB single transport streams. This example illustrates how to create programs and tables, then multiplex them into a transport stream.

## Step 1: Start the Composer




The ISDB and DSM-CC features require the purchase of E6314A and E6315A, respectively. If you have not purchased these licenses, you will not be able to create ISDB single transport streams and DSM-CC data carousels from the Composer, and some user interface items will not be visible.

## Step 2: Enter transport stream parameters




If desired, change the default transport stream parameters.

### Parameters

**Transport Stream ID** — The **transport_stream_id** is a field in the Program Allocation Table (PAT) which identifies the transport stream and distinguishes it from other multiplexes in the network. Default is 0; valid range is from 0 to 65535 (0x0000 to 0xFFFF).

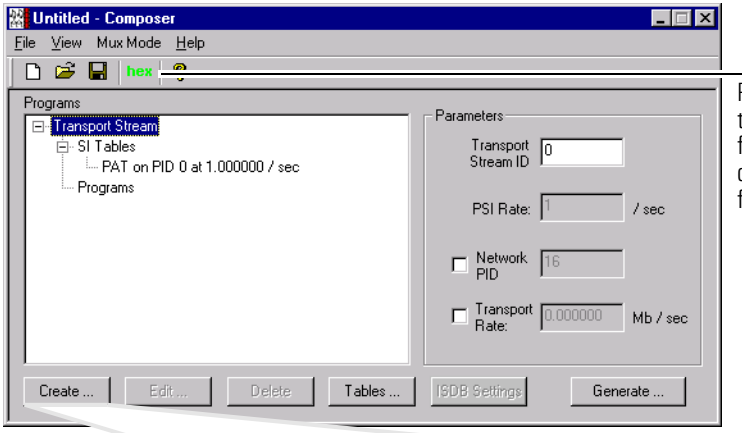
**PSI Rate** — The label for this parameter changes according to the table section selected. The PSI rate is the number of times per second that SI (Service Information) table sections will appear in the transport stream. Default is once per second; valid range is from 0 to 20 times per second.

**Network PID** — The **network_PID** is a field in the PAT which specifies the PID for the optional Network Information Table (NIT). If you are adding a NIT to the transport stream, enable this field and enter the PID value. When you add the NIT, ensure that its PID is the same value you entered here. Default is 16, the correct value for DVB NITs; valid range is from 16 to 8190 (0x0010 to 0x1FFE).

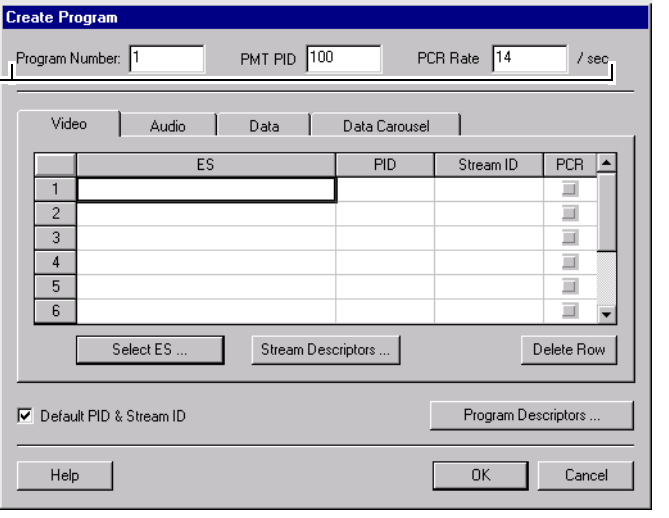
**Transport Rate** — The rate at which to play (transmit) the transport stream file from the Recorder/Player. You can set a specific transport rate or leave the field blank and allow the Composer to calculate the minimum required rate (as determined by the component elementary streams). To set a specific rate, enable this field and enter a value greater than the minimum required rate and less than 90 Mb/s.

**Note**: If you enter a transport rate and later disable the field, the Composer will gray out the value and use the default calculated minimum required rate.

# Step 3: Create a program




Press the **hex** button on the toolbar to toggle field values from decimal to hexadecimal format.



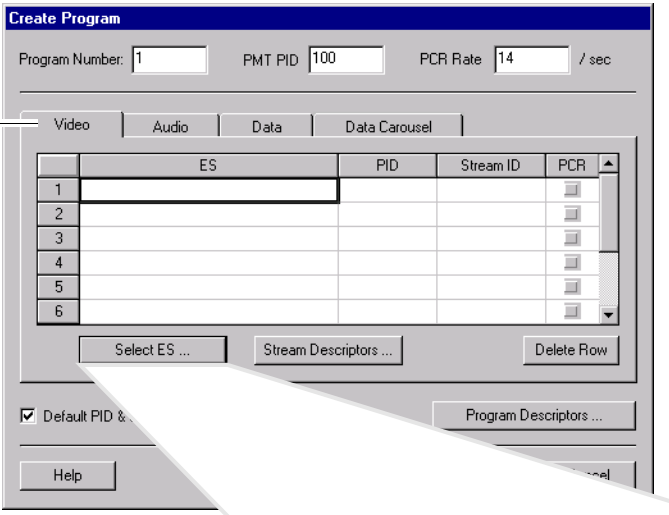

Accept the default program number, PMT PID and PCR rate, or enter different values.

The valid range for the PMT PID is from 16 to 8190 (0x0010 to 0x1FFE).

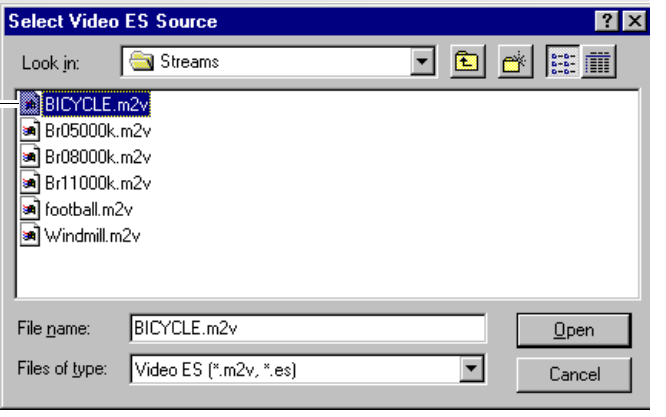
The PCR rate is the number of times per second the PCR will appear in the transport stream. The valid range is from 5 to 30 times per second.

# Step 4: Define the video elementary stream

① Select the **Video** tab.



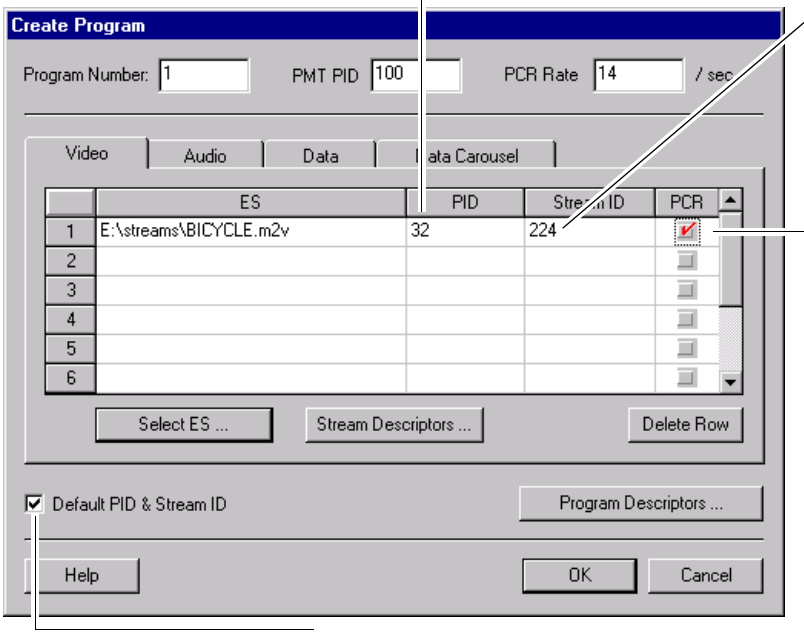

② Double click on the video elementary stream you want to include in the program.




You can add multiple video streams to the program. To add a stream, select a new row marker then include another video stream.

You can also create a program without video elementary streams as long as the transport stream contains at least one video stream.

If you have disabled **Default PID & Stream ID**...




4 ...enter a PID value from 16 to 8190 (0x10 to 0x1FFE). This value cannot be used by any other video, audio, private data or data carousel stream, or by the PMT PID.

5 ...enter a Stream ID value from 224 to 239 (0xE0 to 0xEF).

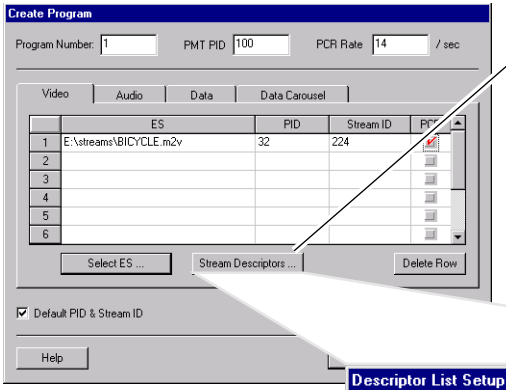
6 **If you want PCR information placed in this stream, check the PCR check box.**

You can insert PCR information on one video PID in each program. If you do not check the PCR box for any video stream, the Composer will automatically assign it to the first video stream in the program. If the program does not contain a video stream, the Composer will assign it to the first audio stream in the program. (A program must contain at least one video or audio elementary stream.)

3 Disable this option if you want to manually assign PID and Stream ID values.

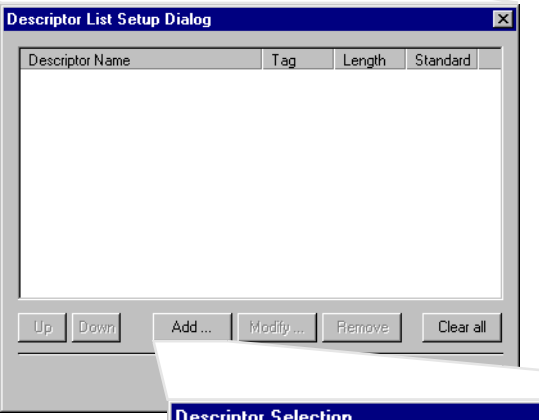
As of the A.05 MPEGscope release, PCR PID information is local to each program. This differs from previous releases where the Composer allowed PCR information on only one stream in the entire transport stream. Previously, if you did not insert PCR information from the user interface, the Composer assigned it during multiplexing to the video or audio stream with the highest bit rate.

If you load a pre-A.05 **.cfg** file without PCR information assigned, the Composer will automatically assign it to the first video stream *in each program*. If a program does not contain a video stream, it will be assigned to the first audio stream.



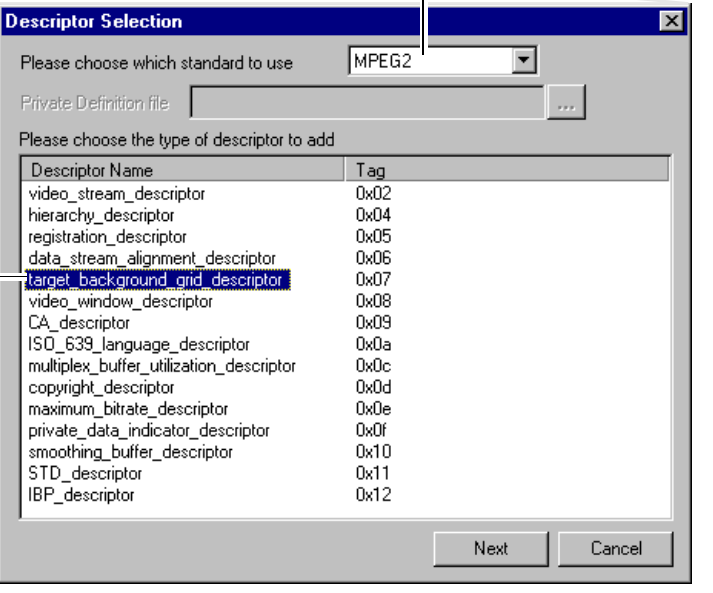

7 **If desired, define descriptors for the video stream.**

The **Descriptor List Setup** dialog shows the list of descriptors that have been defined for this stream. You can add, modify, delete, and reorder descriptor entries.




8 **Select the standard to use from the pulldown list.**

If you select **Private Tables**, you must also specify a **Private Definition** (**.pdv**) file. For information about creating **Private Definition** files, refer to "Analyzing a private table", page 6–38.

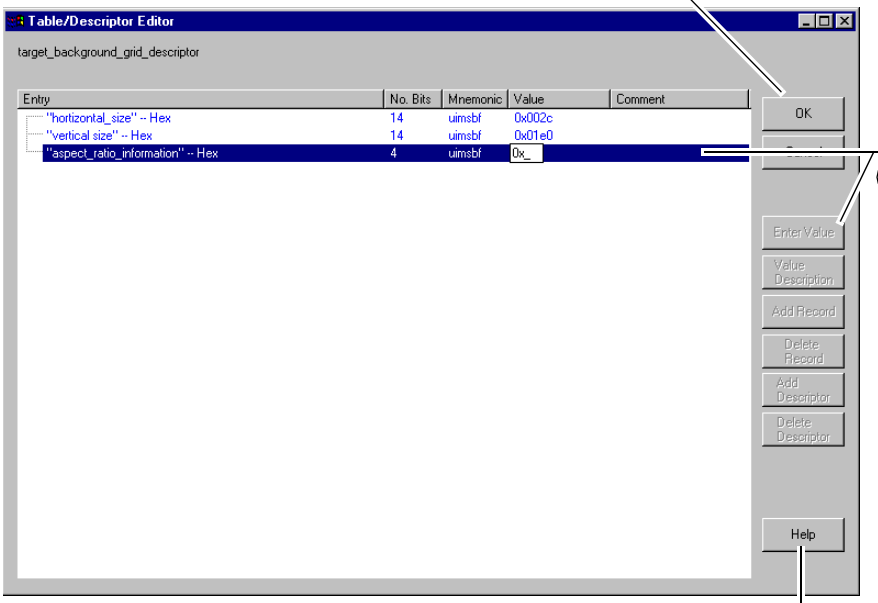



9 **Double click on the video descriptor you want to define.**

Only descriptors that are applicable to video streams will be displayed.

The **Table/Descriptor Editor** dialog displays the fields, length, and field types of the descriptor you have selected. Some fields also display an optional description.

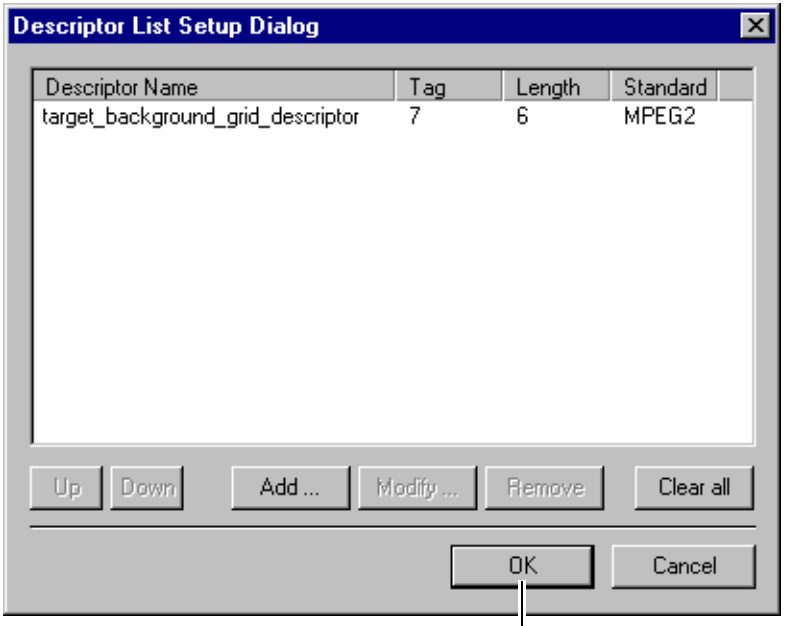
⑪ To add more descriptors, repeat the procedure on page 7–7. Press **OK** when finished.




⑩ **Click on the field and enter a new value.**

You can also select the **Enter Value** button, then enter a new value.

Press the **Help** button for instructions on how to use this dialog.

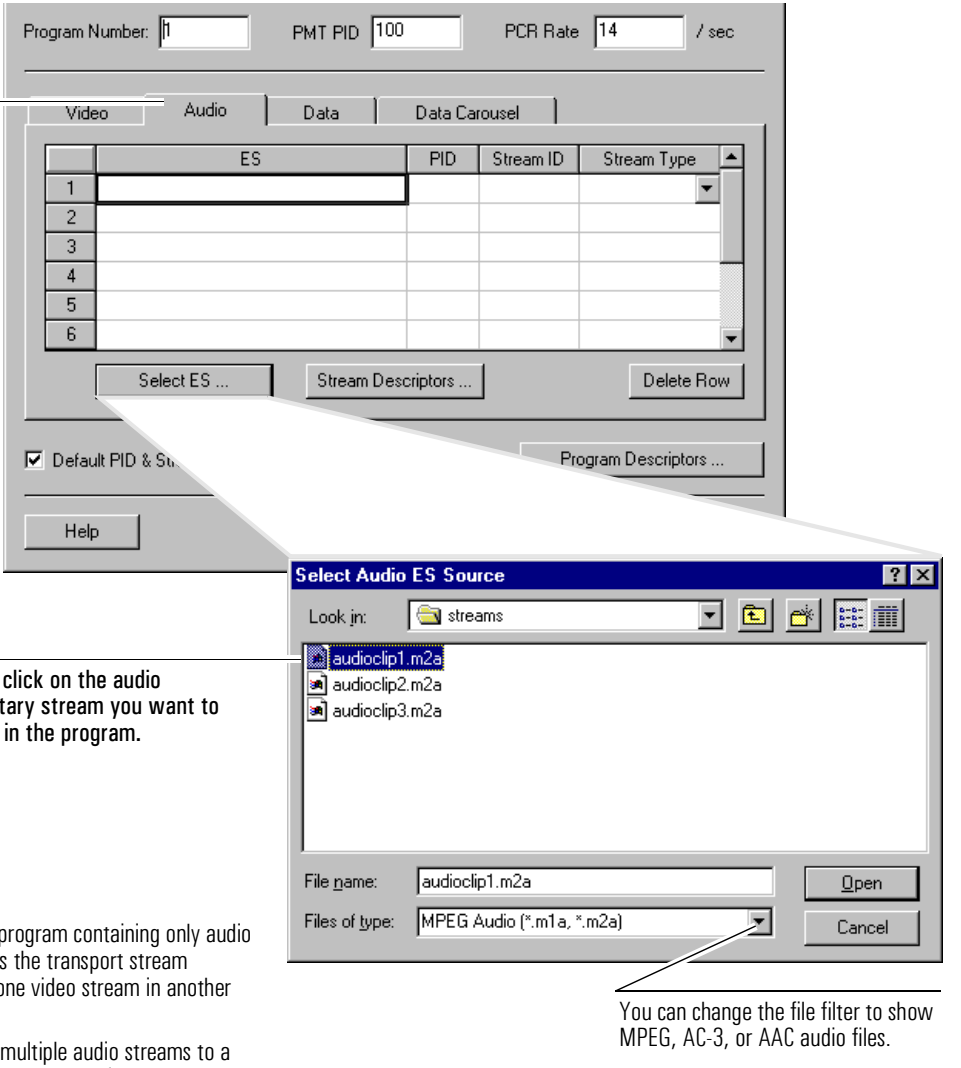



⑫ When you are finished adding descriptors, press **OK**.

If you have included more than one descriptor, you can reorder them by selecting a descriptor, then using the **Up** or **Down** button. Descriptors will be added to the stream in the order you list them in this dialog.

## Step 5: Define the audio elementary stream(s)

The Composer supports MPEG-1, MPEG-2, Dolby AC-3, and AAC (Audio Data Transport Stream (ADTS) file format) audio streams.




① Select the **Audio** tab.

② Double click on the audio elementary stream you want to include in the program.

You can change the file filter to show MPEG, AC-3, or AAC audio files.

You can create a program containing only audio streams as long as the transport stream contains at least one video stream in another program.

You can also add multiple audio streams to a program. To add a stream, select a new row marker then include another audio stream file.

Before you can multiplex an AC-3 audio file with the Composer, it must be aligned on an AC-3 syncword (0x0B77).
If you have demultiplexed an AC-3 audio stream from a transport stream file, it is likely that the AC-3 audio file will not be aligned on the AC-3 syncword You can align the stream by processing the file through a synchronization utility called **AC3_sync.exe** to drop all bytes occurring before the first syncword, as follows:

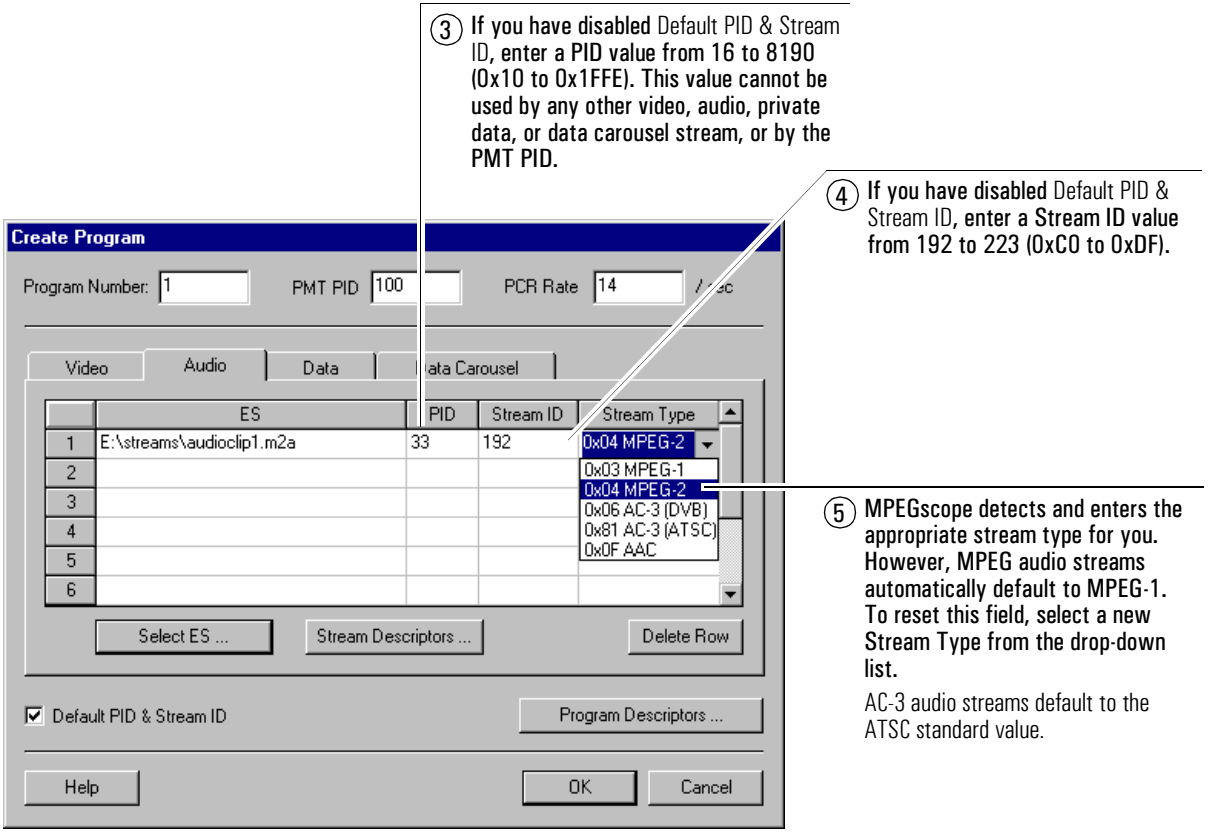
1 From the **Windows NT® Start/Programs** menu, select **MS-DOS Command Prompt.**

2 Change to the **HP-Apps\Resources\bin** directory.

3 Run the utility from the MS-DOS command line, using this syntax:

**ac3_sync -i *inputfilename* -o *outputfilename***

If the input and output files are not in the **C:\HP-Apps\Resources\bin** directory, you must specify the full path names.

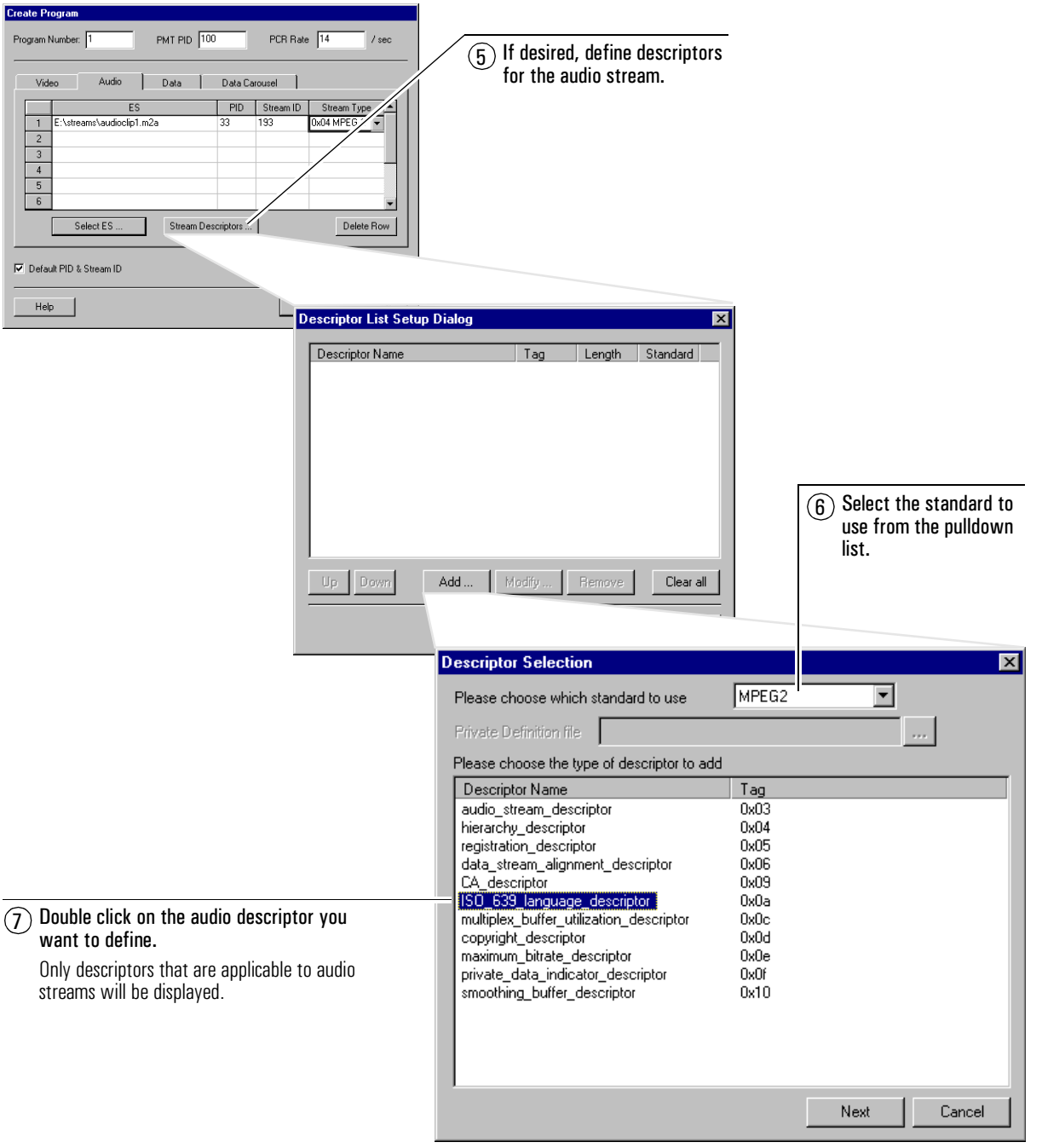



③ **If you have disabled** Default PID & Stream ID**, enter a PID value from 16 to 8190 (0x10 to 0x1FFE). This value cannot be used by any other video, audio, private data, or data carousel stream, or by the PMT PID.**

④ **If you have disabled** Default PID & Stream ID**, enter a Stream ID value from 192 to 223 (0xC0 to 0xDF).**

⑤ **MPEGscope detects and enters the appropriate stream type for you. However, MPEG audio streams automatically default to MPEG-1. To reset this field, select a new Stream Type from the drop-down list.**

AC-3 audio streams default to the ATSC standard value.



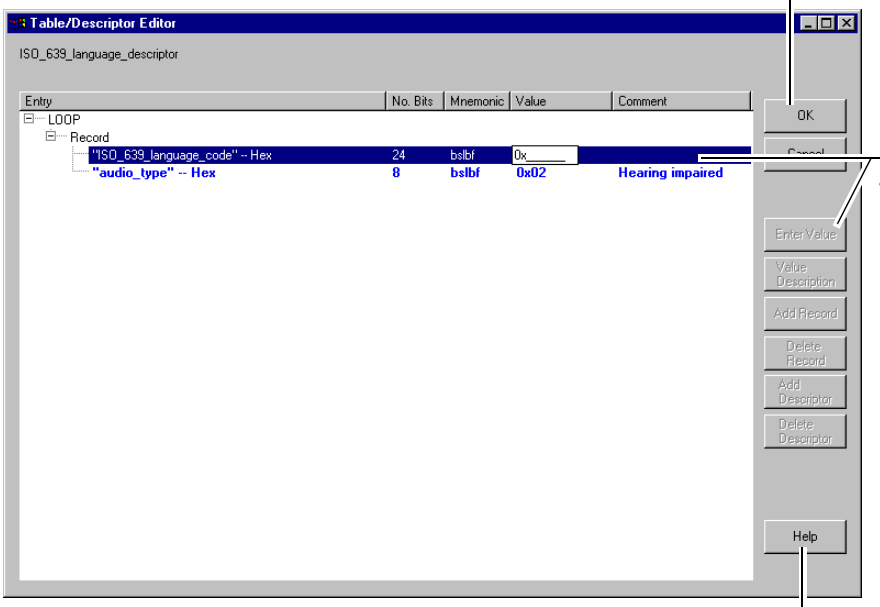

5 If desired, define descriptors for the audio stream.

6 Select the standard to use from the pulldown list.

7 Double click on the audio descriptor you want to define.

Only descriptors that are applicable to audio streams will be displayed.

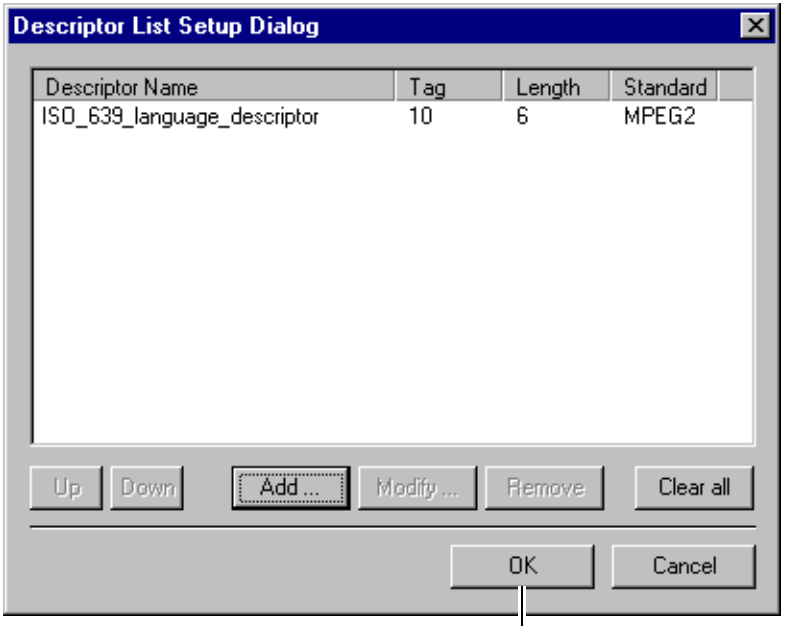
9 To add more descriptors, repeat the procedure on page 7–12. Press **OK** when finished.




8 **Click on the field and enter a new value.**

You can also select the **Enter Value** button, then enter a new value.

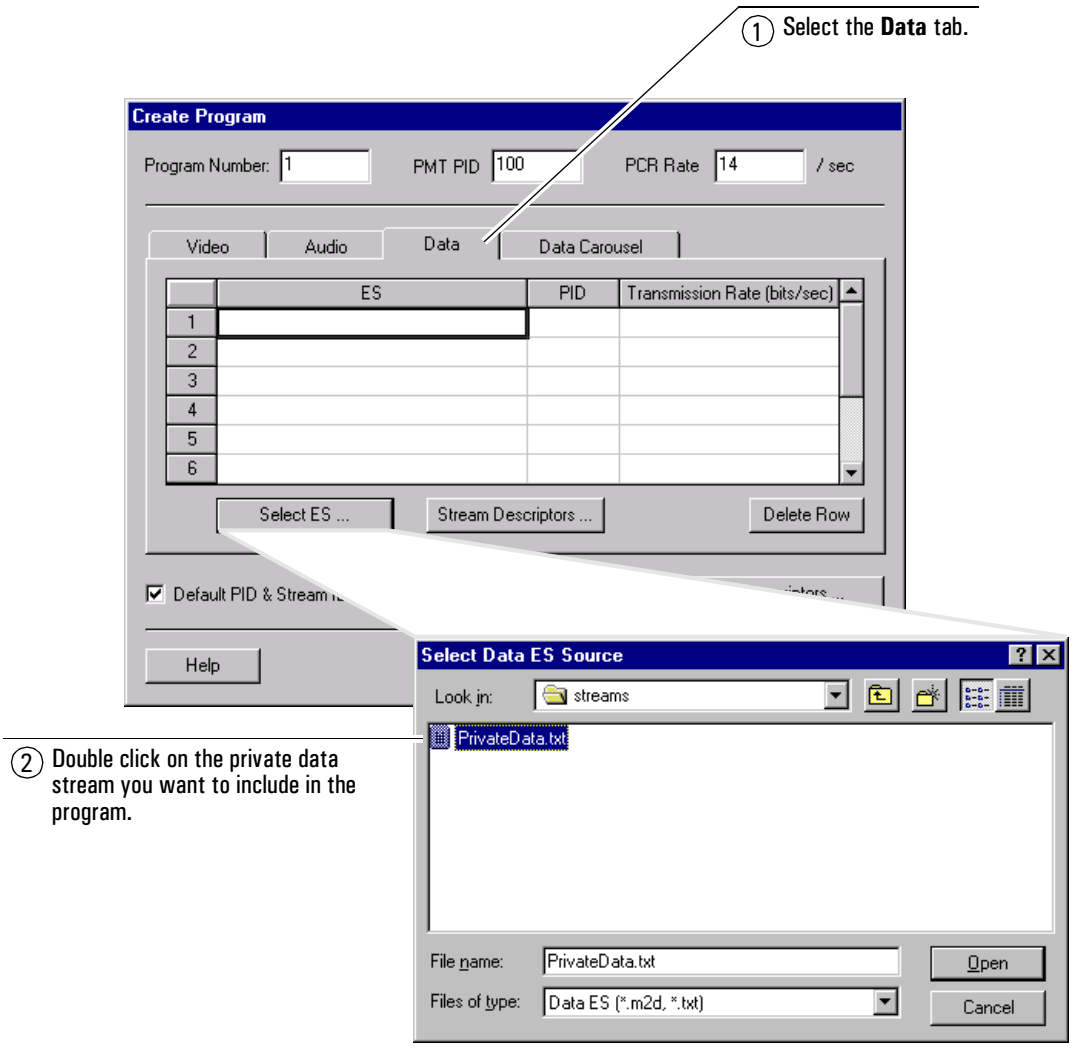
Press the **Help** button for instructions on how to use this dialog.




⑩ When you are finished adding descriptors, press **OK**.

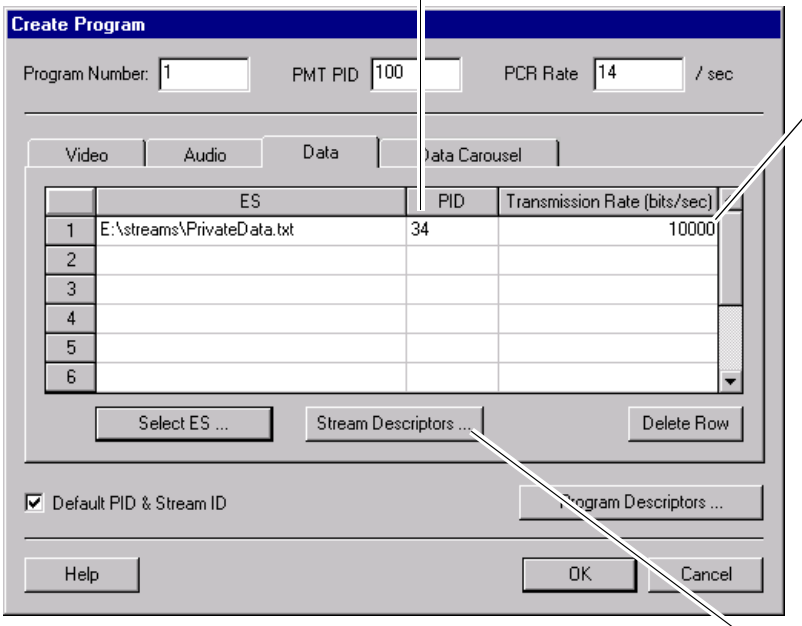
If you have included more than one descriptor, you can reorder them by selecting a descriptor, then using the **Up** or **Down** button. Descriptors will be added to the stream in the order you list them in this dialog.

## Step 6: Define the private data stream(s)




You can add up to 10 private data streams to the entire transport stream. To add a stream, select a new row marker then include another private data stream file.

If you have disabled **Default PID & Stream ID**...




3 ...enter a PID value from 16 to 8190 (0x10 to 0x1FFE). This value cannot be used by any other video, audio, private data or data carousel stream, or by the PMT PID.

4 Enter the bit rate to transfer the private data stream. The valid range is 300 to 5,000,000 bits per second.

5 To add a private data stream descriptor, follow the procedure on pages 7–7 to 7–9 or pages 7–12 to 7–14.

# Step 7: Define a data carousel

This step requires the purchase of the DSM-CC license (E6315A). In the current release, the Composer supports only one-layer data carousels.

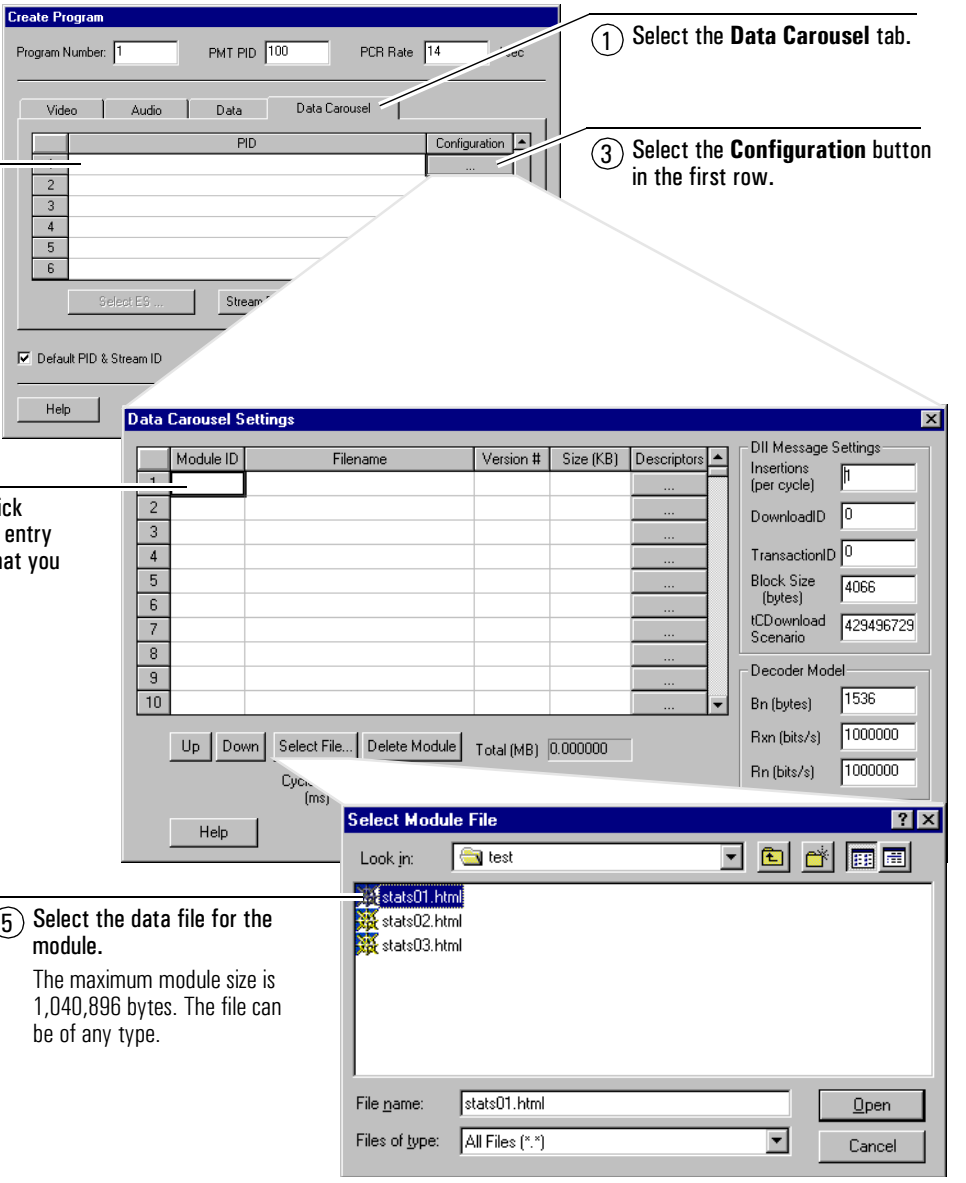
If you have disabled **Default PID & Stream ID**...

(2) ...enter a PID value from 16 to 8190 (0x10 to 0x1FFE). This value cannot be used by any other video, audio, private data or data carousel stream, or by the PMT PID.

(4) Put the focus (single click with the mouse) on the entry in the list of modules that you want to define.

(5) **Select the data file for the module.**

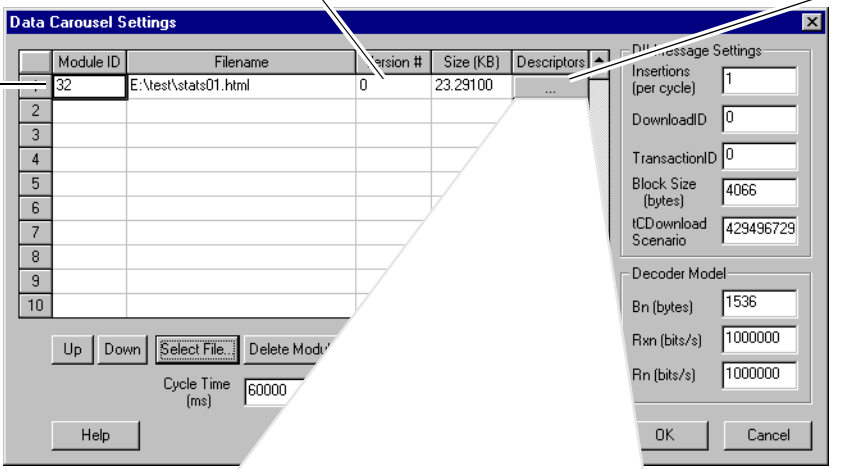
The maximum module size is 1,040,896 bytes. The file can be of any type.

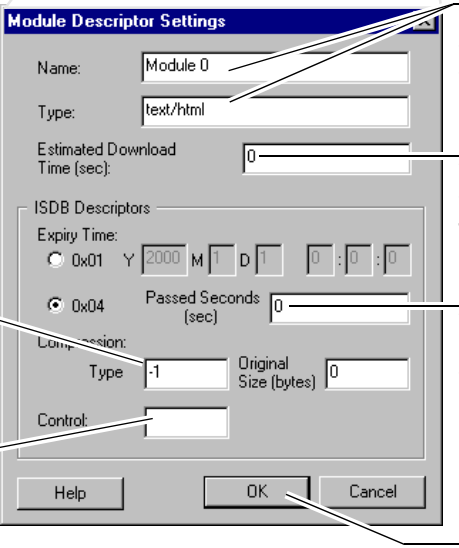


⑦ Enter a **Version #** or accept the default.

⑧ **If desired, add descriptors for the module.**
The Composer automatically includes a **module_link_descriptor** based on the order in which you list the modules in the **Data Carousel Settings** dialog.

⑥ A **Module ID** value is automatically assigned if the **Default PID & Stream ID** is enabled. If this feature is not enabled, enter a **Module ID** value that is unique among module IDs within the data carousel.




Leave the **Name** and **Type** fields blank if you do not want to include these descriptors.

The default value of **0** means the **estimated_download_time_descriptor** is not included.

The default value of **-1** means the **compression_type_descriptor** is not included.

The default value of **0x04 Passed Seconds = 0** means the **expire_descriptor** is not included.

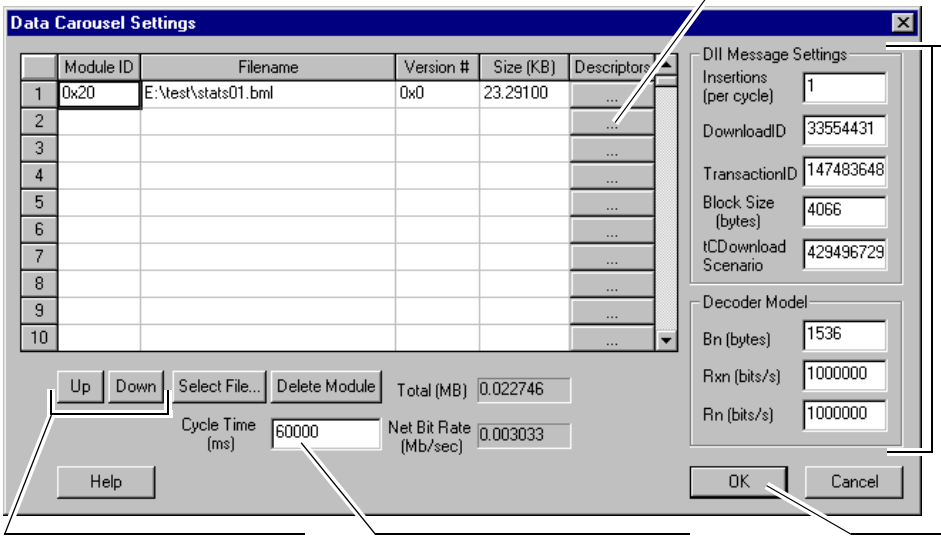
Leave this field blank if you do not want to include a **control_descriptor.**

⑨ **When you are finished defining descriptors, close the dialog.**

Press the **Help** button for more information about each field in this dialog.

⑩ To add more modules, move the focus to the next available row and select the **Descriptors** button. Follow the procedure outlined on pages 7–17 to 7–18.

You can add up to 256 modules to a data carousel.




⑬ Enter values for the DII (DataInfoIndication) Message Settings, and Decoder Model.

⑪ When you have finished adding modules, use the **Up** and **Down** buttons to order the modules in the data carousel. The Composer multiplexes the modules in the order you specify.

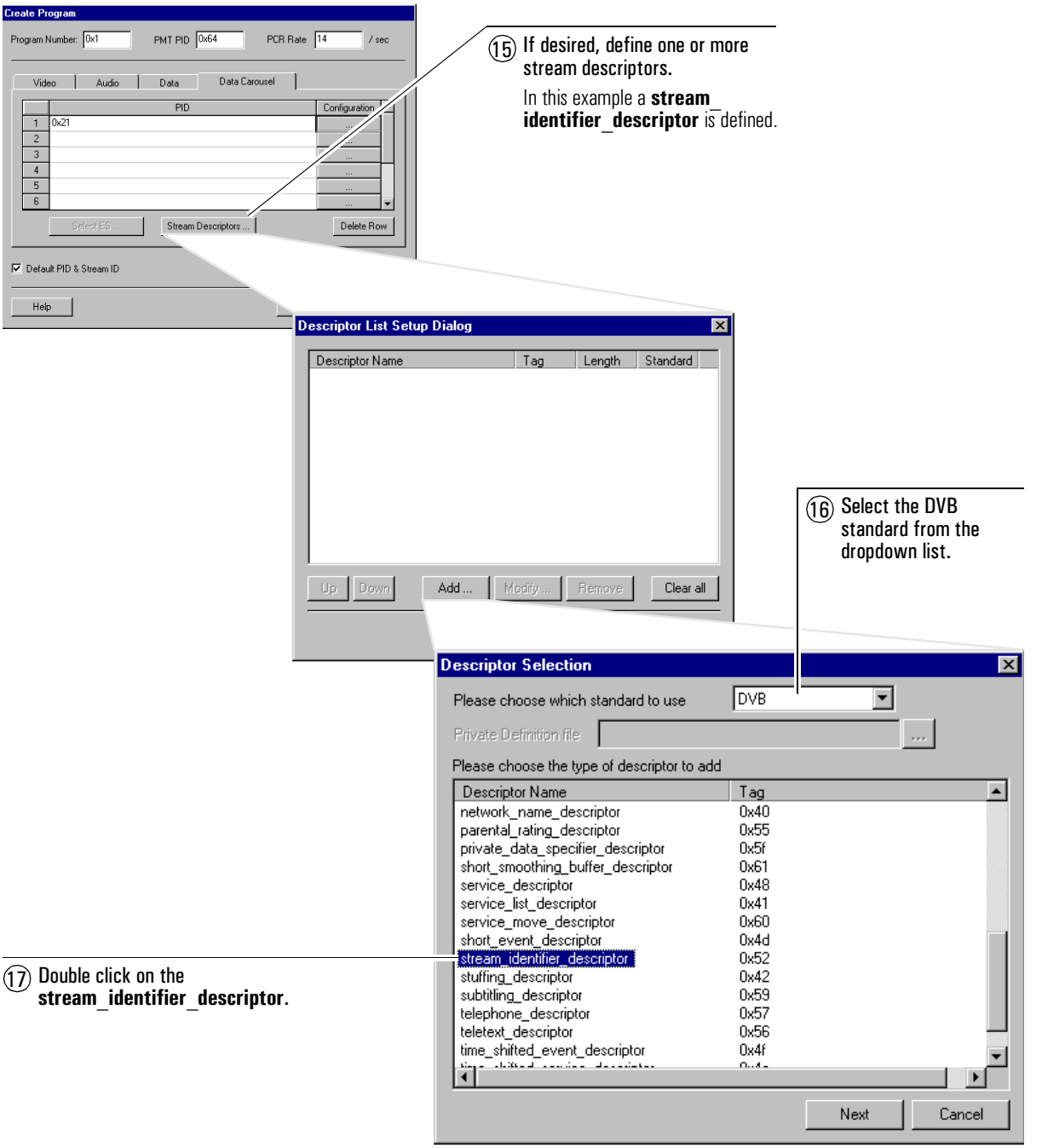
⑫ Enter a **Cycle Time** value.

The **Cycle Time** is the period in which the entire data carousel will be transmitted.

⑭ When you have finished defining the data carousel, close the dialog.

The **Net Bit Rate** value is the bit rate of the *data* portion of the carousel, and does not take headers or other overhead into consideration. It is calculated from the **Cycle Time** and the total data carousel size. **Rx** and **Rxn** are the data transfer rates of a Transport System Target Decoder (T-STD) model. To avoid buffer overflow, ensure that the rates exceed the **Net Bit Rate** by at least 15%.

Press the **Help** button for more information about each field in this dialog.

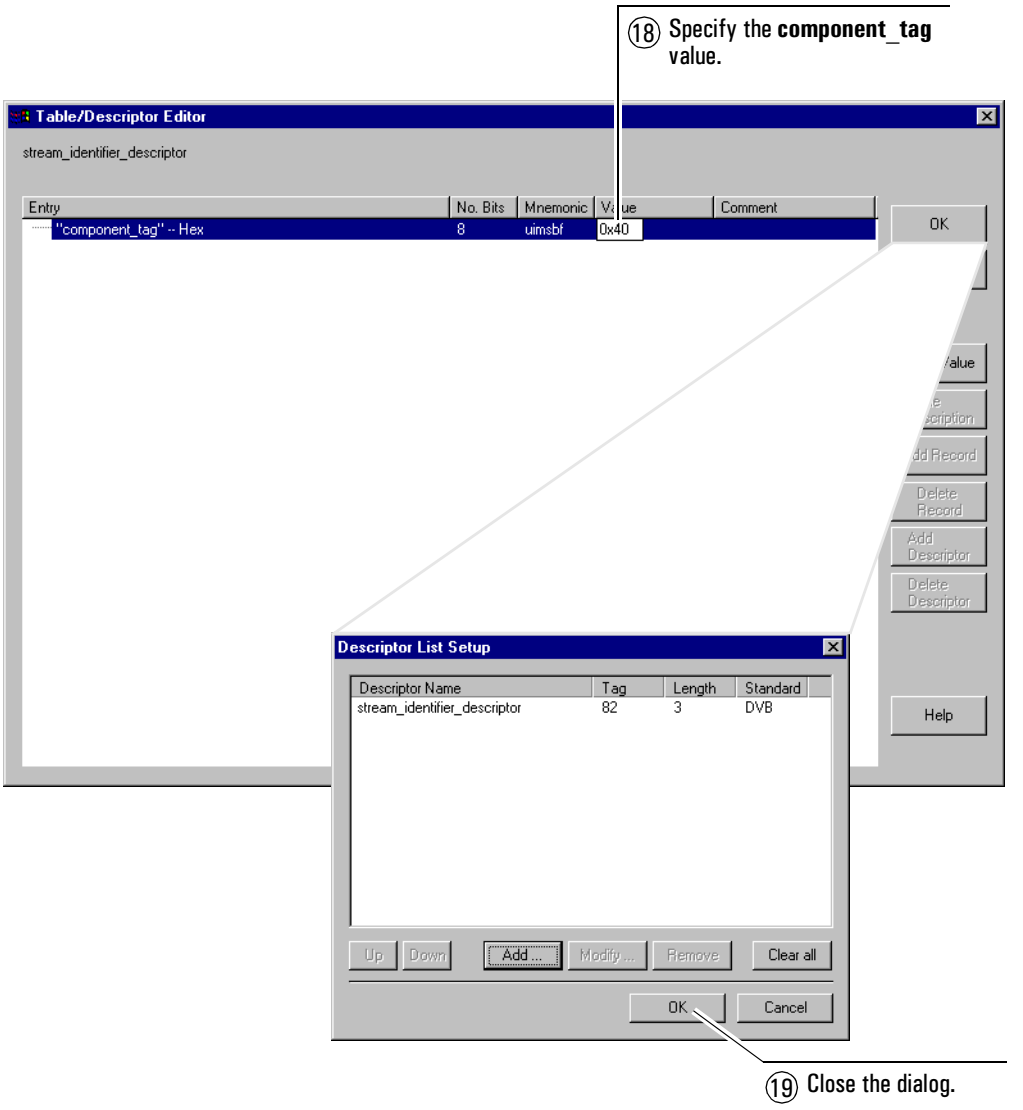



⑮ If desired, define one or more stream descriptors.

In this example a **stream_identifier_descriptor** is defined.

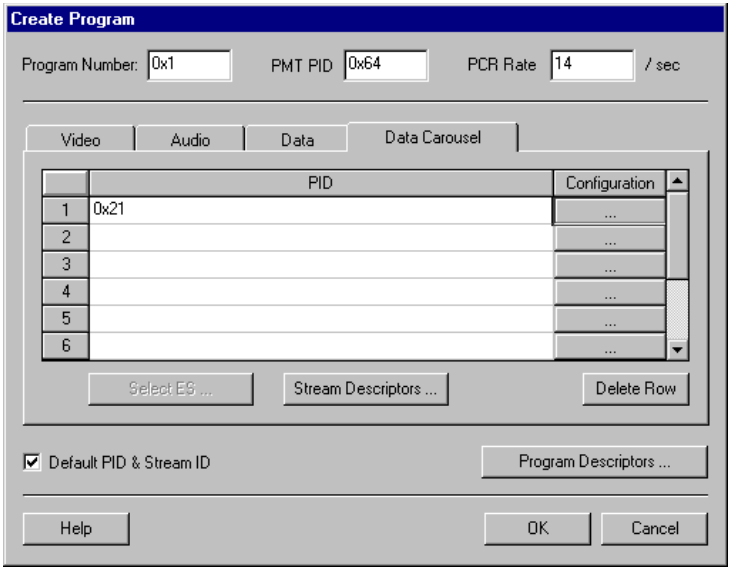
⑯ Select the DVB standard from the dropdown list.

⑰ Double click on the **stream_identifier_descriptor**.


⑱ Specify the **component_tag** value.
Table/Descriptor Editor
stream_identifier_descriptor
Entry
No. Bits
Mnemonic
Value
Comment
"component_tag" -- Hex
8
uimsbf
0x40
OK
Value
Delete Record
Add Descriptor
Delete Descriptor
Help
Descriptor List Setup
Descriptor Name
Tag
Length
Standard
stream_identifier_descriptor
82
3
DVB
Up
Down
Add ...
Modify ...
Remove
Clear all
OK
Cancel
⑲ Close the dialog.

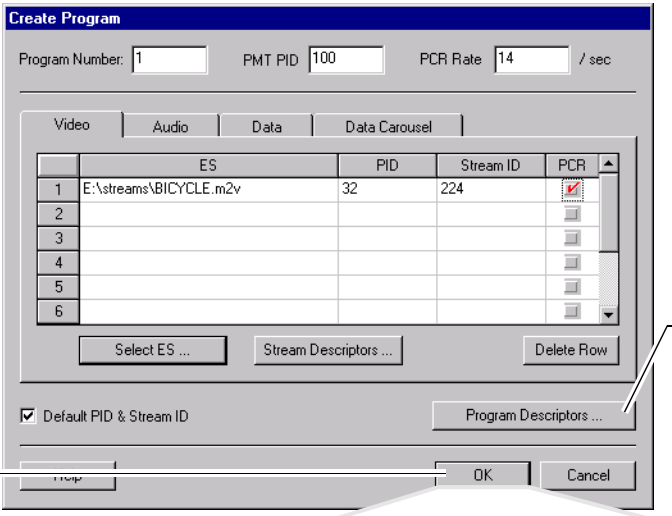
⑳ **If desired, add other data carousels to the program, following the procedure outlined in pages 7–17 to 7–21.**

You can add up to 10 data carousels to each program.



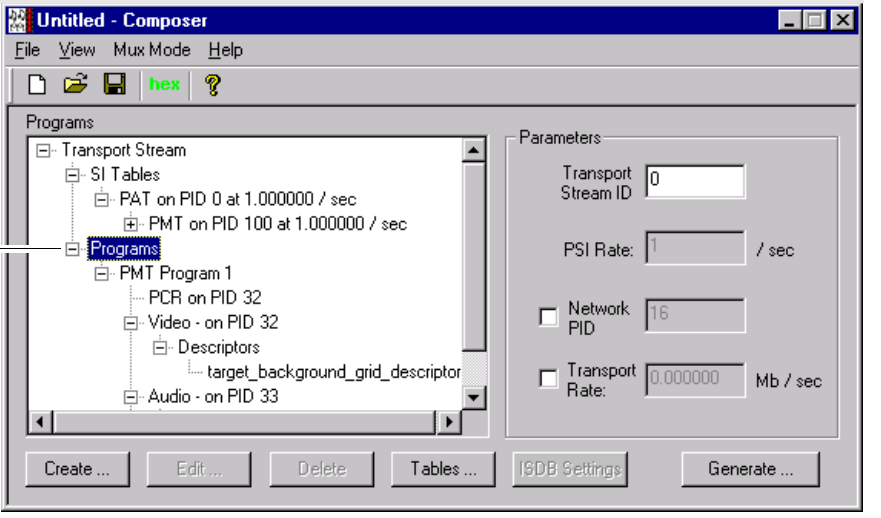

To add an SDT (Service Description Table), EIT (Event Information Table), and data broadcast descriptors to the transport stream, as specified in EN 301 192 (DVB specification for data broadcasting), follow the general procedure outlined in "Define tables", page 7–25.

# Step 8: Save program settings



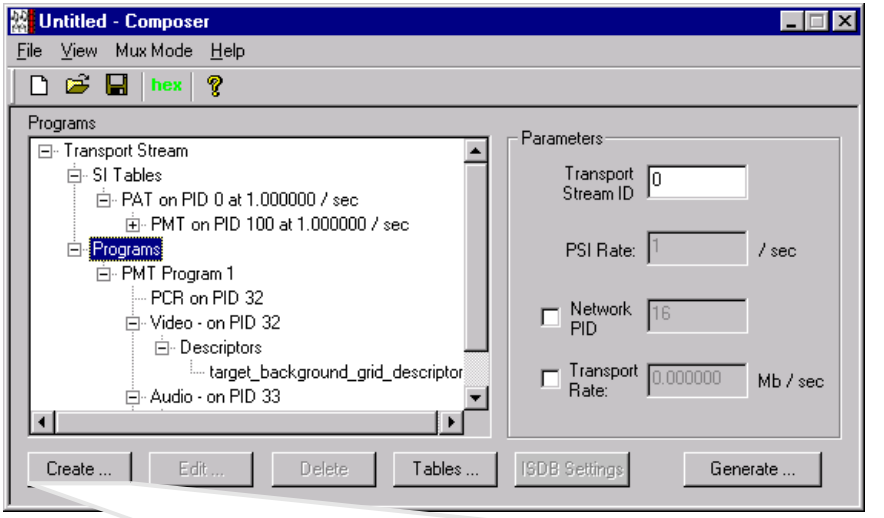

① To add a program descriptor, follow the general procedure illustrated on pages 7–7 to 7–9.

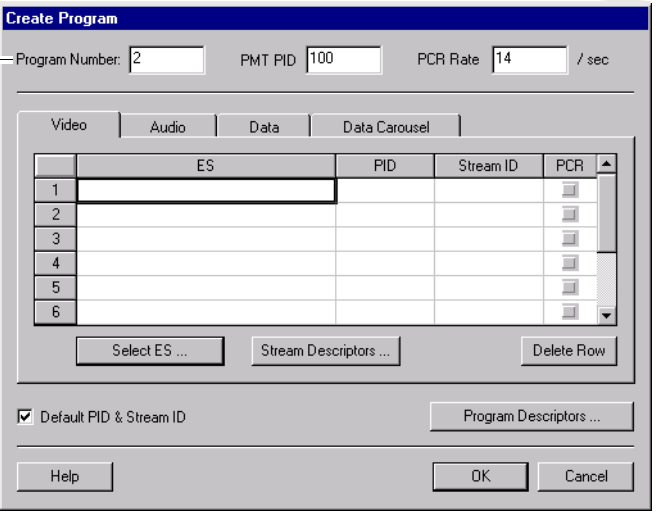
② Select **OK** to save the program settings.




③ Expand the tree display to see program information.

# Step 9: Create additional programs



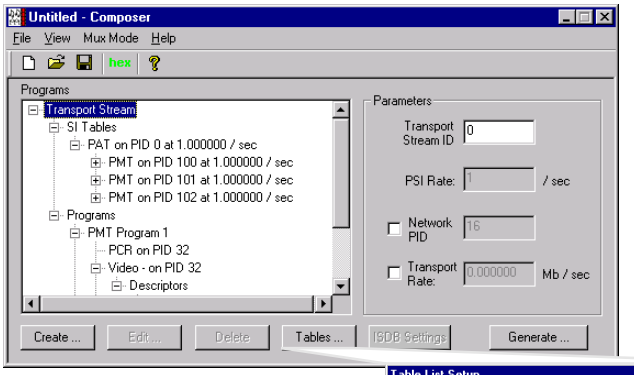




To create another program, follow the steps on pages 7–4 to 7–23.

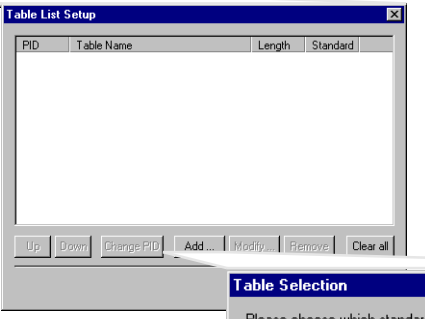
You can insert up to 51 programs into a transport stream. However, the maximum number of elementary streams in a transport stream is also 51. The maximum number of private data streams is 10.

# Step 10: Define tables

The Composer automatically generates a PAT (Program Allocation Table) and a PMT (Program Map Table) for the program(s) you define. You can also define and multiplex DVB, ATSC, or private tables into the transport stream.

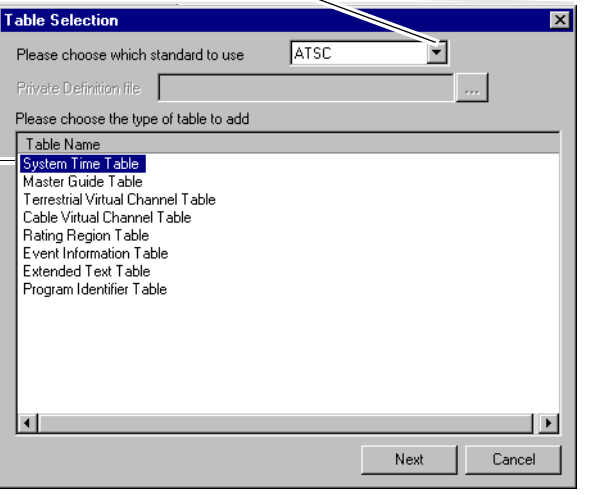



The **Table List Setup** dialog shows the list of tables that have been defined for this stream. You can add, modify, delete, change the table PID, and reorder table entries from this dialog.



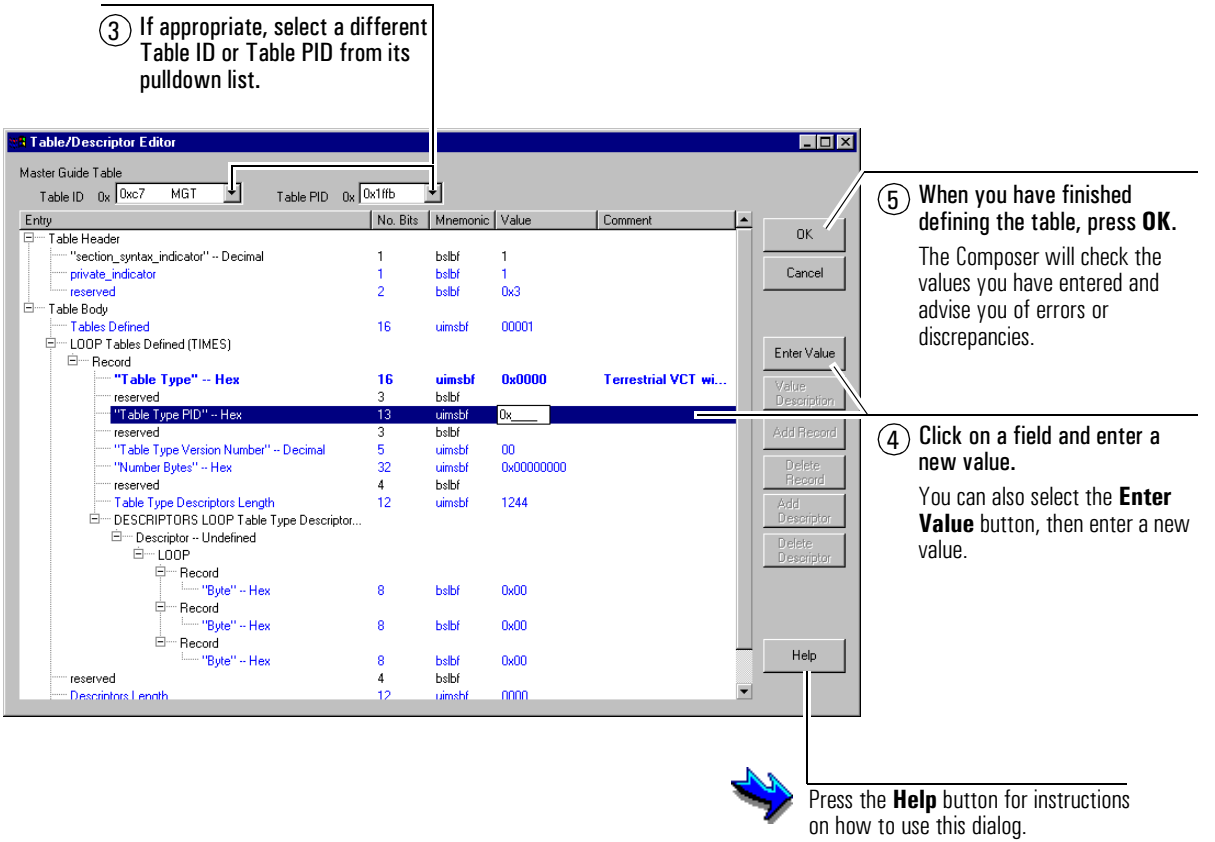

① **Select the standard to use from the pulldown list.**

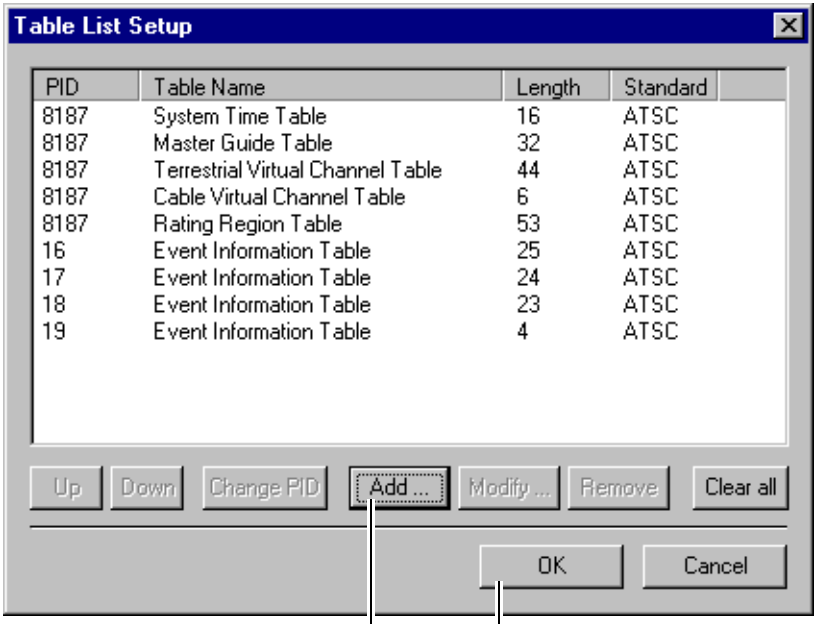
If you want to create a private table, select **Private Tables** from the pulldown list, then enter the name of the compiled private table definition (**.pdv**) file to use. For information on creating a **.pdv** file, refer to "Analyzing a private table", page 6–38.




② **Double click on the table you want to define.**

The **Table/Descriptor Editor** displays the fields, length, and field types of the table or descriptor you have selected. Some fields also display an optional description. Fields for DVB and ATSC tables have default values provided, so you can quickly generate a table or descriptor.



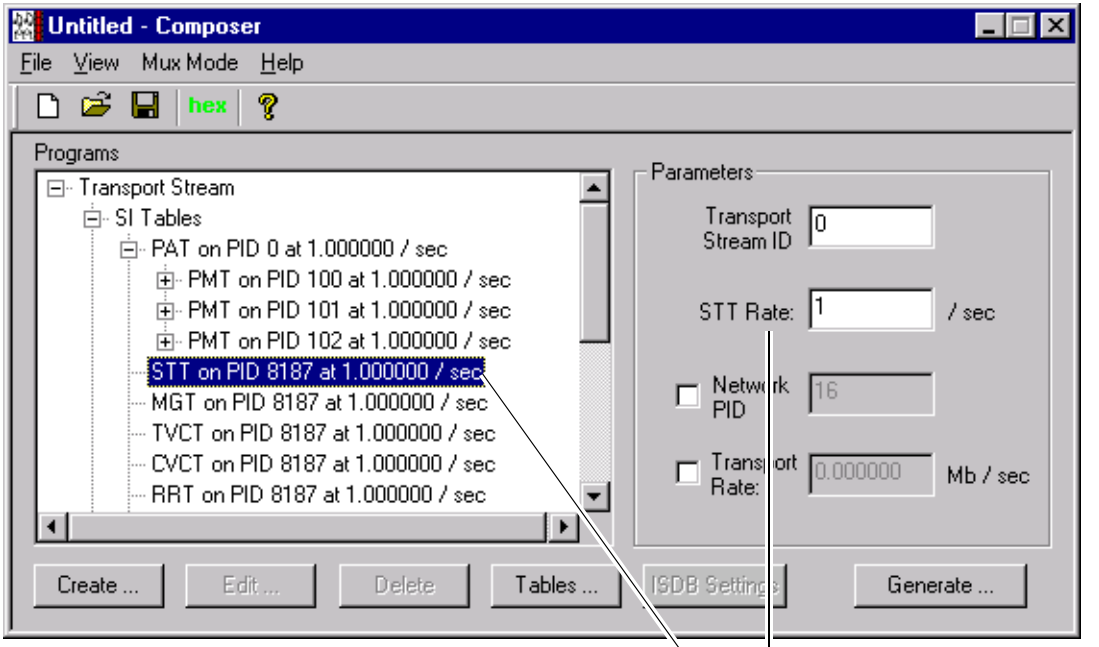



(6) Continue defining the tables you want to include in the transport stream.

(7) When you are finished adding tables, press **OK**.

If you have included more than one table, you can reorder them by selecting a table, then using the **Up** or **Down** button. Tables will be multiplexed into the stream according to their specified insertion rate and the order they were defined. PAT and PMT tables always have the highest priority.

You can also change the table PID from this dialog by selecting the PID, pressing **Change PID**, and entering a new value.



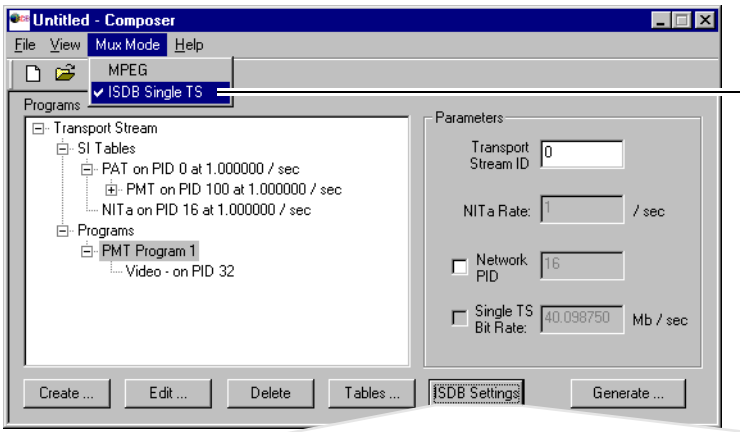

7 **If desired, select a table in the tree display and change the default table insertion rate.**

The label for the table rate changes according to the type of table selected.

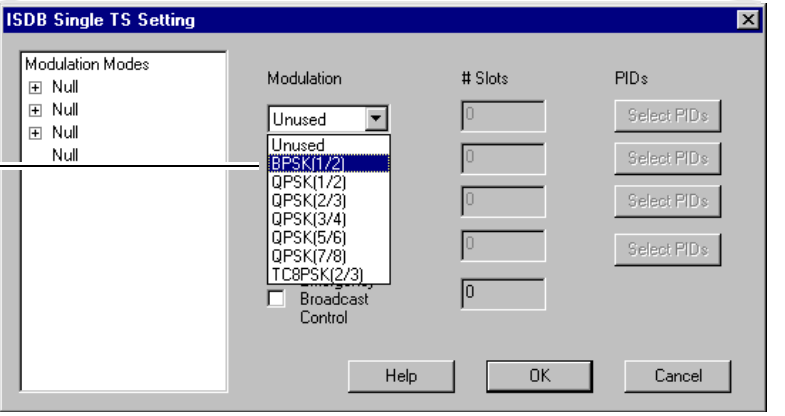
# Step 11: Define an ISDB single transport stream

This step requires the purchase of the ISDB software license (E6314A). You can define an ISDB single transport stream from either the Composer or the ISDB MultiTS Generator. When you use the Composer, as illustrated in this step, you can specify elementary streams and tables along with the TMCC (Transmission Multiplexing Configuration Control) trailer. When you use the ISDB MultiTS Generator, you start with a predefined MPEG transport stream then define the TMCC trailer only. However, the ISDB MultiTS Generator also allows you to define more than one TMCC section and assign different modulation modes and other TMCC information to each. Refer to "Using the ISDB MultiTS Generator", page 8–1, for more information.

① Define the programs (elementary streams, tables, descriptors, and program settings) for the transport stream, as illustrated in the previous steps.



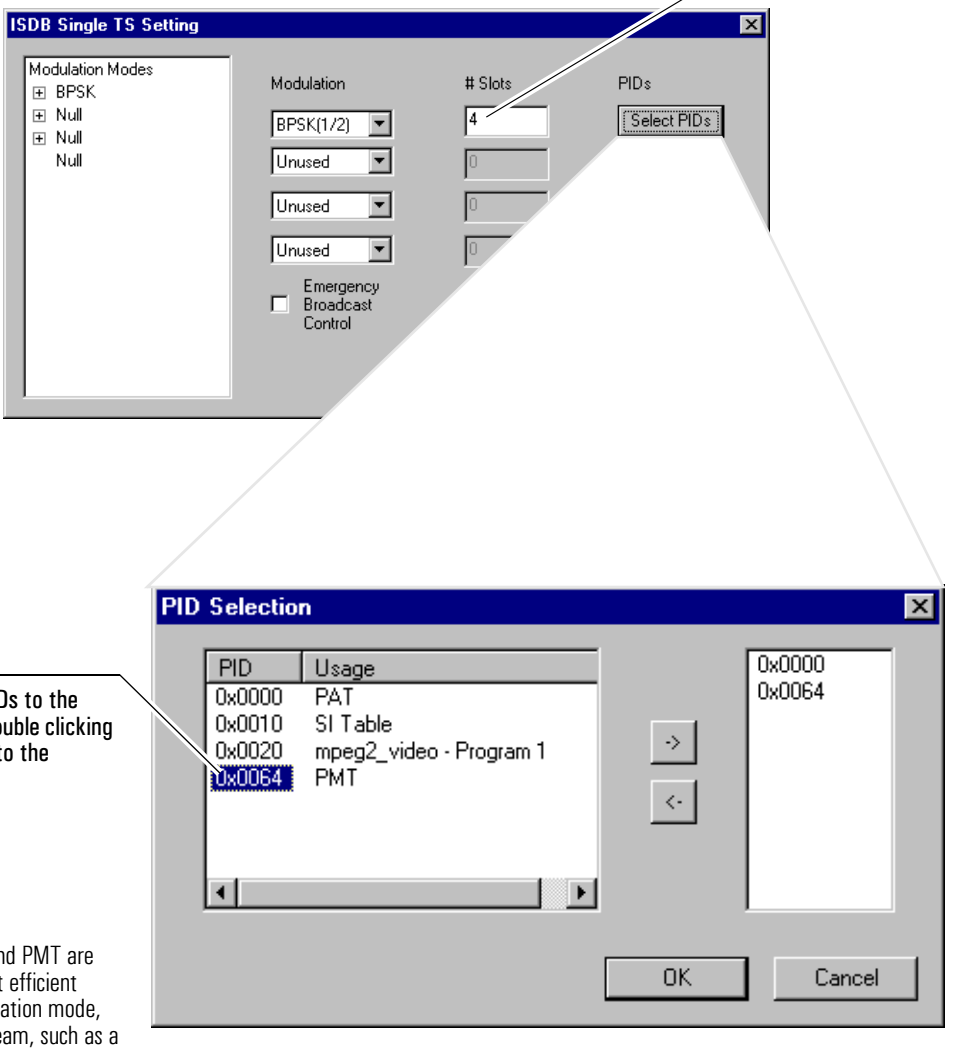

② Select the **ISDB Single TS** multiplex mode to enable the **ISDB Settings** button.




③ **Select the first modulation mode to use in the ISDB single transport stream.**

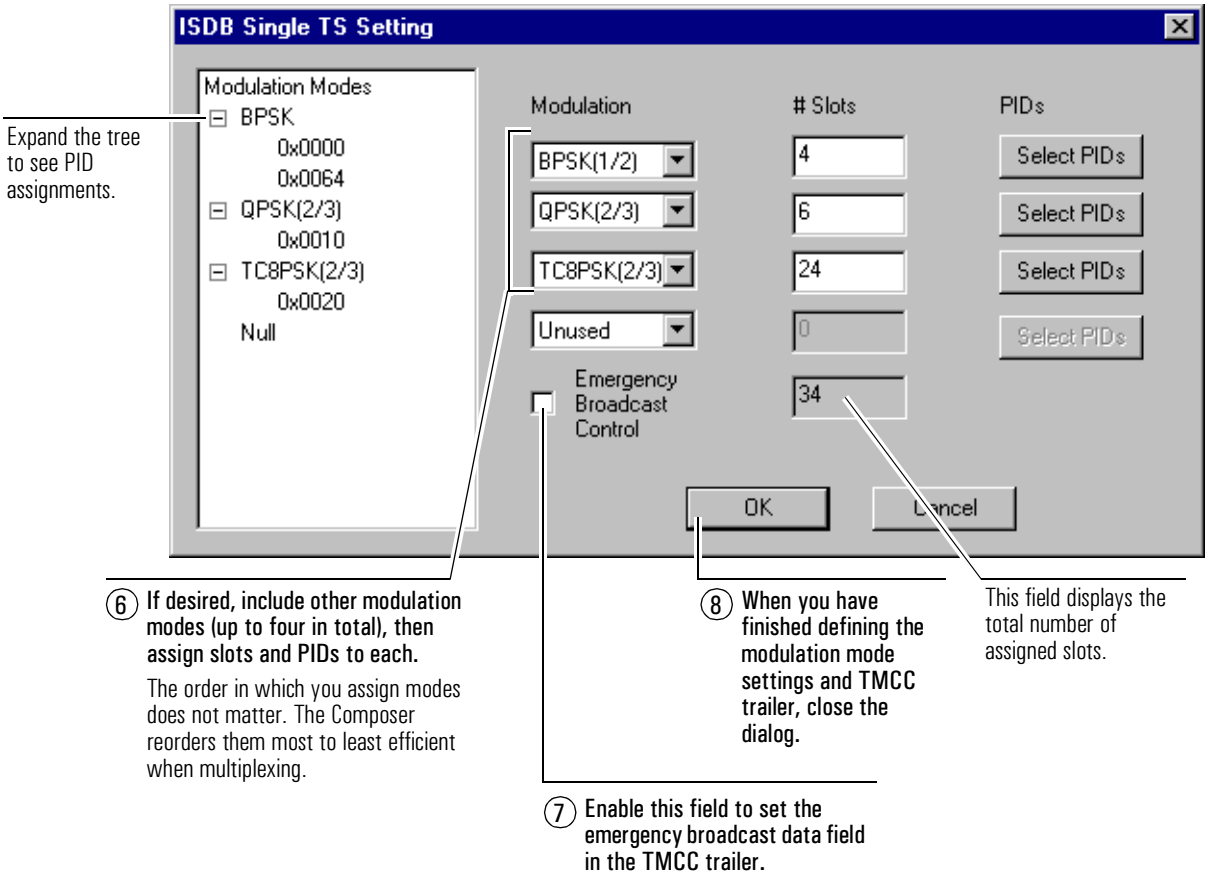
The modulation modes are ordered least to most efficient in the dropdown list.

④ **Select the number of slots to allocate to this modulation mode.**
Except for TC8PSK(2/3), which has no dummy slots, modulation modes are assigned in multiples appropriate to their inner code rate (the number in parenthesis). Refer to "**Rules for slot assignment:**", page 8–14 for more information.




⑤ **Assign one or more PIDs to the modulation mode by double clicking on the PID to move it to the assigned column.**

Normally the PAT and PMT are assigned to the least efficient (most robust) modulation mode, whereas a large stream, such as a video elementary stream, is assigned to a more efficient mode.




The following information may be helpful when selecting the modulation mode and assigning slots for a specific PID:

1 Note that each slot can carry data at approximately 1 Mb/s.

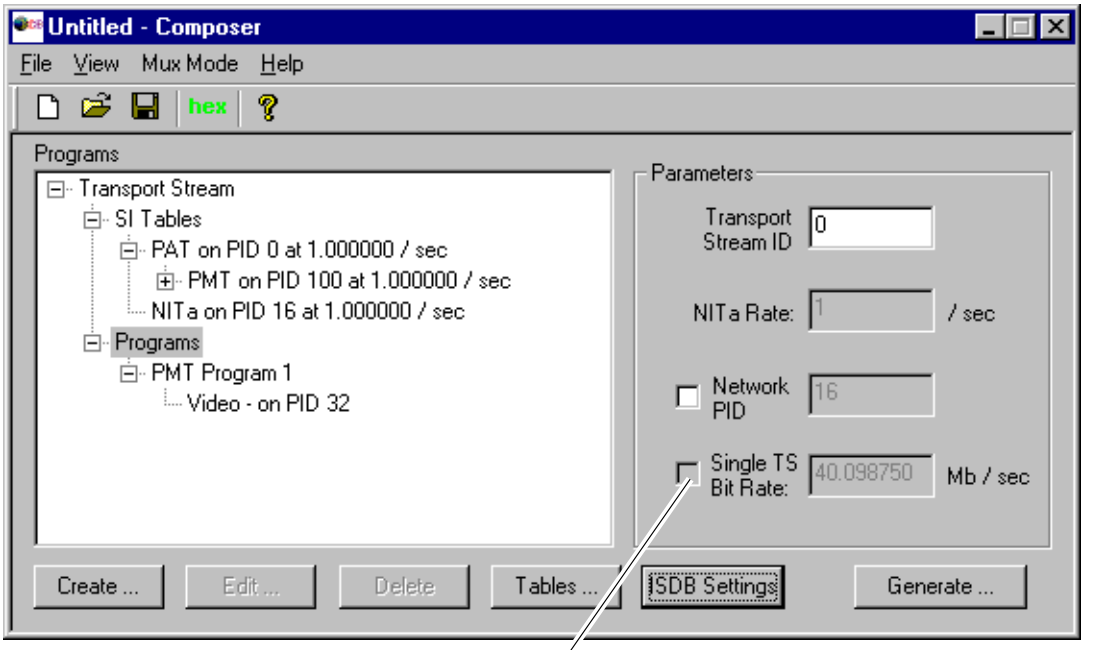
2 Check the size and bit rate of the elementary stream on the PID for which you want to assign a modulation mode.

3 Note the inner code rate of the modulation mode you are selecting. For example, QPSK(2/3) carries PID data on only two out of three slots. The third slot is a dummy slot.

4 Enter the correct number of slots to ensure that the complete file can be multiplexed, taking into consideration the bit rate and size of the file plus the inner code rate of the modulation mode.

If you receive an abnormal termination with a decoder buffer underflow message when you attempt to generate the stream, try increasing the slot size (in increments appropriate to the modulation mode) then regenerating the stream until multiplexing terminates normally. During multiplexing the Composer displays information about the TMCC trailer and transport stream components—such as the bit rate of each elementary stream—which can help you determine the correct number of slots to assign.

⑨ Follow the procedure outlined in the next step to generate the ISDB single transport stream.

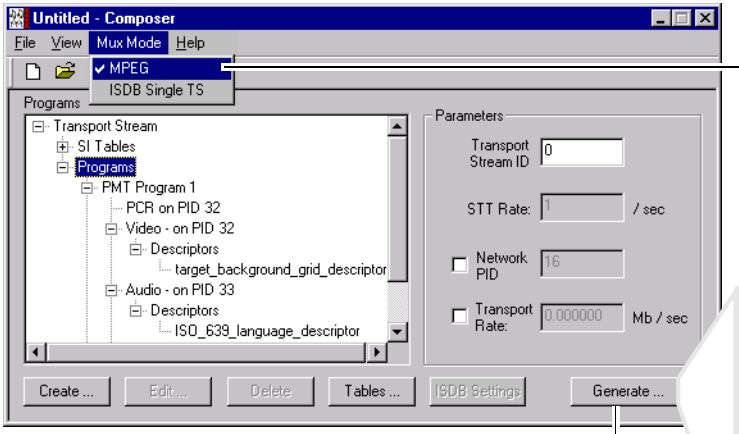



This is a display field only In **ISDB Single TS** mode. The checkbox is disabled and the rate cannot be set directly. The **Single TS Bit Rate** is calculated as follows:

$$56.61\ Mb/s \times \frac{total\ no.\ of\ slots\ used}{48}$$

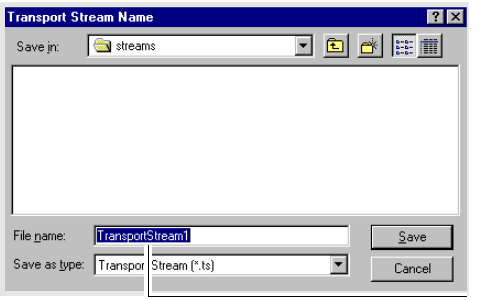
where *48* is the total number of possible slots

# Step 12: Generate a transport stream




1. Select the multiplexing mode.
   Use **ISDB Single TS** mode for ISDB streams. Use **MPEG** mode for all other types.

2. Select **Generate.**

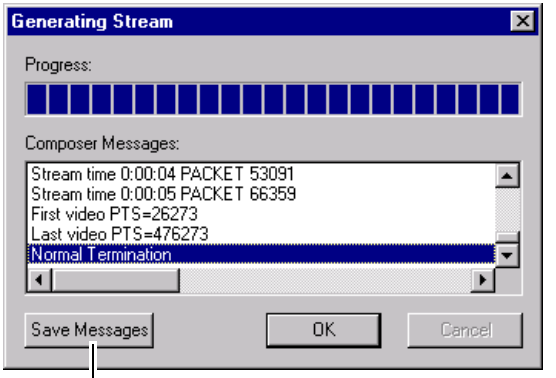



3. Enter a name for the transport stream file, then select **Save** to begin multiplexing.

The Composer provides run-time feedback—both status and error information. Typically, you should see a startup greeting, analysis report of all the programs and elementary streams, some multiplexing statistics, and completion notifications.

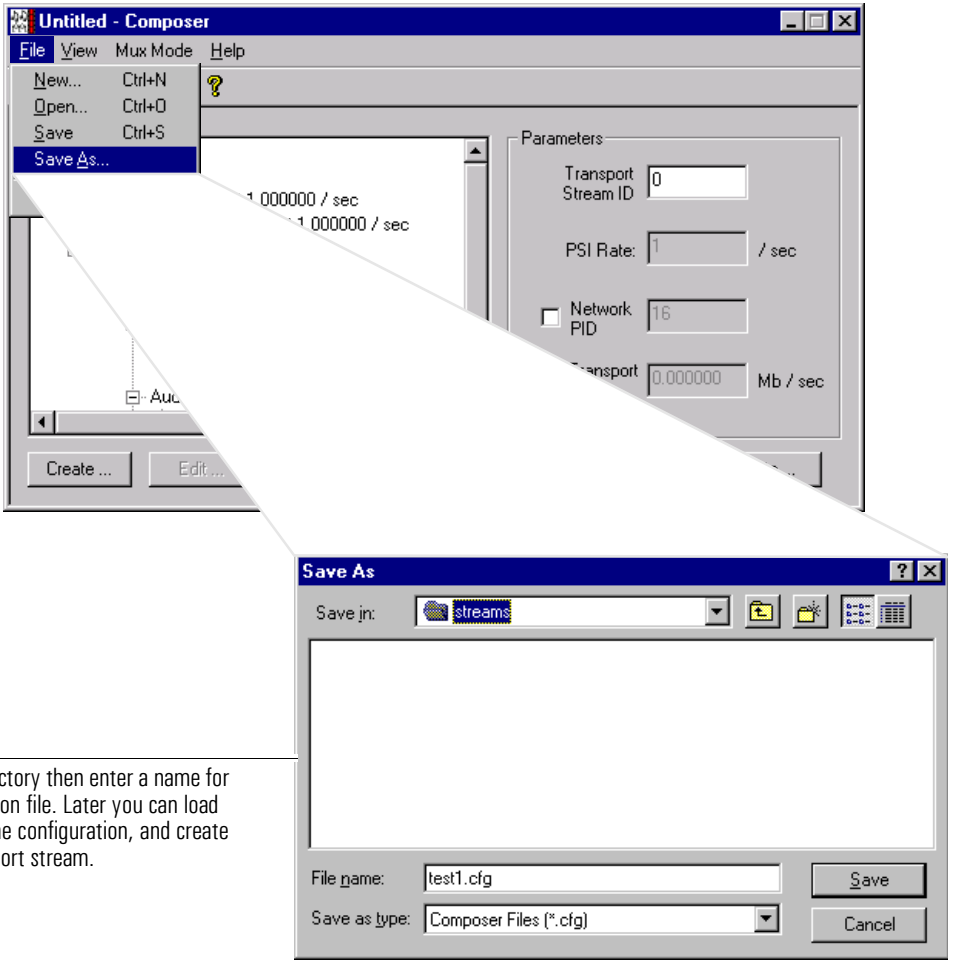
If the Composer encounters an error, such as an invalid elementary stream, multiplexing will stop. If no errors are encountered, the Composer displays a successful completion notification. The **OK** button will be enabled when multiplexing is complete. You can press **Cancel** any time to stop multiplexing.

After the transport stream file is successfully multiplexed, you can play it from the Recorder/Player or analyze it from the Protocol Data Viewer.



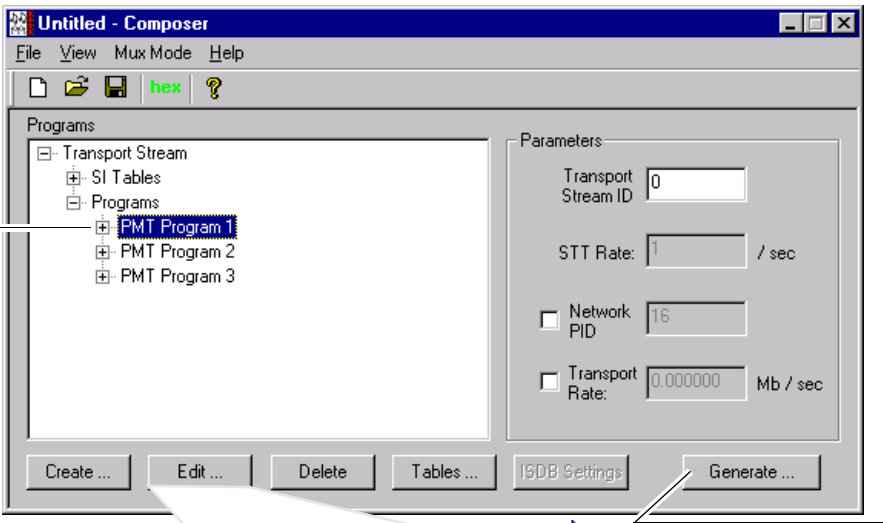

You can save status and error messages to a text file

## Step 13: Save the configuration




Select the directory then enter a name for the configuration file. Later you can load the file, edit the configuration, and create another transport stream.

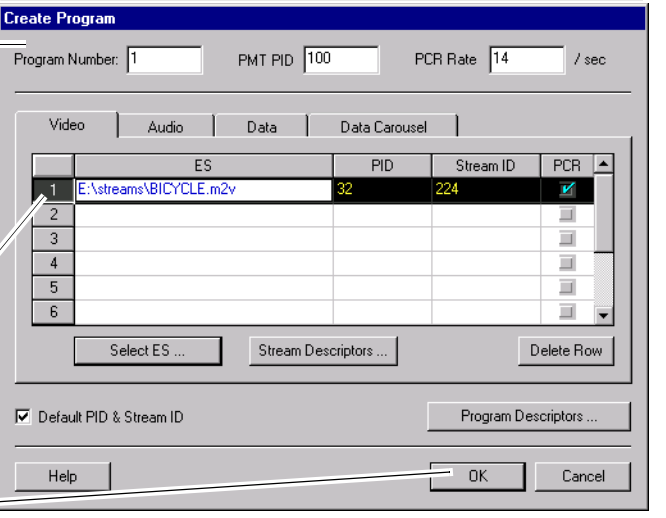
# Step 14: Edit a program




1. Select the PMT Program entry of the program you want to redefine.

IMPORTANT

4. Regenerate the transport stream.

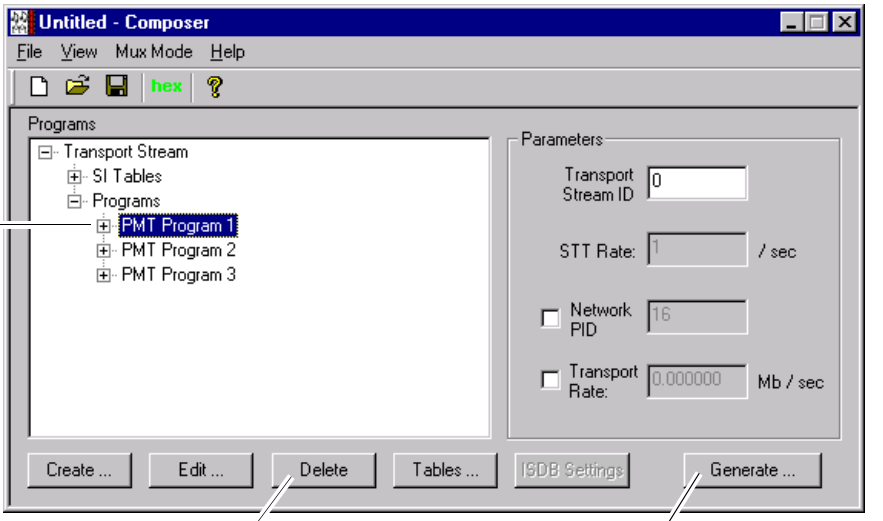



2. Redefine the program by changing the elementary stream files, PIDs, Stream IDs, PMT PID, or any other information you have specified.

To remove an entire stream entry, select the row marker and press **Delete**.

3. Close the dialog.

## Step 15: Delete a program



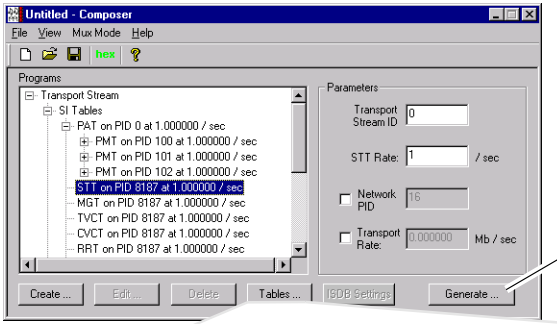

1. Select the PMT Program entry of the program you want to delete.

2. Delete the program.

IMPORTANT

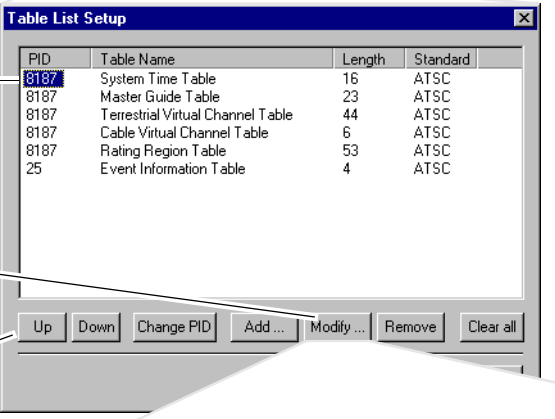
3. Regenerate the transport stream.

# Step 16: Edit a table




IMPORTANT

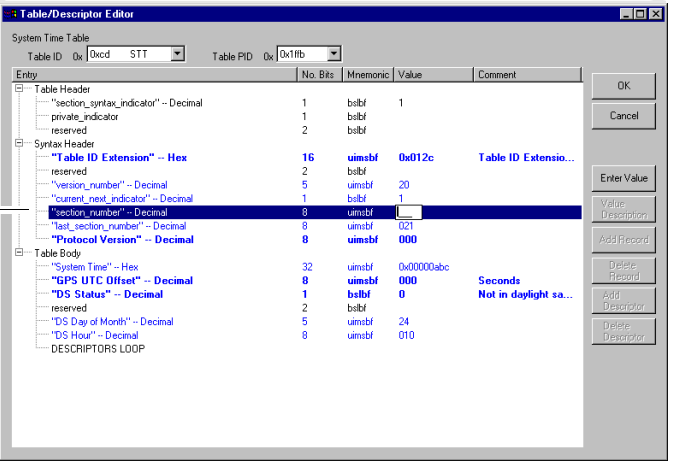
5 Regenerate the transport stream.




1 Select the PID of the table you want to edit. To change the PID, click the **Change PID** button and add a new value.

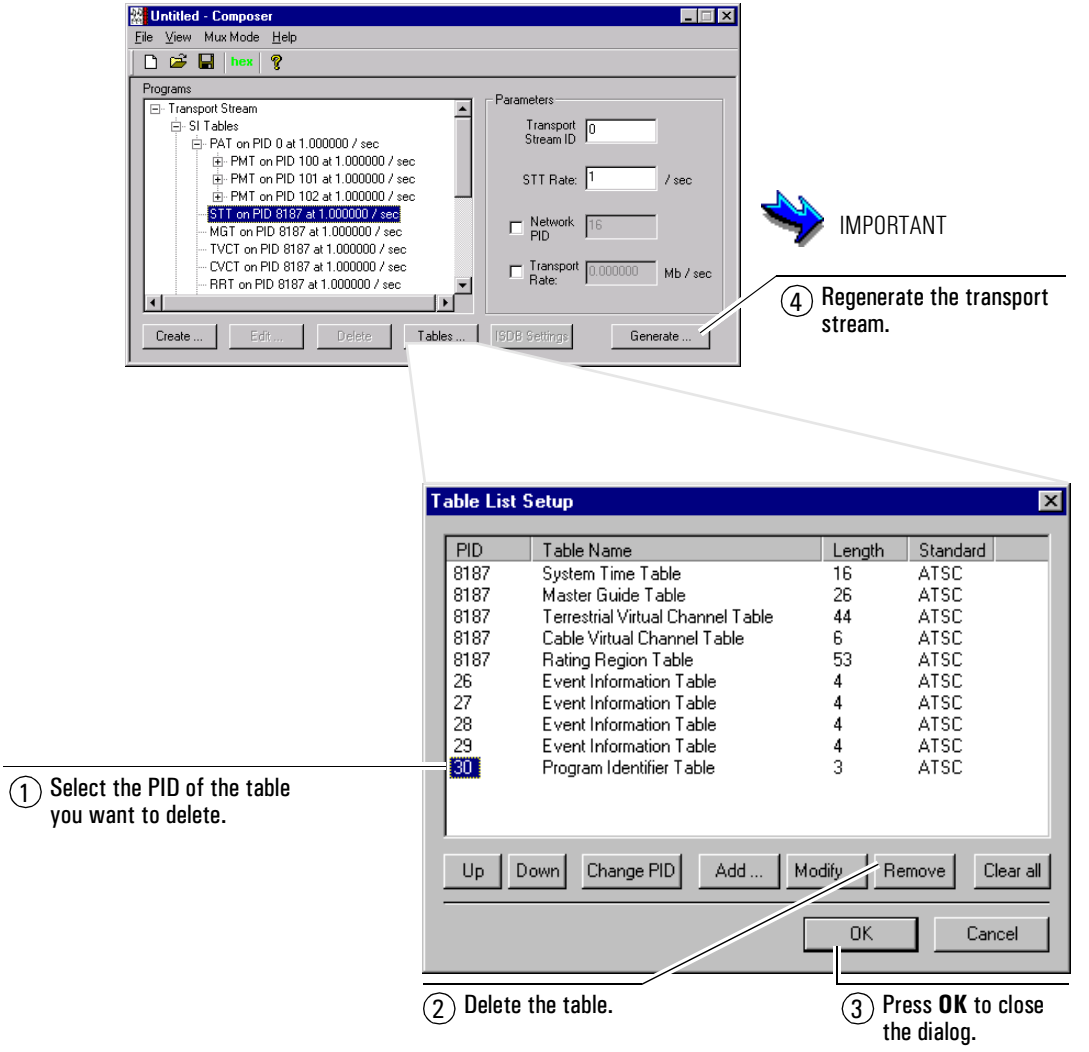
2 Press **Modify** to open the **Table/Descriptor Editor**.

4 After editing the table, you can use the **Up** and **Down** buttons to reorder tables. Press **OK** to close the dialog.




3 Edit the table, then press **OK** to close dialog.

## Step 17: Delete a table



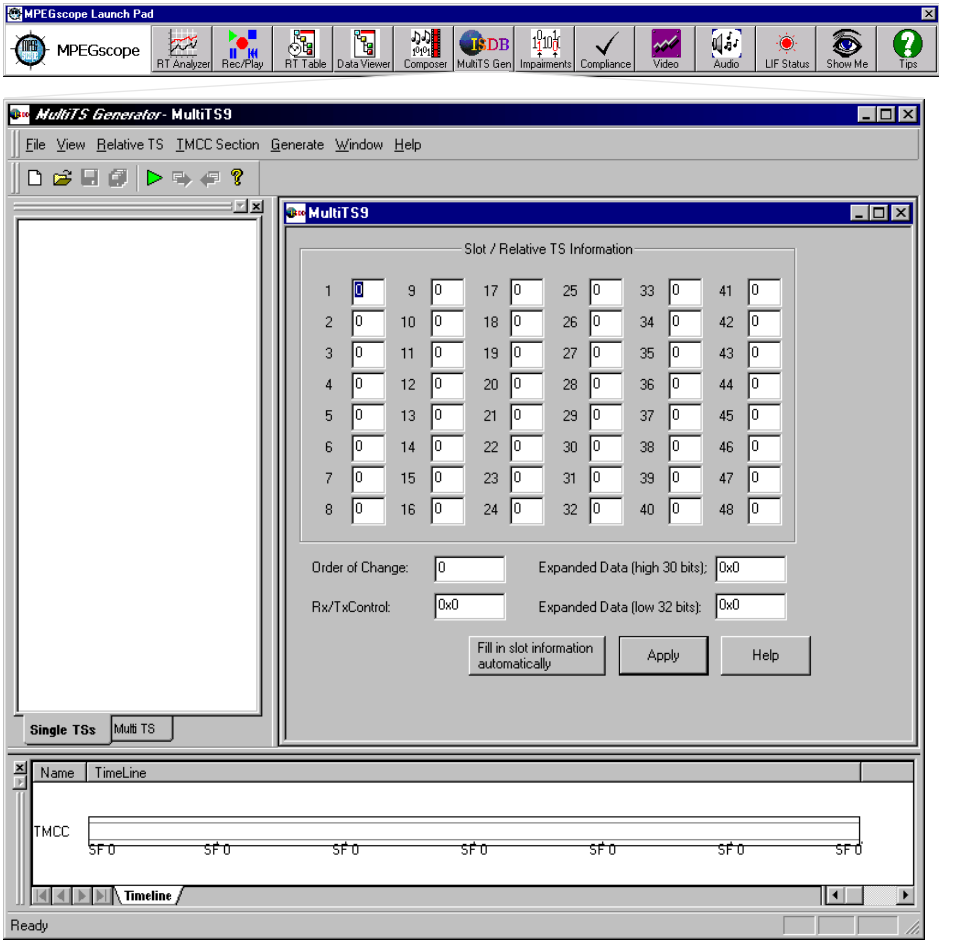
# 8

# Using the ISDB MultiTS Generator

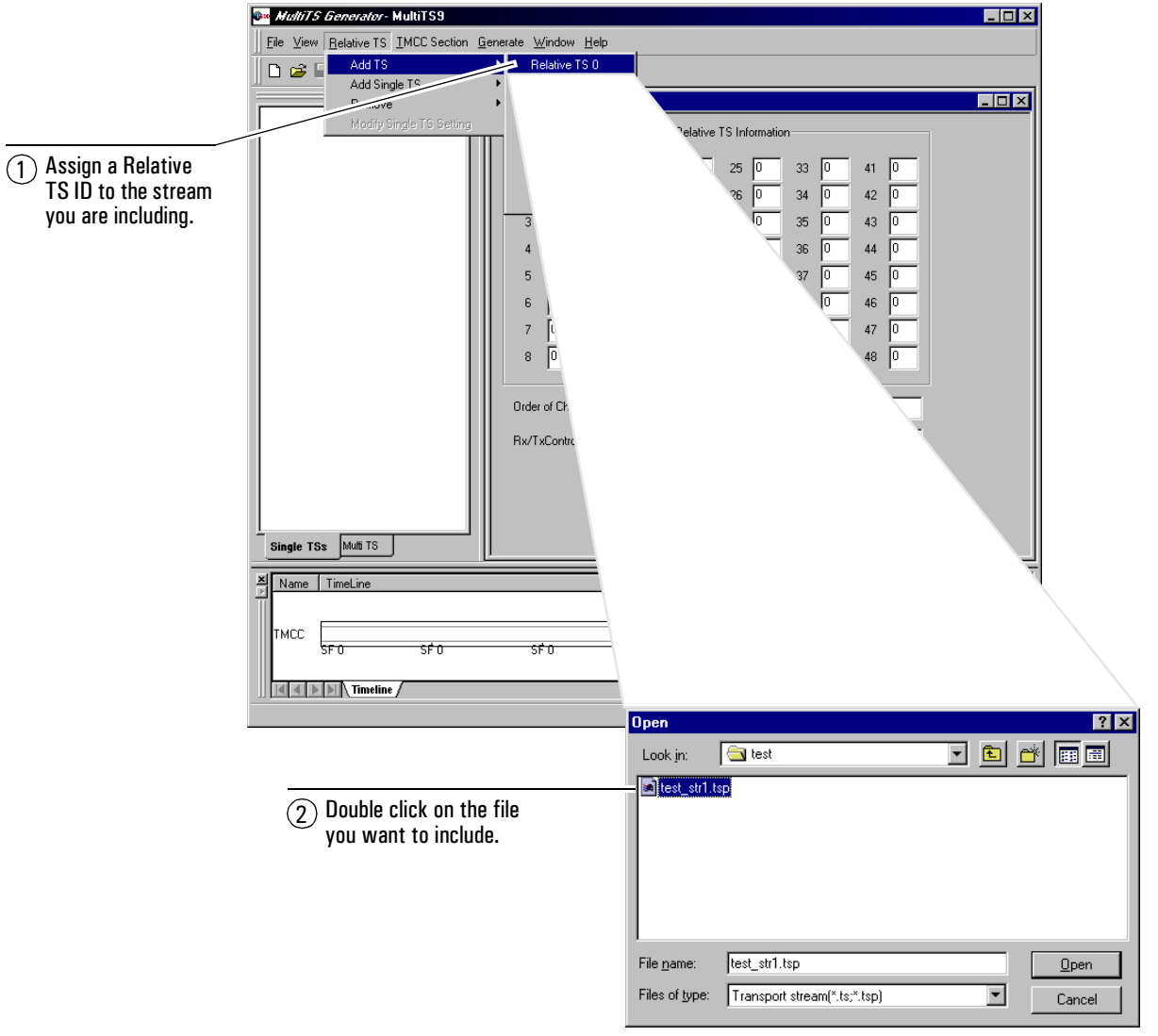
# Creating a single transport stream


This example illustrates how to create an ISDB-S single transport stream from an MPEG-2 transport stream.

## Step 1: Start the ISDB MultiTS Generator



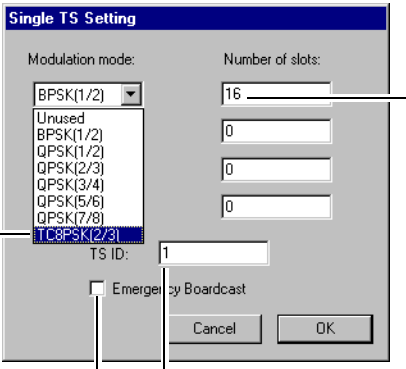
## Step 2: Specify an MPEG-2 transport stream file



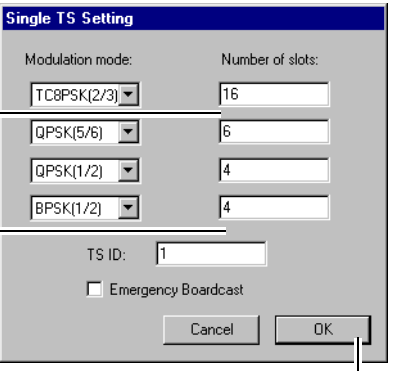

1. Assign a Relative TS ID to the stream you are including.

2. Double click on the file you want to include.

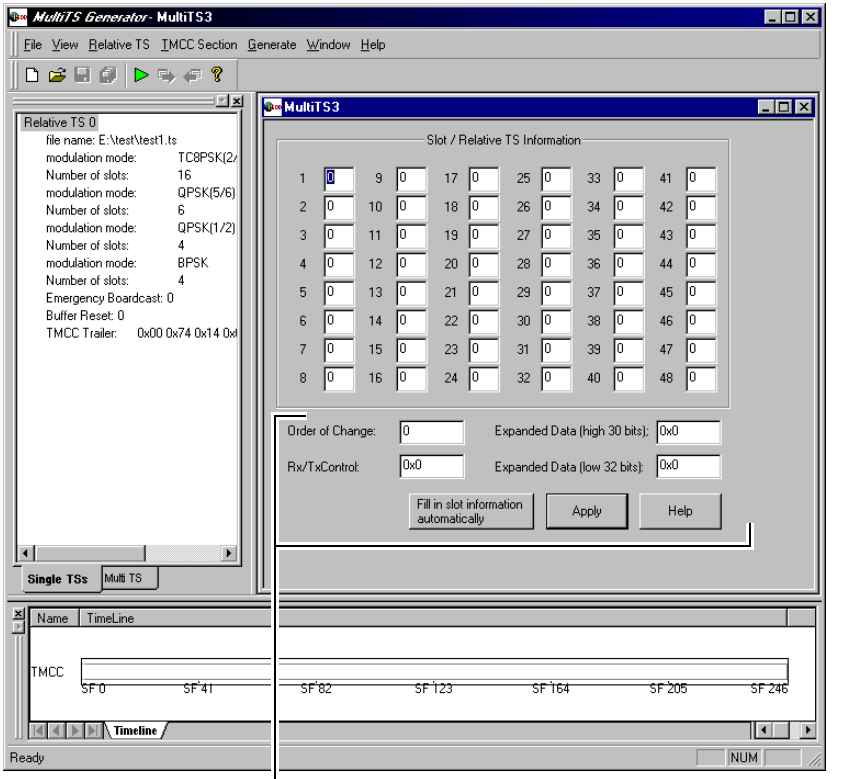
## Step 3: Define TMCC data




1. Select a modulation mode from the drop-down list.
2. Enter the number of slots to allocate to this mode in the framing structure for this stream.
3. Enable this field if the stream is carrying emergency broadcast data.
4. Enter the 16-bit MPEG-2 transport stream ID of the stream.
   This value is specified in the PAT (Program Allocation Table).
5. If desired, select other modulation modes for the stream (up to four different modes) and enter the number of framing slots to use for each.




6. Close the dialog.



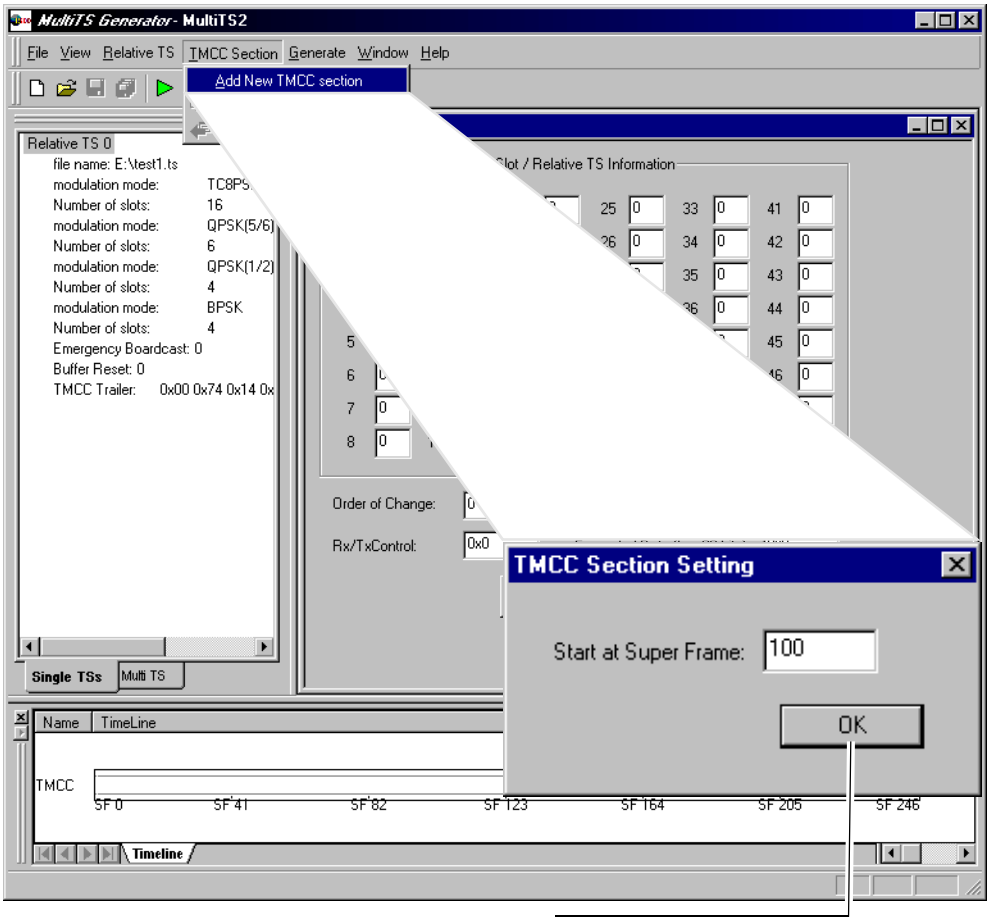

7 If desired, change the default settings for other TMCC data fields, then apply the changes.

The information you enter above and at the **Single TS Setting** dialog is incorporated into the first 8 bytes of the TMCC (Transmission Multiplexing Configuration Control) 16-byte trailer.

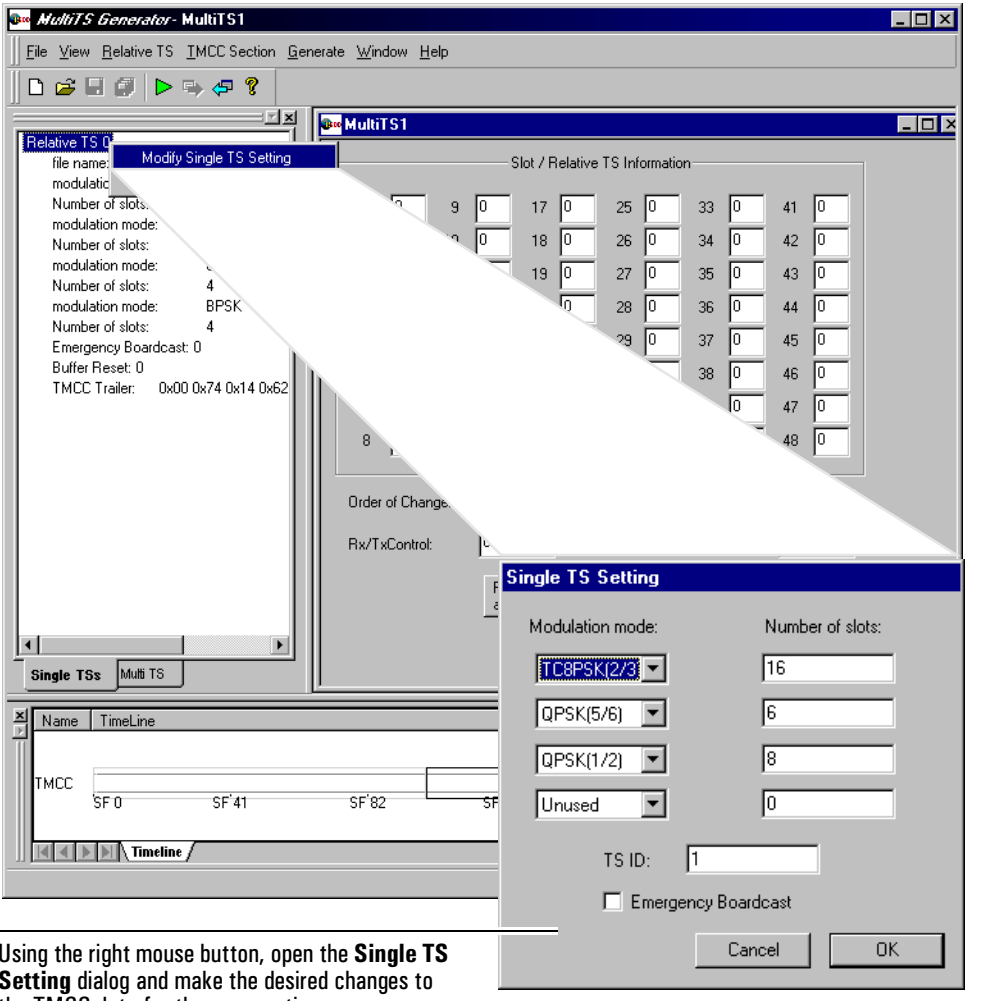
MPEGscope does not add Reed-Solomon coding to the last 8 bytes of the TMCC trailer. Currently these bytes are set to zero.

## Step 4: Define additional TMCC sections

You can define one or more additional TMCC sections, and assign different modulation modes and TMCC data to each, as long as the total number of slots for each Relative TS remains the same throughout all TMCC sections.




① Enter the super frame number where you want the TMCC section to start, then press **OK**.



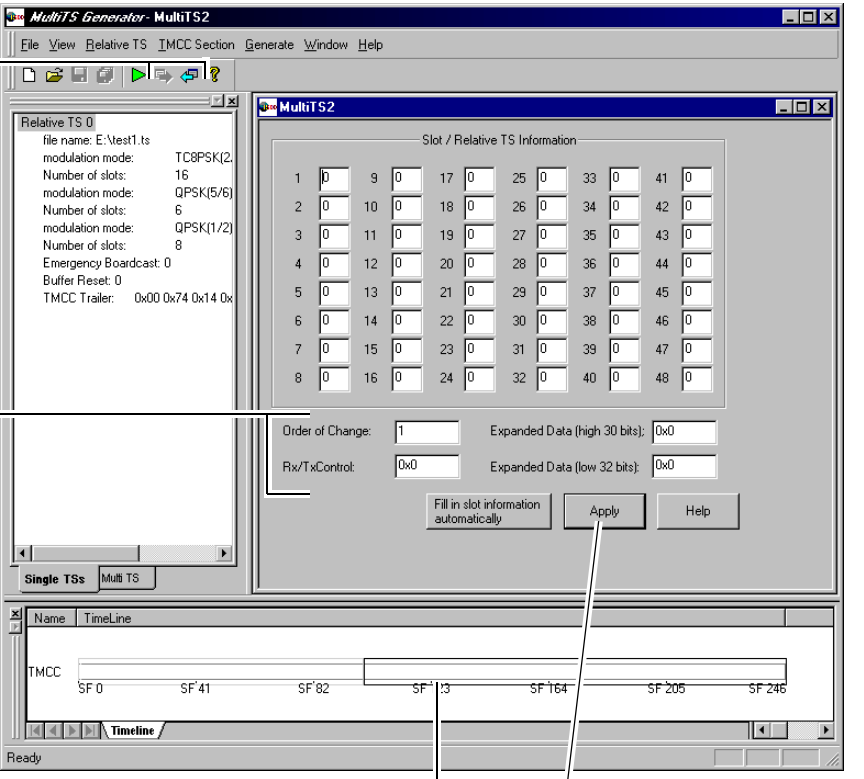

② Using the right mouse button, open the **Single TS Setting** dialog and make the desired changes to the TMCC data for the new section.

When defining new sections, you must adhere to the following rules:

1 You can change the MPEG-2 TS ID, emergency broadcast flag, modulation modes, and slot assignments in each section; however, the total number of slots for each Relative TS must remain the same.

2 You cannot add new Relative TSs to additional sections, although you can substitute different MPEG-2 TS files by removing a Relative TS, then specifying a new file for that Relative TS number.

Press the arrow buttons to view other TMCC sections you have defined.

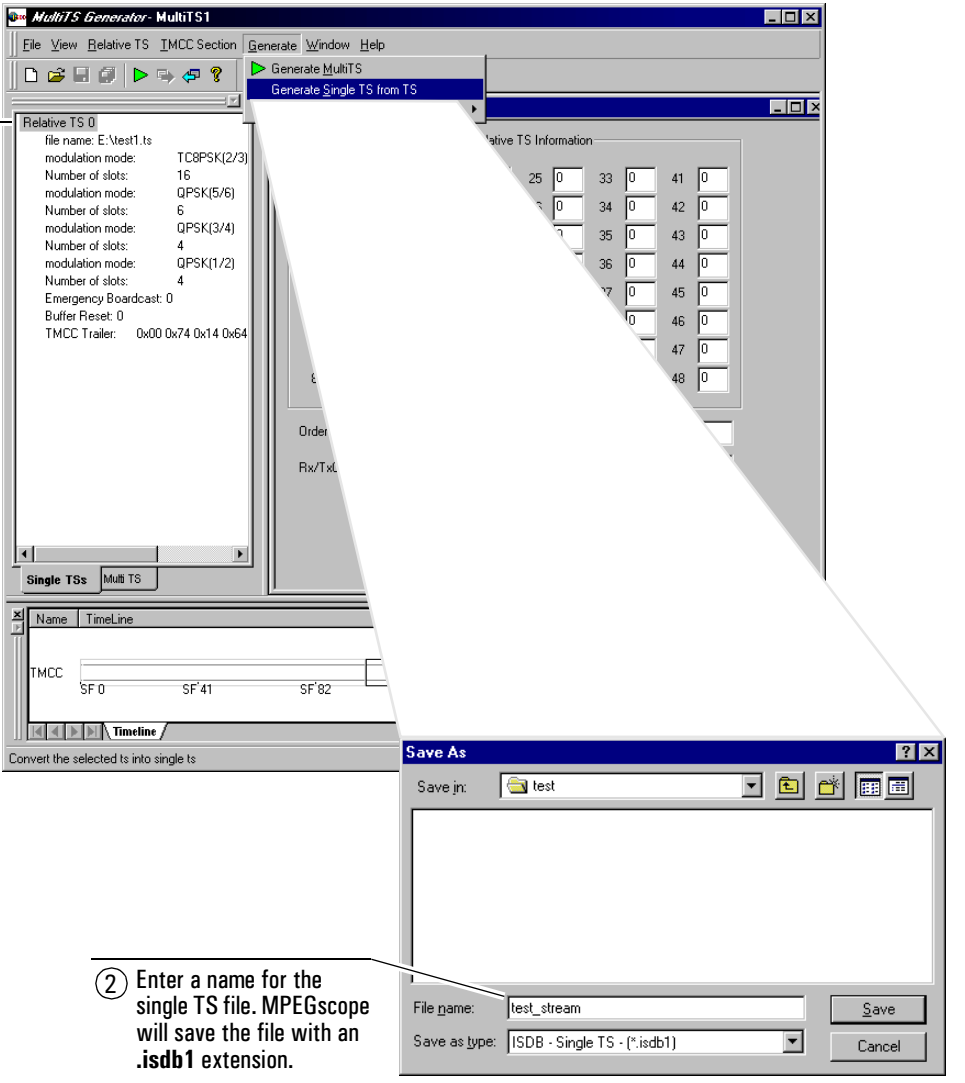



③ Increment the **Order of Change** field each time you create an additional section, then make any other desired changes to TMCC data.

The current TMCC section is highlighted.

④ Press Apply to apply your changes.

## Step 5: Generate a single transport stream



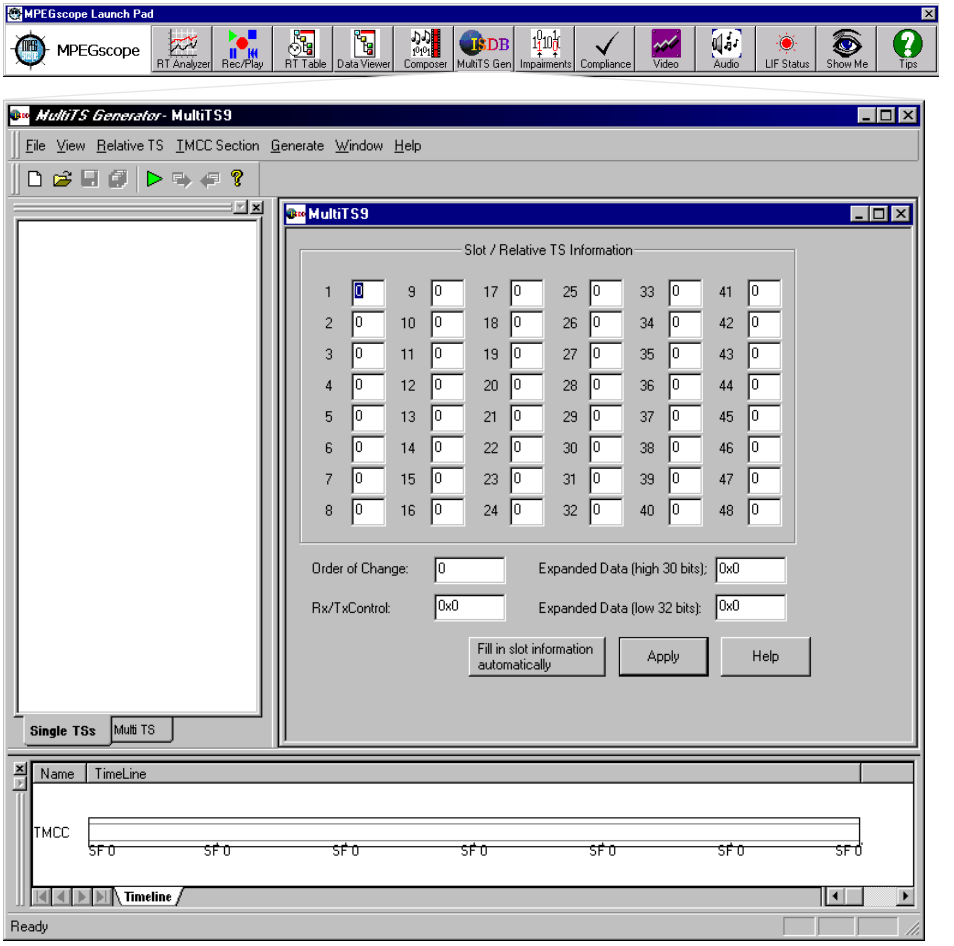

1. Select the Relative TS.

2. Enter a name for the single TS file. MPEGscope will save the file with an **.isdb1** extension.

# Creating a multiple transport stream

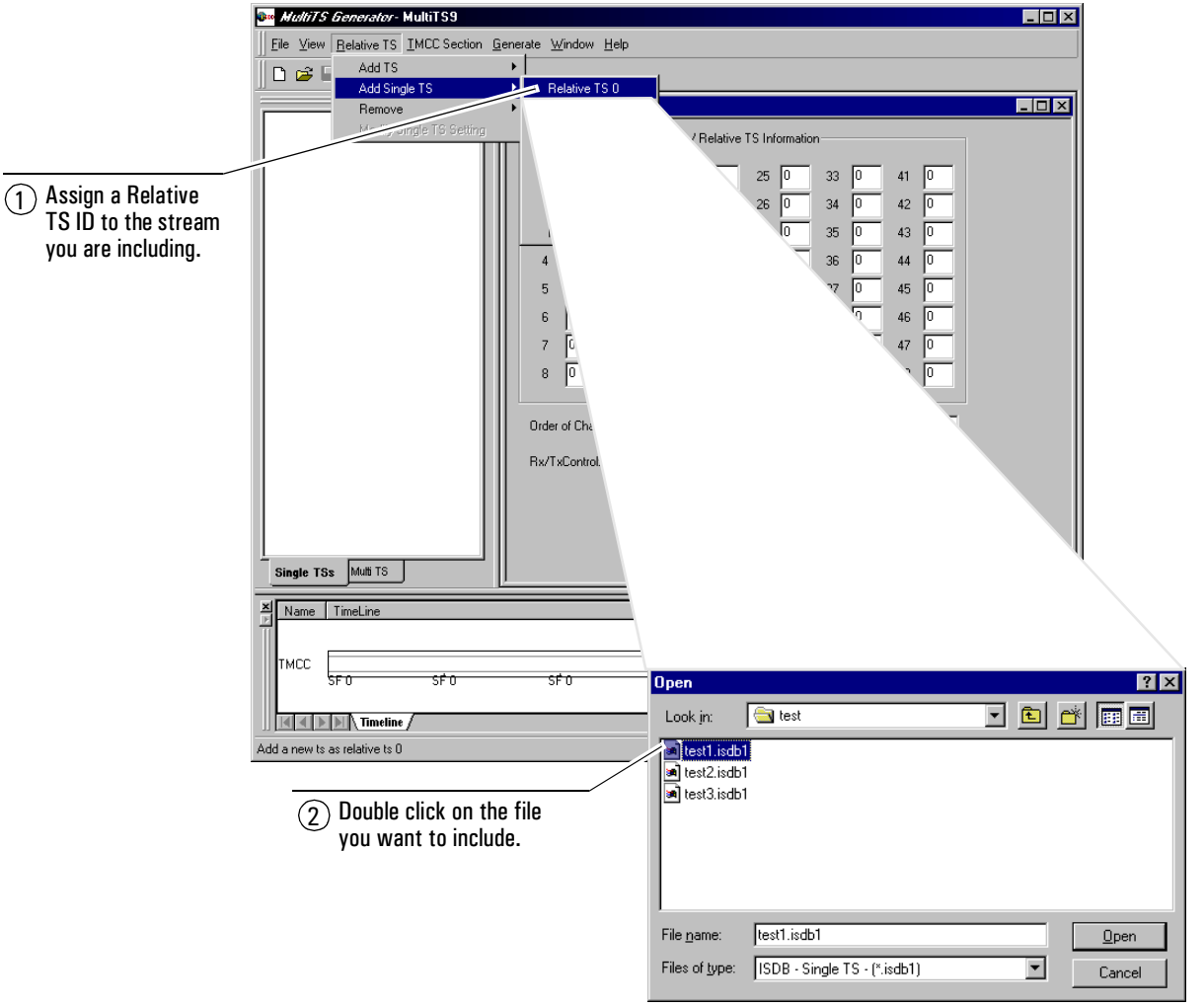

This example illustrates how to define an ISDB-S super frame structure and create an ISDB-S multiple transport stream.

## Step 1: Start the ISDB MultiTS Generator

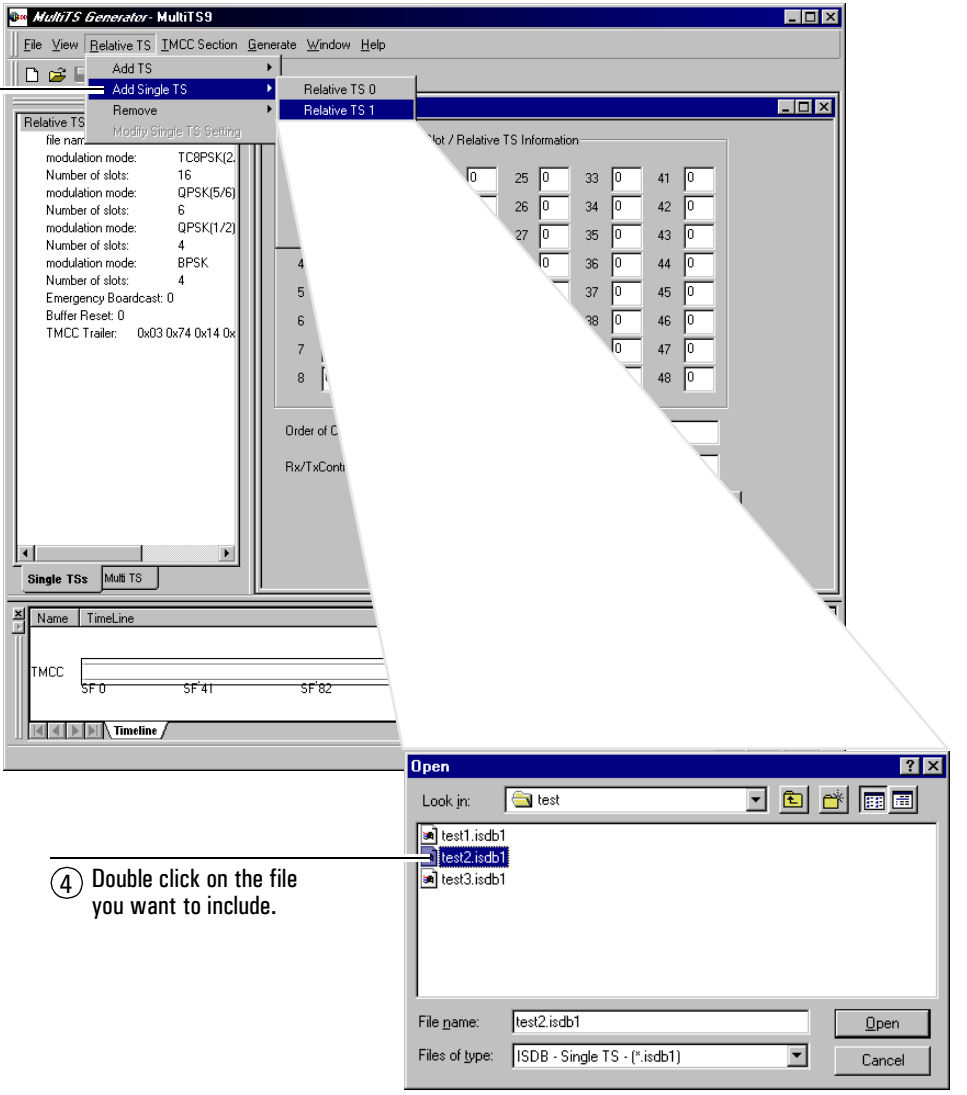


## Step 2: Select the transport stream files

You can generate an ISDB-S multiple transport stream from either MPEG-2 TS files or ISDB-S single TS files. This example uses ISDB-S single TS files. If you use MPEG-2 TS files, the **Single TS Setting** dialog will appear as soon as you select a file, and you will need to enter TMCC data, as illustrated on page 8–4.




1. Assign a Relative TS ID to the stream you are including.
2. Double click on the file you want to include.

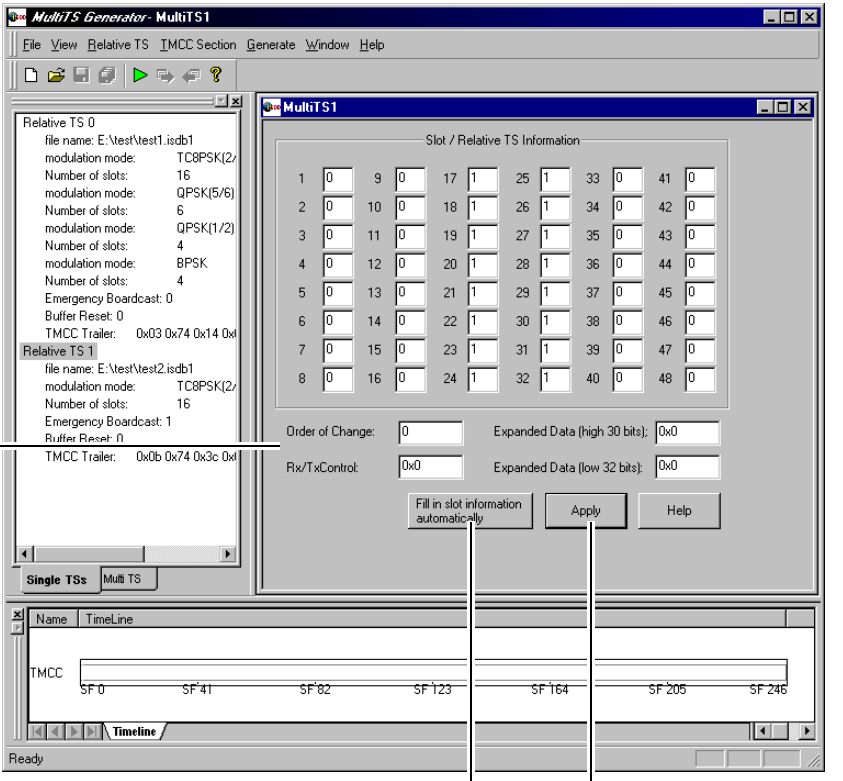



3. If desired, add another single TS file, using the next Relative TS number.

   You can add up to eight files.

4. Double click on the file you want to include.

# Step 3: Define TMCC data




1. If desired, change the default settings for other TMCC data fields.

2. Press this button to allow MPEGscope to define the framing structure automatically. Or you can define it manually by entering the Relative TS numbers of the TS stream into the appropriate slots.

   For more information on defining the framing structure manually, refer to page 8–14.

3. Apply the changes.

**Rules for slot assignment:**

1 If more than one modulation mode is used, the most efficient mode is ordered first in the super frame, regardless of the TS stream's Relative TS number. The table below lists the modulation modes in order of most to least efficient.

2 Except for TC8PSK, which has no dummy slots, modulation modes are assigned to slots in multiples appropriate to their inner code rate (the number in parentheses), as shown in the table below.

| Modulation mode | Assign to slots in multiples of... |
|---|---|
| TC8PSK (2/3) | — |
| QPSK (7/8) | 8 |
| QPSK (5/6) | 6 |
| QPSK (3/4) | 4 |
| QPSK (2/3) | 3 |
| QPSK (1/2) | 2 |
| BPSK (1/2) | 4 |

**Example**

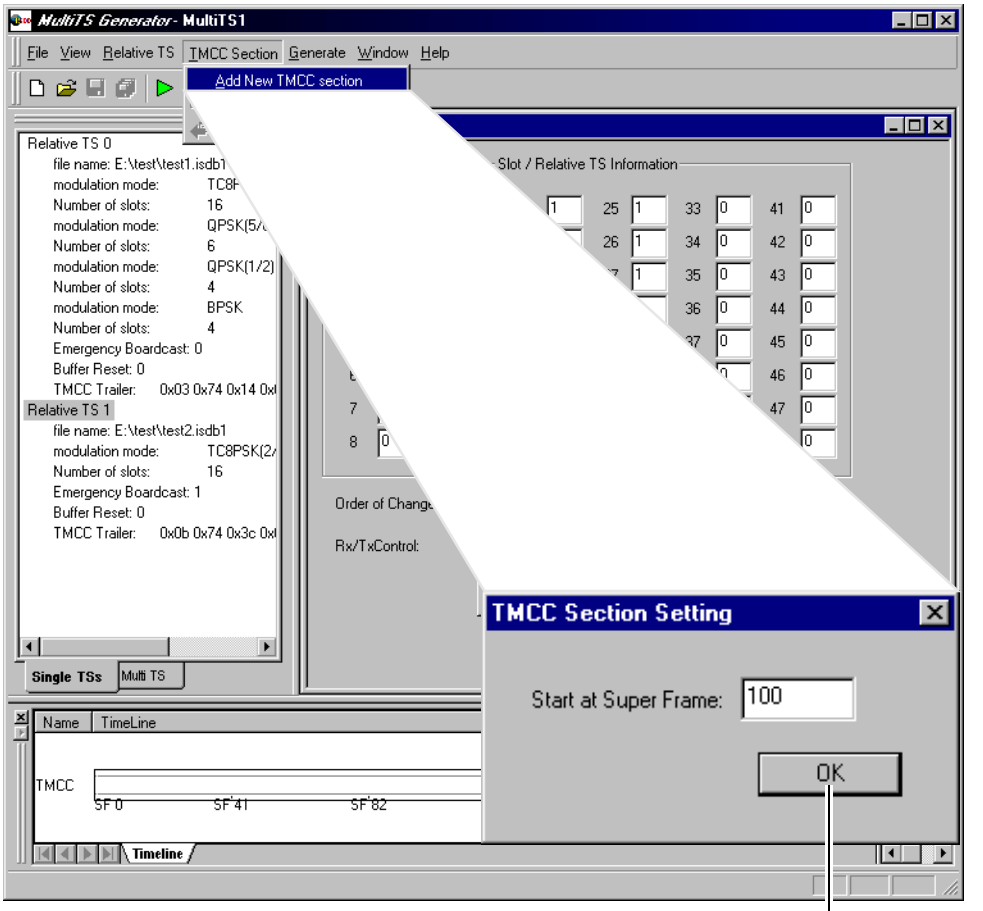
In the example on page 8–13, the most efficient mode is TC8PSK, selected for **Relative TS 0** and **Relative TS 1**. Both streams have 16 slots assigned to this mode. A "**0**" (for **Relative TS 0**) is therefore entered into the first 16 slots, and a "**1**" (for **Relative TS 1**) is entered into slots 17 to 32.

The second most efficient mode is QPSK (5/6), selected for **Relative TS 0** with 6 slots assigned. A "**0**" (for **Relative TS 0**) is therefore entered into slots 33 to 38.

The least efficient mode is BPSK (1/2), also selected for **Relative TS 0** with 4 slots assigned. A "**0**" is entered into slots 39 to 42.

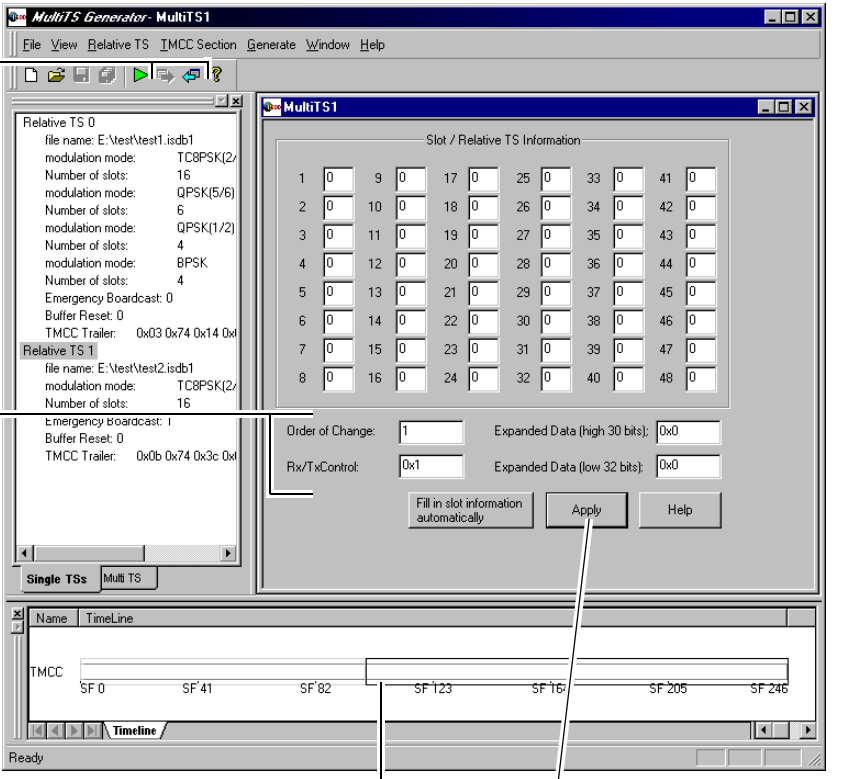
## Step 4: Define additional TMCC sections

You can define one or more additional TMCC sections, and assign different modulation modes and TMCC data to each, as long as the total number of slots for each Relative TS remains the same throughout all TMCC sections.




(1) Enter the super frame number where you want the TMCC section to start, then press **OK**.

Press the arrow buttons to view other TMCC sections you have defined.




② Increment the **Order of Change** field each time you create an additional section, then make any other desired changes to TMCC data.

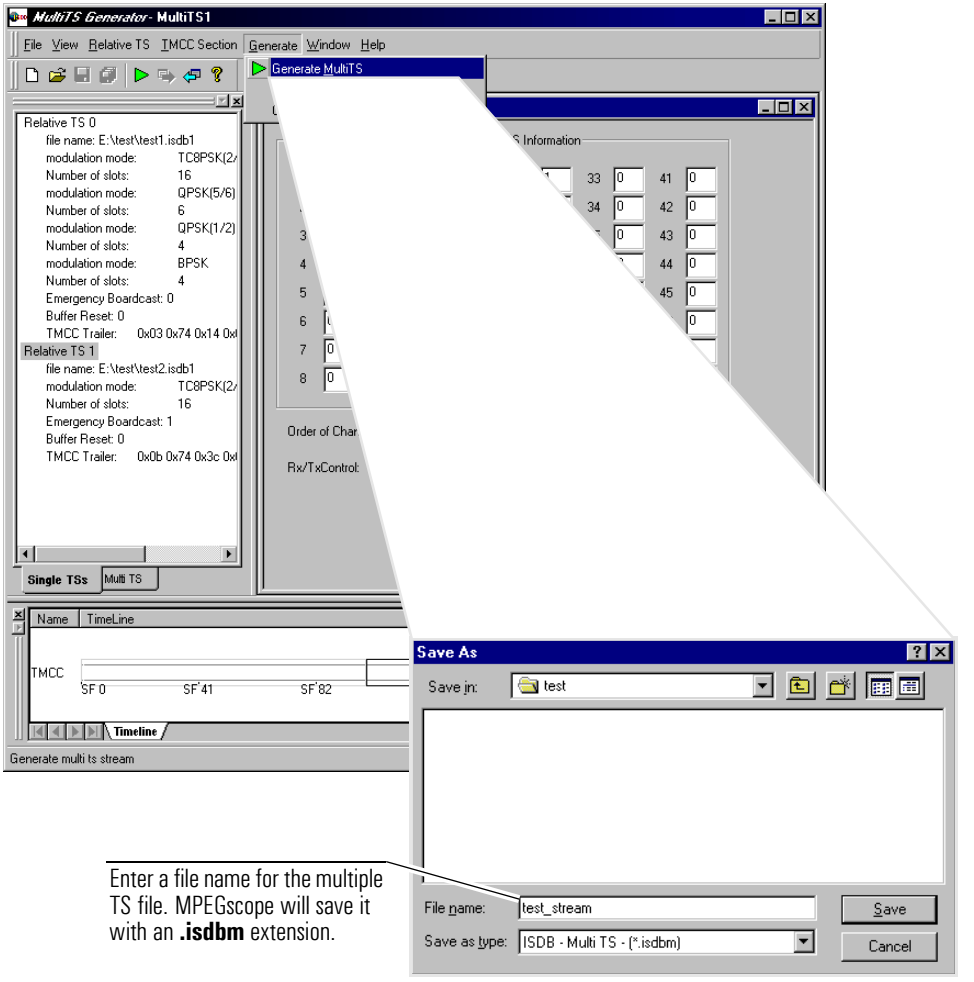
The current TMCC section is highlighted.

③ Press Apply to apply your changes.

You cannot add new Relative TSs to additional sections, although you can substitute different single TS files by removing a Relative TS, then specifying a new file for that Relative TS number.

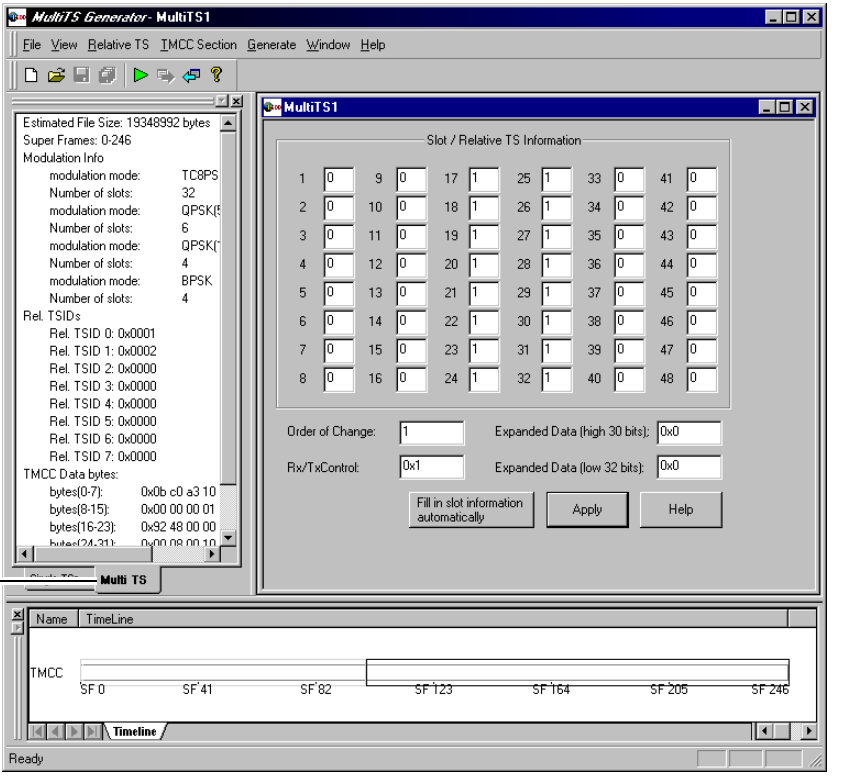
If you are creating an ISDB-S multiple transport stream from MPEG-2 TS files instead of ISDB-S single TS files, you can also change the MPEG-2 TS ID, emergency broadcast flag, modulation modes, and slot assignments in each section; however, the total number of slots for each Relative TS must remain the same. For an example of changing modulation modes, refer to page 8–7. If you change slot allocations, don't forget to update slot assignments by pressing the **Fill in slot information automatically** button, then applying your changes.

## Step 5: Generate the ISDB-S multiple transport stream




Enter a file name for the multiple TS file. MPEGscope will save it with an **.isdbm** extension.

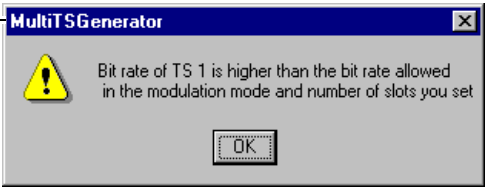
# Step 6: View Multi TS data

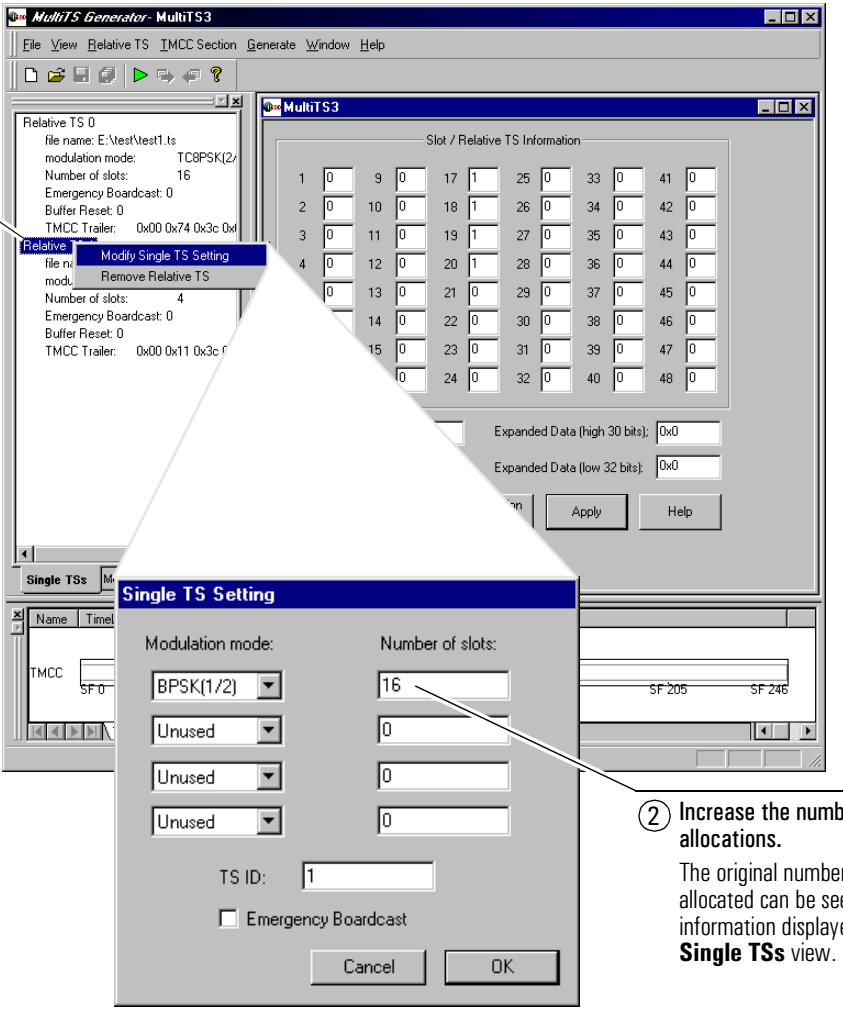



Select the **Multi TS** tab to see information about the multiple transport stream.

# Step 7: Troubleshooting

If you encounter this error message while attempting to generate an ISDB-S multiple transport stream from MPEG-2 TS files, follow the steps outlined below.

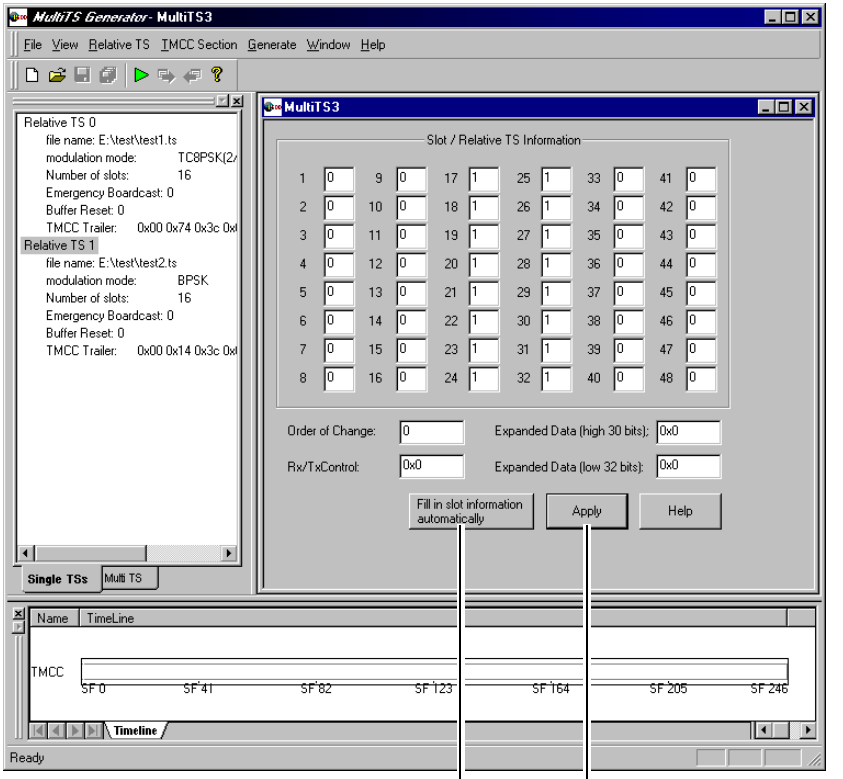



① **Using the right mouse button, highlight the Relative TS number cited in the error message, then select** Modify Single TS Setting **from the short cut menu.**

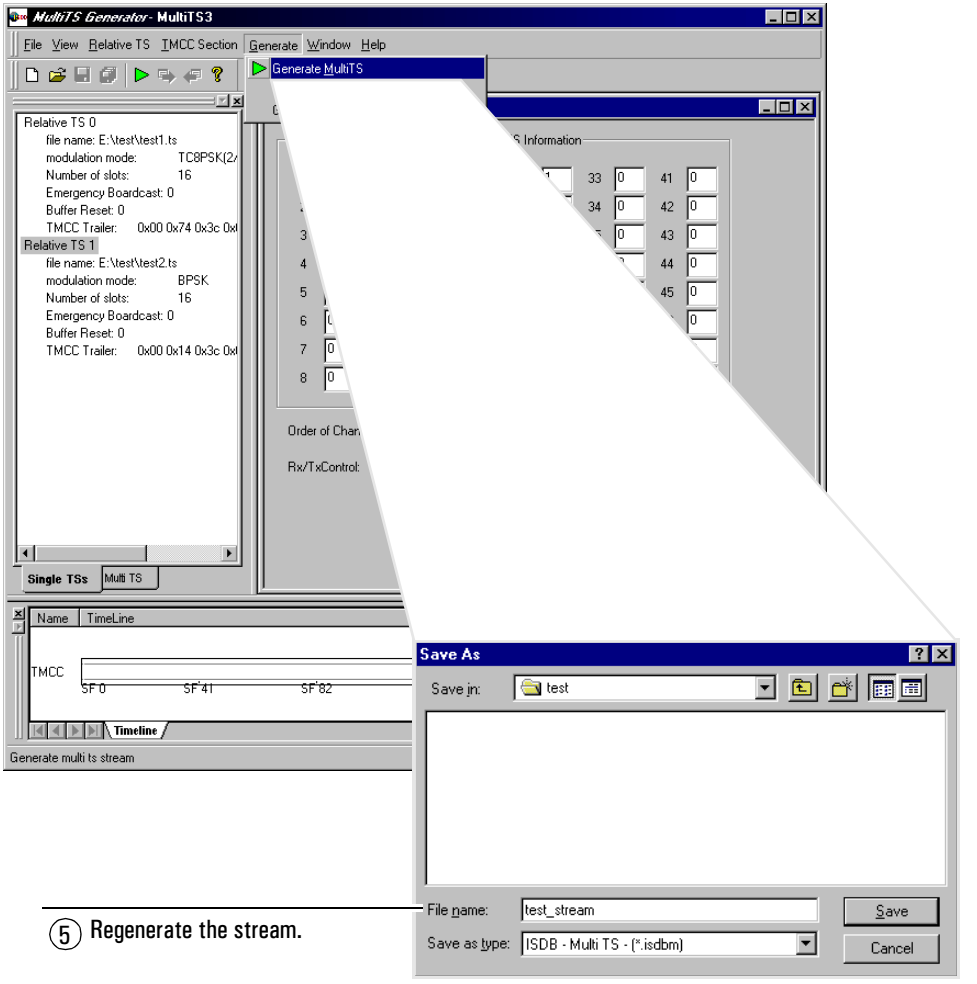
② **Increase the number of slot allocations.**

The original number of slots allocated can be seen from the information displayed in the **Single TSs** view.




3 Adjust the slot allocation in the framing structure.

4 Apply the changes.



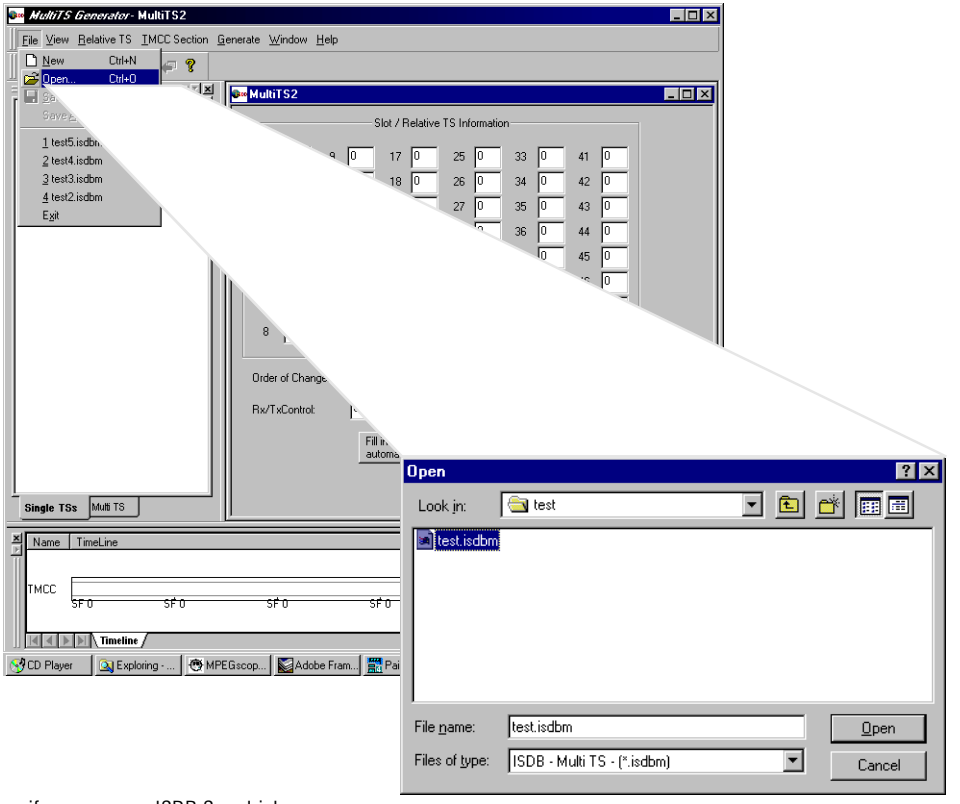

5 Regenerate the stream.

# Demultiplexing ISDB-S multiple transport streams


This example illustrates how to demultiplex an ISDB-S multiple transport stream into its constituent single transport streams. After demuxing, you can then use the single transport streams to create new multiple transport stream files.

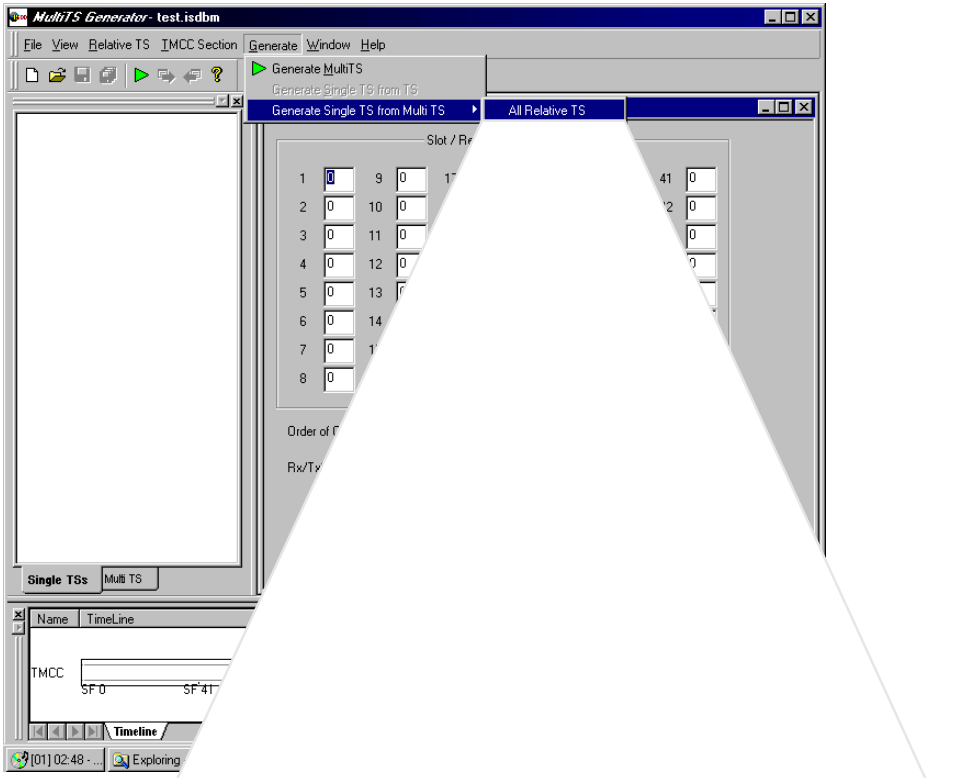
## Step 1: Open an ISDB-S multiple transport stream

Open the multiple transport stream file (**.isdbm**) you want to demultiplex.



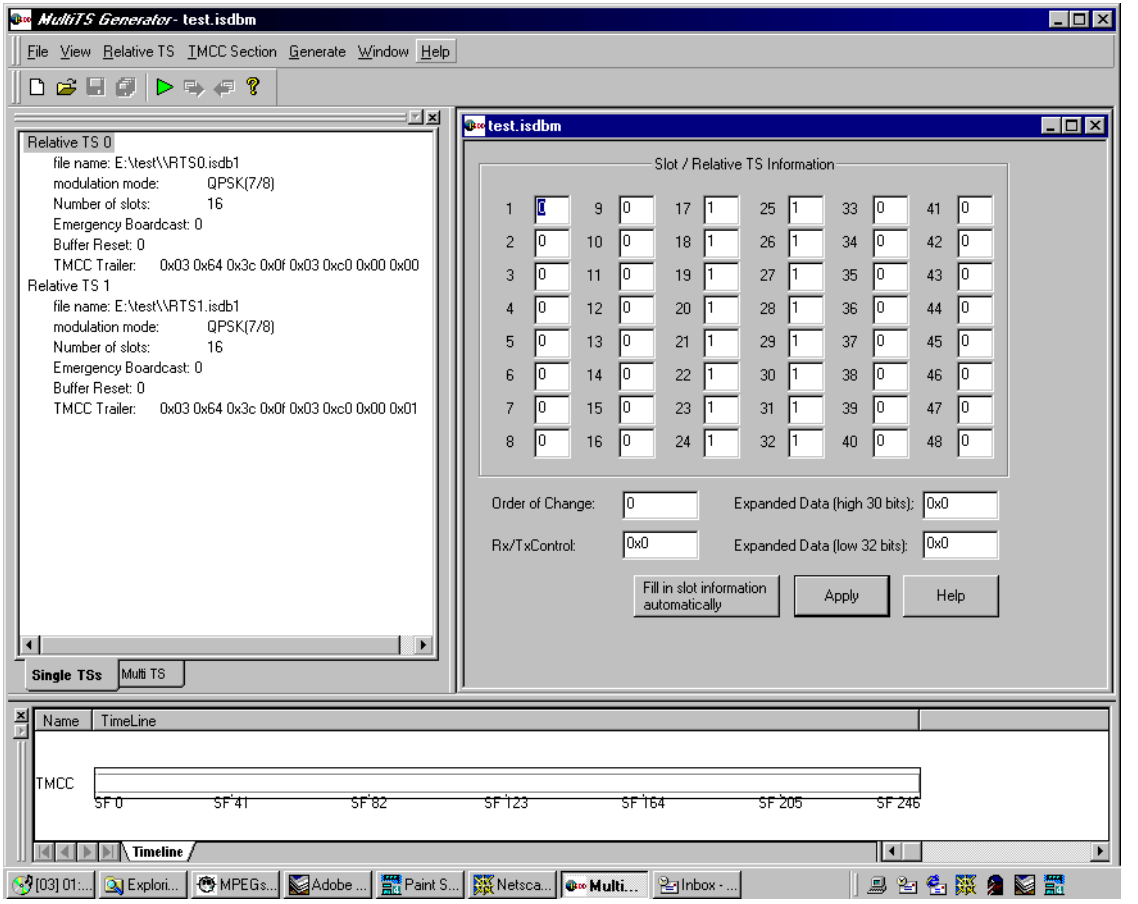

In the current release, if you open an ISDB-S multiple transport stream that contains more than one TMCC section, only the first TMCC section is displayed.

## Step 2: Demultiplex the stream




(1) Specify the output directory.

(2) Click OK to begin demultiplexing.




After demultiplexing, the single transport streams are displayed in the main dialog. The streams are located in the output directory as RTS0.isdb1 – RTS7.isdb1.

# Using interfaces with ISDB streams

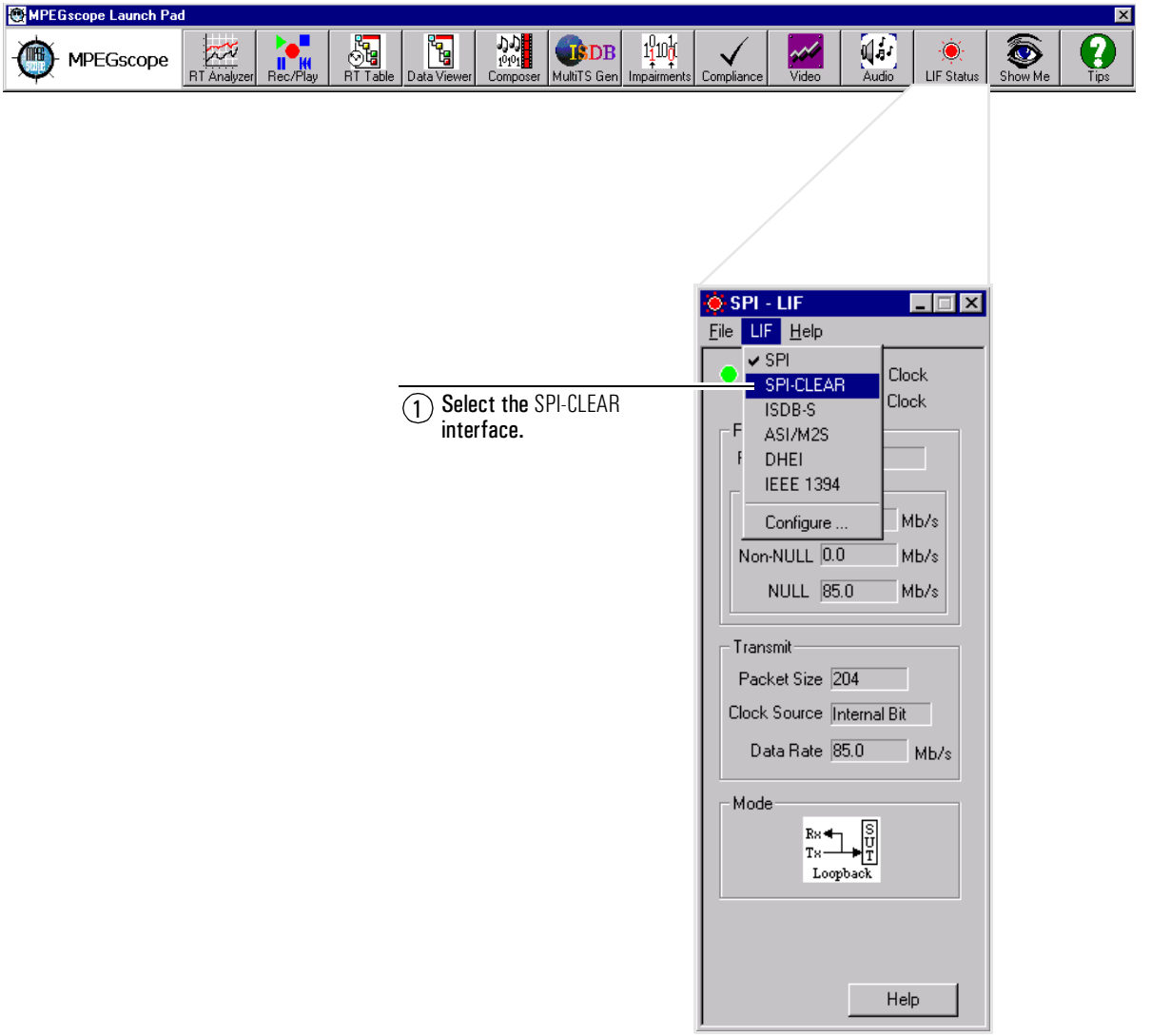
MPEGscope includes two interfaces, SPI-Clear and ISDB-S, for use with the ISDB Multi TS Generator. These interfaces are variants of the SPI interface, so input and output cables should be connected to the SPI interface's **DVB-SPI Rx** and **DVB-SPI Tx** ports, respectively.

## Configuring the interface for single transport streams

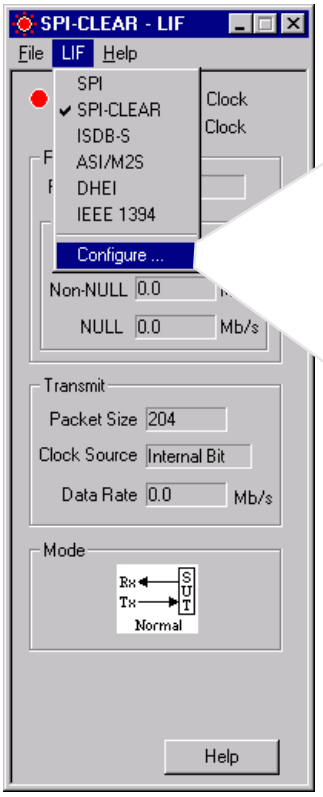
The SPI-Clear interface is used when you send or receive ISDB-S single transport streams with the Recorder/Player. It is similar to the SPI interface except as follows:

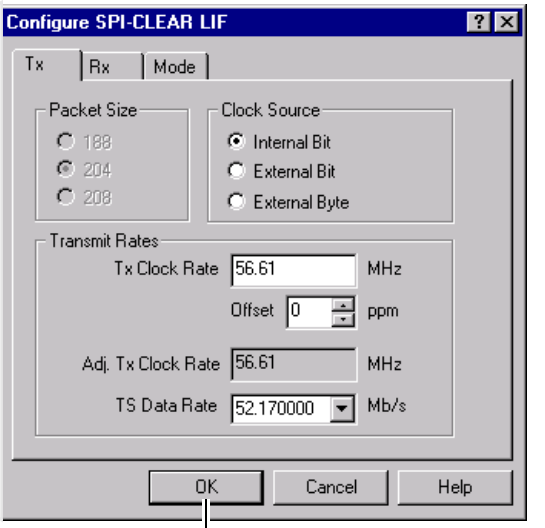
| | |
|---|---|
| Packet size | The MPEGscope SPI-Clear interface supports only 204-byte packets. |
| Packet trailer | The SPI-Clear interface transmits the entire 204 bytes without modifying the 16-byte trailer. In contrast, the SPI interface pads the 16-byte trailer of a 204-byte packet with zeros. |
| Opening files | You must select the SPI-Clear interface before you can open an ISDB-S single transport stream file from the Recorder/Player. ISDB-S single transport stream files must have **.isdb1** extensions. |
| Filtering | NULL-packet filtering (from the LIF Status **Configure SPI-CLEAR LIF** dialog) and PID filtering (from the Recorder/Player's **PID Filter Setup** dialog) are not permitted when using the SPI-Clear interface. |
| Discontinuity indicator | The **Set Discontinuity Indicator** feature (from the Recorder/Player's **File** menu) is not permitted with ISDB-S single transport stream files. |

This example illustrates how to configure the SPI-Clear interface for sending or receiving ISDB-S single transport streams.



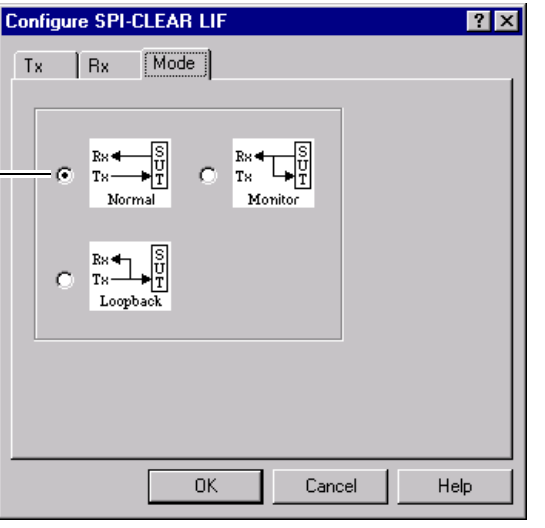

1. **Select the** SPI-CLEAR **interface.**




(2) **If you are transmitting an ISDB-S single transport stream, set transmitter options.**

Packet size is always fixed at 204 bytes.




(3) **Select the mode to use from the Mode tab.**

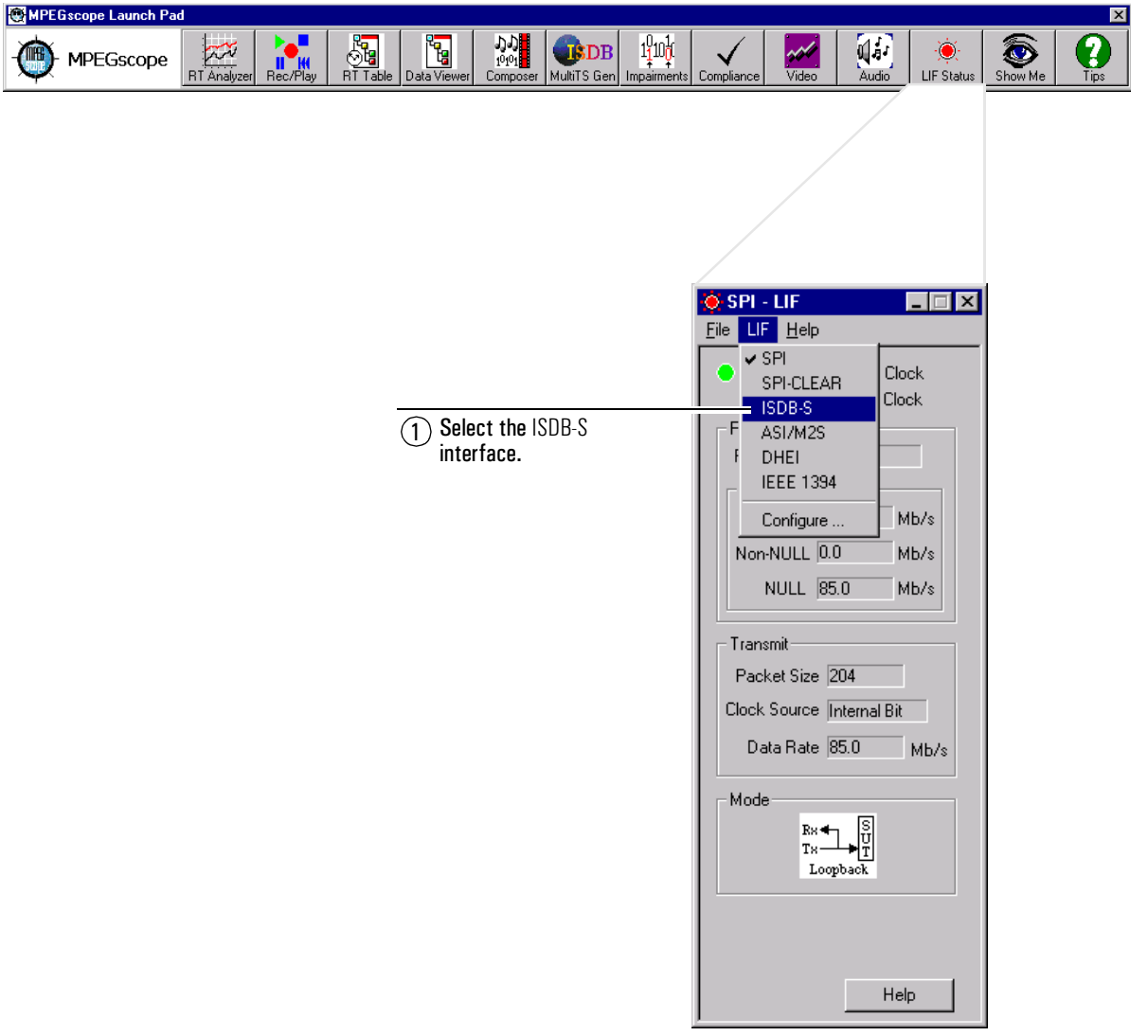
There are no user-configurable **Rx** options when testing with the SPI-Clear interface, since packet size is set to 204 bytes and NULL packet filtering is not allowed.

## Configuring the interface for multiple transport streams

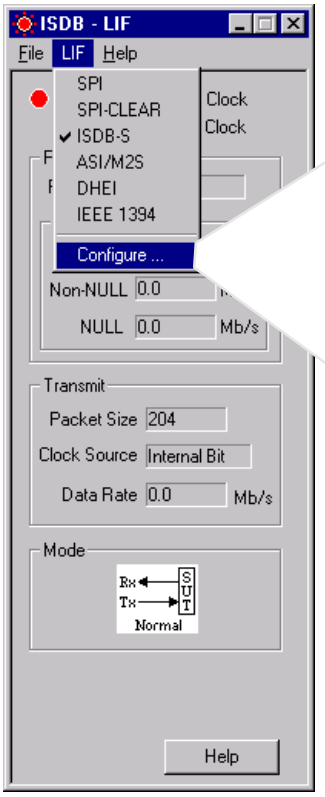
The ISDB-S interface is used when you send or receive ISDB-S multiple transport streams with the Recorder/Player. It is similar to the SPI interface except as follows:

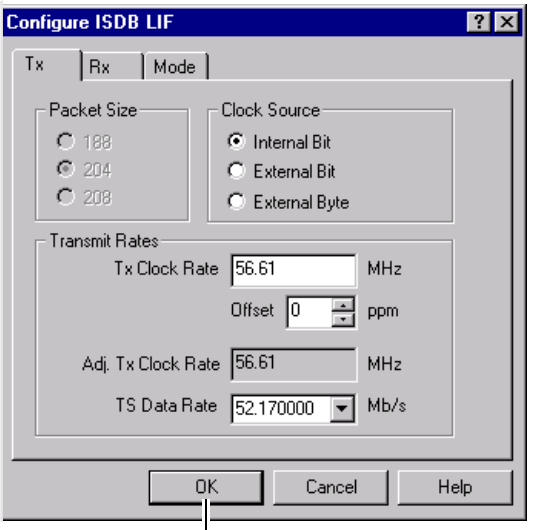
| | |
|---|---|
| Packet size | The MPEGscope ISDB-S interface suppports only 204-byte packets. |
| Packet trailer | The ISDB-S interface transmits the entire 204 bytes without modifying the 16-byte trailer. In contrast, the SPI interface pads the 16-byte trailer of a 204-byte packet with zeros. |
| Framing | With the ISDB-S interface, DVALID and PSYNC pin functionality has changed. The DVALID pin now outputs an FS (frame sync) pulse every 48 packets, and the PSYNC pin outputs an SF (super-frame sync) pulse every 384 packets. |
| Opening files | You must select the ISDB-S interface before you can open an ISDB-S multiple transport stream file from the Recorder/Player. ISDB-S multiple transport stream files must have **.isdbm** extensions. |
| Analyzing files | You cannot analyze ISDB-S multiple transport stream files from the Protocol Data Viewer. |
| Filtering | NULL-packet filtering (from the LIF Status **Configure SPI-CLEAR LIF** dialog) and PID filtering (from the Recorder/Player's **PID Filter Setup** dialog) are not permitted when using the ISDB-S interface. |
| Discontinuity indicator | The **Set Discontinuity Indicator** feature (from the Recorder/Player's **File** menu) is not permitted with multiple transport stream files. |

This example illustrates how to configure the ISDB-S interface for sending or receiving ISDB-S single transport streams.



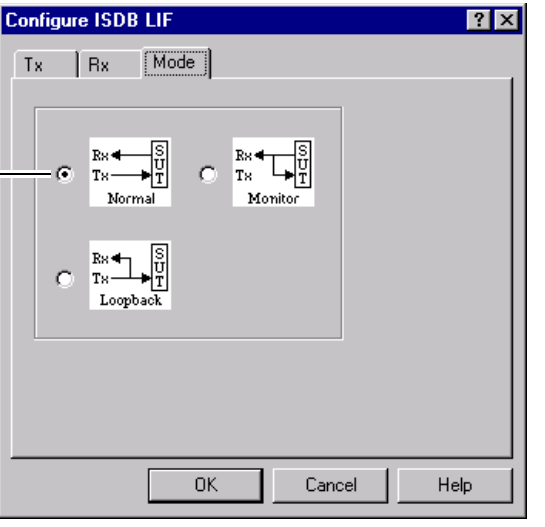

1. Select the ISDB-S interface.




2 **If you want to transmit an ISDB-S multiple transport stream, set transmitter options.**

Packet size is always fixed at 204 bytes.




3 **Select the mode to use from the Mode tab.**

There are no user-configurable **Rx** options when testing with the ISDB-S interface, since packet size is set to 204 bytes and NULL packet filtering is not allowed.

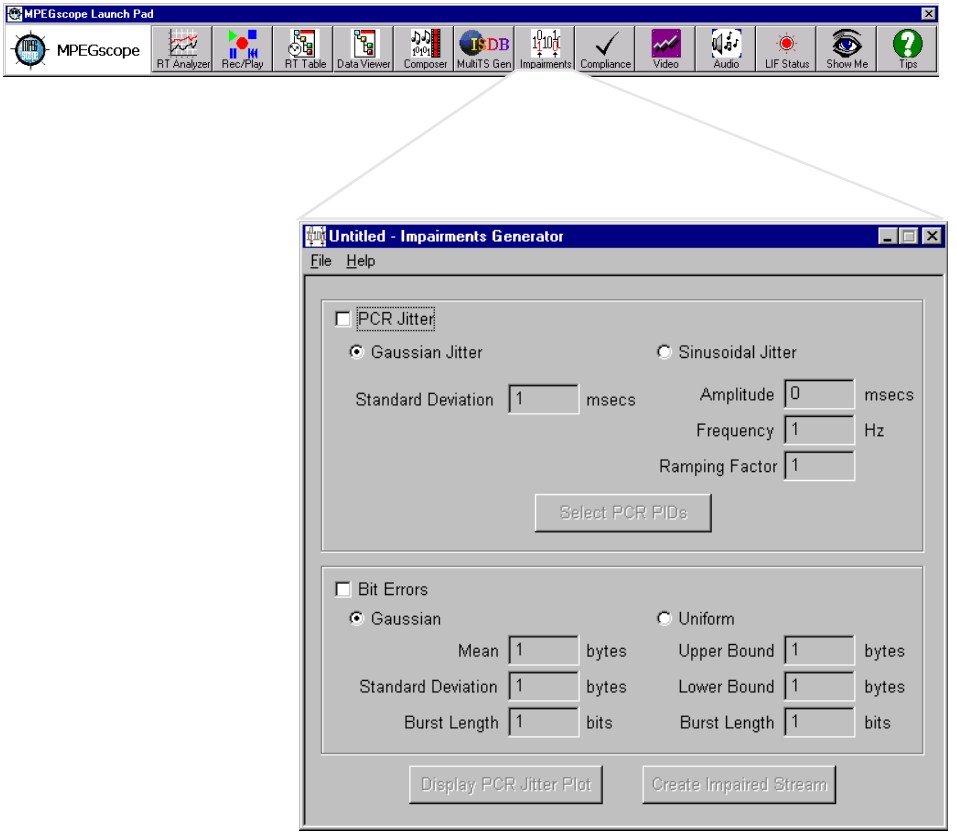
# 9

# Using the Impairments Generator

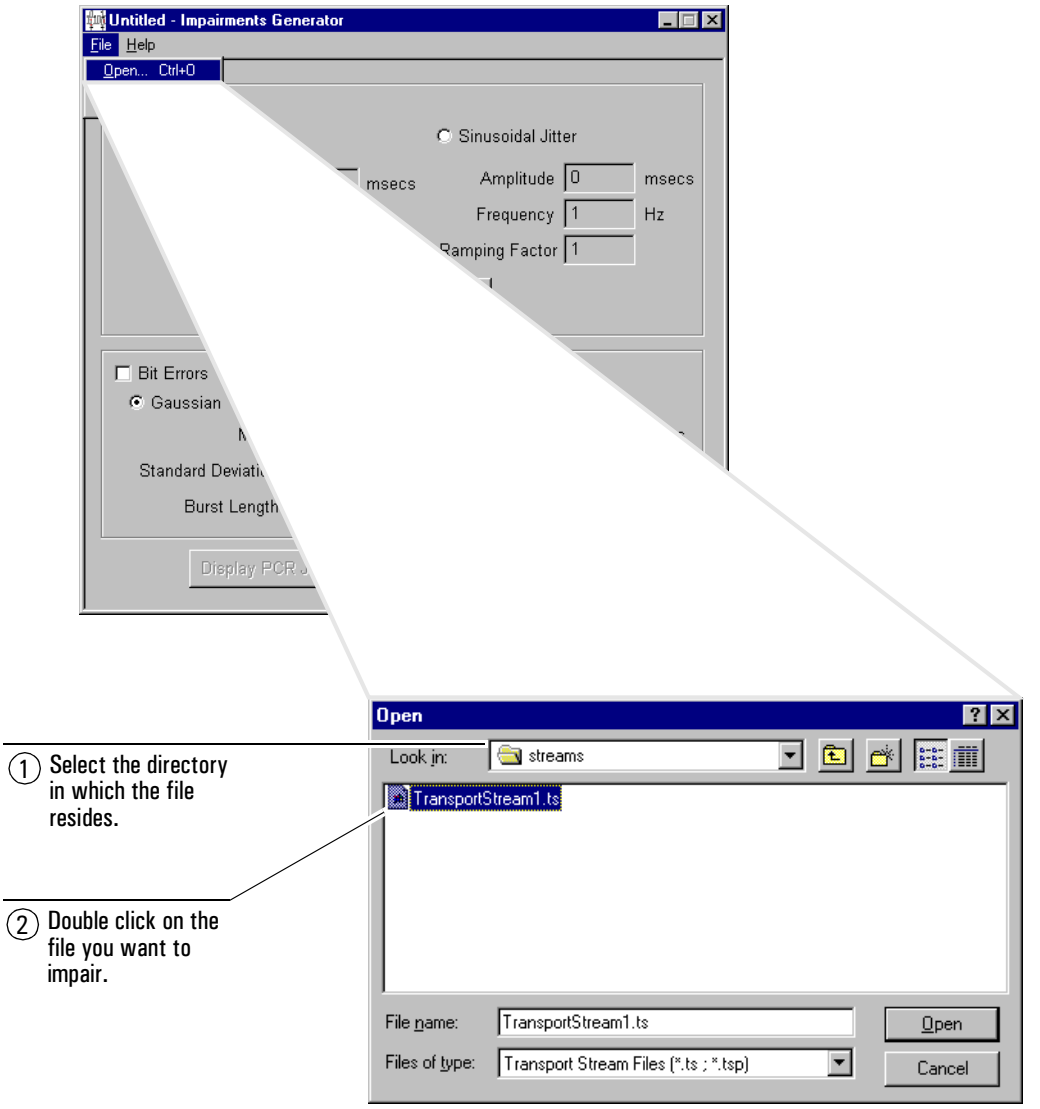
# Impairing a transport stream


This example illustrates how to impair a transport stream with PCR jitter and bit errors.

## Step 1: Start the Impairments Generator

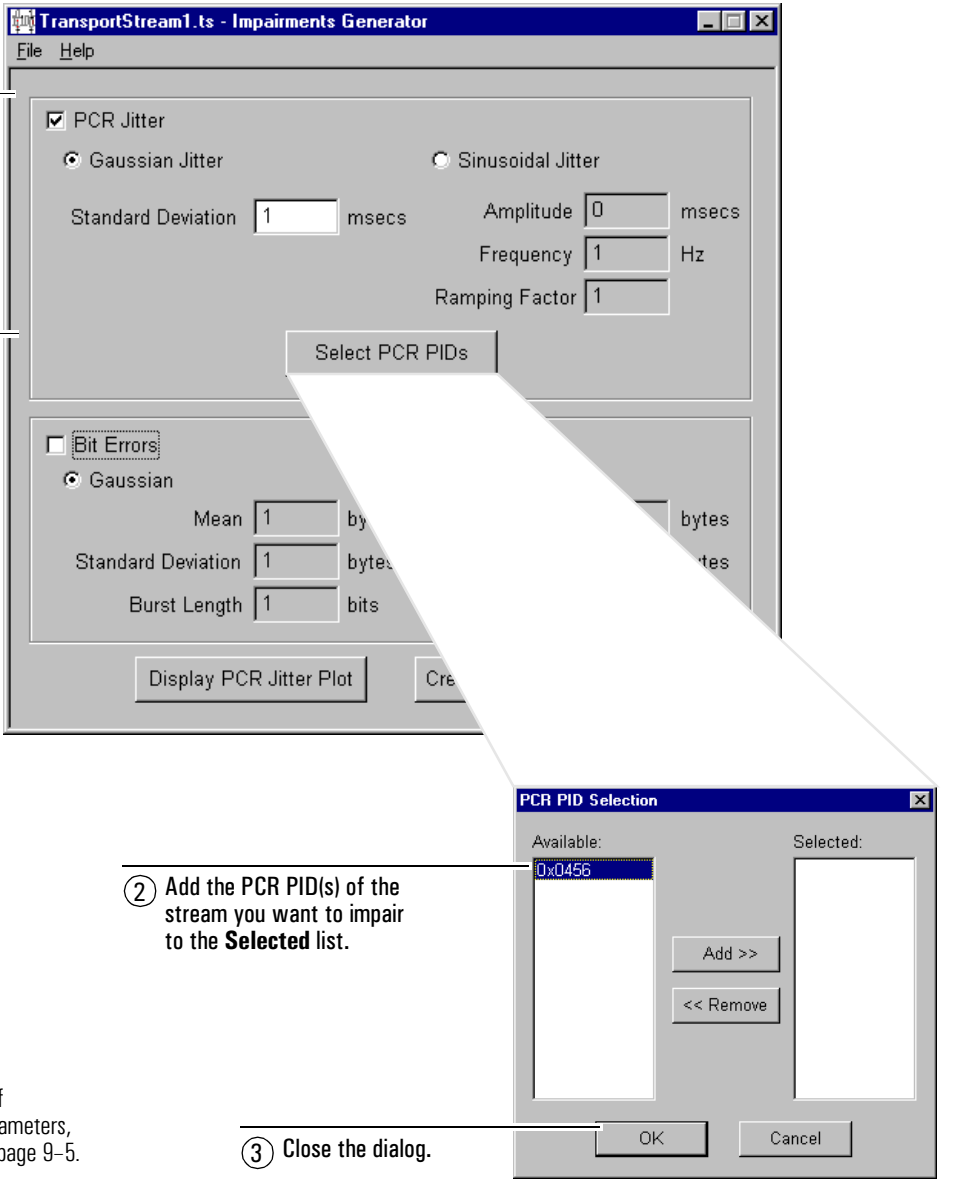


## Step 2: Open a transport stream file




1. Select the directory in which the file resides.
2. Double click on the file you want to impair.

## Step 3: Select impairments

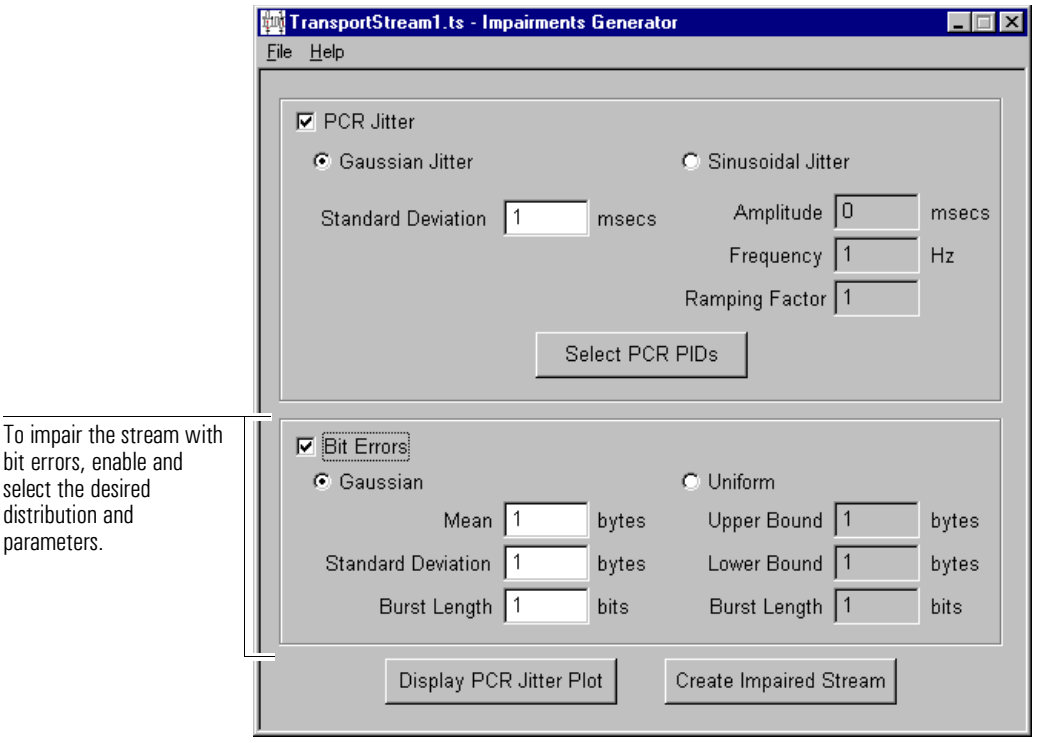



1. To impair the stream with PCR jitter, enable and select the desired distribution and parameters.

2. Add the PCR PID(s) of the stream you want to impair to the **Selected** list.

3. Close the dialog.

For an explanation of distributions and parameters, refer to the table on page 9–5.



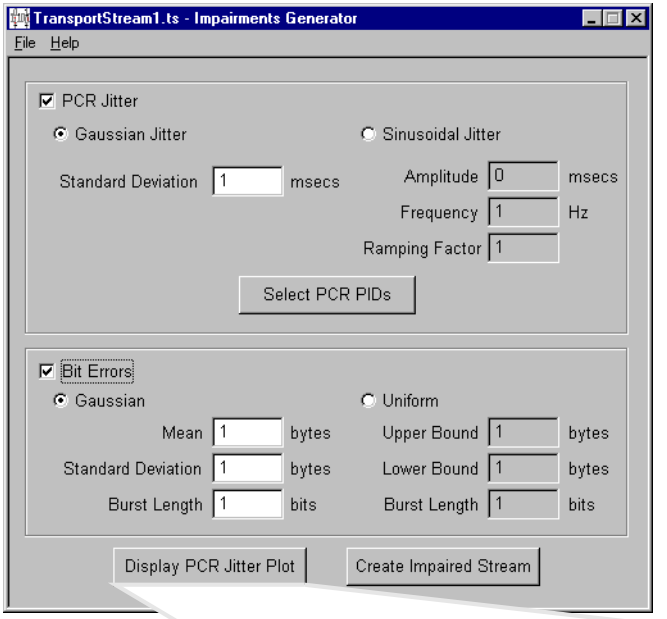

To impair the stream with bit errors, enable and select the desired distribution and parameters.

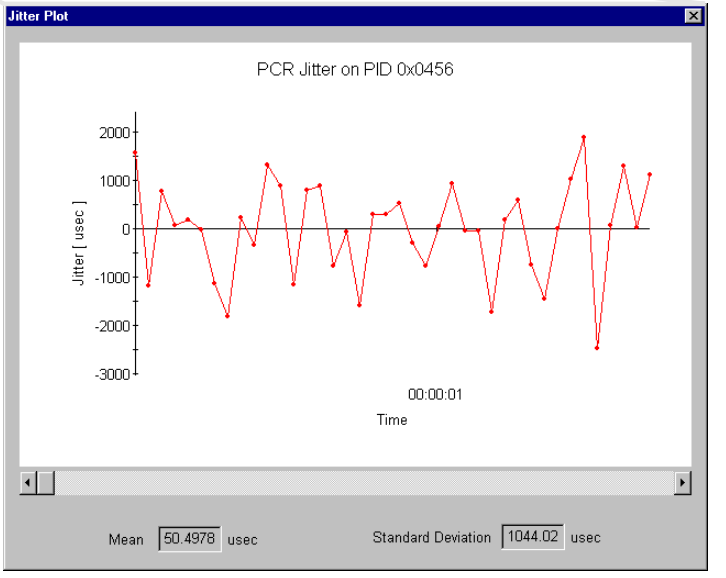
## Impairments

| | | | | |
|---|---|---|---|---|
| **PCR Jitter** | **Gaussian Jitter:** | Introduces PCR jitter that follows a normal distribution. | **Standard Deviation:** | Standard deviation of the Gaussian distribution, specified in milliseconds. Floating point values are permitted.<br>**Note**: By definition, the PCR jitter mean is 0. |
| | **Sinusoidal Jitter:** | Introduces PCR jitter that follows a sinusoidal waveform (i.e., that has the form of a sine wave) | **Amplitude:** | Amplitude of the sine function, specified in milliseconds. Floating point values are permitted. |
| | | | **Frequency:** | Frequency of the sine function, specified in Hertz. Floating point values are permitted. |

| | | | | |
|---|---|---|---|---|
| | | | **Ramping Factor:** | Factor by which the amplitude of the sine function increases in time. Floating point values are permitted. |
| **Bit Errors** | **Gaussian:** | Introduces bit errors that follow a Gaussian distribution. | **Mean:** | Mean number of bytes between errors in the Gaussian distribution. |
| | | | **Standard Deviation:** | Standard deviation in the number of bytes between errors in the Gaussian distribution. |
| | | | **Burst Length:** | Number of consecutive bits to be errored. |
| | **Uniform:** | Introduces a uniform probability of bit errors, defined by upper and lower limits. | **Upper Bound:** | Maximum number of bytes between errors in the uniform distribution. |
| | | | **Lower Bound:** | Minimum number of bytes between errors in the uniform distribution. |
| | | | **Burst Length:** | Number of consecutive bits to be errored. |

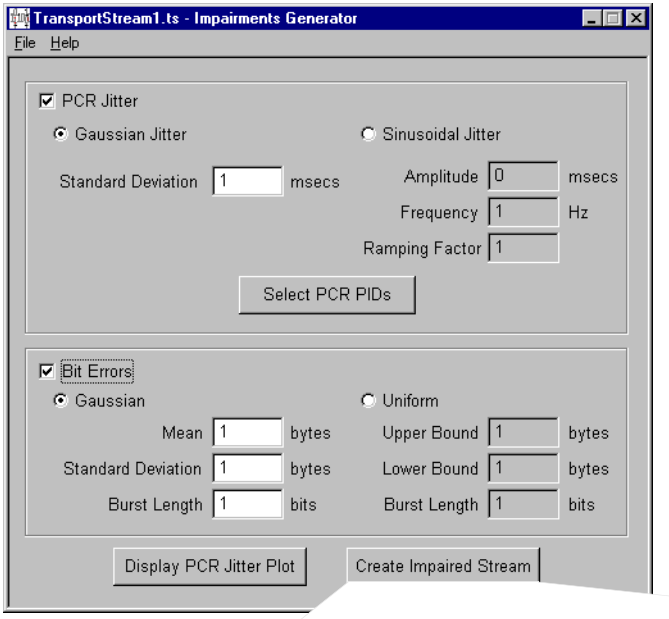
# Step 4: Display PCR jitter—before impairments

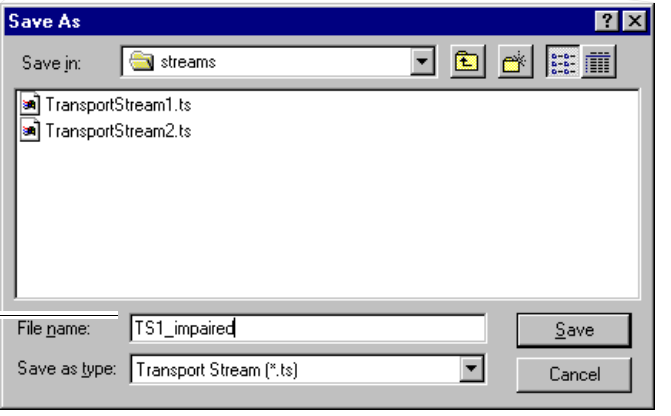



You can check the amount of PCR jitter on the stream before introducing impairments.

## Step 5: Create an impaired transport stream

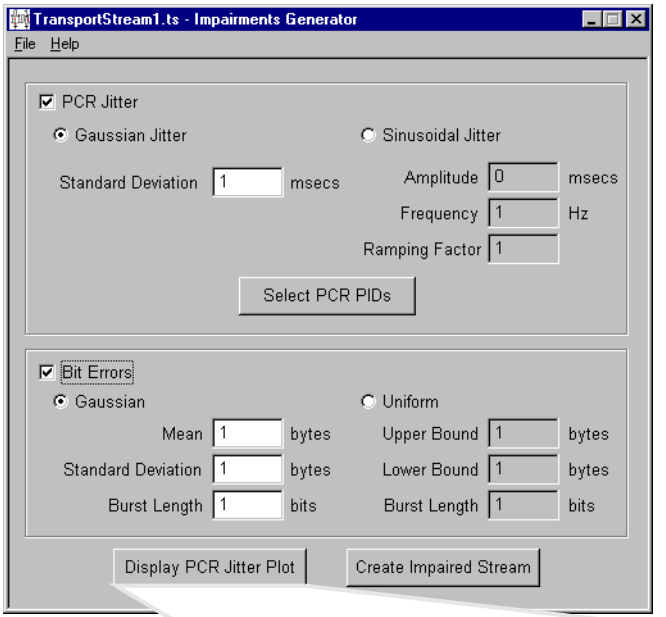


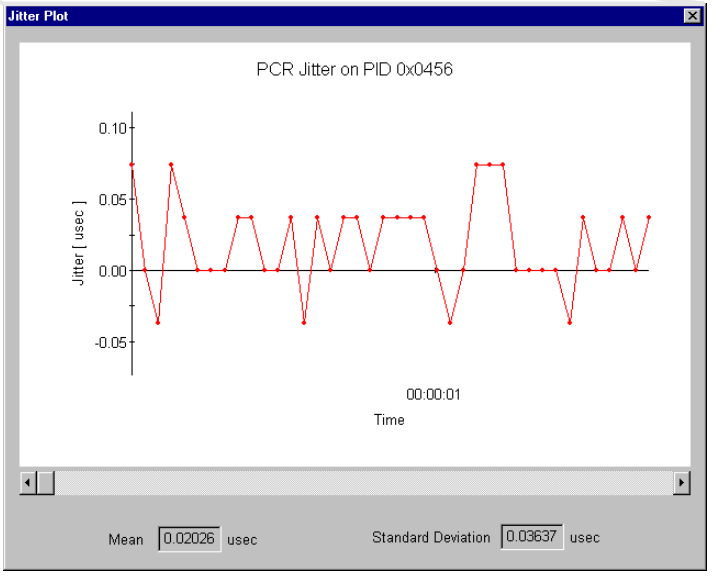




Enter a file name for the impaired stream, or overwrite the current stream.

Depending on the size of the file and the number of impairments, this process may take a few minutes.

## Step 6: Display PCR jitter—after impairments




You can check the amount of PCR jitter on the stream after introducing impairments.

# 10

# Using the Compliance Verifier

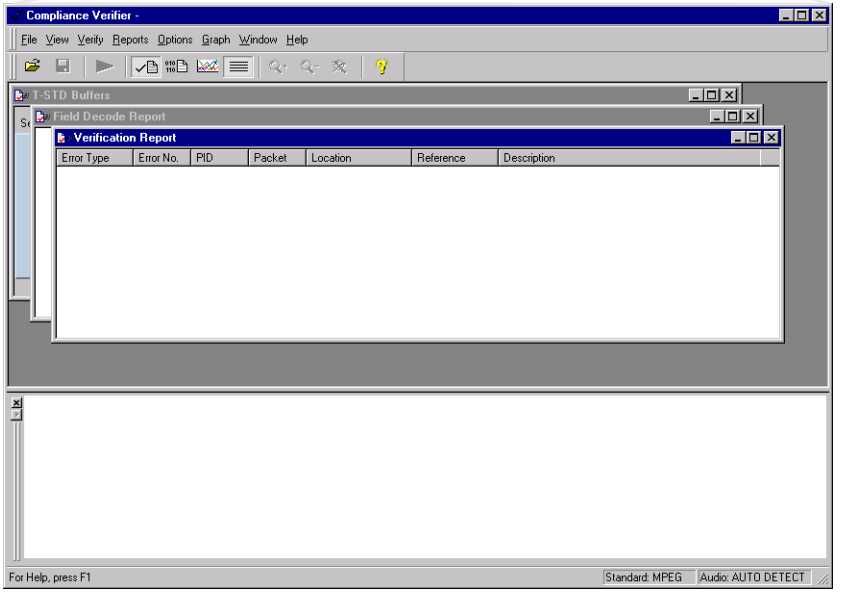
# Verifying a transport stream


This example requires the purchase of E6311A, E6312A, or E6313A. It shows how to use the Compliance Verifier to analyze a transport stream and its components, view reports, and plot the dynamic behavior of the T–STD buffers.

## Step 1: Start the Compliance Verifier

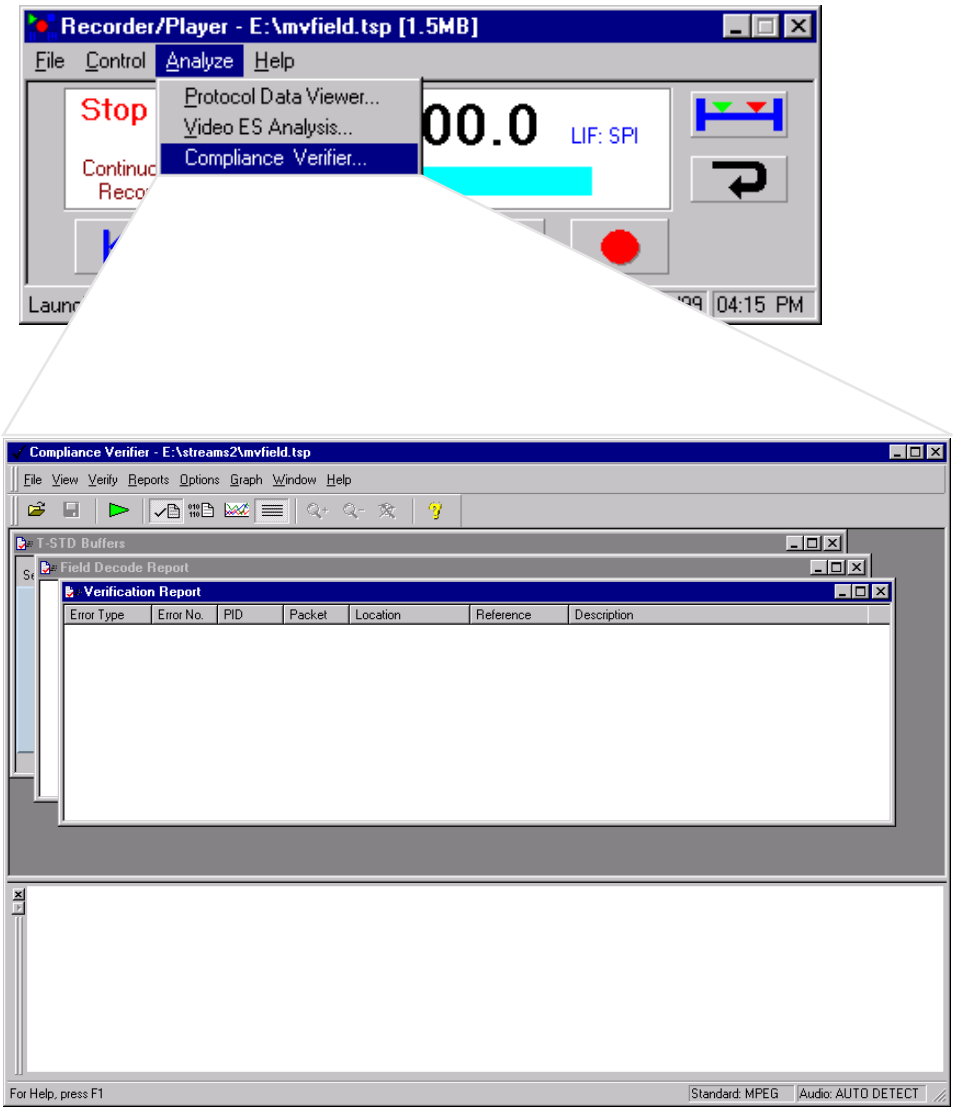
There are two ways to start the Compliance Verifier.

From the Launch Pad:


Use this method when you want to open and analyze a file you have saved to disk.

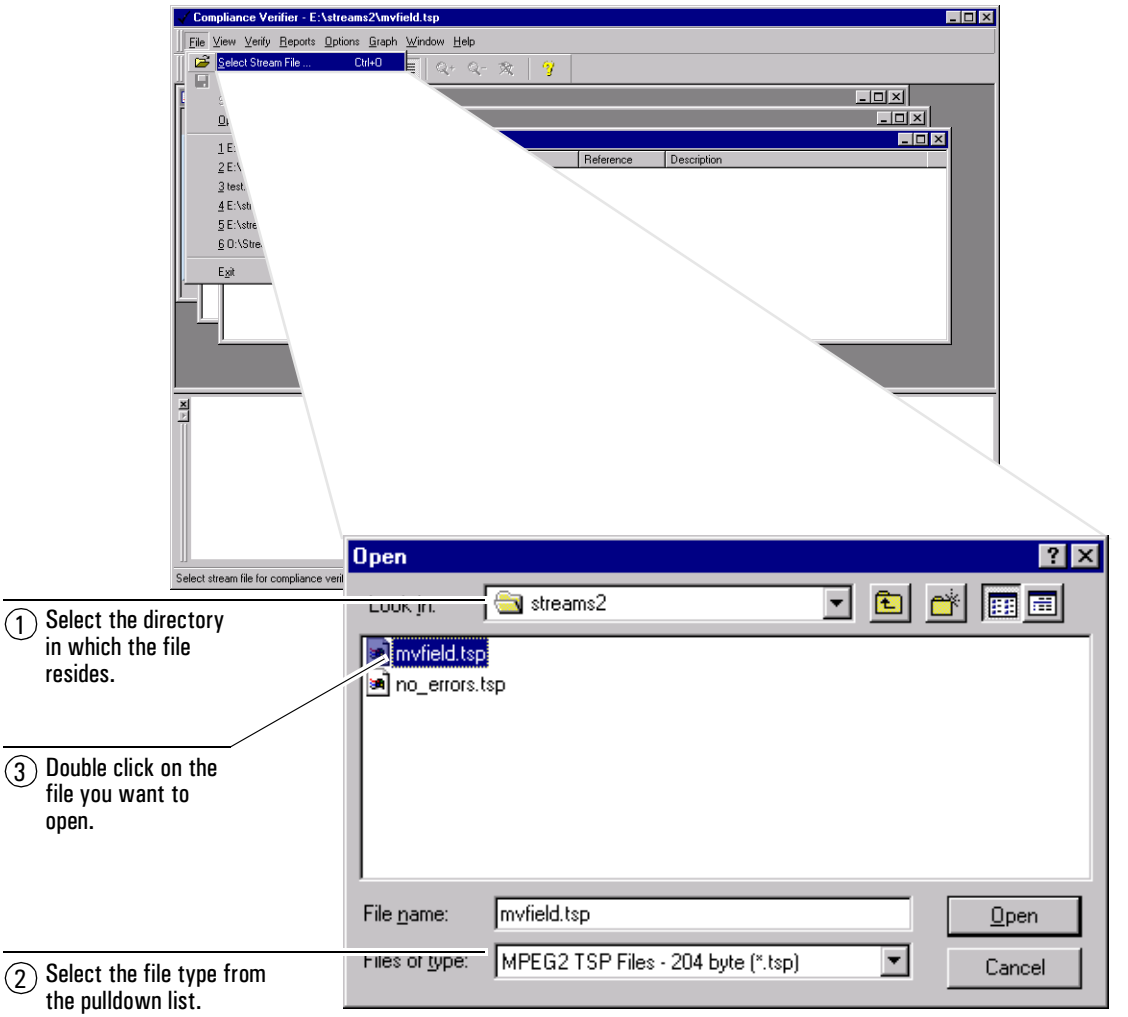
From the Recorder/Player:




Use this method when you want to analyze a file you have opened or recorded from the Recorder/Player.

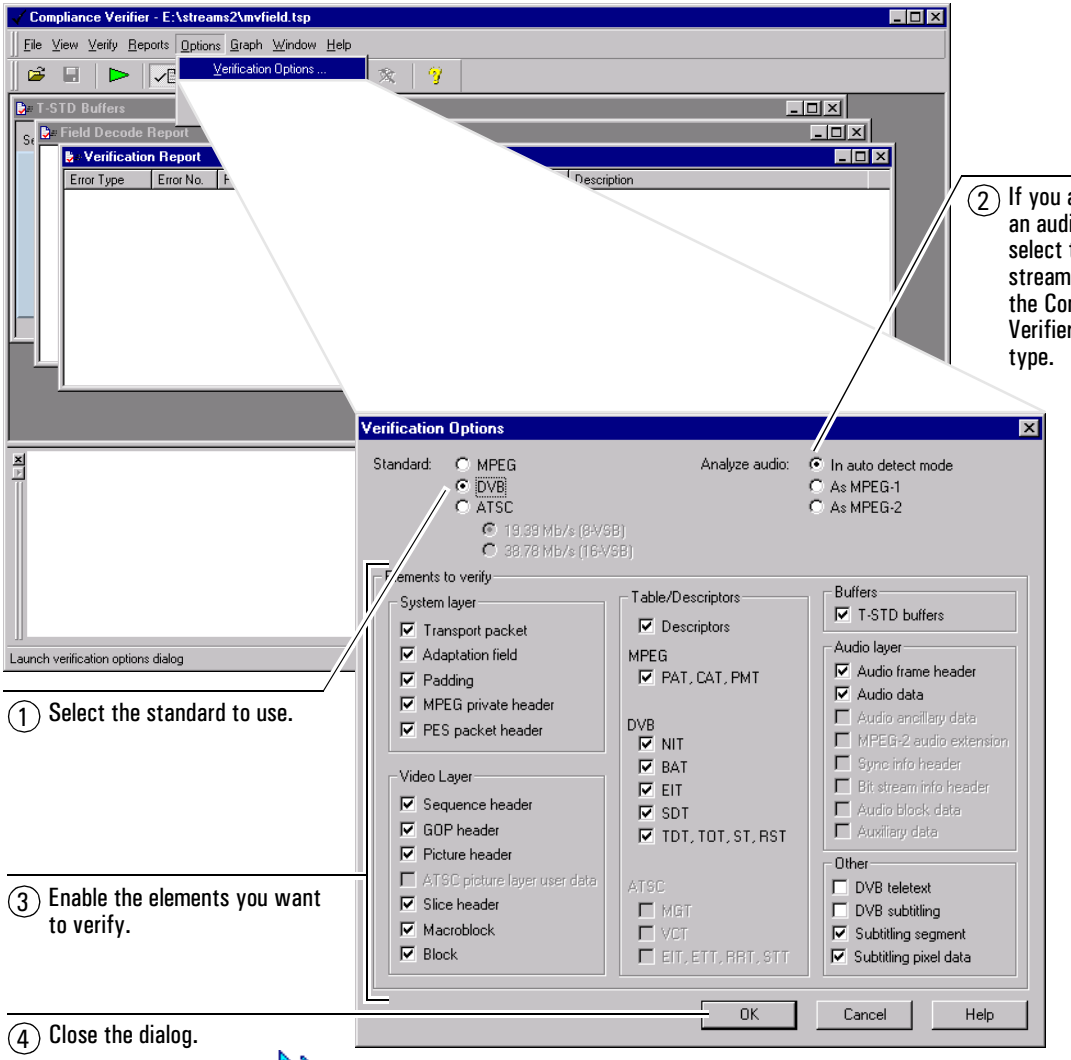
## Step 2: Open the input file

Complete this step if you have opened the Protocol Data Viewer from the Launch Pad.




If you are opening an MPEG-2 audio file *and* its accompanying extension file, the Compliance Verifier will prompt you to open the extension file name after you open the audio file.

## Step 3: Select **Verification** options




(2) If you are verifying an audio stream, select the audio stream type or let the Compliance Verifier detect the type.

(1) Select the standard to use.

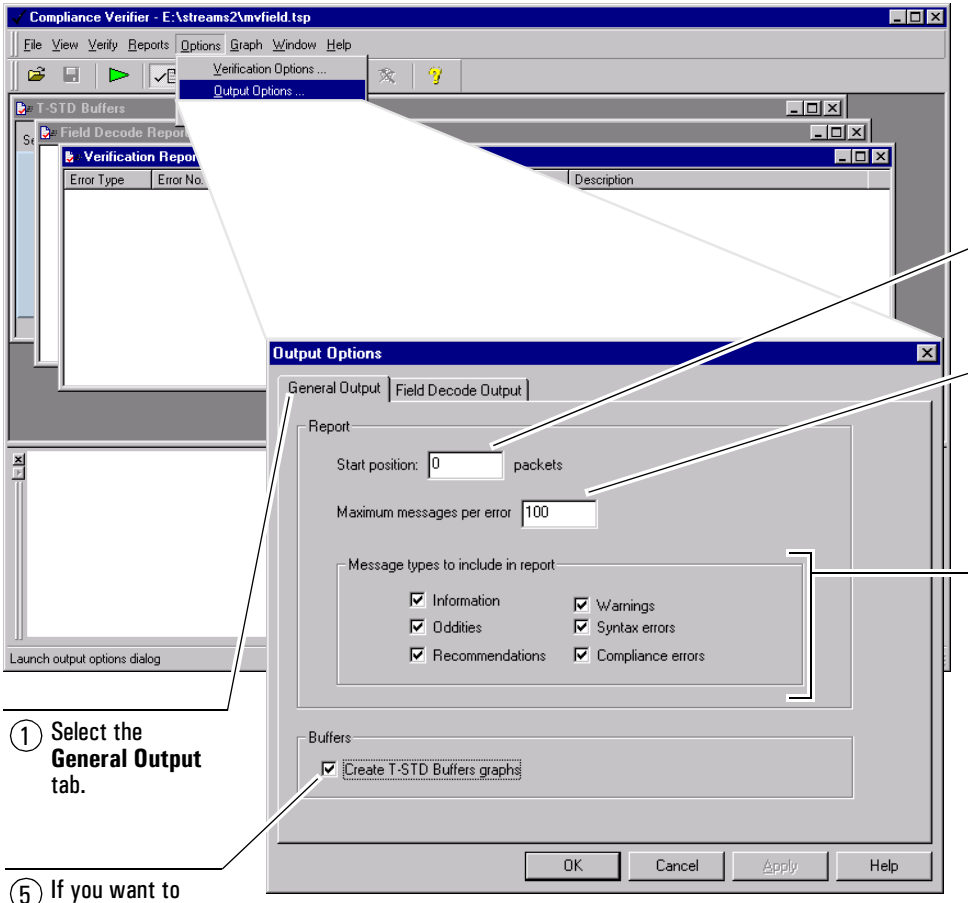
(3) Enable the elements you want to verify.

(4) Close the dialog.

The Compliance Verifier checks the ATSC sequence header, sequence extension header, display extension header, picture header, and user_data only.

If you forget which elements are multiplexed in a specific transport stream file, you can open the file from the Protocol Data Viewer then create a **TS Hierarchy** view to see the contents.

## Step 4: Select **Output** options



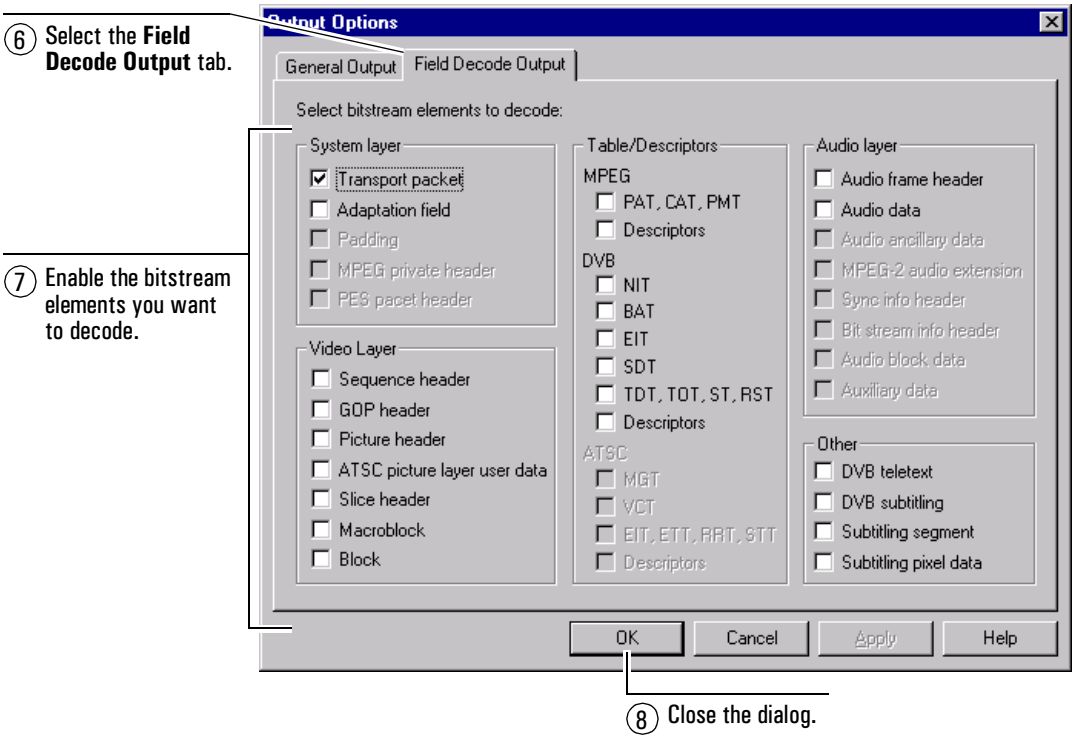

(2) Enter the packet number at which to start verifying.

(3) Enter the maximum number of messages per error to include in the report.

(4) Select which message types to include in the report.

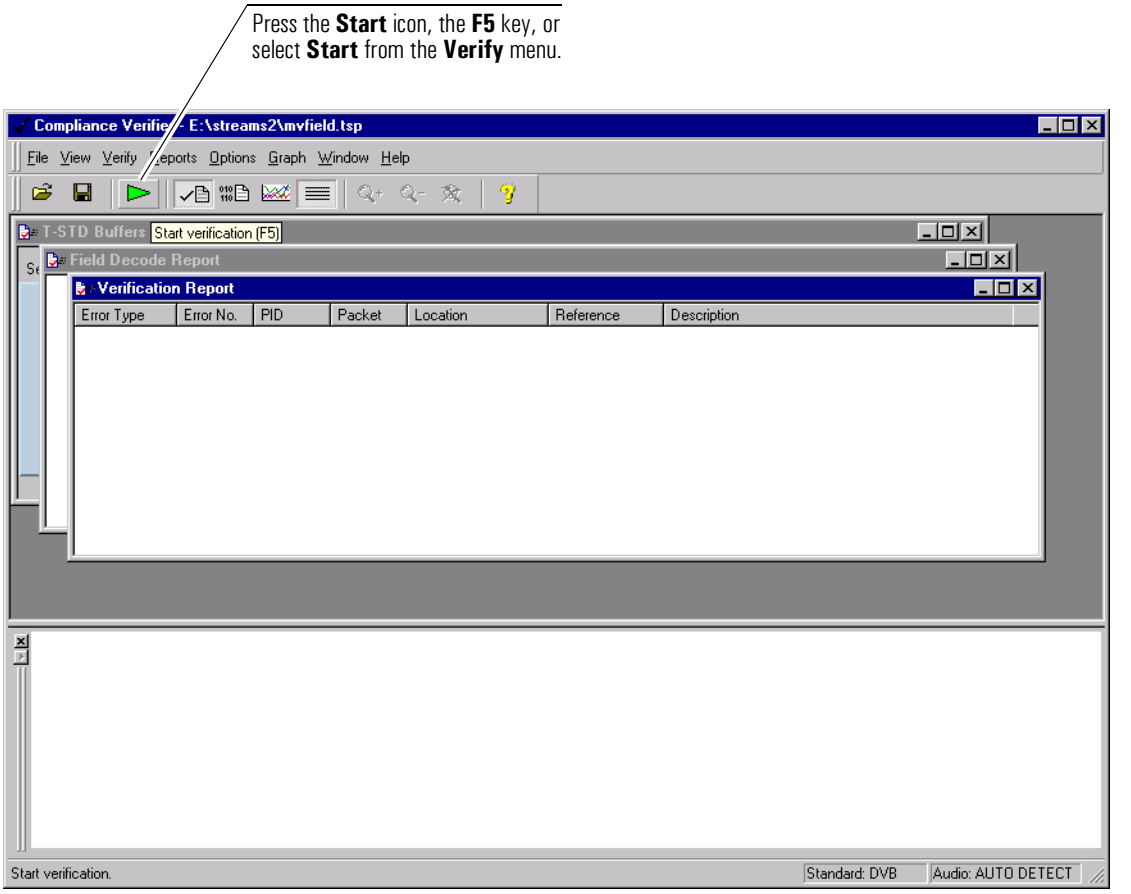
(1) Select the **General Output** tab.

(5) If you want to verify the T-STD buffers, ensure this option is selected.

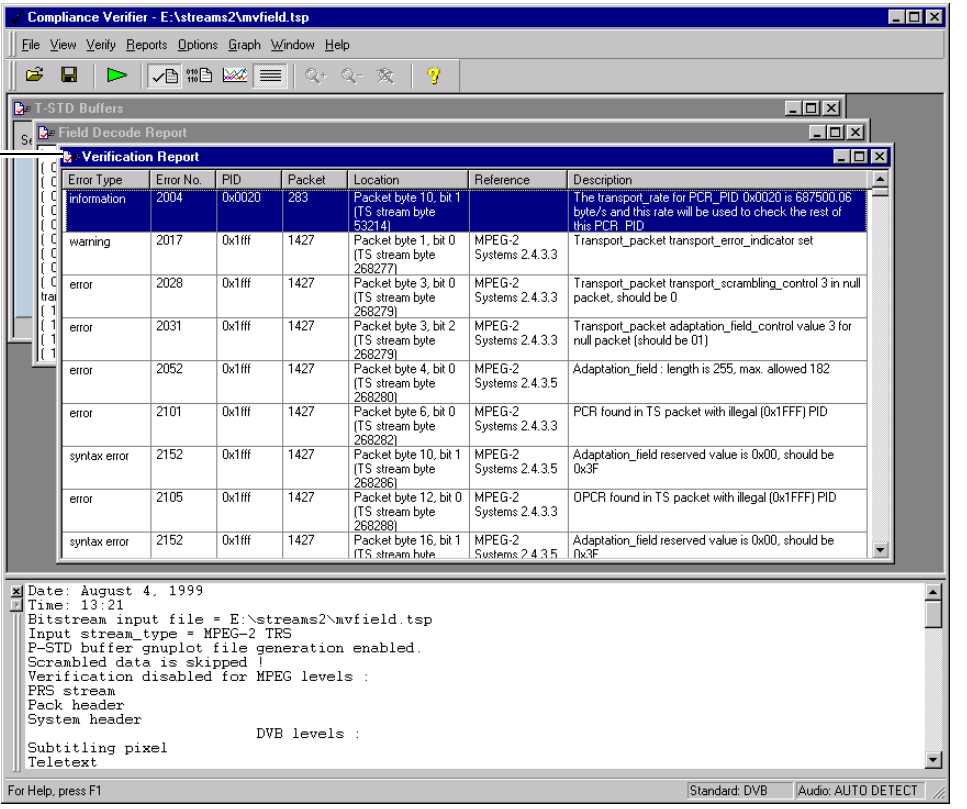



Depending on the frequency at which the selected information occurs in the stream, enabling bitstream elements may significantly slow down the verification process. You can speed up processing time by shortening the stream before verification.

## Step 5: Verify the stream



## Step 6: View reports




The **Verification Report** describes problems encountered and gives a reference to the part of the standard that was violated.

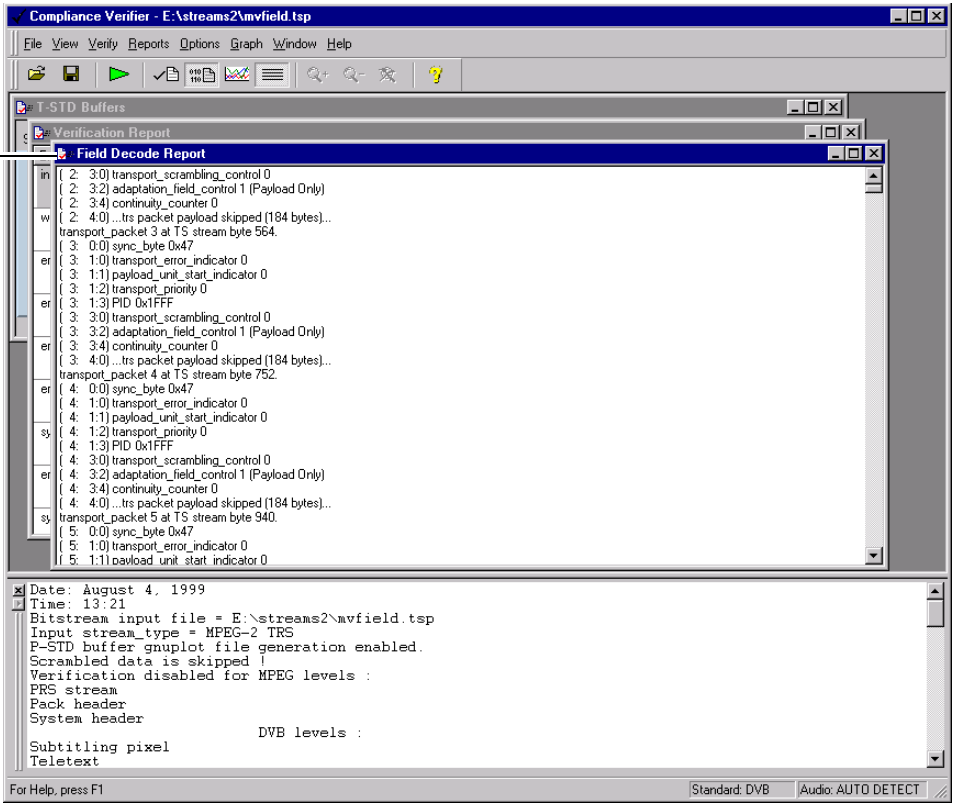
For a detailed description of how to read report messages. refer to "Sample report messages", page 10–17.

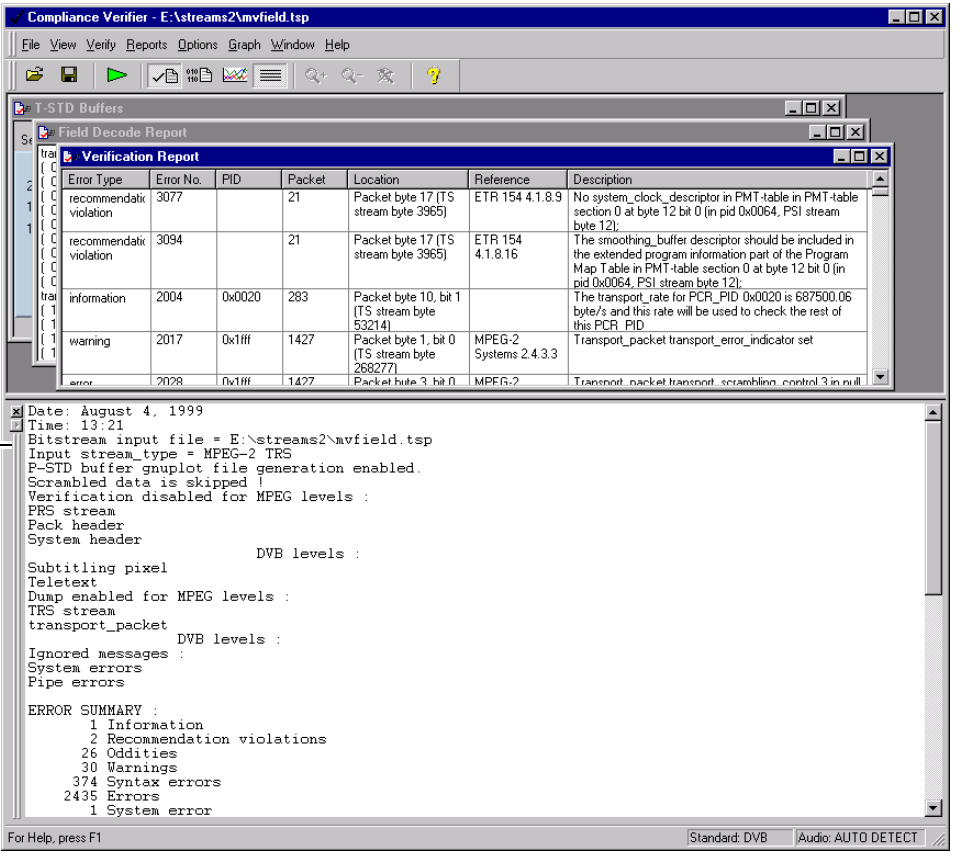
If you are verifying a file with a **.ts** or **.tsp** extension, you can double click on any entry in the **Verification Report** to open a **PDU Details** view for the packet in question from the Protocol Data Viewer. The error will be highlighted in red in the hexadecimal display (the middle pane) of the **PDU Details** view. If the error is an SI section or PES error, the PDU Details view will open at the SI Sections or PES Packets level, respectively. If the error is of any other type, the PDU Details view will open at the Transport Stream level.

If you check the **Timeline** at the bottom of the Protocol Data Viewer's **Substream View**, you will see the range of packets included in the **Substream View**. You can open more **PDU Details** views from the **Verification Report** for other packets within this range by double clicking on the errored packet from the **Verification Report**. (Check the packet number in the fourth column.) However, if you want to open a **PDU Details** view for a packet *outside* this range, you will first have to close the Protocol Data Viewer application. After the application is closed, you can double click on the new errored packet from the **Verification Report**. The Protocol Data Viewer will then launch a new **Substream View** with an appropriate packet range and will display a **PDU Details** view for the errored packet.

For more information on the **PDU Details** view, refer to "Check the PDU Details view", page 6–11.

The **Field Decode Report** displays a decode of the parameters and flags of specified bitstream elements. It shows field names and values as well as the location in the stream (element number, byte number, and bit number).

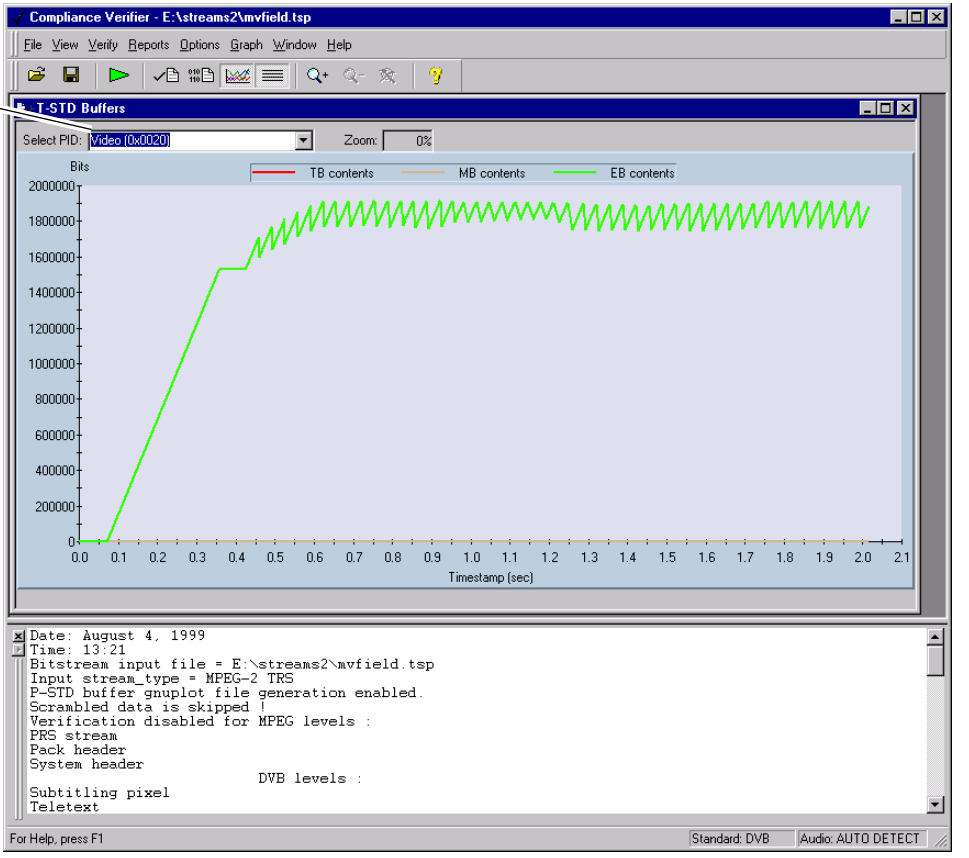




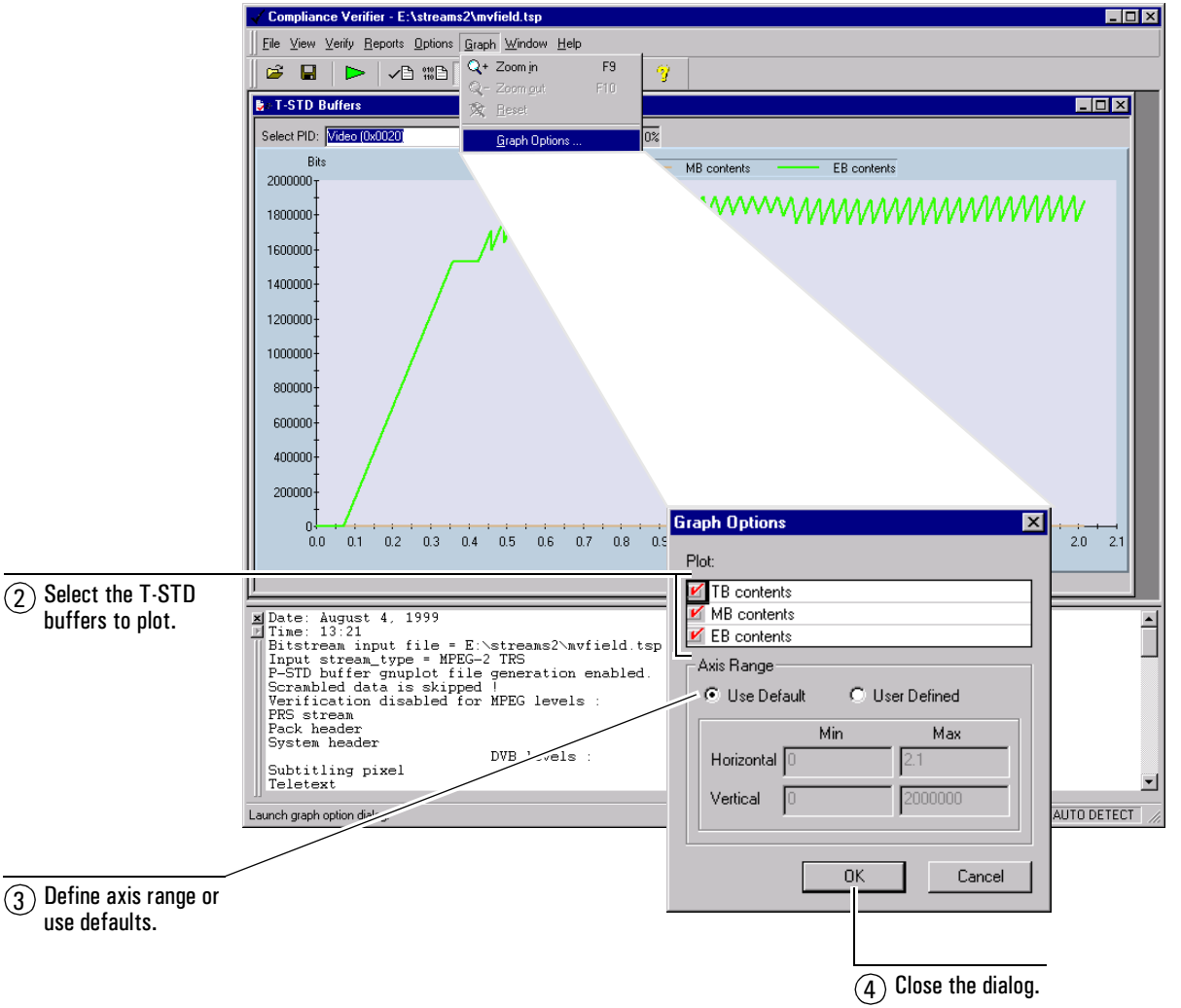

The **Summary Report** provides a general and detailed error summary as well as information about the time and date the stream was verified, the file name and stream type, and the options enabled or disabled.

## Step 7: **View T-STD Buffers** graphs

The **T-STD Buffers** dialog displays the contents of the T-STD (Transport Stream System Target Decoder) buffers plotted over time for the selected PID.




1. Select the PID of the video, audio or PSI stream for which you want to view buffer data.

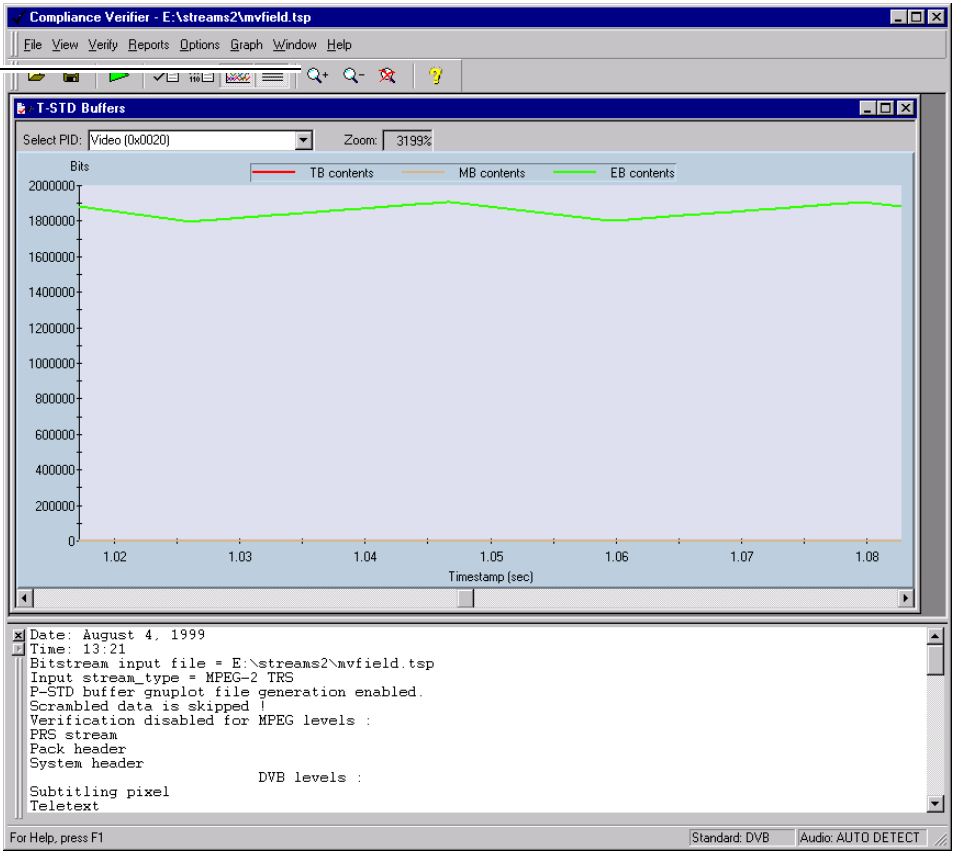



(2) Select the T-STD buffers to plot.

(3) Define axis range or use defaults.

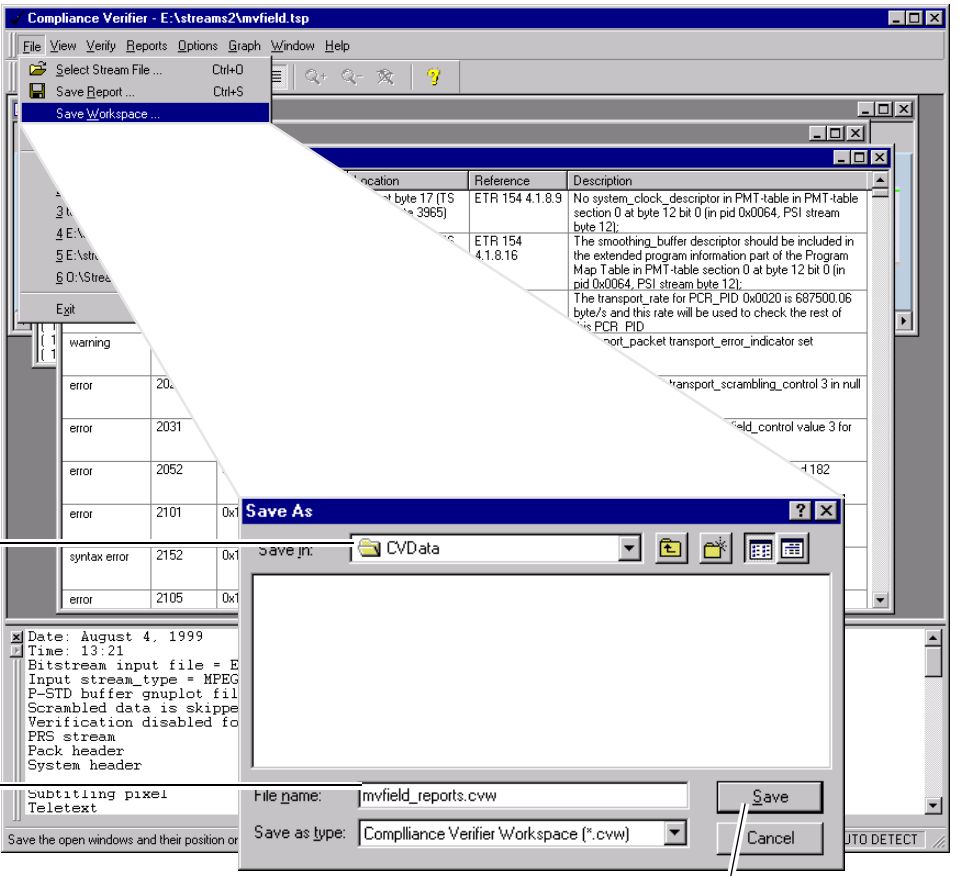
(4) Close the dialog.

⑤ Use the **Zoom** button to enlarge a portion of the graph.



# Step 8: Save reports

You can save all the reports from a test into a workspace file, then restore it to view test results at a later date.

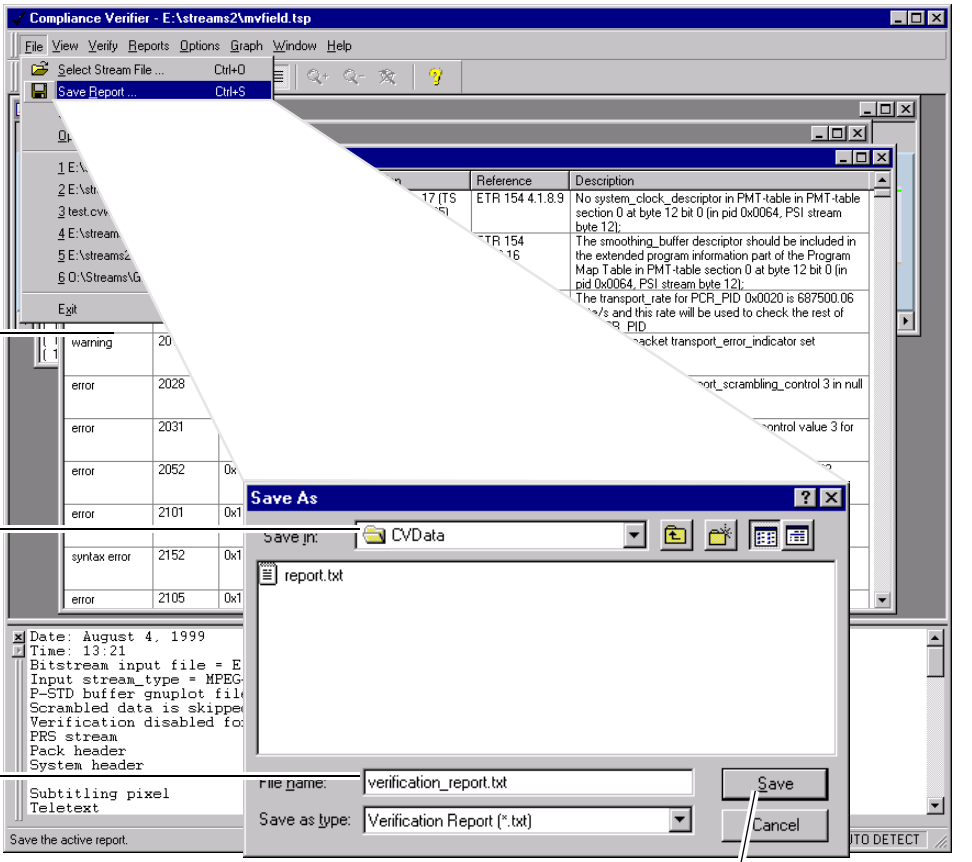



① **Select the directory in which to save the file.**

② **Enter the file name.**
Workspaces are saved as **.cvw** files.

③ Save the workspace.

You can also save any individual report to a text file.



## Sample report messages

**Error numbers**

Each error message logged to the report file has an accompanying number.

| Number range | Description |
|---|---|
| 0 to 999 | System errors |
| 1100 to 1949 | MPEG-1 checks |
| 2300 to 2999 | MPEG-2 checks |
| 3000 to 4199 | DVB checks |
| 4500 to 4649 | ATSC checks |

**Error types**

Errors are classified into eight main types. You can apply filters to ignore one or more of these error types. When you apply a filter, the MPEG/DVB Compliance Verifier will not log the error messages to the report file.

| Error types | Description |
|---|---|
| Information | Describe a notable event in the stream. |
| Oddity | Signal an odd inconsistency in the stream, such as a superfluous or useless setting, and do not necessarily indicate that an error has occurred. |
| Recommendation Violation | Issued when a specification recommendation is not respected. |
| Warning | Signal potential problems with the reliability of the bitstream and indicate a potential cause of errors. |
| Syntax Error | Indicate syntax violations of the protocol standards and are generated during parsing. They cause the Compliance Verifier to enter **Recover** mode as it attempts to resynchronize with the bitstream. |
| Error | A semantic error, or type of error other than syntactic. |
| System and Pipe Errors | Report serious problems with the system on which the Compliance Verifier is running, such as memory problems, file creation problems, internal program problems, etc. |

## Example of Report File Messages

The following example shows report file messages from a Compliance Verifier analysis and provides a more detailed explanation of each problem. To confirm some of the error messages, the file was also analyzed in the Protocol Data Viewer and relevant sections copied from the **PDU Details** view.

### Information message

[MPEG-2] INFORMATION 2015 (ref. MPEG-2 Systems 2.4.3.3) : Transport_packet sync_byte (0x47) emulation at header bit 22 at byte 188 bit 0 of t_packet 94 (pid 0x0029; TS stream byte 17672).

[MPEG-2] INFORMATION 2015 (ref. MPEG-2 Systems 2.4.3.3) : Transport_packet sync_byte (0x47) emulation at header bit 22 at byte 188 bit 0 of t_packet 131 (pid 0x0029; TS stream byte 24628).

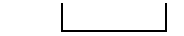
[MPEG-2] INFORMATION 2015 (ref. MPEG-2 Systems 2.4.3.3) : Transport_packet sync_byte (0x47) emulation at header bit 22 at byte 188 bit 0 of t_packet 295 (pid 0x0029; TS stream byte 55460).

[MPEG-2] INFORMATION 2015 (ref. MPEG-2 Systems 2.4.3.3) : Transport_packet sync_byte (0x47) emulation at header bit 22 at byte 188 bit 0 of t_packet 301 (pid 0x0029; TS stream byte 56588).

[MPEG-2] INFORMATION 2015 (ref. MPEG-2 Systems 2.4.3.3) : Transport_packet sync_byte (0x47) emulation at header bit 22 at byte 188 bit 0 of t_packet 312 (pid 0x0029; TS stream byte 58656). MPEG-2 INFORMATION 2015 will no longer be reported !

These information messages are caused by a pattern in the transport stream header of four packets that matches the sync byte sequence **0100 0111**.

For example, the packet in the first error has a header value of **0x 47 00 29 1C**. Converted to binary, the value is **0100 0111 0000 0000 0010 1001 0001 1100**.



Sync byte pattern appears within the sequence.

### Recommendation

[[DVB] RECOMMENDATION VIOLATION 3077 (ref. ETR 154 4.1.8.9) : No system_clock descriptor in PMT-table in PMT-table section 0 at byte 12 bit 0 (in pid $0020, PSI stream byte 12); byte 17 of t_packet 438 (TS stream byte 82361).

[DVB] RECOMMENDATION VIOLATION 3094 (ref. ETR 154 4.1.8.16) : The smoothing_buffer descriptor should be included in the extended program information part of the Program Map Table in PMT-table section 0 at byte 12 bit 0 (in pid $0020, PSI stream byte 12); byte 17 of t_packet 438 (TS stream byte 82361).

[DVB] RECOMMENDATION VIOLATION 3077 (ref. ETR 154 4.1.8.9) : No system_clock descriptor in PMT-table in PMT-table section 1 at byte 12 bit 0 (in pid $0020, PSI stream byte 38); byte 43 of t_packet 438 (TS stream byte 82387).

[DVB] RECOMMENDATION VIOLATION 3094 (ref. ETR 154 4.1.8.16) : The smoothing_buffer descriptor should be included in the extended program information part of the Program Map Table in PMT-table section 1 at byte 12 bit 0 (in pid $0020, PSI stream byte 38); byte 43 of t_packet 438 (TS stream byte 82387).

These messages refer to a recommendation in the DVB specification which explains which descriptors to include in a PSI table. The text below shows the decoded PSI table from TS packet #438, copied from the Protocol Data Viewer's PDU Details view. It confirms there are no descriptors in the table.

```
TSP #438   0:0.016 449 24

Program 1 Program Map Table
Section Syntax Indicator                    1
Version Number                              7
Current/Next Indicator                      Current
PCR PID                                     0x0026
Stream Type                                 0x02
(Video-MPEG2)
   Elementary PID                           0x0026
Stream Type                                 0x04
(Audio-MPEG2)
   Elementary PID                           0x0027
```

**Error message**

[MPEG] SYNTAX ERROR 1504 (ref. MPEG Video 2.4.2.3 | 6.2.2) : Sequence_header_code expected for video sequence (0) at byte 0 bit 0; PES (0xE0) byte 28 (byte 28 of packet 0); byte 40 of t_packet 535 (pid 0x002C; TS stream byte 100620). [Look Ahead : 0x00 0x00 0x01 0x00 (len : 32 bit)]

[MPEG] ERROR 1666 (ref. MPEG Video 2.4.1 | 6.1.1.7) : First picture in GOP has type B, should be an I-picture for video sequence (0) at byte 5 bit 2; PES (0xE0) byte 33 (byte 33 of packet 0); byte 45 of t_packet 535 (pid 0x002C; TS stream byte 100625).

These errors occur because the beginning of the first PES packet in TS packet #535 does not start with the sequence header start code—**00 00 01 B3**, as shown below.

```
TSP #535   0:0.020 092 14

Start Code                                  0x000001
Stream Id                                   0xE0
(Video Stream 0)
Packet Length                               0
PES Scramble Control                        0
(Not Scrambled
```

```
PES Priority                                0
Data Alignment Indicator                    1
Copyright                                   0
Original Indicator                          1
(Original)
PES Header Data Length                      19
PTS/DTS Flags                               0x2
(PTS)
PTS                                         13:26:16:90565555
Payload :00 00 01 00 00 18 AA 1B B8 00 00 01 B5 87 ...
```

No sequence header start code.

The Sequence Header Start code does not appear until TS packet #11006, the start of the next sequence.

```
TSP #11006   0:0.413 334 69

Start Code                                  0x000001
Stream Id                                   0xE0
(Video Stream 0)
Packet Length                               0
PES Scramble Control                        0
(Not Scrambled)
PES Priority                                0
Data Alignment Indicator                    1
Copyright                                   0
Original Indicator                          1
(Original)
PES Header Data Length                      19
PTS/DTS Flags                               0x3
(PTS and DTS)
PTS                                         13:26:17:46565555
DTS                                         13:26:17:34565555
Payload :00 00 01 B3 2C 02 40 23 24 9F 23 ...
```

Sequence header start code.

### Information message

[MPEG-2] INFORMATION 2004 : The transport_rate for PCR_PID 0x02C is 5005952.09 byte/s and this rate will be used to check the rest of this PCR_PID at byte 10 bit 1 of t_packet 1383 (pid 0x002C; TS stream byte 260014).

[MPEG-2] INFORMATION 2004 : The transport_rate for PCR_PID 0x0029 is 5005952.46 byte/s and this rate will be used to check the rest of this PCR_PID at byte 10 bit 1 of t_packet 1724 (pid 0x0029; TS stream byte 324122).

[MPEG-2] INFORMATION 2004 : The transport_rate for PCR_PID 0x0026 is 5005947.48 byte/s and this rate will be used to check the rest of this PCR_PID at byte 10 bit 1 of t_packet 1958 (pid 0x0026; TS stream byte 368114).

These information messages explain what the Compliance Verifier bases its timing analysis on.

### Error message

[MPEG-2] ERROR 2856 (ref. MPEG-2 Audio 2.5.3.1) : Not all fields of mc_header fit in base frame for audio ES, AU 1, at byte 1535 bit 7; PES (0xC0) byte 1549 (byte 1549 of packet 0); byte 81 of t_packet 3069 (pid 0x002A; TS stream byte 577053).

[MPEG-2] SYNTAX ERROR 2853 (ref. MPEG-2 Audio) : The default setting (MPEG-2) for audio_stream 0xC0 parsing proved to be incorrect, due to the mc_header that does not fit into the base_frame. The audio parsing will be switched to MPEG-1 parsing for audio ES, AU 1, at byte 1535 bit 7; PES (0xC0) byte 1549 (byte 1549 of packet 0); byte 81 of t_packet 3069 (pid 0x002A; TS stream byte 577053). [Look Ahead : 0x7F 0xFE 0x62 0x00 (len : 32 bit)]

These messages explain that the Compliance Verifier originally defaulted to MPEG-2 audio, but is now switching to MPEG-1 audio.

### Error message

[MPEG] ERROR 1642 (ref. MPEG Video 2.4.1 | Compl 9.2.1.3) : GOP ends with too few B-pictures for video sequence (0) at byte 50133 bit 4; PES (0xE0) byte 50273 (byte 41 of packet 4); byte 53 of t_packet 4452 (pid 0x0029; TS stream byte 837029).

This message reports that the Compliance Verifier did not find the expected coding sequence, **B B P B I**, at the end of this group of pictures.

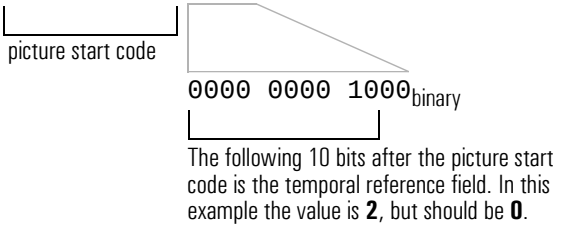
### Error message

[MPEG] ERROR 1667 (ref. MPEG Video 2.4.3.4 | 6.3.9) : Picture has type I, temporal reference of picture 1 indicates it should be B for video sequence (0) at byte 50221 bit 2; PES (0xE0) byte 50361 (byte 129 of packet 4); byte 141 of t_packet 4452 (pid 0x0029; TS stream byte 837117).

This message reports that an I frame was encountered when a B frame was expected. The Compliance Verifier expected a B frame because the temporal reference value for this picture is **2**, as shown below. Since this is the first picture after a group of pictures header, the temporal reference value should be **0**.

```
TSP #4452   0:0.167 196 60

Start Code                                  0x000001
Stream Id                                   0xE0
(Video Stream 0)
Packet Length                               0
PES Scramble Control                        0
(Not Scrambled)
PES Priority                                0
Data Alignment Indicator                    1
Copyright                                   0
Original Indicator                          1
(Original)
PES Header Data Length                      19
PTS/DTS Flags                               0x3
(PTS and DTS)
PTS                                         13:26:39:99450000
DTS                                         13:26:39:87449999
Payload :00 00 01 B3 22 02 40 23 24 9F 23 81 10 11 11 12
         12 12 13 13 13 13 14 14 14 14 14 15 15 15 15 15
         15 16 16 16 16 16 16 16 17 17 17 17 17 17 17 17
         18 18 18 19 18 18 18 19 1A 1A 1A 1A 19 1B 1B 1B
         1B 1B 1C 1C 1C 1C 1E 1E 1E 1F 1F 21 00 00 01 B5
         14 82 00 01 00 00 00 00 00 00 01 B8 01 AE D6 80
         00 00 01 00 00 88 C9 D0 00 00 00 01 B5 8F FF FB
...
```



### Oddity message

[MPEG-2] ODDITY 2036 (ref. MPEG-2 Systems 2.4.3.5) : An I Picture is present in the TS packet, but the elementary_stream_priority_indicator flag is not set at byte 192 bit 0 of t_packet 4453 (pid 0x0029; TS stream byte 837168).

[MPEG-2] ODDITY 2036 (ref. MPEG-2 Systems 2.4.3.5) : An I Picture is present in the TS packet, but the elementary_stream_priority_indicator flag is not set at byte 192 bit 0 of t_packet 4469 (pid 0x0029; TS stream byte 840176).

[MPEG-2] ODDITY 2036 (ref. MPEG-2 Systems 2.4.3.5) : An I Picture is present in the TS packet, but the elementary_stream_priority_indicator flag is not set at byte 192 bit 0 of t_packet 4482 (pid 0x0029; TS stream byte 842620).

[MPEG-2] ODDITY 2036 (ref. MPEG-2 Systems 2.4.3.5) : An I Picture is present in the TS packet, but the elementary_stream_priority_indicator flag is not set at byte 192 bit 0 of t_packet 4493 (pid 0x0029; TS stream byte 844688).

[MPEG-2] ODDITY 2036 (ref. MPEG-2 Systems 2.4.3.5) : An I Picture is present in the TS packet, but the elementary_stream_priority_indicator flag is not set at byte 192 bit 0 of t_packet 4515 (pid 0x0029; TS stream byte 848824). MPEG-2 ODDITY 2036 will no longer be reported !

These 5 oddities report that the elementary stream priority indicator bit in the TS header adaptation field is not set to **1**. When the packet payload contains one or more bytes from an intra coded slice (an I frame), the elementary stream priority indicator should be set to **1** to indicate that the payload has a higher priority than the payloads of other TS packets.

### Error message

[MPEG] ERROR 1445 (ref. MPEG Systems 2.4.4.3 | 2.4.3.7) : Previous I/P picture's PTS - DTS offset is 10799, should be 10800 in PES stream 0xE0 at byte 122617 bit 0 (byte 17 of packet 11); byte 29 of t_packet 12669 (pid 0x0026; TS stream byte 2381801).

This message reports that the difference between the PTS (presentation time stamp) and DTS (decoding time stamp) in this packet is incorrect.

The frame rate for this stream, as indicated in the sequence header, is 25 frames/second, and the coding structure is I B B P B B P. Because I and P frames are required to decode B frames, three frames need to be buffered. The time required for an I or P frame to remain in the buffer is therefore 3/25 seconds (0.12000000), or 10,800 clock ticks of a 90 kHz system time clock. The time for the frame to remain in the buffer is also equal to the difference between the PTS (presentation time stamp) and DTS (decoding time stamp).

Below are the PTS and DTS fields for packet #12669 copied from the Protocol Data Viewer's PDU Details view. The difference between the two fields is 0.11998889 (10,799 / 90 kHz), but should be 0.12000000 (10,800 / 90 kHz).

```
TSP #12669   0:0.475 789 32

Start Code                                0x000001
Stream Id                                 0xE0
(Video Stream 0)
Packet Length                             0
PES Scramble Control                      0
(Not Scrambled)
PES Priority                              0
Data Alignment Indicator                  1
Copyright                                 0
Original Indicator                        1
(Original)
PES Header Data Length                    19
PTS/DTS Flags                             0x3
(PTS and DTS)
PTS                                       13:27:01:12557777
DTS                                       13:27:01:00558888
```

**Error message**

[MPEG] ERROR 1622 (ref. MPEG Video 2.4.3.3 | 6.3.8) : GOP marker_bit in time_code is 0 for video sequence (0) at byte 2025684 bit 0; PES (0xE0) byte 2029520 (byte 120 of packet 136); byte 132 of t_packet 144535 (pid 0x0029; TS stream byte 27172712).

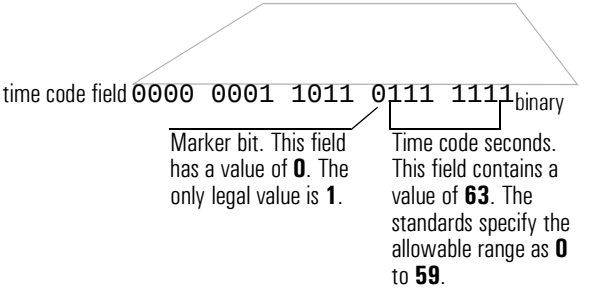
[MPEG] ERROR 1633 (ref. MPEG Video 2.4.3.3 | 6.3.8) : GOP time_code_seconds is 63, should be in 0..59 for video sequence (0) at byte 2025684 bit 0; PES (0xE0) byte 2029520 (byte 120 of packet 136); byte 132 of t_packet 144535 (pid 0x0029; TS stream byte 27172712).

These messages describe illegal values in the time code, a 25–bit field in the GOP (group of pictures) header made of five subfields. In the decoded packet below, the PES packet payload contains the 25 bits of the time code in the sequence **01 B7 FA 0**.

```
TSP #144535   0:5.428 069 17

Start Code                                0x000001
Stream Id                                 0xE0
(Video Stream 0)
Packet Length                             0
PES Scramble Control                      0
(Not Scrambled)
PES Priority                              0
Data Alignment Indicator                  1
```

```
Copyright                                   0
Original Indicator                          1
(Original)
PES Header Data Length                      19
PTS/DTS Flags                               0x3
(PTS and DTS)
PTS                                         13:26:45:27451111
DTS                                         13:26:45:15449999
Payload :00 00 01 B3 22 02 40 23 24 9F 23 81 10 11 11 12
         12 12 13 13 13 13 14 14 14 14 14 15 15 15 15 15
         15 16 16 16 16 16 16 16 17 17 17 17 17 17 17 17
         18 18 18 19 18 18 18 19 1A 1A 1A 1A 19 1B 1B 1B
         1B 1B 1C 1C 1C 1C 1E 1E 1E 1F 1F 21 00 00 01 B5
         14 82 00 01 00 00 00 00 00 00 01 B8 01 B7 FA 00
         00 00 ....
```



# 11

# Using the Video ES Analyzer

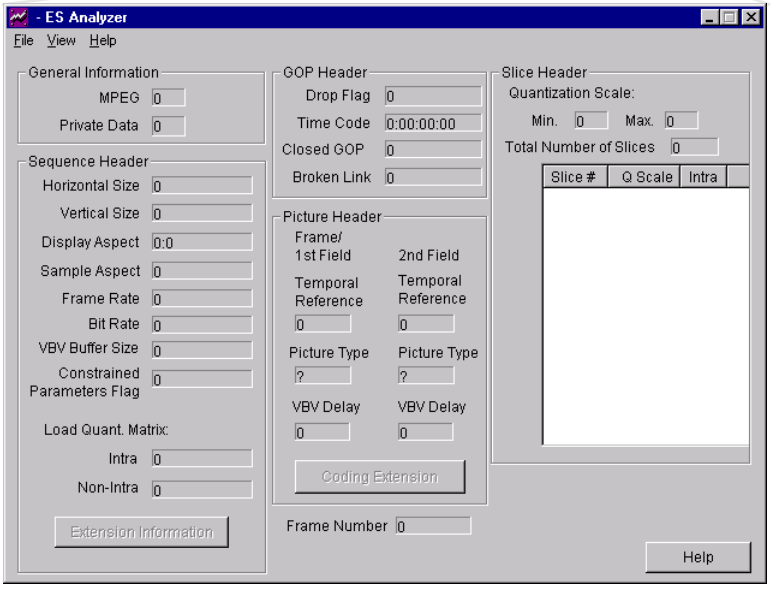
# Analyzing a video elementary stream


This example requires the purchase of E6310A. It shows how to use the Video ES Analyzer to play back and analyze a video elementary stream.

## Step 1: Start the Video ES Analyzer

There are two ways to start the Video ES Analyzer.

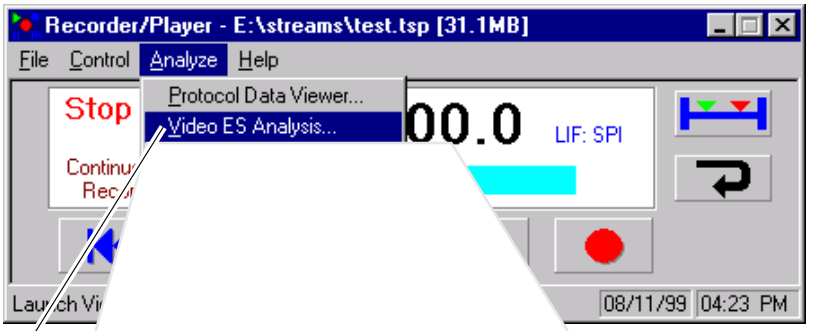
From the Launch Pad:

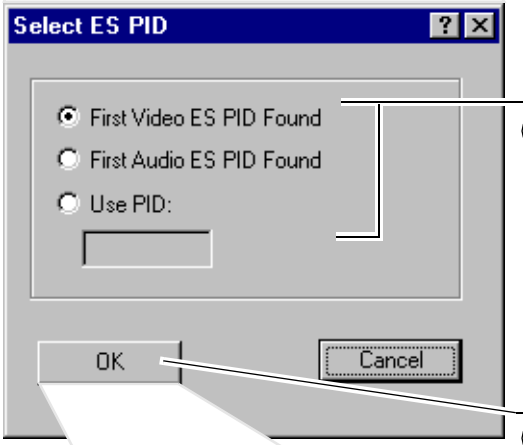


Use this method when you want to open and analyze a file you have saved to disk.

From the Recorder/Player:



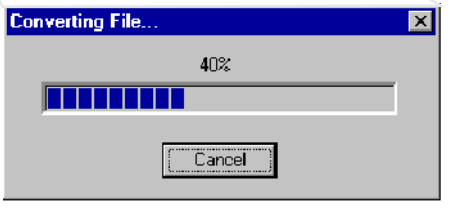

1. Start the Video ES Analyzer.




2. Select the first video elementary stream PID (Program Identifier) found, or enter the PID of the elementary stream to analyze.

3. Extract the video elementary stream from the file.

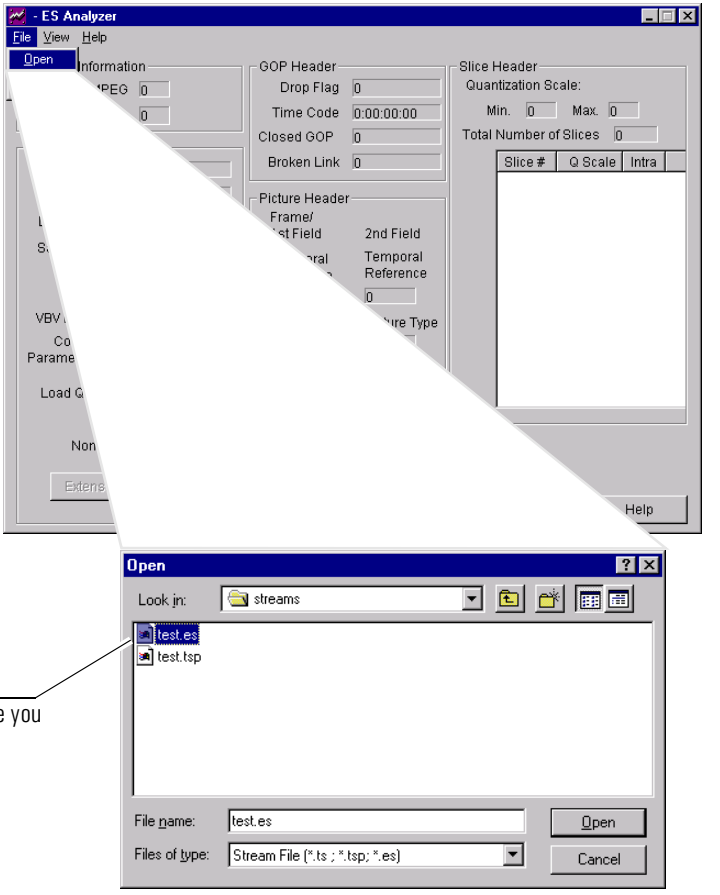
   When MPEGscope has finished demultiplexing the file, the Video ES Analyzer will launch with the file loaded.




Use this method when you want to extract and analyze an elementary stream from a transport stream file you have opened or recorded from the Recorder/Player.

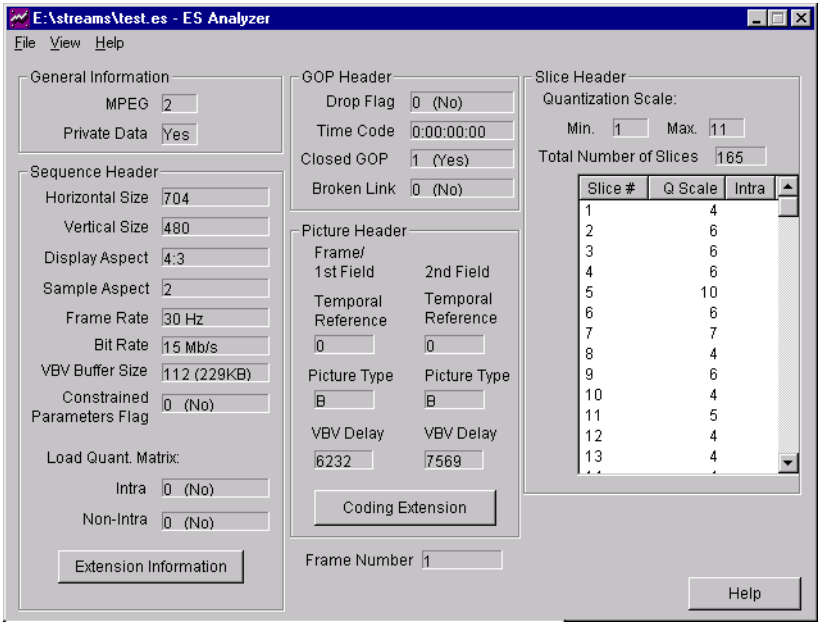
## Step 2: Open a file

Complete this step if you have opened the Video ES Analyzer from the Launch Pad.




Double click on the file you want to open.

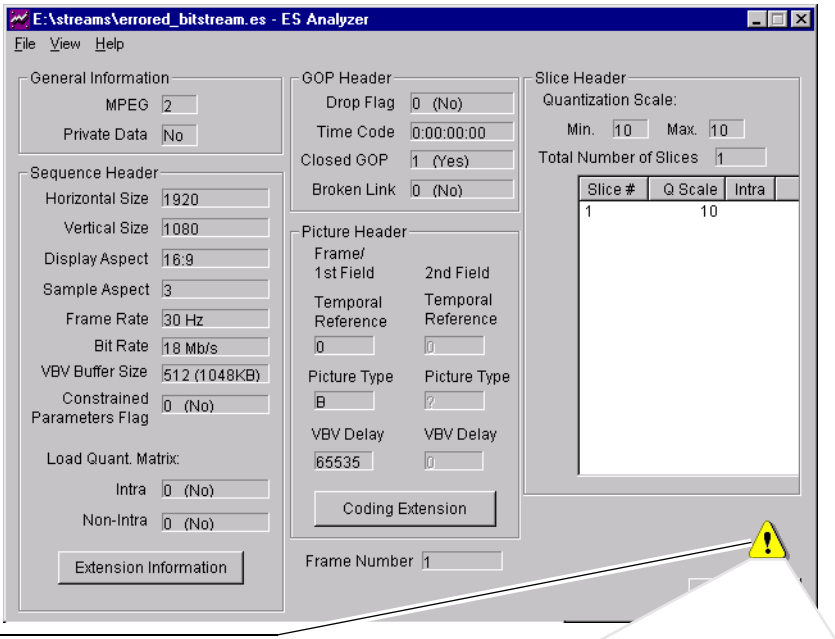
The Video ES Analyzer can open elementary stream (**.es**), transport stream (**.ts**), and transport stream plus (.**tsp**) files for analysis. If you select a **.ts** or a **.tsp** file, specify a video elementary stream PID (Program Identifier) number, as illustrated on page 11–3.

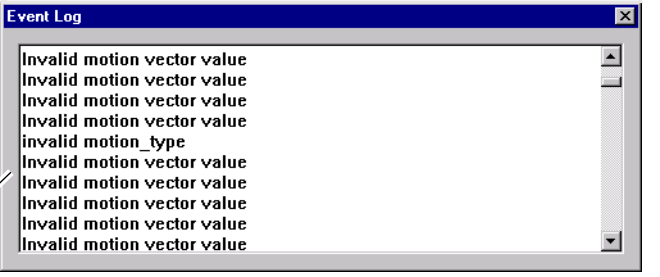



With the exception of the **Buffer Occupancy** and **Bit Rates** graphs, information and statistics are based on the *display* order of frames in the bitstream, as defined in Section 6.1.1.11, "Frame re-ordering", of ISO/IEC 13818-2. The Video ES Analyzer does not use the **temporal_reference** field to determine display order.

## Step 3: Troubleshoot bitstream errors

If the Video ES Analyzer finds errors when opening or playing a video stream, a warning icon will display in the bottom right corner. You can use **Trace Mode** to view the decoded bitstream elements located immediately before an error.

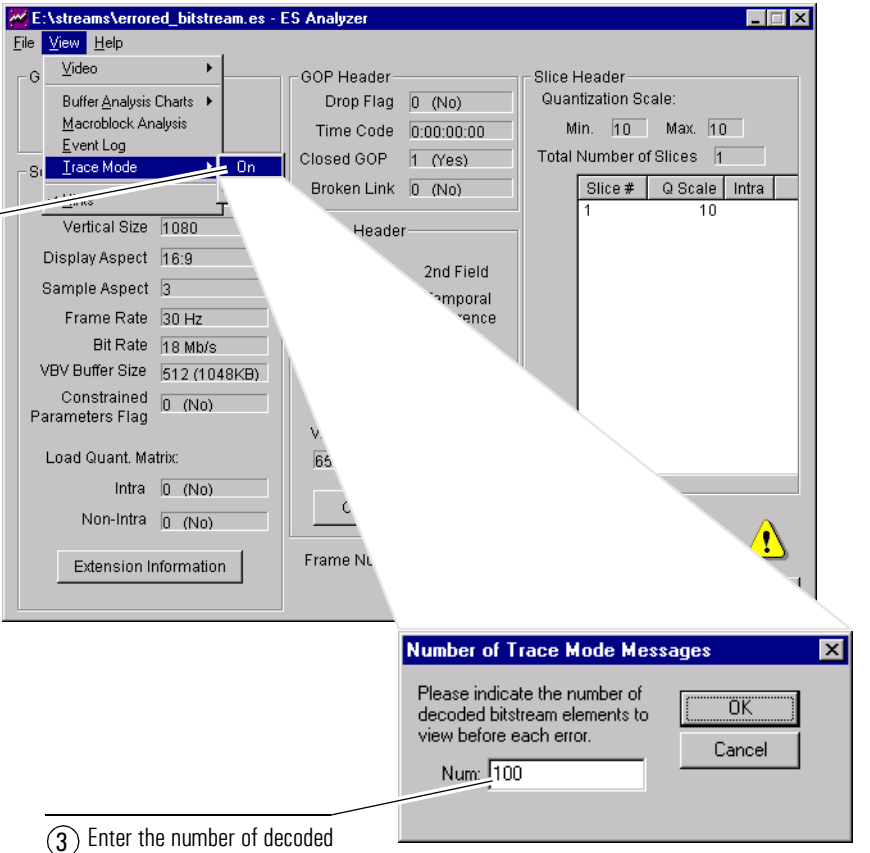



(1) **Click on the warning icon to open the Event Log. You** can also open it from the **View/Event Log** menu.

The warning icon will display when the Video ES Analyzer finds an error while decoding. This can happen when you first open a file or when you navigate forwards or backwards in the file using the **Play Controls**.

The Event Log provides a brief description of the errors encountered.

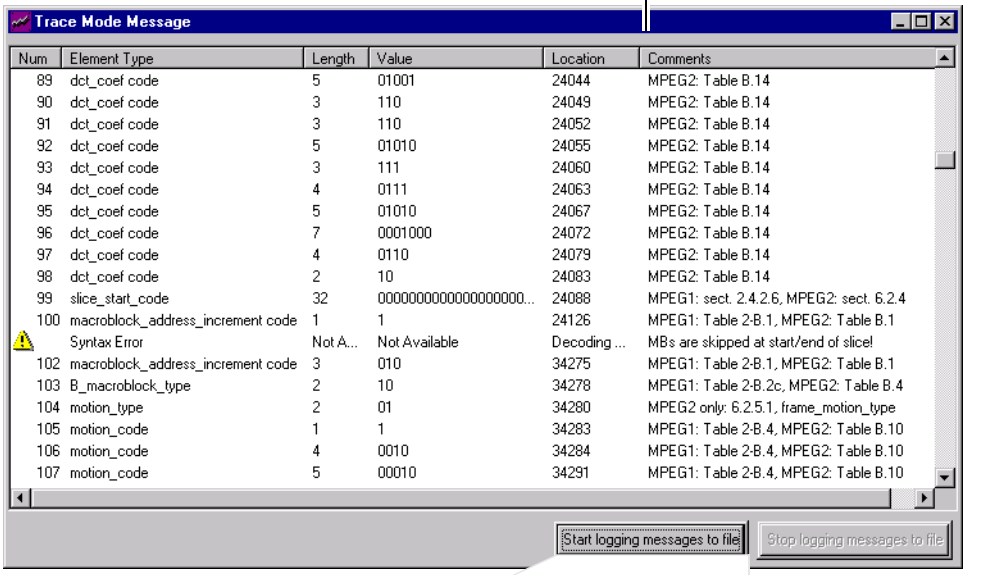


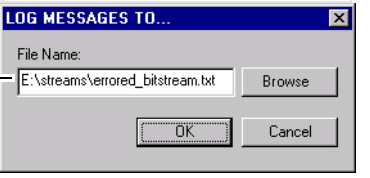

② To view the decoded bitstream elements that occurred immediately before each error, turn **Trace Mode** on.

③ Enter the number of decoded elementary stream fields to display before each error, then press **OK**.


④ Use the **Play Controls** to step back one frame to the beginning of the frame where errors were first encountered.

Errors in the frame are displayed along with the decoded bitstream elements that occurred immediately before.




5 **If desired, log the messages to a file.**

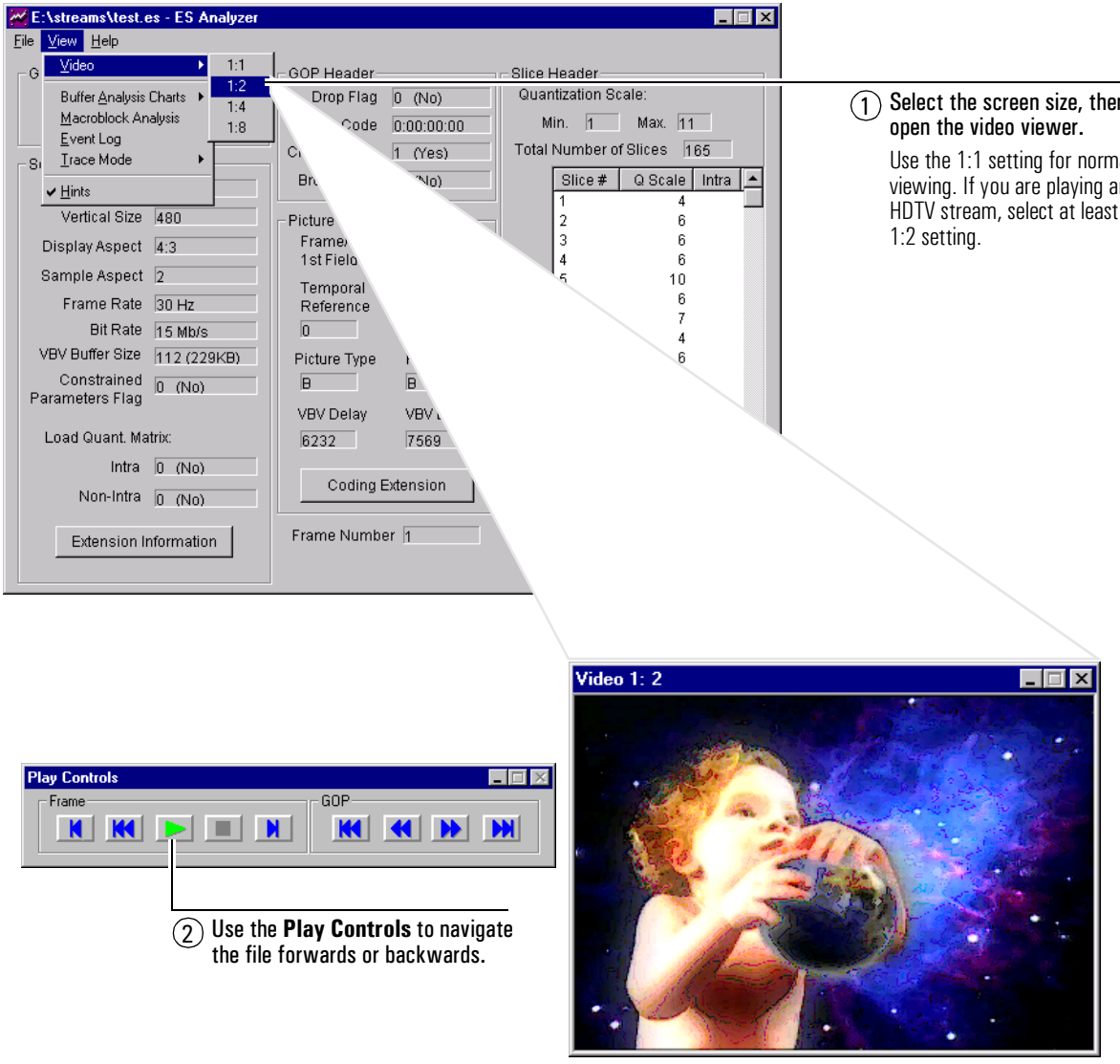
You can start or stop logging at any time.

6 Play the rest of the stream. Errors and bitstream elements will display in the **Trace Mode Message** dialog as successive frames are decoded and


When you turn **Trace Mode** on, the stream will be analyzed in *decode* order and will not correspond to the display in the video viewer.

You can turn **Trace Mode** on or off at any time from the **View/Trace Mode** menu. You can also turn **Trace Mode** off by closing the **Trace Mode Message** dialog.

# Step 4: View the video elementary stream

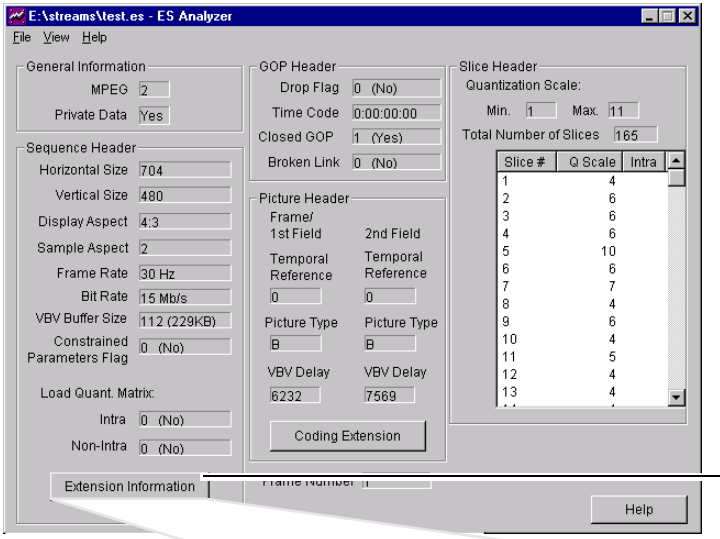



1. **Select the screen size, then open the video viewer.**

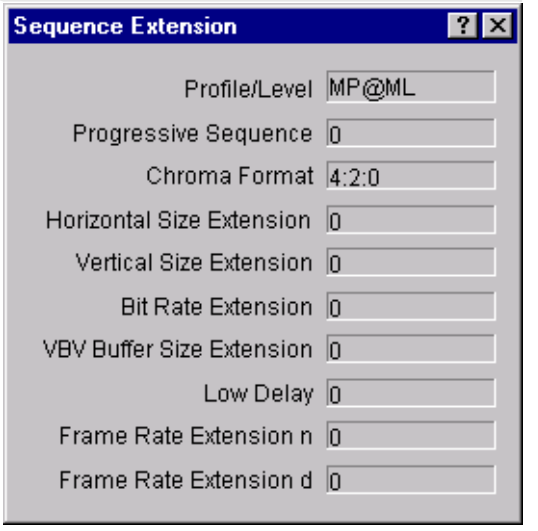
   Use the 1:1 setting for normal viewing. If you are playing an HDTV stream, select at least a 1:2 setting.

2. Use the **Play Controls** to navigate the file forwards or backwards.

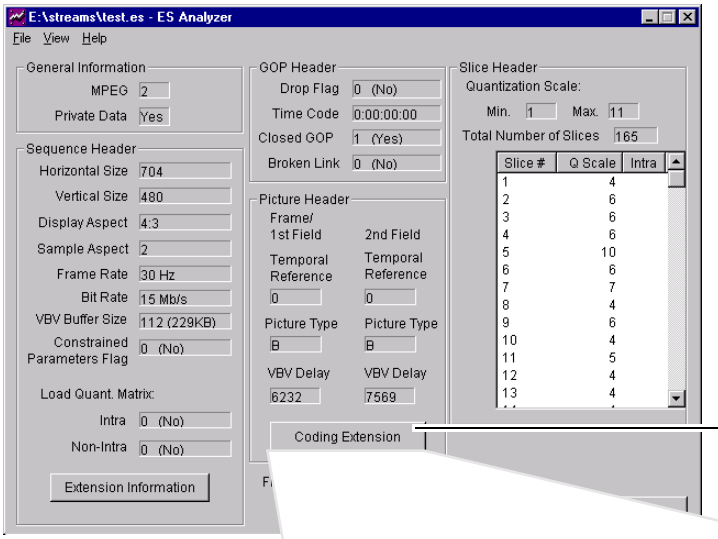
## Step 5: View MPEG-2 extension information



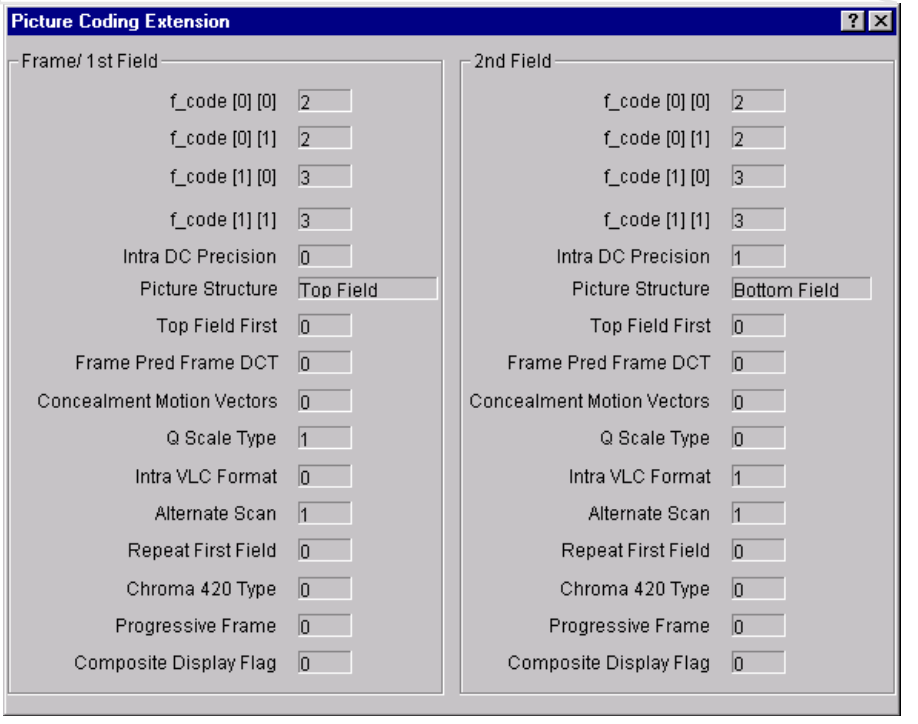

1. View MPEG–2 sequence extension information.




The header extension buttons are automatically disabled for MPEG–1 elementary streams.




② View MPEG–2 picture extension information.

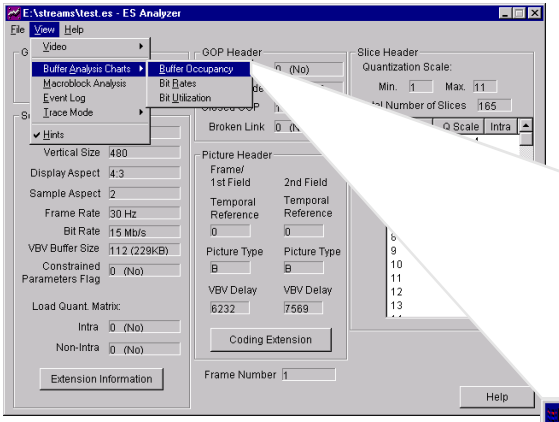



This example shows a field-coded picture. Picture coding extension values for the first field are displayed in the left column. Values for the second field are displayed in the right column.

If you are viewing a frame-coded picture, the **Frame/1st Field** column shows the picture coding extension values for the frame. The **2nd Field** column, which applies only to field-coded pictures, is grayed out.

# Step 6: View the **Buffer Occupancy** graph

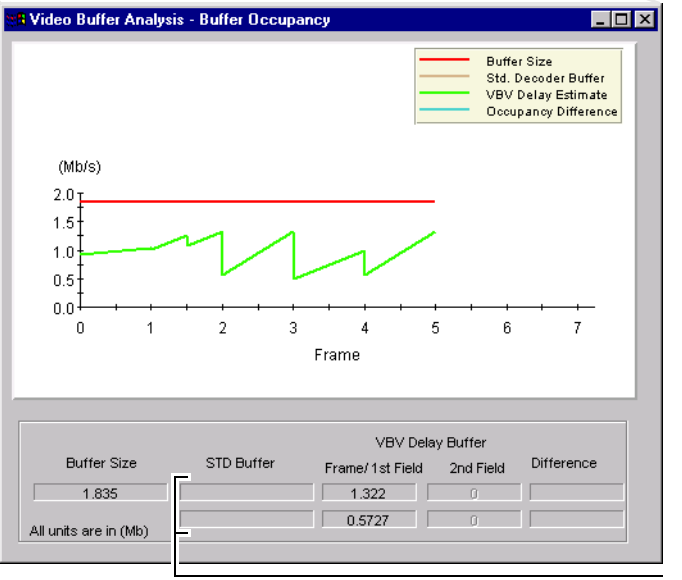
The **Buffer Occupancy** graph displays buffer size, buffer occupancy estimated by both the standard decoder method (MPEG–1 only) and the VBV delay method (MPEG-1 and MPEG-2). If you are analyzing an MPEG-1 file, it also displays the buffer occupancy difference between the two methods.





① Select **Play** to start graph analysis.

② Select **Stop** to end graph analysis.




Top fields show buffer occupancy after the current picture enters the buffer. Bottom fields show buffer occupancy after the previous picture is removed but before the current picture enters the buffer.

The **STD Buffer** on this graph refers to the **B** (input elementary stream) buffer in the STD (System Target Decoder) model of the ISO/IEC 11172-1 and 11172-2 MPEG-1 standards.

The **VBV Delay Buffer** on this graph refers to the **EB** (video elementary stream) buffer in the T-STD (Transport Stream System Target Decoder) model of the ISO/IEC 13812-1 MPEG-2 standard as well as the VBV buffer in Annex C of the ISO/IEC 13812-2 MPEG-2 standard.

The **Buffer Occupancy** graph and statistics are based on the *decode* order of frames in the bitstream. No correlation therefore exists with the video display or other statistics, which are based on *display* order.

Field-coded pictures have four graph points for each frame (two for each field). All display within one frame interval on the graph. Frame-coded pictures have only two graph points.

## Buffer Occupancy Formulas

**Buffer Size**

$B = 16 * 1024 * vbv\_buffer\_size$ [ISO/IEC 13818–2, 6.3.3]

where

| | | |
|---|---|---|
| B | = | buffer size (in bits) |
| vbv_buffer_size | = | sequence_header().vbv_buffer_size |

**Standard Decoder Buffer (MPEG–1 only)**

**Top:** $SDBtop(n) = bit\_rate * vbv\_delay(n) / 90000$

**Bottom:** $SDBbottom(n) = SDBtop(n) - d_n$

where

| | | |
|---|---|---|
| SDBtop(n) | = | Standard Decoder Buffer: value (in bits) for $n^{th}$ picture before removing picture from buffer |
| SDBbottom(n) | = | Standard Decoder Buffer: value (in bits) for $n^{th}$ picture after removing picture from buffer |
| bit_rate | = | sequence_header().bit_rate |
| vbv_delay(n) | = | picture_header().vbv_delay for $n^{th}$ picture |
| $d_n$ | = | number of picture bits removed from buffer during decode of picture *n* |

**VBV Delay Method**

**Top:** $VBVtop(n) = VBVtop(n-1) - d_{n-1} + (I * R(n))$

**Bottom:** $VBVbottom(n) = VBVtop(n-1) - d_{n-1}$

where

| | | |
|---|---|---|
| VBVtop(n) | = | VBV Buffer: value (in bits) for *n*th picture before removing picture from buffer |
| VBVbottom(n) | = | VBV Buffer: value (in bits) for *n-1*th picture after removing picture from buffer |
| I | = | time interval (in seconds) [ISO/IEC 13818–2, C.9 – C.12] |
| R(n) | = | buffer input rate for *n*th picture) (refer to "Bit Rate Formulas", page 11–16 |
| $d_n$ | = | number of picture bits removed from buffer during decode of picture *n* |

**Occupancy Difference (MPEG–1 only)**

**Top:** $ODtop(n) = | VBVtop(n) - SDBtop(n) |$

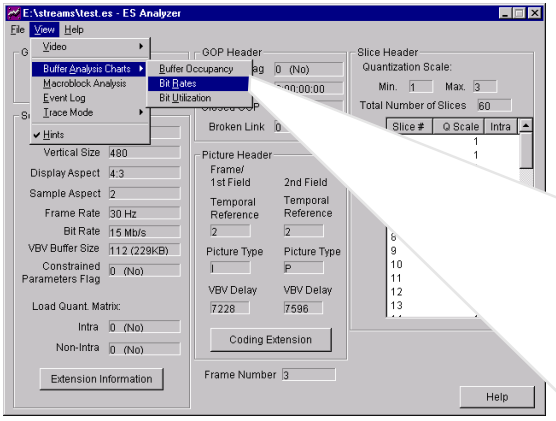
**Bottom:** $ODbottom(n) = | VBVbottom(n) - SDBbottom(n) |$

where

| | | |
|---|---|---|
| ODtop(n) | = | occupancy difference (in bits) for top coordinate |
| VBVtop(n) | = | VBV buffer value for top coordinate |
| SDBtop(n) | = | standard decoder buffer value for top coordinate |
| ODbottom(n) | = | occupancy difference (in bits) for bottom coordinate |
| VBVbottom(n) | = | VBV buffer value for bottom coordinate |
| SDBbottom(n) | = | standard decoder buffer value for bottom coordinate |

# Step 7: View the **Bit Rates** graph

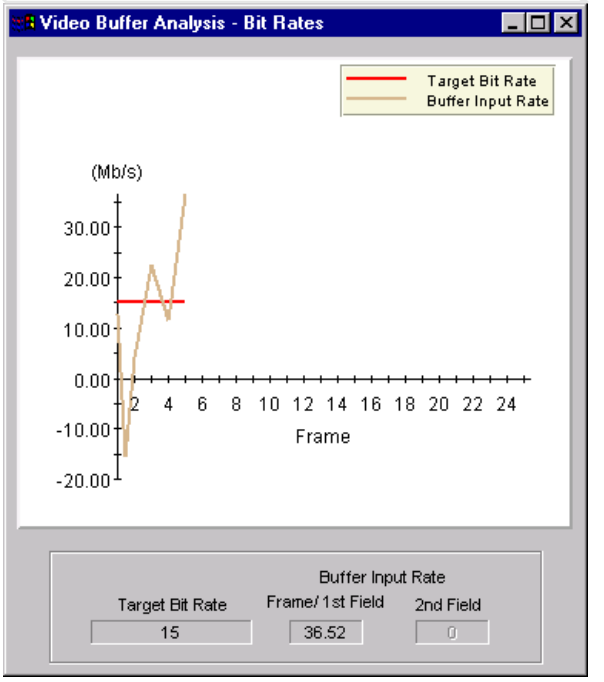
The **Bit Rates** graph displays the target bit rate, as given in the sequence header, along with the actual buffer input rate.





(1) Select **Play** to start graph analysis.

(2) Select **Stop** to end graph analysis.




Statistics are based on the *decode* order of frames in the bitstream.

Field-coded pictures have two graph points for each frame (one for each field). Both display within one frame interval on the graph. Frame-coded pictures have only one graph point.

## Bit Rate Formulas

**Target Bit Rate**

$TBR(n) = bit\_rate$

where

| | | |
|---|---|---|
| TBR(n) | = | target bit rate (in bits per second) for $n^{th}$ picture |
| bit_rate | = | sequence_header().bit_rate |

**Buffer Input Rate (MPEG-1, CBR)**

$$R(n) = frame\_rate * \sum_{i=1}^{n} d_i \; / \; n$$

where

| | | |
|---|---|---|
| R(n) | = | input frame rate (in bits per second) for $n^{th}$ picture |
| frame_rate | = | sequence_header().frame_rate |
| $\sum_{i=1}^{n} d_i$ | = | sum of picture bits to the $n^{th}$ picture |
| n | = | picture number |
| $d_i$ | = | number of bits for the $i^{th}$ picture |

**Buffer Input Rate (MPEG-2, VBR)**

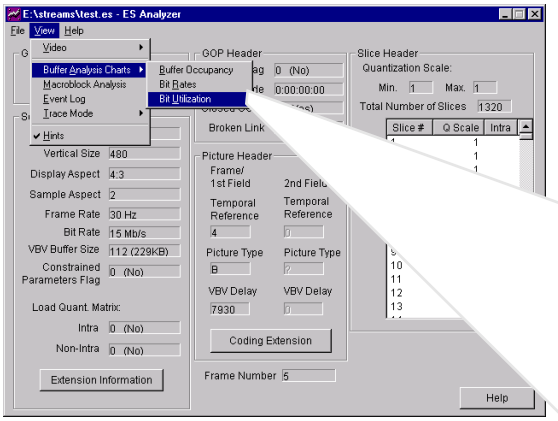
$R(n) = d_n / (\tau(n) - \tau(n + 1) + t(n + 1) - t(n))$[ISO/IEC 13818-2, C.3.1]

where

| | | |
|---|---|---|
| R(n) | = | input frame rate (in bits per second) for $n^{th}$ picture |
| $d_n$ | = | number of bits in the $n^{th}$ picture |
| $\tau(n)$ | = | vbv_delay for the $n^{th}$ picture |
| t(n) | = | time when $n^{th}$ picture is removed from buffer |

## Step 8: View the **Bit Utilization** graph

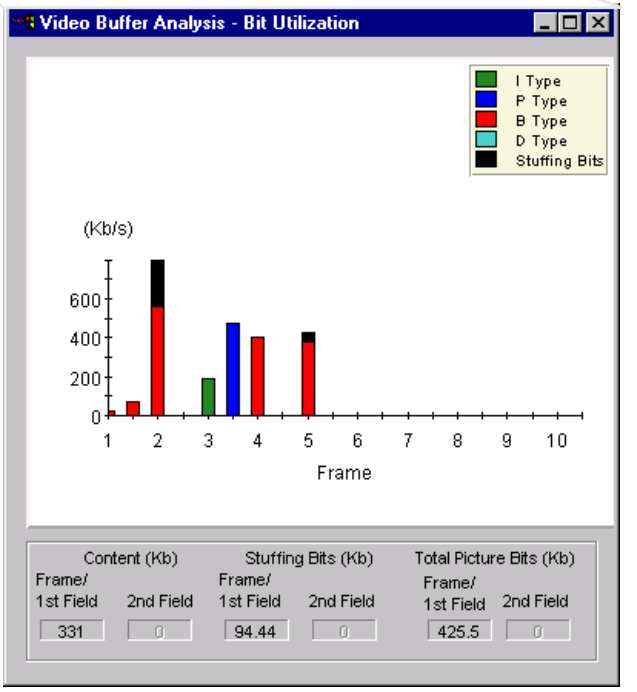
The **Bit Utilization** graph displays the number of bits used per frame. The frames are color–coded to show I, P, and B frames. If MPEG–1 is used, D frames are also shown. Stuffing bits are displayed in black on the graph.





① Select **Play** to start graph analysis.

② Select **Stop** to end graph analysis.

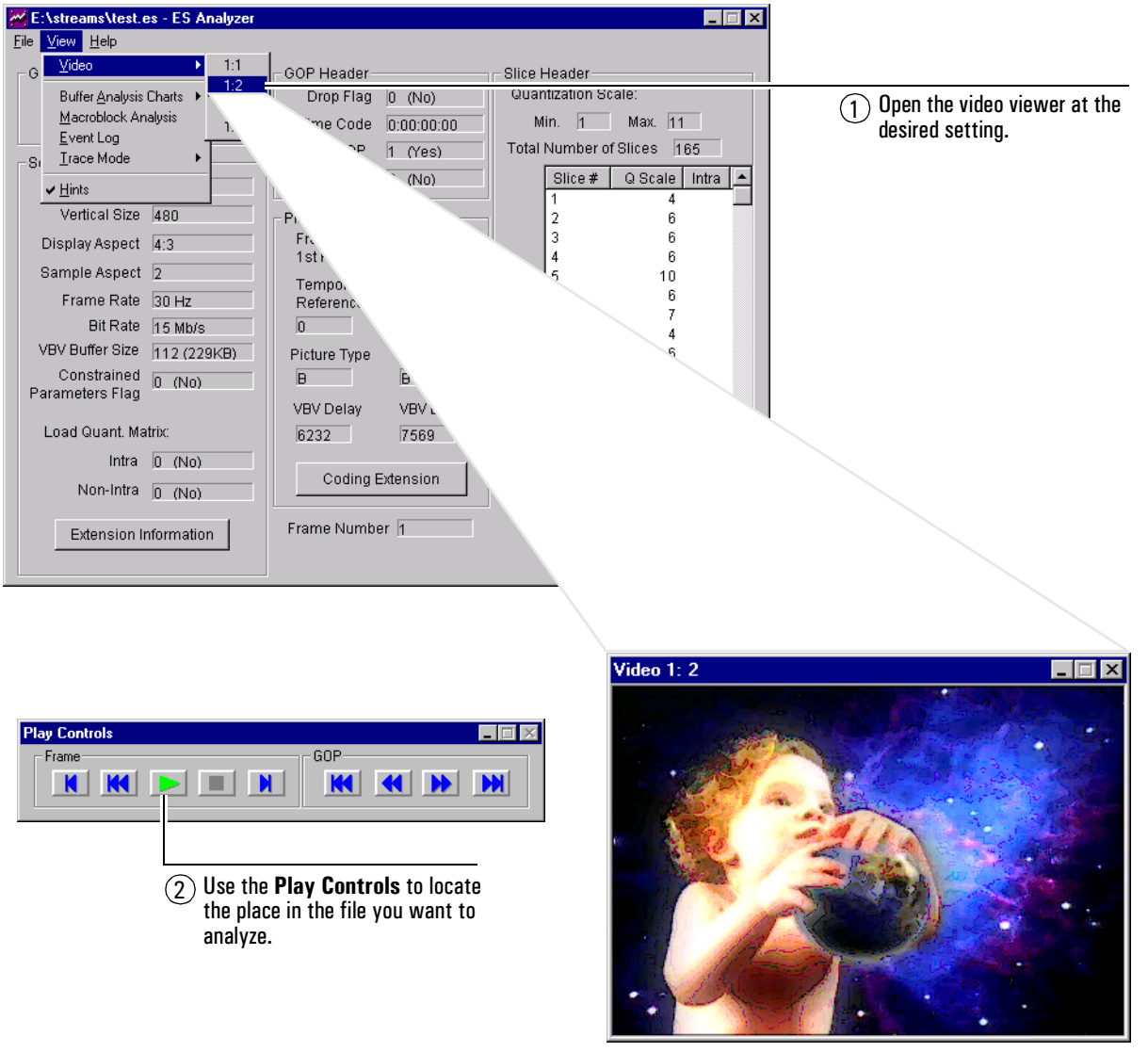



Statistics are based on the *display* order of frames in the bitstream.

Field-coded pictures have two graph points for each frame (one for each field). Both display within one frame interval on the graph. Frame-coded pictures have only one graph point.

## Step 9: View macroblock information

The Video ES Analyzer provides a graphical representation of the macroblock types, bits per macroblock, macroblock vectors, macroblock motion types, and quantization scales.

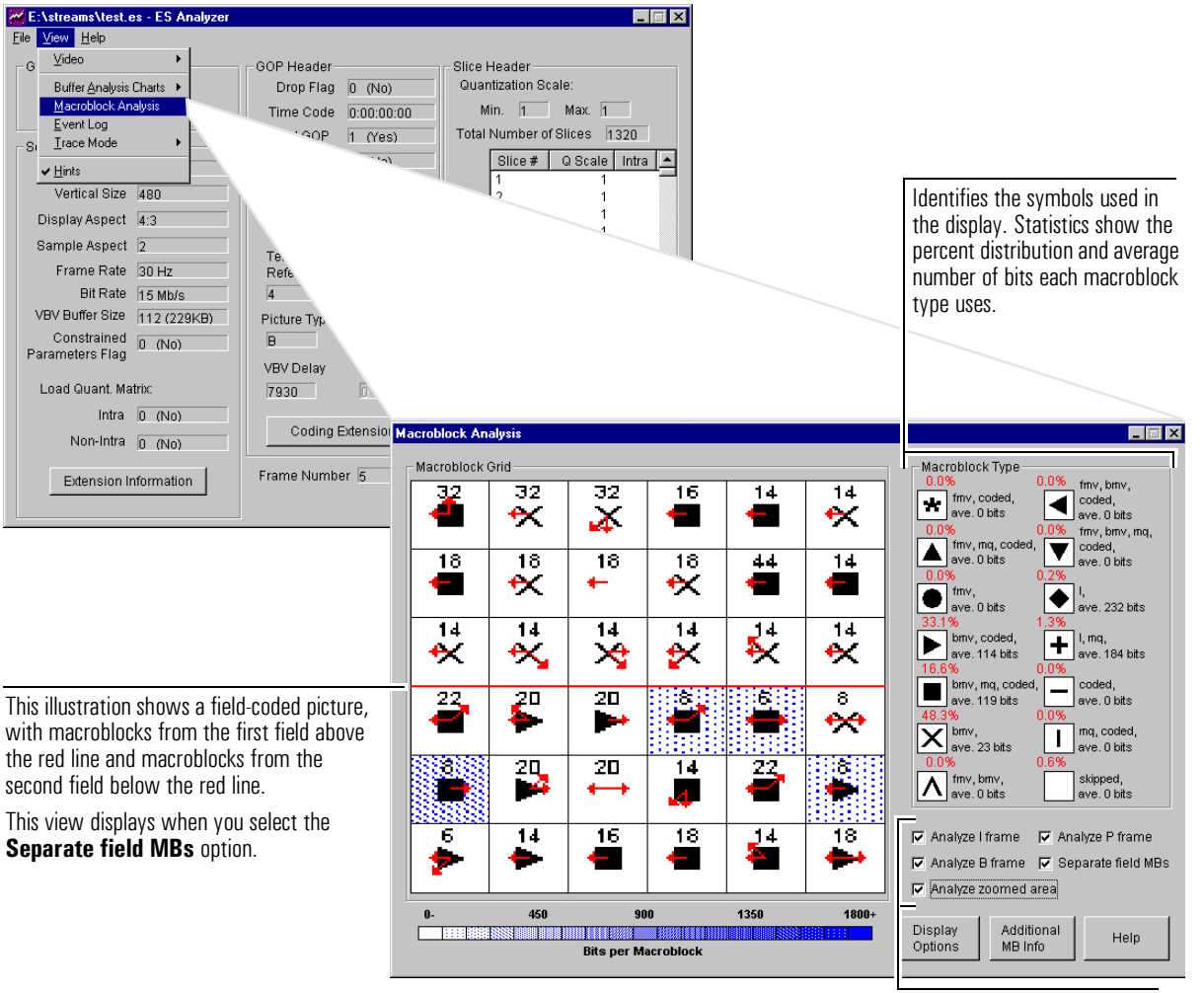



3 **Holding down the left mouse button, select the specific zoom area to analyze.**

When you release the mouse button, the area will automatically resize to a perfect square.

When you select a zoom area from the video viewer, the Video ES Analyzer analyzes the macroblocks in this area. If you do not select a zoom area, the Video ES Analyzer analyzes all macroblocks in the current frame.

**To analyze the selected zoom area:**



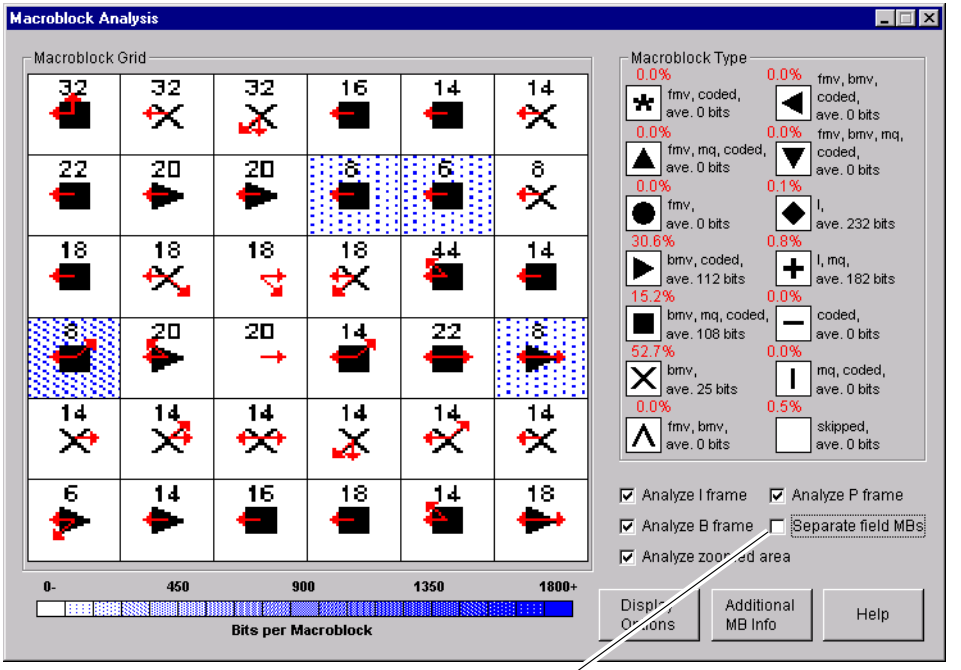

Identifies the symbols used in the display. Statistics show the percent distribution and average number of bits each macroblock type uses.

This illustration shows a field-coded picture, with macroblocks from the first field above the red line and macroblocks from the second field below the red line.

This view displays when you select the **Separate field MBs** option.


④ Select the frame types you want to observe, then select **Analyze Zoomed Area**.

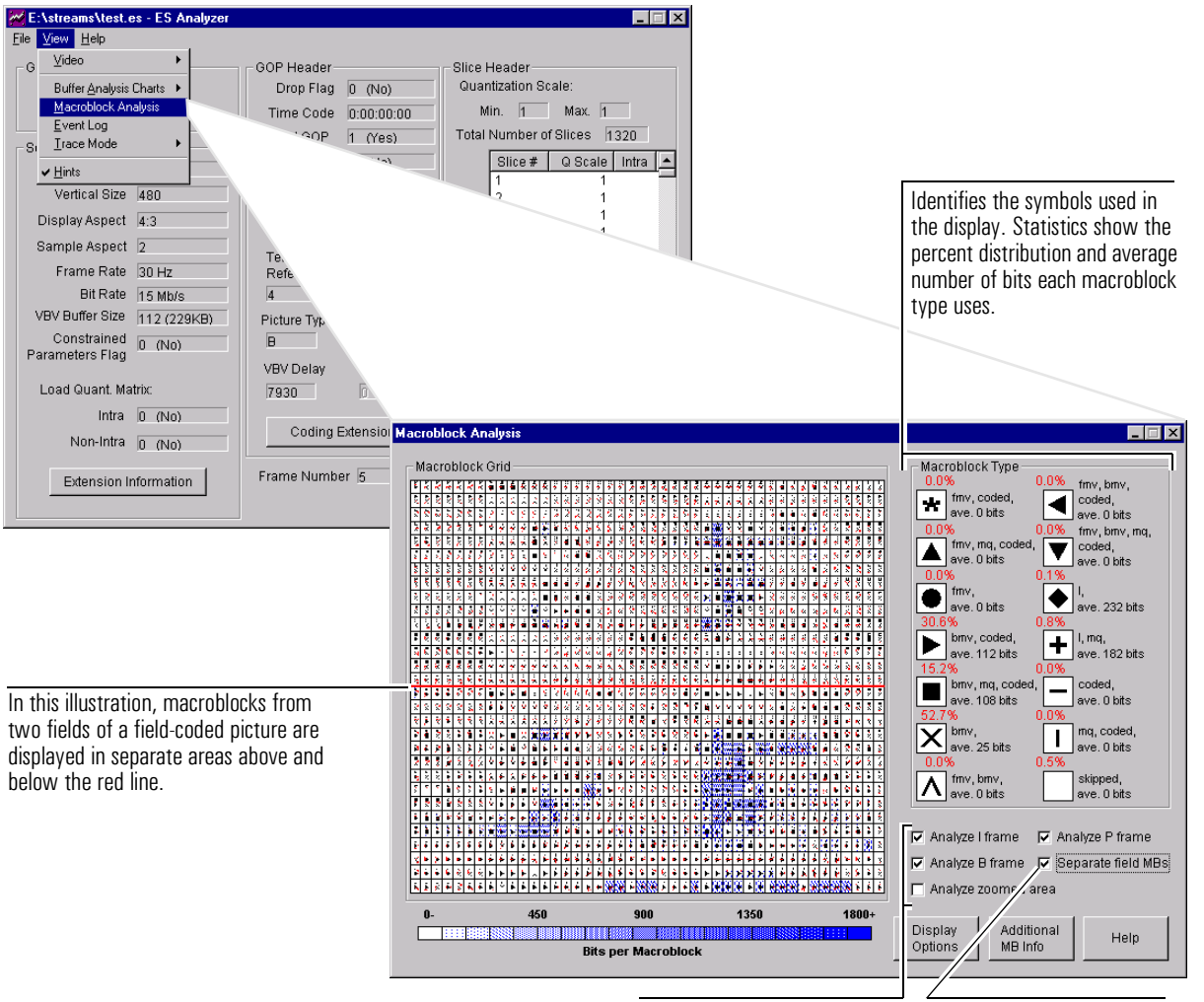
The graphical macroblock display updates to the current frame as you navigate the file.




This illustration displays macroblocks from the same zoomed-in, field-coded picture as in the previous page; however, macroblocks from each field are displayed on alternate lines.

This view makes it easier to relate the area you are analyzing to the video display, and occurs when you deselect the **Separate field MBs** option.

**To analyze all macroblocks in the frame:**




Identifies the symbols used in the display. Statistics show the percent distribution and average number of bits each macroblock type uses.

In this illustration, macroblocks from two fields of a field-coded picture are displayed in separate areas above and below the red line.

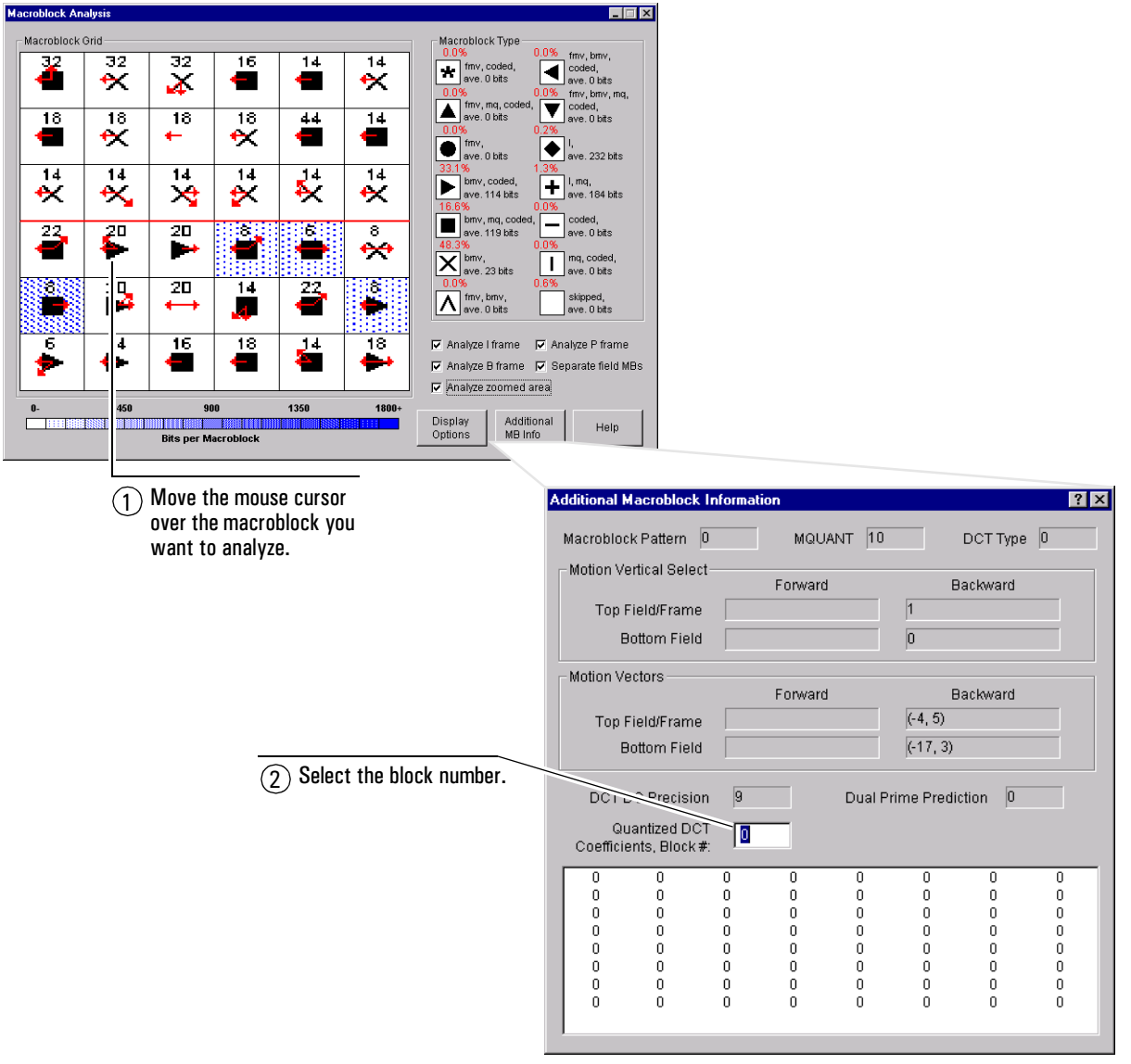

The graphical macroblock display updates to the current frame as you navigate the file.

5 **Select the frame types you want to observe, then unselect Analyze Zoomed Area.**

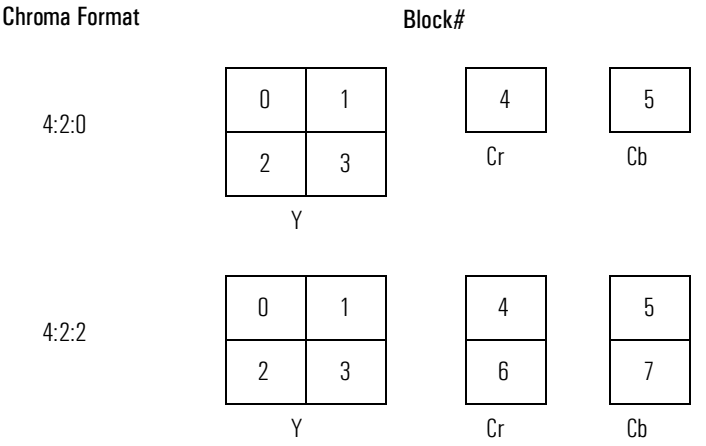
To view the macroblocks on alternate lines instead of in separate areas, deselect **Separate field MBs**.

# Step 10: View additional macroblock information

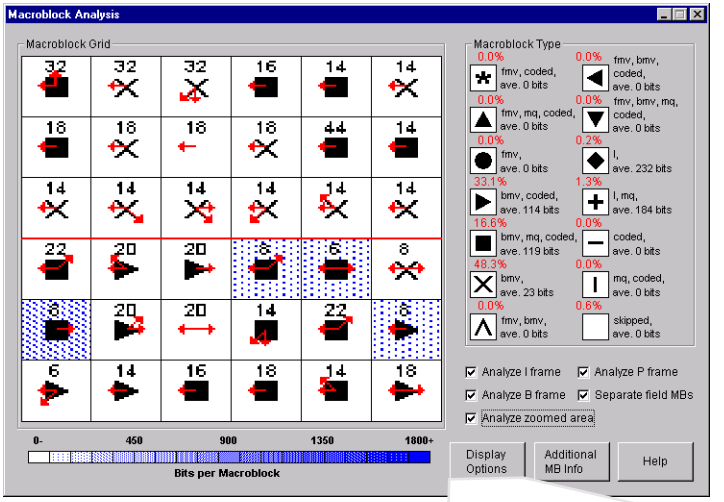
You can also view additional macroblock information, such as motion vectors, DCT type, and quantized DCT coefficients.

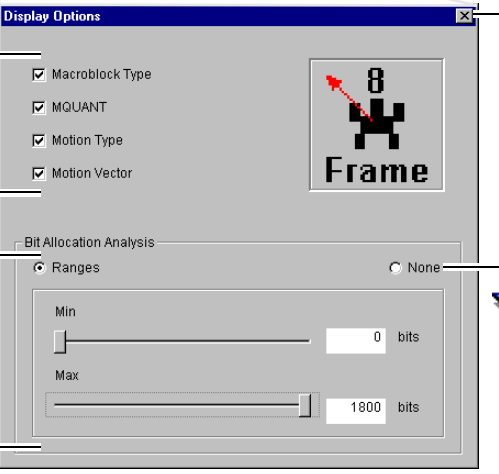


The DCT coefficients for a given chroma block are shown at the bottom of the window. To view the DCT coefficients for a different chroma block, change the block number. The relationship between block numbers and chroma blocks is shown below for 4:2:0 and 4:2:2 macroblocks. You can view the chroma format for an elementary stream in the sequence header extension information.



# Step 11: Change the display options




(1) Select any combination of options.

(2) Select **Ranges** and move the slider bars to set minimum and maximum values. Only macroblocks within the specified range will display a bits–per–macroblock color.

(3) Close the dialog to apply changes.

To disable the color display, select **None**.

# 12

# Using the Audio ES Analyzer

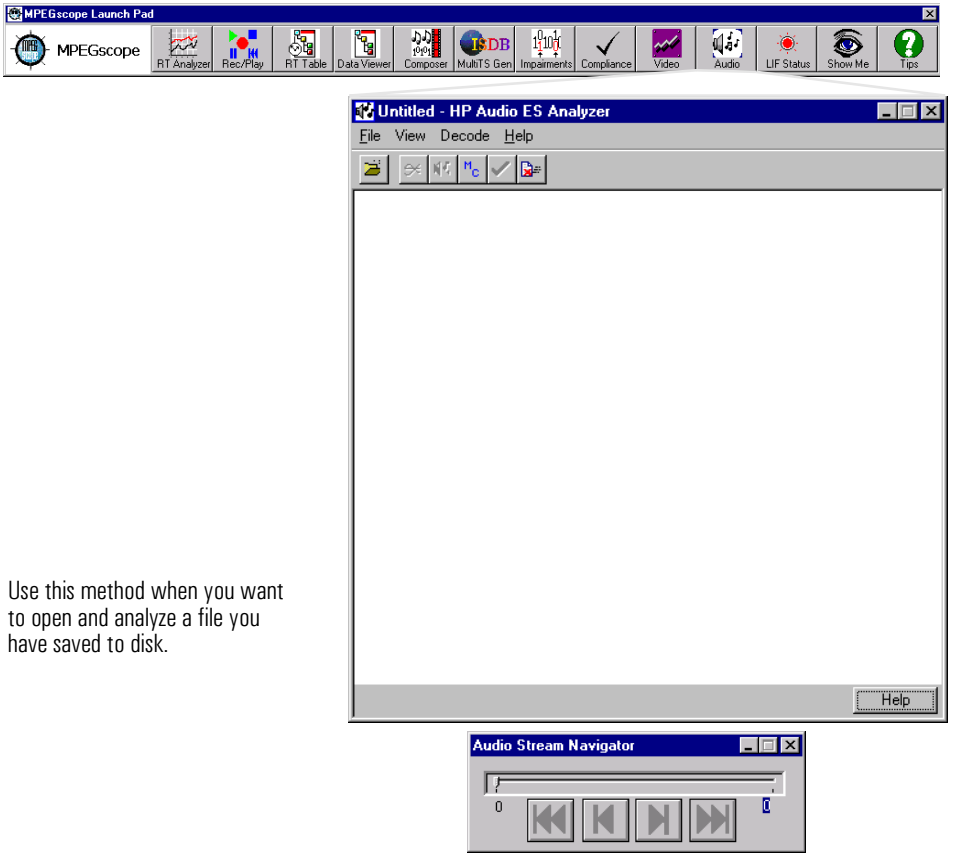
# Analyzing an audio elementary stream


This example requires the purchase of the E6309A Audio ES Analyzer. It shows how to analyze and play out audio elementary streams. The Audio ES Analyzer supports MPEG-1 Layer II, MPEG-2 Layer II, and Dolby Digital AC-3. As of version A.05.03, the Audio ES Analyzer also includes AAC (Audio Data Transport Stream (ADTS) file format) header decodes and stream information.

## Step 1: Start the Audio ES Analyzer

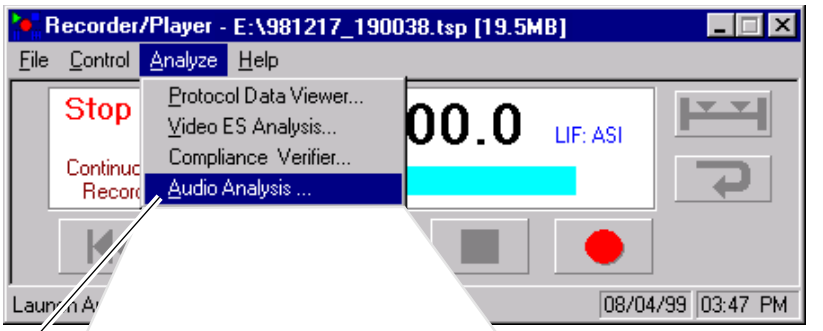
There are two ways to start the Audio ES Analyzer.

From the Launch Pad:

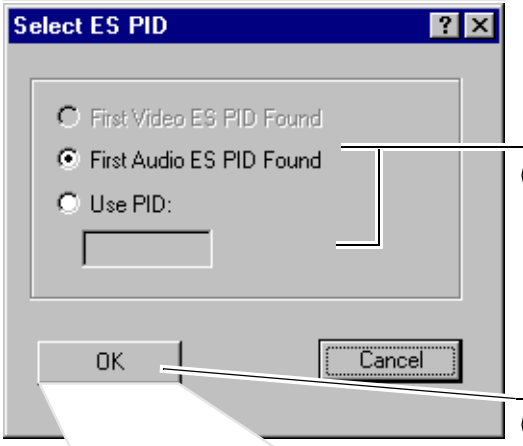



Use this method when you want to open and analyze a file you have saved to disk.

From the Recorder/Player:



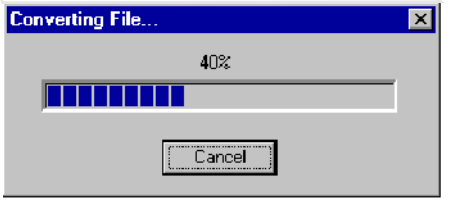

① Start the Audio ES Analyzer.




② Select the first audio elementary stream PID (Program Identifier) found, or enter the PID of the elementary stream to analyze.

③ Extract the audio elementary stream from the file.

When MPEGscope has finished demultiplexing the file, the Audio ES Analyzer will launch with the file loaded.

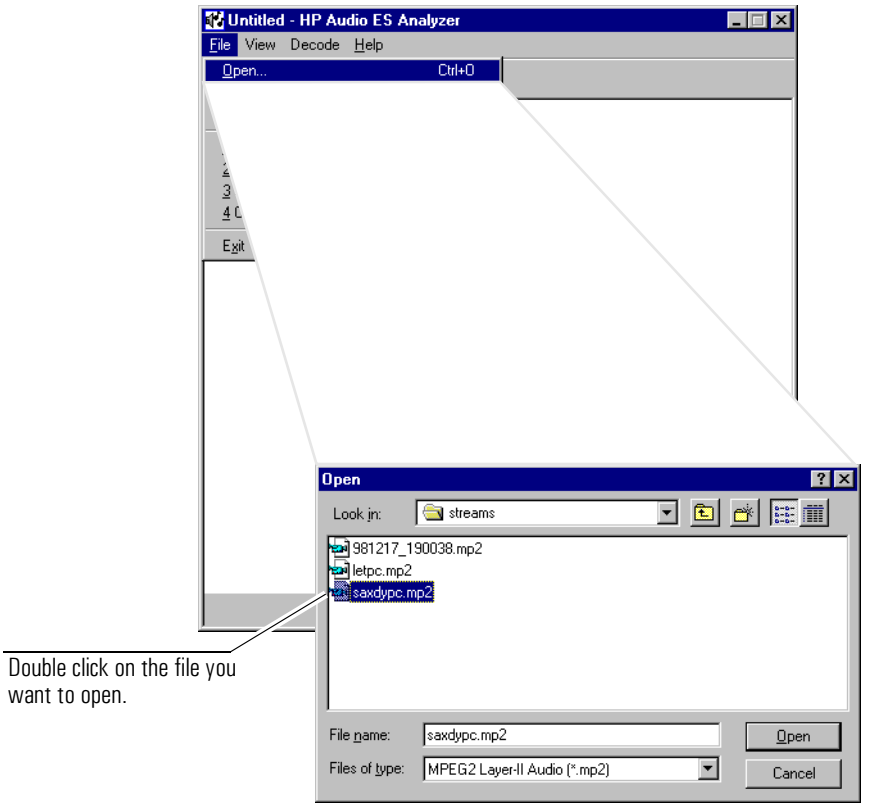



Use this method when you want to extract and analyze an elementary stream from a transport stream file you have opened or recorded from the Recorder/Player.

The Audio ES Analyzer automatically detects the audio file type from the **stream_type** field in the Program Map Table.

## Step 2: Open a file

Complete this step if you have opened the Audio ES Analyzer from the Launch Pad.




When you open an MPEG-1 or MPEG-2 elementary stream file, the file's extension must be **.mp1** or **.mp2**, respectively. If the file has any other extension, the Audio ES Analyzer assumes it is an AC-3 file and will attempt to detect the AC-3 sync word. If this is unsuccessful, you will then receive an "unknown file type" error message. For fastest results, give your audio elementary stream files **.mp1**, **.mp2**, or **.ac3** extensions before opening them from the Audio ES Analyzer.

Note

Before you can analyze an AC-3 audio file with the Audio ES Analyzer, it must be aligned on an AC-3 syncword (0x0B77).

If you have demultiplexed an AC-3 audio stream from a transport stream file, it is likely that the AC-3 audio file will not be aligned on the AC-3 syncword You can align the stream by processing the file through a synchronization utility called **AC3_sync.exe** to drop all bytes occurring before the first syncword, as follows:

1 From the Windows® NT Start/Programs menu, select **MS-DOS Command Prompt**.

2 Change to the HP-Apps\Resources\bin directory.

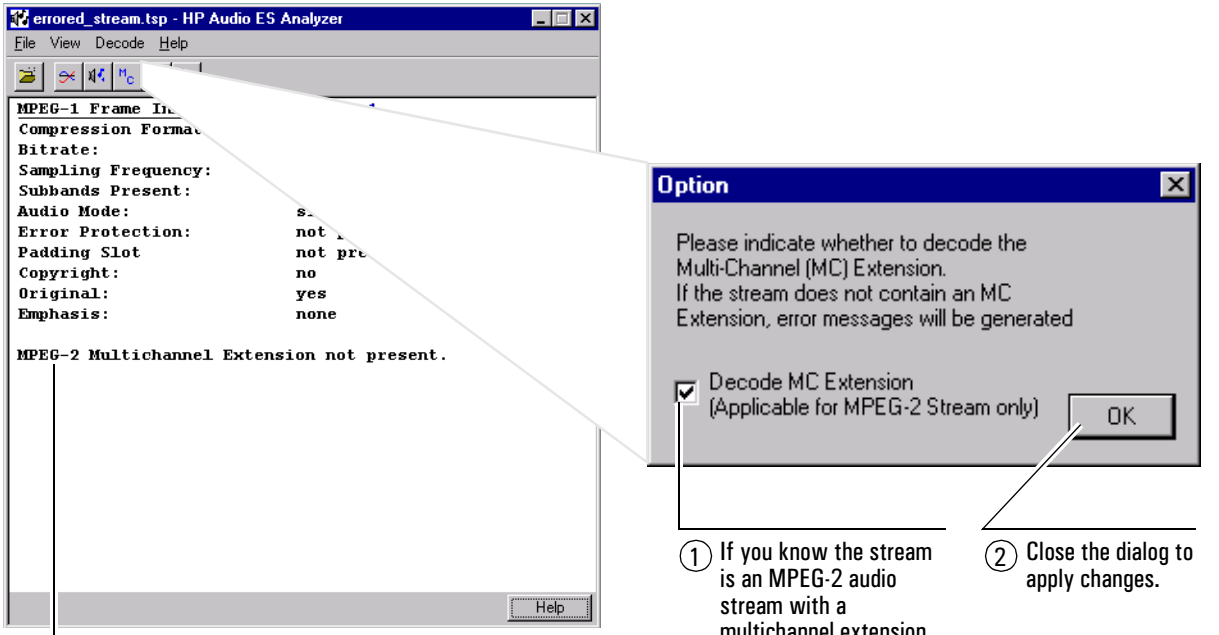
3 Run the utility from the MS-DOS command line, using this syntax:

```
ac3_sync -i inputfilename -o outputfilename
```

If the input and output files are not in the C:\HP-Apps\Resources\bin directory, you must specify the full path names.

## Step 3: Force MC Extension decode

If you open an **.mp1** file or demultiplex an MPEG audio file from a **.ts** or **.tsp** file, the Audio ES Analyzer automatically assumes the stream does not contain a multichannel extension and will not attempt to decode it, as illustrated in the example below. After demultiplexing an MPEG-2 audio stream, you can override this decision by specifically selecting to decode the multichannel extension.




Because it has been demultiplexed from a **.tsp** file, the Audio ES Analyzer assumes this audio stream contains no multichannel extension.

① If you know the stream is an MPEG-2 audio stream with a multichannel extension, select to decode the multichannel extension.

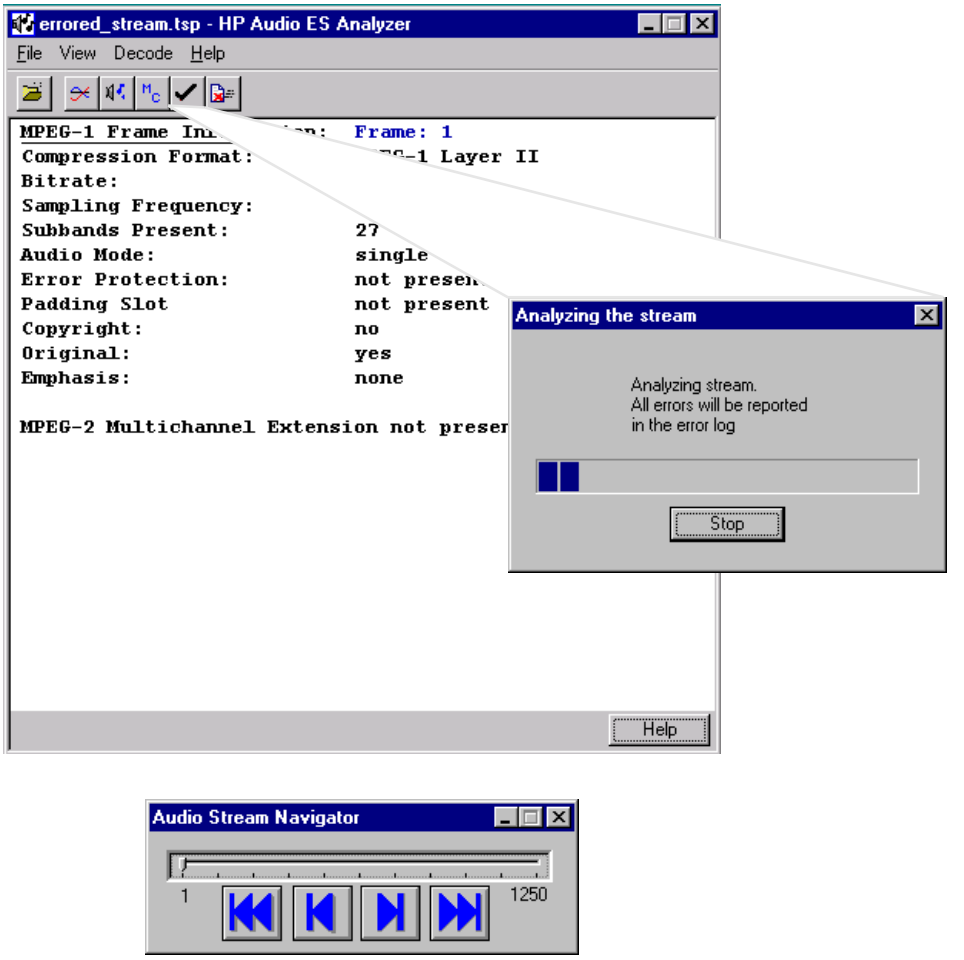
② Close the dialog to apply changes.

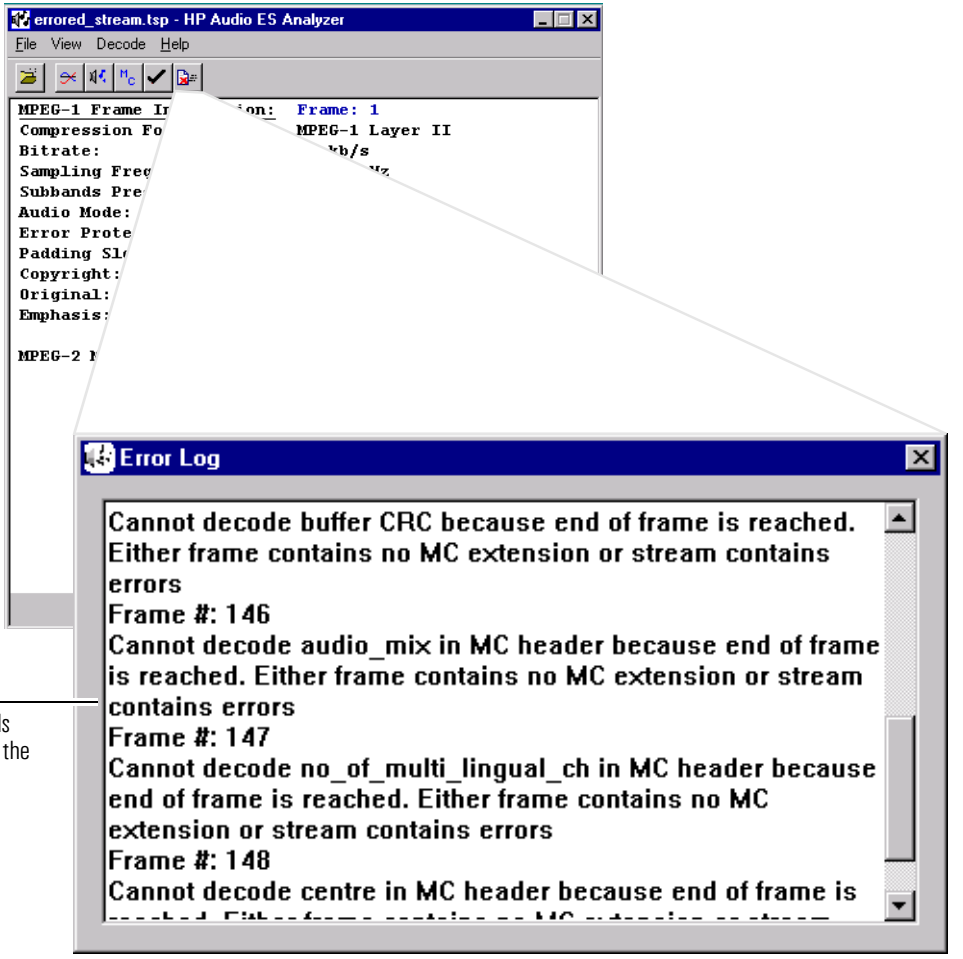
If the **stream_type** field in the Program Map Table defines the audio stream as an **ISO/IEC 11172-3 Audio** stream (MPEG-1), the Audio ES Analyzer will not decode the multichannel extension, even if you select this option.

## Step 4: Analyze the stream

You can analyze the entire stream by clicking the **Analyze** button. Errors will be reported in the **Error Log**.

Alternatively, you can use the **Audio Stream Navigator** to go forwards or backwards in the file. The Audio ES Analyzer will decode and analyze each frame as you navigate the file, then report any errors encountered in the **Error Log**.





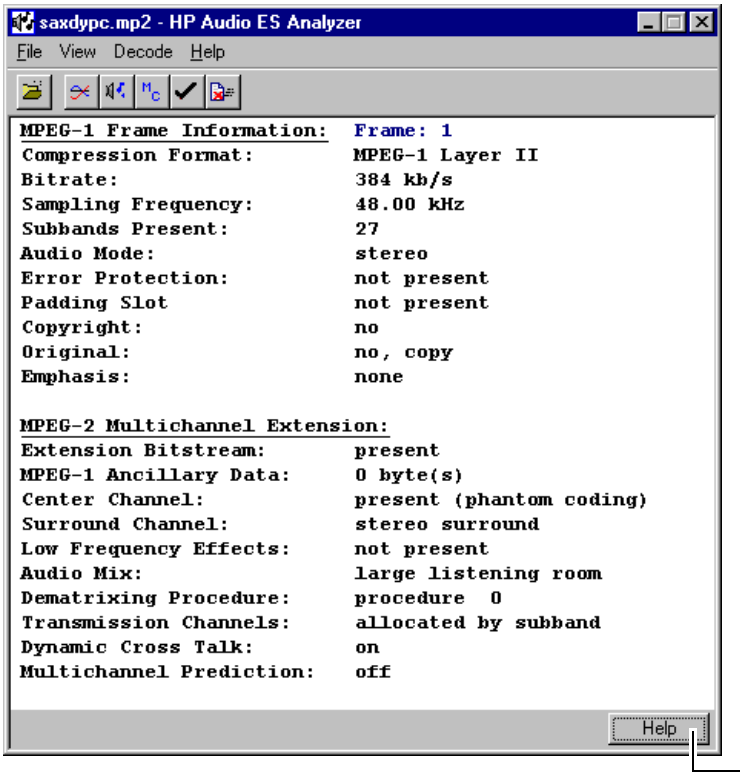

The Audio ES Analyzer records any problems encountered in the **Error Log**.

The errors shown in this example have occurred because a multichannel extension decode was selected when the stream did not contain a multichannel extension. For information on choosing to decode the multichannel extension, refer to "Force MC Extension decode", page 12–6.

## Step 5: View audio header information

After opening a file, the Audio ES Analyzer automatically displays either MPEG, AC-3, or AAC header information in the main dialog, depending on the type of audio file you are analyzing.

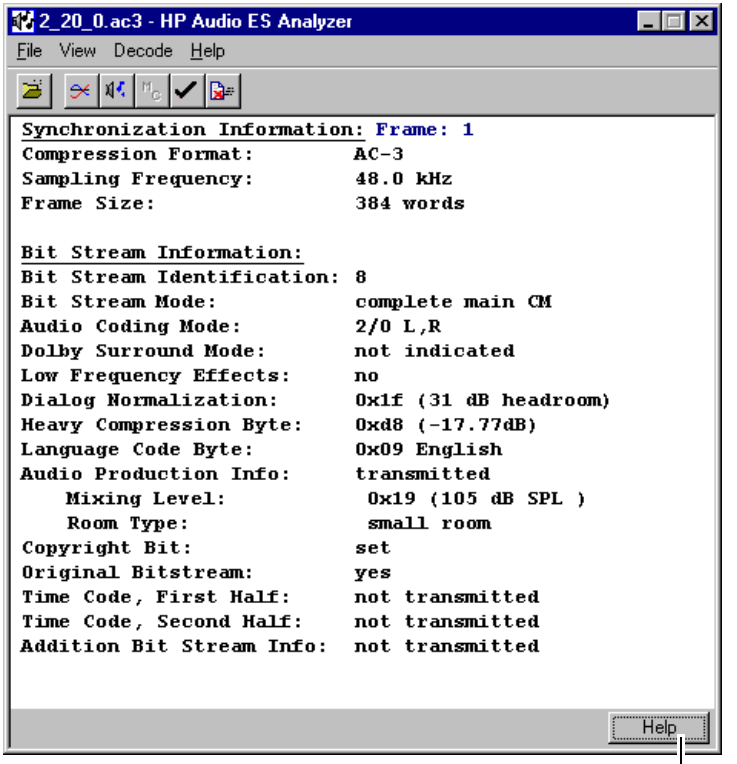
Example of main dialog with MPEG-2 audio file loaded.



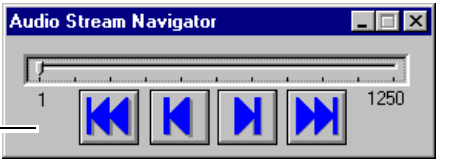

Press the **Help** button for information about the Audio ES Analyzer main dialog.


Use the navigation buttons to move forwards or backwards in the file.

Example of main dialog with AC-3 audio file loaded.

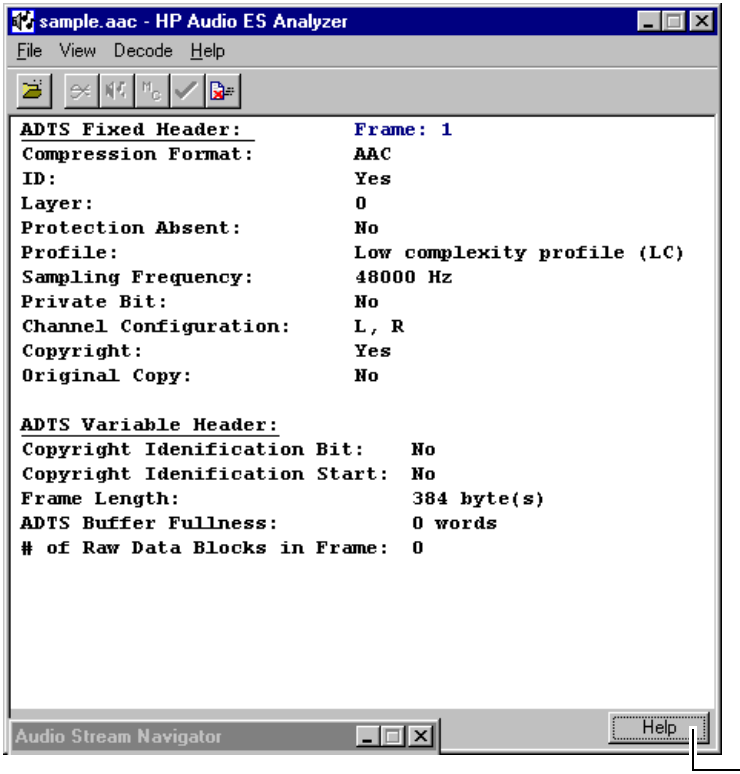



Press the **Help** button for information about the dialog.




Use the navigation buttons to move forwards or backwards in the file.

Example of main dialog with AAC audio file loaded.

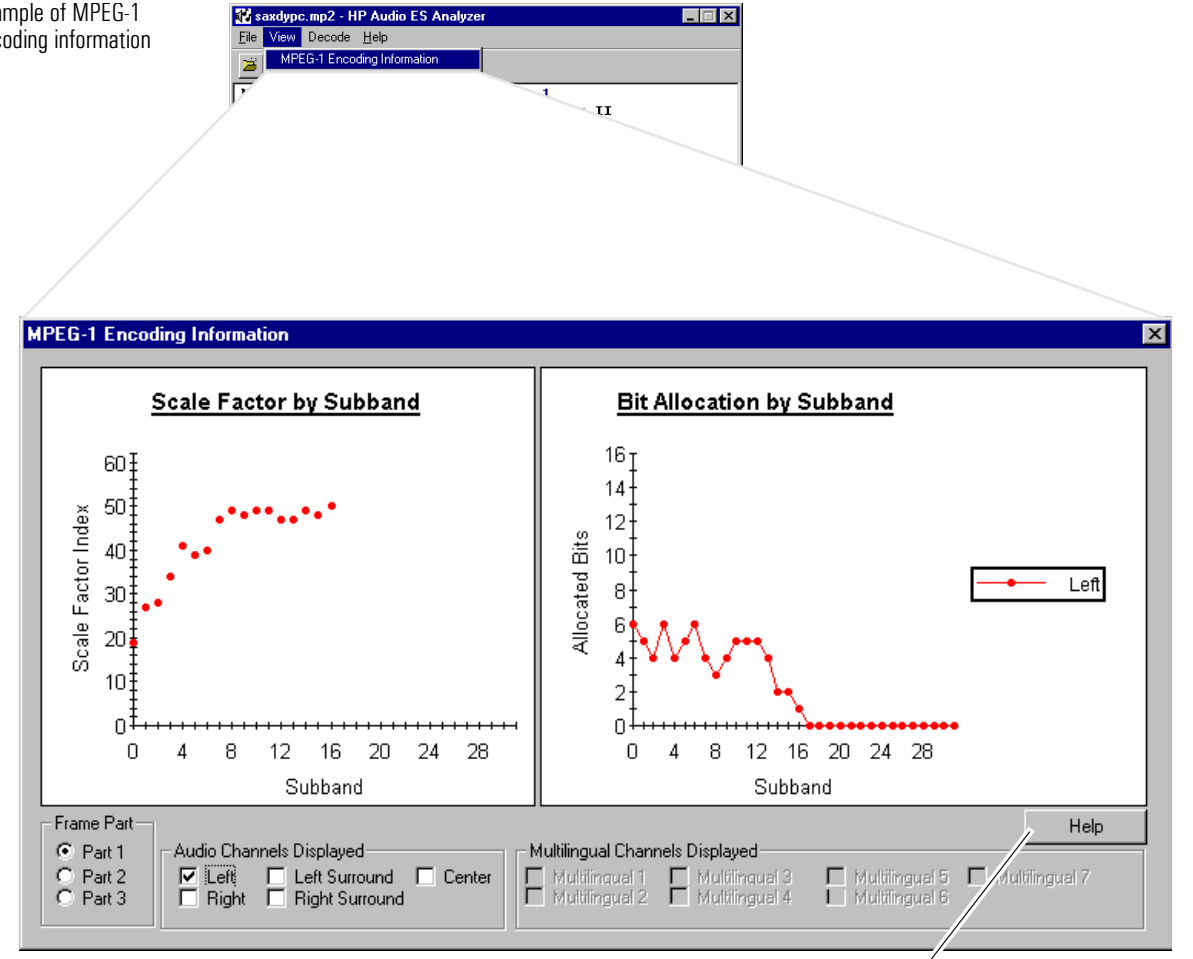



Press the **Help** button for information about the Audio ES Analyzer main dialog.


Use the navigation buttons to move forwards or backwards in the file.

## Step 6: View encoding or stream information

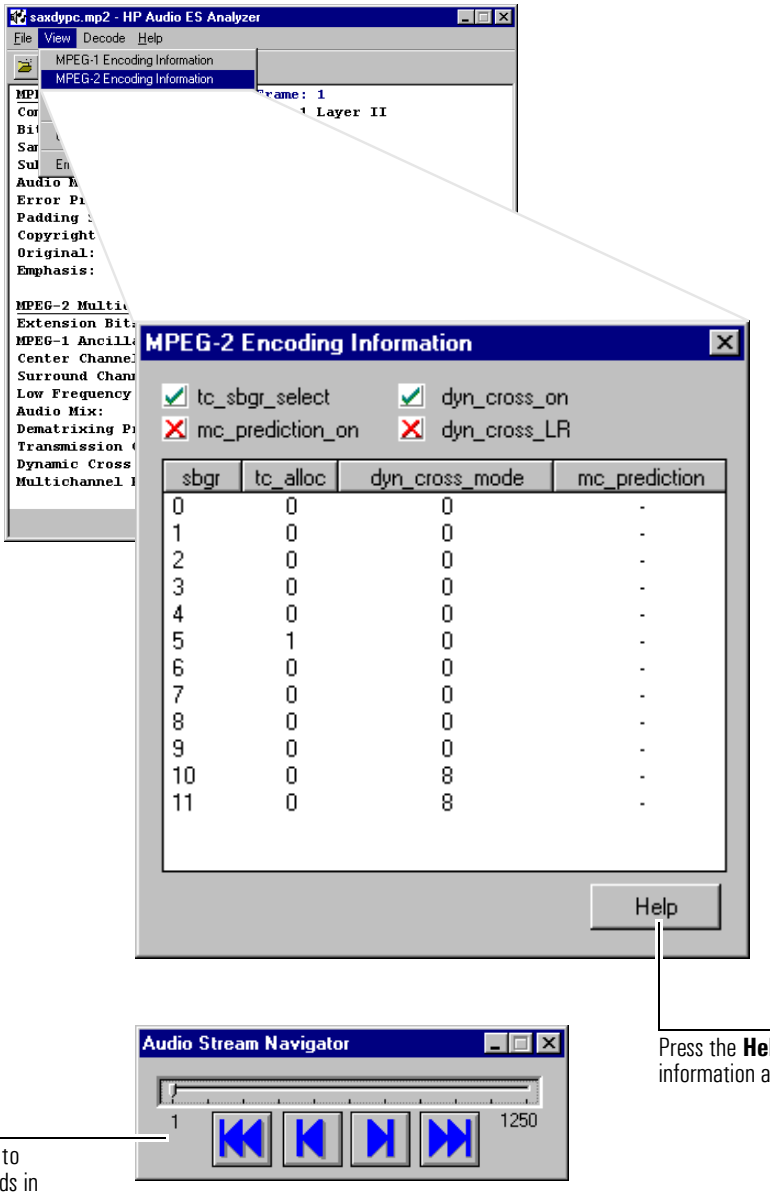
Example of MPEG-1 encoding information




Press the **Help** button for information about the dialog.

Use the navigation buttons to move forwards or backwards in the file.

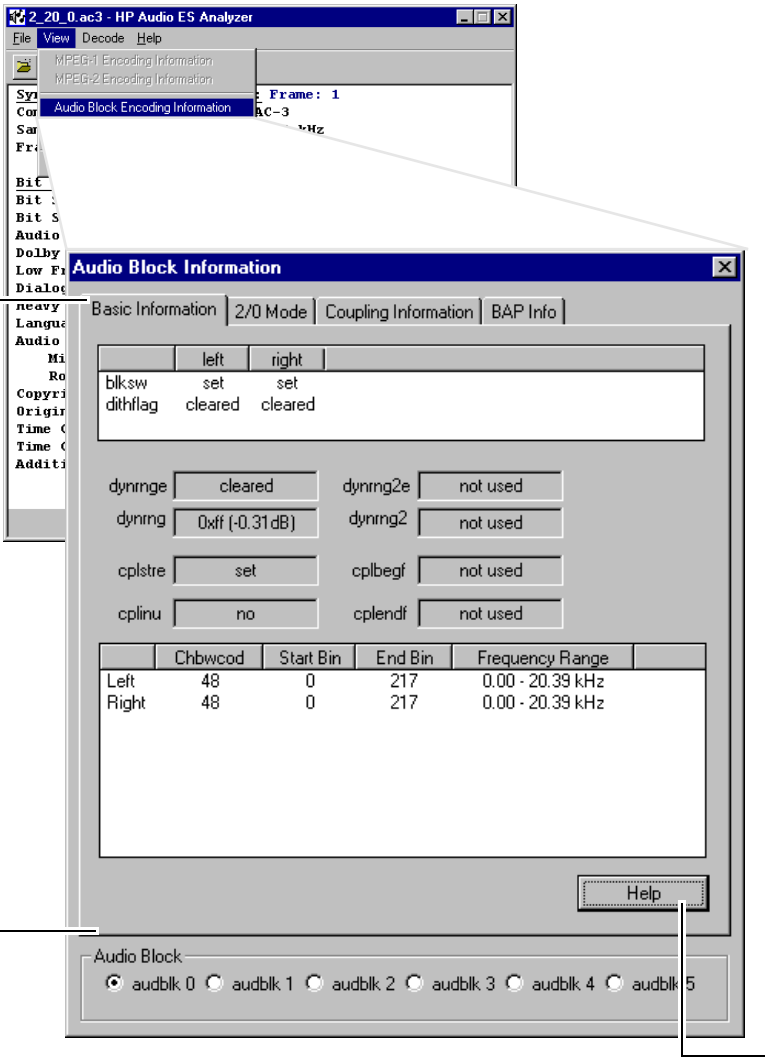
Example of MPEG-2 encoding information




Press the **Help** button for information about the dialog.

Use the navigation buttons to move forwards or backwards in the file.

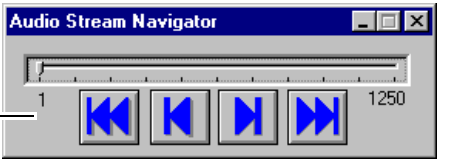
Example of AC-3 encoding information




① Select the information you want to view.

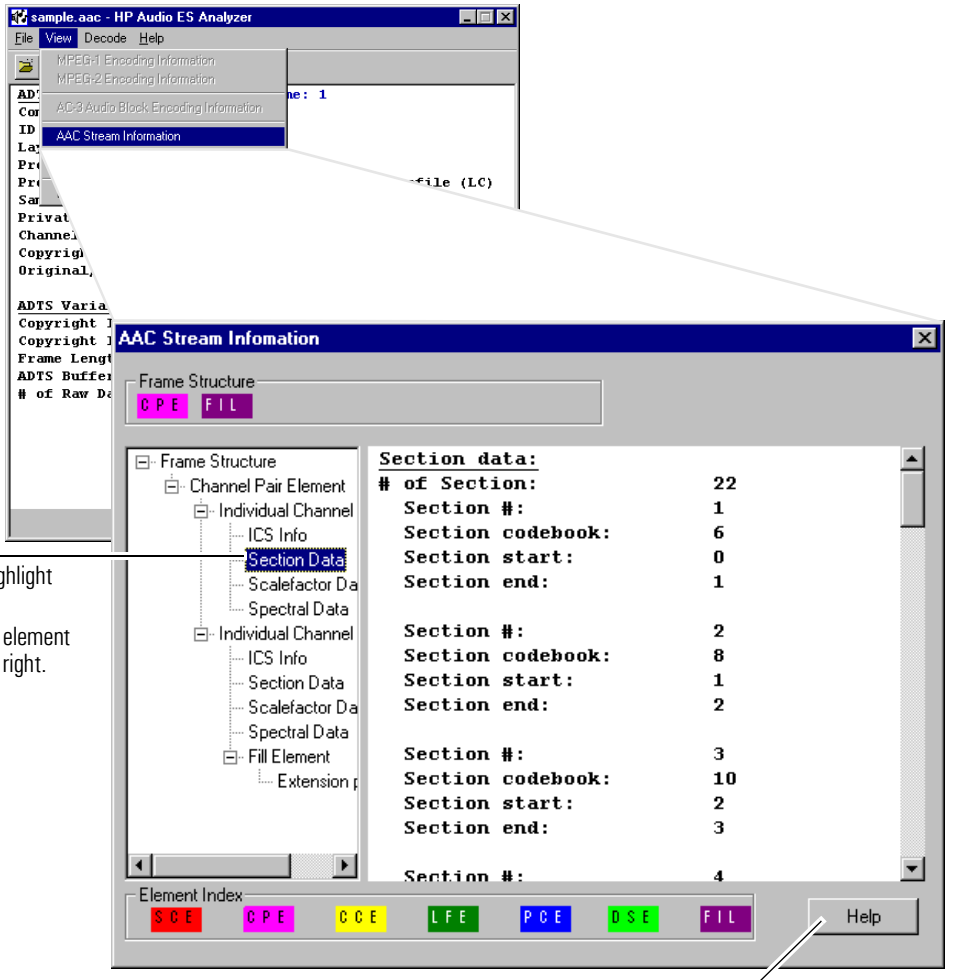
② Select the audio block.

Press the **Help** button for information about the dialog.




Use the navigation buttons to move forwards or backwards in the file.

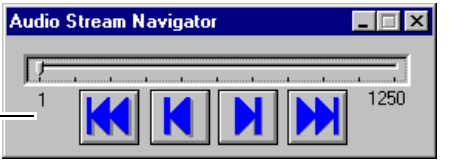
Example of AAC stream information




Expand the tree view, then highlight the element to view.

Stream information about the element displays in the window to the right.

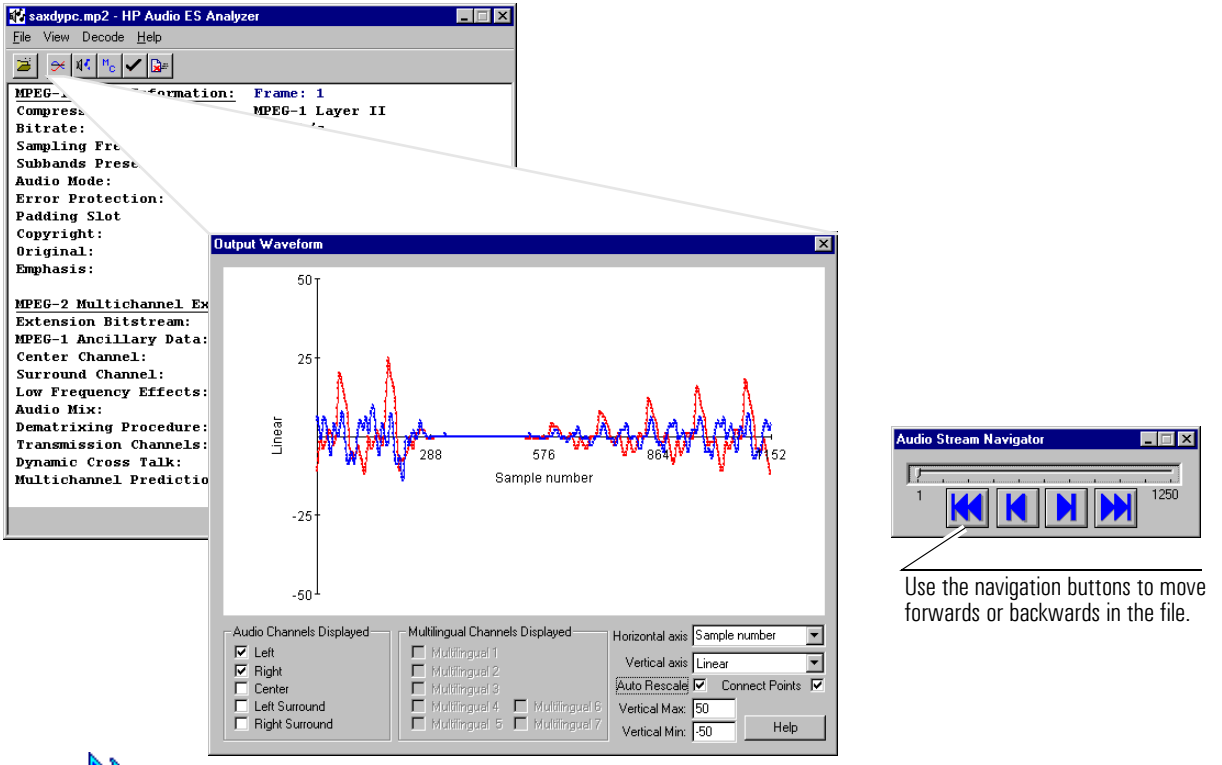
Press the **Help** button for information about the dialog.




Use the navigation buttons to move forwards or backwards in the file.

## Step 7: View PCM samples

The Output Waveform graph shows the PCM (Pulse Code Modulation) sample values of the current frame after decoding.



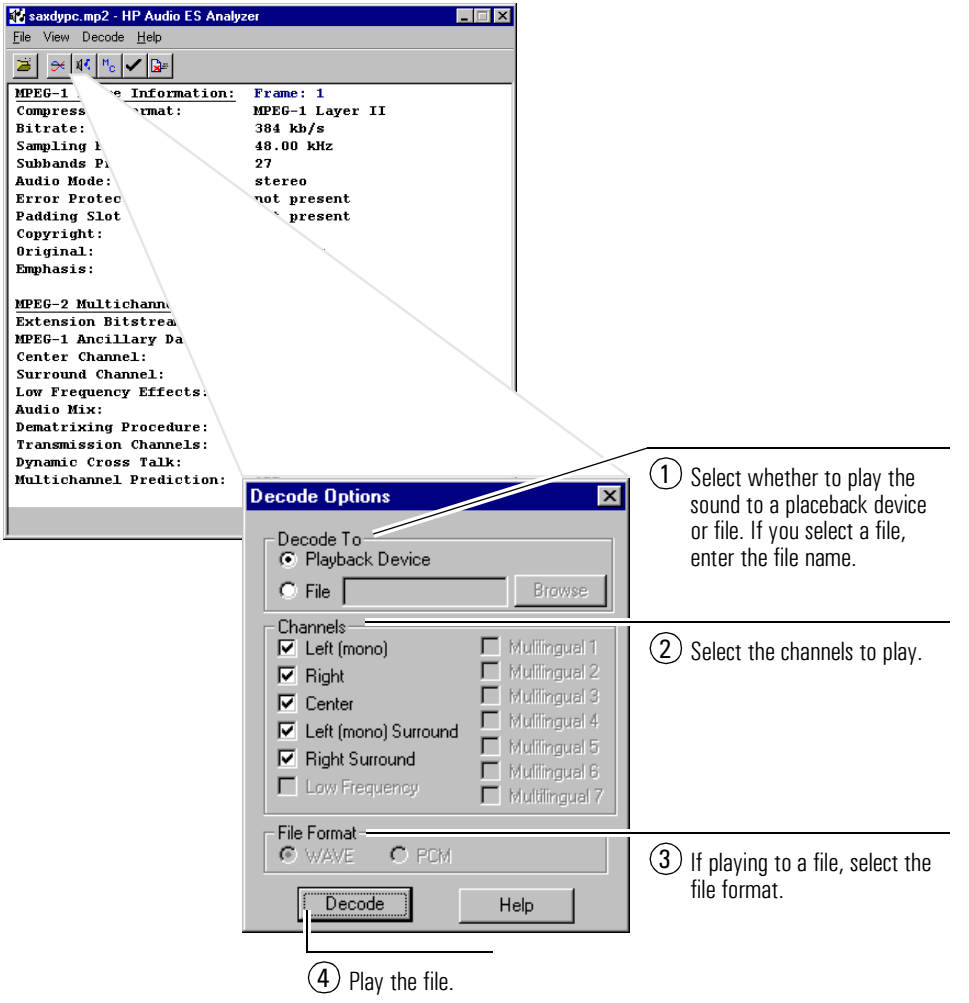

Use the navigation buttons to move forwards or backwards in the file.

### Graph Options

| | |
|---|---|
| **Audio Channels Displayed** | Select the audio channels to display simultaneously on the graph. Individual channels will display in separate colours. |
| **Multilingual Channels Displayed** | Select the multilingual channels to display simultaneously on the graph. Individual channels will display in separate colours. |
| **Horizontal axis** | Select sample number or sample time (in seconds). |
| **Vertical axis** | Select a linear or logarithmic scale. |
| **Auto Rescale** | When enabled, the y-axis automatically rescales as the amplitude range changes. |
| **Connect Points** | When enabled, PCM sample points are connected. |
| **Vertical Max/Vertical Min** | Enter maximum and minimum values to manually rescale the y-axis to set values. |

# Step 8: Play out the stream

Decoding MPEG streams



Decoding AC-3 streams