ANT-20, ANT-20E
Advanced Network Tester

CATS Test Sequencer
Testcase Library
Transmission

BN 3035/95.90, BN 3035/95.91,
BN 3035/95.92, BN 3035/95.93
and for BN 3045/01, BN 3045/93.43

Software Version 4.00

Operating Manual
Developers Guide

Wavetek
Wandel
Goltermann
Communications Test Solutions

Please direct all enquiries to your
local Wavetek Wandel Goltermann sales
company. The addresses are given at the
end of this handbook.

Wavetek Wandel Goltermann
Eningen GmbH & Co.
Mühleweg 5, 72800 Eningen u. A.

Author: W. Besse, A. Baier, H. Bez, MDD/TD
Translator: John Nutley

Order no.: 3035/98.31
Edition: 09/00.07 (V4.00)

Previous edition:
08/00.01 (V4.00)

Printed in Germany

Operating Manual

For latest development changes please read file “readme.txt” of installed software.

Operating Manual

7 Appendix: Frequently asked questions

Notes:

1 Introduction

Automatic testing has become essential for achieving a high degree of test depth, test speed and test repeatability. To save test time and to make full use of expensive test equipment, it is often necessary to run long-term automatic test campaigns overnight or even over the weekend.

Wavetek Wandel Goltermann has selected the popular National Instruments LabWindows CVI / Test Executive product as a platform for the development of a family of test automation products, called CATS (**C**omfortable **A**utomated **T**est**s**).

1.1 Variants of the CATS Test Sequencer

This section summarizes all the variants of the CATS Test Sequencer that are described in this operating manual.

1.1.1 Overview

Variant / BN number	Brief description
CATS (BN 3045/01)	32 bit version. For complex applications with integration of various external test equipment. Allows the ANT-20 to be operated from a separate PC.
CATS DWDM (BN 3045/93.43)	16 bit version. Basically the same as ANT-20 CATS Test Sequencer (BN 3035/95.90). For fully automatic quality analysis of DWDM systems using ANT-20 and OSA-155. Additionally contains optical test cases for the OSA-155.
ANT-20 CATS Test Sequencer (BN 3035/95.90)	Standard version of the ANT-20 CATS Test Sequencer. “All-in-one” package. The CATS Test Sequencer resides directly on the ANT-20’s built-in PC and controls the ANT-20 measurement hardware via an internal interface.
ANT-20 CATS Test Sequencer with TCP/IP interface (BN 3035/95.91)	Special version of the ANT-20 CATS Test Sequencer with TCP/IP option. Allows remote control of the ANT-20 via a TCP/IP link.
ANT-20 CATS Test Sequencer with DDE interface (BN 3035/95.92)	Special version of the ANT-20 CATS Test Sequencer with a standard Windows DDE interface. Allows integration of the ANT-20 into another Windows application, such as MS Excel or MS Access.
BN 3035/95.43	Upgrade to ANT-20 CATS Test Sequencer (BN 3035/95.90)

Table 1-1 CATS Test Sequencer variants

1.1.2 CATS (BN 3045/01)

32 bit version.

The CATS Test Sequencer toolkit is an add-on package for creating automated test systems providing test sequencing (based on single test cases) and test result logging. The software may be used for complex applications that demand the integration of several different test instruments as in a full-blown ATE (Automatic Test Equipment) system.

Test cases are built upon standard LabWindows CVI device drivers. The CATS Test Sequencer therefore allows easy integration of additional instruments.

The software package runs on almost any Windows PC. Controlled devices that use standard SCPI commands and VISA standard drivers can be connected via an IEEE 488.2 or V.24 cable.

A number of CATS Test Sequencer test case packages are available for integrating SDH / PDH / SONET / ATM test equipment (e.g. ANT-20/ANT-20E, ANX-920, ANX-930, ANX-950, etc.) with optical test equipment (e.g. laser sources, attenuators, level meters), test point selectors and other devices. The test cases cover the full range of typical SDH/SONET/ATM and optics measurements and are ready to use at the click of a mouse button.

1.1.3 CATS DWDM (BN 3045/93.43)

16 bit version.

Basically the same as ANT-20 CATS Test Sequencer (BN 3035/95.90).

CATS DWDM is a software application for fully automatic quality assessment measurements on DWDM systems during commissioning and troubleshooting. It combines the optical measurement features of the OSA-155 with the SONET/SDH test facilities of the ANT-20 Network Tester.

CATS DWDM is based on the ANT-20 CATS Test Sequencer concept, extending the original application to include DWDM measurement functions.

CATS is installed on an ANT-20, the OSA-155 is controlled by the ANT-20.

1.1.4 ANT-20 CATS Test Sequencer (BN 3035/95.90)

The ANT-20 CATS Test Sequencer is a special “all-in-one” package in the CATS Test Sequencer Test Automation family. The package includes the LW CVI based CATS Test Sequencer and a complete range of test cases for the ANT-20/ANT-20E and DominoCom, including PDH, SDH, SONET and ATM tests.

Instead of controlling the test instrument from an external PC via remote control interfaces such as the IEEE488.2 bus or V.24 interface, the CATS Test Sequencer resides directly on the ANT-20’s built-in PC and controls the ANT-20 measurement hardware via an internal interface. It replaces the usual interactive ANT-20 graphical user interface and allows completely automatic sequences of pre-defined and user-definable test cases to be run.

This special “low-cost” version does not provide for integration of external test sets. If test equipment other than the ANT-20 is to be controlled from the ANT-20’s built in PC, the “full” CATS Test Sequencer (BN3045/01) and its test case packages should be used.

All test cases developed for running on an external PC will also run on the ANT-20 itself.

There is an upgrade for the ANT-20 CATS Test Sequencer (BN 3035/95.90). The order number is BN 3035/95.93.

1.1.5 ANT-20 CATS Test Sequencer with TCP/IP interface (BN 3035/95.91)

The ANT-20 CATS Test Sequencer is also available in a special version that allows it to be remote-controlled via a TCP/IP Ethernet interface.

The software package includes all the functions of the ANT-20 CATS Test Sequencer, i.e. all available test cases, and it can be run on the ANT-20 in standard mode just like the normal package (BN 3035/95.90). In addition to these existing functions, test cases can also be called via Ethernet (TCP/IP) using simple commands like “execute_test” or “set_parameters”. The test case called is then automatically executed and the results including a PASS/FAIL statement are transferred back to the calling platform.

As the package is based on the standard socket mechanism (which means it can be called from most of the software platforms used in manufacturing and network management), it allows you to integrate the ANT-20 very easily and effectively into virtually any system environment. Instead of using “low level” programming and SCPI commands, you can instantly use test cases that have been prepared and fully debugged by WWG. All tests prior to integrating the ANT-20 into the user’s environment can be done easily and effectively on the ANT-20 using the standard ANT-20 CATS Test Sequencer. The CATS Test Sequencer with TCP/IP interface is a unique solution for integrating the ANT-20 into user environments quickly.

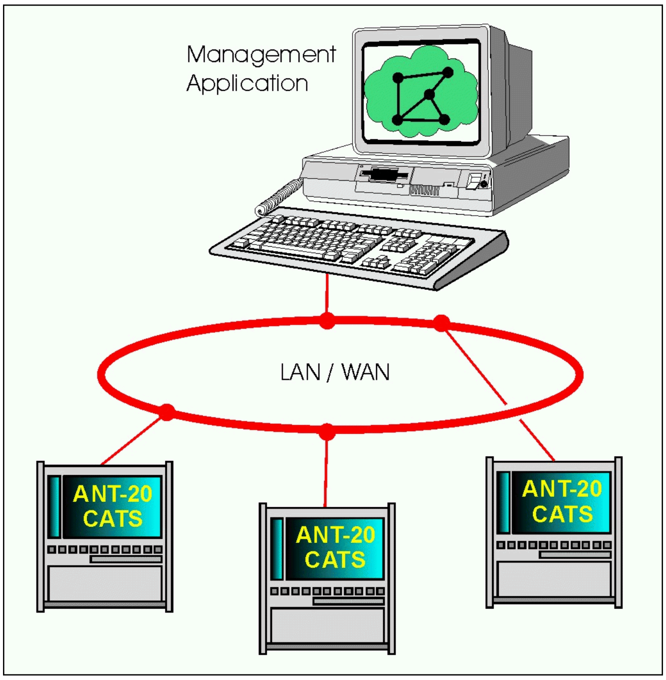


Fig. 1-1 ANT-20 CATS Test Sequencer with TCP/IP interface (integrated into a management system)

1.1.6 ANT-20 CATS Test Sequencer with DDE interface (BN 3035/95.92)

This package provides exactly the same functions as the ANT-20 CATS Test Sequencer with TCP/IP interface (BN3035/95.91), but uses the standard Windows DDE mechanism to communicate instead of sockets. There are no differences in the formats of the command syntax and the results that are returned.

This software allows the ANT-20 CATS Test Sequencer to be integrated easily into other Windows applications running on the ANT-20, e.g. MS EXCEL or MS ACCESS. Test cases can then be fully integrated and called using macros written using these standard Windows applications.

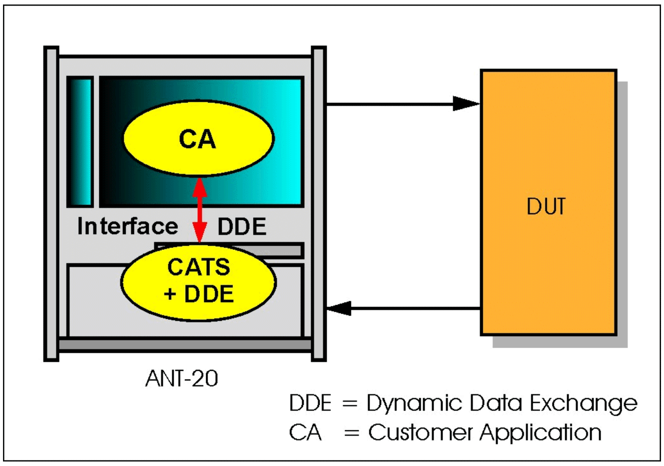


Fig. 1-2 ANT-20 CATS Test Sequencer with DDE interface

1.2 About this operating manual

This operating manual describes the operation and functions of the following CATS Test Sequencer variants:

- ANT-20 CATS Test Sequencer (BN 3035/95.90)
- ANT-20 CATS Test Sequencer with TCP/IP interface (BN 3035/95.91)
- ANT-20 CATS Test Sequencer with DDE interface (BN 3035/95.92)
- CATS (BN 3045.01)
- CATS DWDM (BN 3045/93.43)

It also includes all the information required for updating and creating test cases and test sequences.

2 Test sequences

This section contains a brief explanation of the terms test sequence and test case, together with descriptions of the sequences supplied by Wavetek Wandel Goltermann with the CATS Test Sequencer.

2.1 General information

The CATS Test Sequencer test automation products provide the framework for running, creating or modifying sequences of test cases. Comprehensive libraries of standard test cases for PDH, SDH, SONET, ATM, optical communications and test point selectors are available. The ANT-20 CATS Test Sequencer is supplied with a complete set of standard test cases. Test case libraries can be ordered as options for the CATS Test Sequencer.

Note: The CATS Test Sequencer can also be used to run test cases that have been created using the developer's environment of LabWindows CVI and CATS Test Sequencer. See CATS Test Sequencer developer's manual for details.

Although it is possible to create completely new sequences by selecting “Sequence”, “Edit Sequence” in the main menu, we recommend that you use the sequences provided by Wavetek Wandel Goltermann as a basis for your own developments unless you are fully acquainted with the functions of the CATS Test Sequencer.

2.1.1 What is a sequence?

A CATS Test Sequencer sequence:

- is a collection of test cases, which are executed when the sequence is executed
- is defined using the CATS Test Sequencer sequence editor
- is stored as a standard file using the file extension *.SQU

A typical sequence includes a list of test cases, setup/cleanup functions, preconditions for flow control based on Pass/Fail results, test case report file information, description of the sequence, and database information.

The sequences supplied are stored in the directory **./sequence**. Preconditions can be used to define how the test cases of a sequence are executed.

2.1.2 What is a test case?

A CATS Test Sequencer test case consists of:

- Test algorithm
- Graphical user interface to set test parameters

2.1.2.1 Test algorithm

The test algorithm allows the performance of specific actions, such as setting up the instrument, interacting with the user, starting a measurement, evaluating test results against thresholds, etc. This test algorithm is a C-function that is found in a LabWindows CVI object file (file extension

".obj"). You do not need to know the details of this C function to create and modify test sequences, but you must know the name of the function and the name of the ".obj" file where it can be found.

Note: All test case names, object files and detailed descriptions of what they actually do can be found in section 7.

2.1.2.2 Test case user interface

The user interface of each test case in a sequence can be opened by double-clicking on the test case in the "Sequence" display or by clicking the [Set param.] button in the "Test Sequencer" window. Most CATS Test Sequencer test cases are "generic", which means that they can be used for a number of different signal structures, etc.

Note: Each test case in a sequence is independent of any preceding test cases.

2.1.2.3 Test case relevance

The CATS Test Sequencer distinguishes between test cases that are relevant to the status of the test sequence (such as a bit error check or an alarm sensor test) and those which do not affect the outcome of the test sequence, such as those that set up an instrument or send a message to the user.

- Test cases that are relevant to the test sequence status are displayed in red (status can be "PASS" or "FAIL")
- Test cases that are not relevant to sequence status are displayed in blue (status will always be "DC" = Don't Care)

Note: If a test case marked as "Don't Care" fails, the sequence is stopped automatically. This is an illegal situation and should never occur in a debugged sequence.

2.2 Sequence types

The following types of sequence are available. They are distinguished by the start of the sequence name:

Sequence type	Start of sequence name
Demo sequence	"_" (underscore)
Sequence for ANSI bit rates	"a_"
Sequence for ITU-T bit rates	"i_"

Table 2-1 Sequence types

2.3 Changes made to sequences after this manual was printed

The descriptions of sequences in this operating manual were correct at the time of going to press.

Any later changes to sequences and any new sequences are described in the "readme" file of the software.

2.4 Overview of all sequences supplied

This section lists all the test sequences supplied with the software together with their brief descriptions. Detailed descriptions, including descriptions of the test cases included in the sequences, are found in the following three sections: Sec. 2.5, Page 1-10, Sec. 2.6, Page 1-21, and Sec. 2.7, Page 1-40.

Sequence name	Brief description	For exact description, see:
_demo_e1.squ	Line-up of an E1 2Mbit/s PCM30CRC circuit, unbalanced connection	Sec. 2.5.1, Page 1-10
_demods1	Includes a transparency check and a number of alarm sensor checks for a DS-1 signal.	Sec. 2.5.2, Page 1-12
_dem_c12.squ	Line-up of a STM1 VC12 ADM for a list of tributaries	Sec. 2.5.3, Page 1-14
_demooc1.squ	Line-up of an OC-1 circuit that carries a DS-1 framed SF	Sec. 2.5.4, Page 1-15
_demooc3.squ	OC-3 with embedded STS-1 and DS-1 SF	Sec. 2.5.5, Page 1-17
_toolkit.squ	Demonstrates most non-instrument related test cases.	Sec. 2.5.6, Page 1-20
a_almdyn.squ	Insertion of dynamic alarms.	Sec. 2.6.1, Page 1-21
a_atmtra.squ	Transparency check of a number of different ATM VPI and VCI pairs.	Sec. 2.6.2, Page 1-21
a_aps.squ	APS time measurement.	Sec. 2.6.3, Page 1-22
a_atmqos.squ	QoS measurement for ATM 149 Mbit/s with embedded OC-3.	Sec. 2.6.4, Page 1-23
a_atmsca.squ	Scan of different VCI/VPI of ATM 149 Mbit/s with embedded OC-3c.	Sec. 2.6.5, Page 1-24
a_delay.squ	Shows how to measure round trip delay for STS-1 DS-3.	Sec. 2.6.6, Page 1-25
a_ds1.squ	Includes a transparency check and a number of alarm sensor checks for a DS-1 signal.	Sec. 2.6.7, Page 1-25
a_cj_ds1.squ	Combined jitter measurement.	Sec. 2.6.8, Page 1-25
a_dwdm.squ	Combination of ANT-20 and OSA-155.	Sec. 2.6.9, Page 1-26
a_jit_48.squ	Shows how to setup jitter measurements for OC-48.	Sec. 2.6.10, Page 1-27
a_oc48.squ	Tests for OC-48	Sec. 2.6.11, Page 1-28
a_jitter.squ	Shows how to setup jitter measurements.	Sec. 2.6.12, Page 1-29
a_list.squ	Shows how to use the “LIST” feature.	Sec. 2.6.13, Page 1-30
a_mj_ds1.squ	Shows how to find critical mapping offset for DS-1 1.5 Mbit/s.	Sec. 2.6.14, Page 1-31
a_oc3_mi.squ	Test of maintenance interactions at the line, path and VT level of an OC-3 signal	Sec. 2.6.15, Page 1-33
a_oc192.squ	Tests for OC-192	Sec. 2.6.16, Page 1-35
a_sts3.squ	Typical tests for STS-3	Sec. 2.6.17, Page 1-36
a_scan.squ	Scans lists of selected tributaries embedded in an STS-1.	Sec. 2.6.18, Page 1-38
i_almdyn.squ	Shows how to insert dynamic alarms.	Sec. 2.7.1, Page 1-40

Table 2-2 Sequence overview

Sequence name	Brief description	For exact description, see:
i_aps.squ	Shows APS time measurement.	Sec. 2.7.2, Page 1-41
i_atmdel.squ	Shows how to measure cell delay variation of an ATM signal.	Sec. 2.7.3, Page 1-42
i_atmqos.squ	Quality of Service measurement for an ATM 149 Mbit/s signal with embedded STM-1.	Sec. 2.7.4, Page 1-43
i_atmsca.squ	Scan of different VCI / VPI of an ATM 149 Mbit/s signal with embedded STM-1.	Sec. 2.7.5, Page 1-44
i_atmtra.squ	Transparency check of a number of different ATM VPI and VCI pairs	Sec. 2.7.6, Page 1-45
i_c4.squ	Typical measurements for an STM-1 AU4 signal	Sec. 2.7.7, Page 1-46
i_c4_bis.squ	Bringing into service of an STM-1 VC4 circuit	Sec. 2.7.8, Page 1-48
i_c4_mi.squ	Checks all the maintenance interactions (defects and anomalies) of a STM-1 VC4 path.	Sec. 2.7.9, Page 1-50
i_c3.squ	Typical measurements for an STM-1 AU4 VC3 signal	Sec. 2.7.10, Page 1-52
i_delay.squ	Shows how to measure round trip delay.	Sec. 2.7.11, Page 1-53
i_dwdm.squ	Combination of ANT-20 and OSA-155.	Sec. 2.7.12, Page 1-54
i_e1_bis.squ	Bringing into service of an E1–2 Mbit/s transparent circuit	Sec. 2.7.13, Page 1-54
i_e1_crc.squ	Line-up of a 2 Mbit/s PCM30CRC terminated circuit	Sec. 2.7.14, Page 1-56
i_e1_trs.squ	Basis for line-up of a 2 Mbit/s non-terminated transparent circuit	Sec. 2.7.15, Page 1-58
i_cj_e1.squ	Combined jitter measurements	Sec. 2.7.16, Page 1-59
i_e1_jit.squ	Jitter measurements for E1 2 Mbit/s.	Sec. 2.7.17, Page 1-59
i_mj_e1.squ	Mapping jitter measurement	Sec. 2.7.18, Page 1-59
i_list.squ	Shows how to use the “LIST” feature.	Sec. 2.7.19, Page 1-59
i_o_172.squ	Assesses the mapping jitter and pointer jitter of a 2 Mbit/s signal derived from an STM-1 C12 signal.	Sec. 2.7.20, Page 1-60
i_g826.squ	Checks correct operation of the performance monitoring sensors of an STM-1 C4 card.	Sec. 2.7.21, Page 1-61
i_jitter.squ	Jitter measurements for STM-1	Sec. 2.7.22, Page 1-64
i_4jitt.squ	Jitter measurements for STM-4	Sec. 2.7.23, Page 1-64
i_stm4.squ	Tests for STM-4	Sec. 2.7.24, Page 1-64
i_jit16.squ	Shows how to use the STM-16 jitter module with CATS and includes jitter measurements, MTJ, FTJ and F-MTJ.	Sec. 2.7.25, Page 1-64
i_stm16.squ	Tests for STM-16	Sec. 2.7.26, Page 1-66
i_stm64.squ	Tests for STM-64	Sec. 2.7.27, Page 1-67
i_pdhsca.squ	Scans E4 140 Mbit/s PDH of selected embedded tributaries.	Sec. 2.7.28, Page 1-67

Table 2-2 Sequence overview

Sequence name	Brief description	For exact description, see:
i_scan.squ	Scans list of TUG3, TUG2, TU11and AU3 tributaries of a SDH signal.	Sec. 2.7.29, Page 1-68
i_wantol.squ	Tests the wander tolerance of a STM-1 circuit.	Sec. 2.7.30, Page 1-68
synch.squ	Measures the time delay between a stimulus of one specific SDH overhead byte (ANT-20 TX) and the subsequent reaction of another (or the same) overhead byte at the ANT-20 RX.	Sec. 2.7.31, Page 1-69

Table 2-2 Sequence overview

2.5 Demo sequences

This section describes the demo sequences that are provided with the software. Demo sequences demonstrate various aspects of the CATS software as examples.

2.5.1 _demo_e1.squ

This sequence is a demo example for the line-up of a E1 2 Mbit/s PCM30CRC circuit using an unbalanced connection. It includes a bit error test, a pulling range test, various alarm sensor tests and G.826 long-term analysis.

To demonstrate the behavior of G.826 analysis when defects or anomalies occur, this demo sequence also contains test cases that can inject such events while the G.826 analysis is running.

To be useful in a “real-life” environment, the parameters for the alarm sensor tests need to be adapted (e.g. LOS expected when LOS is sent will work fine with a coaxial cable between RX and TX of the ANT-20. With a real unit under test (UUT) between the two, AIS would be expected when LOS is sent).

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
Show Bitmap _show_image	Displays a bit map.
Enter UUT Data _show_info_field	Opens a window for entering data.
TX=RX 2 Mb/s CRC _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX HDB3 PRS11 _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX HDB3 PRS11 _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level. Additional parameters are sensitivity and equalizer settings.

Table 2-3 “_demo_e1.squ” test sequence: test cases used and their functions

Test case, function name	Function
View Alarm _read_actual_alarms	Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval). This test case does not have automatic PASS/FAIL evaluation.
Continuity Check _check_transparency	Checks the transparency of a tributary or channel that was defined by a previous signal structure and tributary setup test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Check RDI _check_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms.
Check LOS _check_alarm	See “Check RDI” test case.
Check AIS _check_alarm	See “Check RDI” test case.
FAS Continuity _check_stimulus_response	This test stimulates the error indication sensors of SDH or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site.
Pulling Range _check_clock_pulling_range	Checks the pulling range of a UUT by transmitting a signal with an increasing clock offset and checking for bit errors and alarms during the measurement.
G.826 Setup/Start _set_and_start_g826	Sets the parameters of a G.826 evaluation.
Inject Errors _set_stimulus	Inserts an error ratio or frequency offset until switched off by the “Error Inject OFF” test case.
wait _wait	Wait for a specified period.
Error Inject. OFF _set_stimulus	Stops the “Inject Errors” test case.
G.826 Analysis _check_g826	Checks the results of the G.826 evaluation.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-3 “_demo_e1.squ” test sequence: test cases used and their functions

2.5.2 _demods1.squ

This sequence includes a transparency check and a number of alarm sensor checks for a DS1 signal. It also includes long-term monitoring (check all events) and a check of pulling range and Alarm N in M.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX=RX DS1 FRM SF _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, etc. Additional parameters are sensitivity and equalizer settings.
TX=RX Tributaries _set_sonet_tributaries	Sets the tributary/channel number of an ANSI/SONET signal. All subsequent test cases are performed on the tributary/channel defined here.
Monitor Alarm _read_sonet_actual_alarms	Reads the current SONET/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval).
Check Alarm LOF-DS1 _check_sonet_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. You can easily change the test case parameters to suit specific test needs. This test case switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by “_set_alarm” and “_set_stimulus” will continue until switched off specifically.
Payl.-Transprncy _check_sonet_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Clock Pull. Range _check_clock_pulling	Checks the pulling range of a UUT by transmitting a signal with an increasing clock offset and checking for bit errors and alarms during the measurement.

Table 2-4 “_demods1.squ” test sequence: test cases used and their functions

Test case, function name	Function
Set Stimulus _set_sonet_stimulus	Inserts an error ratio or frequency offset until switched off by the next test case. This test case can be combined with the "_wait" test case to define insertion time. To switch off error insertion, "_set_sonet_stimulus" must be called again with the stimulus set to "Switch off error insertion".
Check All Events _check_trouble	Checks the ANT-20's event queue for events (alarms, errors, ...).
Set Stimulus OFF _set_sonet_stimulus	Ends the "Set Stimulus" test case.
Check AIS-DS1 _check_sonet_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. You can easily change the test case parameters to suit specific test needs. This test case switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by "_set_alarm" and "_set_stimulus" will continue until switched off specifically.
Check Ratio FE-DS1 _check_sonet_stimulus_response	This test stimulates the error indication sensors of SONET or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site. You can specify an upper/lower limit for the expected result.
Check N * TSE _check_sonet_stimulus_response	See "Check Ratio FE-DS1" test case.
Set Alarm N in M _set_sonet_alarm_N_in_M	Inserts specific alarm conditions for a specific number of frames, followed by a specific number of frames without alarms. This sequence is repeated periodically until switched off by calling this test case with the parameter "all alarm insertions off".
Check Events _check_trouble	Checks the ANT-20's event queue for events (alarms, errors, ...).
Alarm N in M OFF _set_sonet_alarm_N_in_M	Ends the "Set Alarm N in M" test case.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on "OK".

Table 2-4 "_demods1.squ" test sequence: test cases used and their functions

2.5.3 _dem_c12.squ

This sequence is a demo example for the line-up of an STM-1 VC12 ADM. The connection is made via an unbalanced electrical line (STM-1). A transparency check and a number of alarm sensor tests are carried out for all 63 or for a user defined number of VC-12 channels.

To be useful in a “real-life” environment, the parameters for the alarm sensor tests need to be adapted (e.g. TU-LOP expected when TU-LOP is sent will work fine with a coaxial cable between RX and TX of the ANT-20. With a real unit under test (UUT) between the two, TU-AIS would be expected when TU-LOP is sent).

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
check for options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
option error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
Show Bitmap _show_image	Displays a bitmap.
TX STM1 AU4 C12 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX STM1 AU4 C12 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Define VC-12 List _setup_mscan_trib	Defines a list of tributaries which will be set with the _mscan_set_inc_trib test case. This list can be used In combination with a GOTO to scan several tributaries and carry out any number of additional test cases for each tributary from the defined list.
Increment VC12-Trib _mscan_set_inc_trib	Selects the next tributary from the list defined with the _setup_mscan_trib test case. When the end of the list is reached, no new tributary will be set and the test case status is PASS.
Transp’cy VC-12 _check_transparency	Checks the transparency of a tributary or channel that was defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.

Table 2-5 “_dem_c12.squ” test sequence: test cases used and their functions

Test case, function name	Function
Check TU-AIS _check_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms.
Check TU-LOP _check_alarm	See “Check TU-AIS”. test case.
Check LP-UNEQ _check_alarm	See “Check TU-AIS”. test case.
Check LP-BIP _check_stimulus_response	This test stimulates the error indication sensors of SDH or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site.
Goto Increment VC12-Trib	Jumps to the “Increment VC12-Trib” test case.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-5 “_dem_c12.squ” test sequence: test cases used and their functions

2.5.4 _demooc1.squ

This sequence is a demo example of a line-up of an OC-1 circuit that carries a DS1 framed SF. Connection to the ANT-20 is optical. Tests include a quick look at the alarm situation, a payload transparency check, a long-term error test, an alarm sensor test and some examples of injecting different stimuli like errors and alarms (permanent or N in M).

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX=RX STS-1 DS1 SF _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].

Table 2-6 “_demooc1.squ” test sequence: test cases used and their functions

Test case, function name	Function
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the "_set_sonet_structure" test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
TX=RX Tributaries _set_sonet_tributaries	Sets the tributary/channel number of an ANSI/SONET signal. All subsequent test cases are performed on the tributary/channel defined here.
Check Optical Level _read_rx_optical_level	Reads the current optical level and compares it with the user-defined limits.
Monitor Alarm _read_sonet_actual_alarms	Reads the current SONET/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval).
Payl.-Transprncy _check_sonet_transparency	Checks the transparency of a tributary or channel that was defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Clock Pull. Range _check_clock_pulling_range	Checks the pulling range of a UUT by transmitting a signal with an increasing clock offset and checking for bit errors and alarms during the measurement.
Set Stimulus _set_sonet_stimulus	Inserts an error ratio or frequency offset until switched off by the next test case. This test case can be combined with the "_wait" test case to define insertion time. To switch off error insertion, "_set_sonet_stimulus" must be called again with the stimulus set to "Switch off error insertion".
Check All Events _check_trouble	Checks the ANT-20's event queue for events (alarms, errors, ...).
Set Stimulus OFF set_sonet_stimulus	Ends the "Set Stimulus" test case.
Check AIS-DS1 _check_sonet_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. You can easily change the test case parameters to suit specific test needs. This test case switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by "_set_alarm" and "_set_stimulus" will continue until switched off specifically.
Check Ratio FE-DS1 _check_sonet_stimulus_response	This test stimulates the error indication sensors of SONET or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site. You can specify an upper/lower limit for the expected result.
Check N * TSE _check_sonet_stimulus_response	See "Check Ratio FE-DS1" test case.
Set Alarm N in M _set_sonet_alarm_N_in_M	Inserts specific alarm conditions for a specific number of frames, followed by a specific number of frames without alarms. This sequence is repeated periodically until switched off by calling this test case with the parameter "all alarm insertions off".

Table 2-6 "_demooc1.squ" test sequence: test cases used and their functions

Test case, function name	Function
Check Events _check_trouble	Checks the ANT-20's event queue for events (alarms, errors, ...).
Alarm N in M OFF _set_sonet_alarm_N_in_M	Ends the "Set Alarm N in M" test case.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on "OK".

Table 2-6 "_demooc1.squ" test sequence: test cases used and their functions

2.5.5 _demooc3.squ

This sequence is a demo example of tests of the maintenance interactions at all hierarchy levels of an OC-3 optical signal with embedded STS-1 and DS-1 SF. Tests include LOS and LOF alarm sensors, high and low B1 error rate, AIS line and BIP line for the OC-3 line level as well as AIS path, LOP path, UNEQ path and PLM path for the HO path level.

After these tests, a full test sequence is performed on all 84 (or a user-defined number) different VT 1.5. The tests include a payload transparency check for the embedded DS-1, an AIS-VT, LOP-VT, UNEQ-VT and PLM-VT alarm check and a high/low BIP-VT error rate check.

To be useful in a "real-life" environment, the parameters of the alarm sensor tests need to be adapted. This demo sequence is written in such a way that it will deliver a pass result when an optical loop is present between the TX and RX of the ANT-20.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on "OK".
Check needed options _check_options_one_of	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the "Check needed options" test case are not available. The "Goto End" test case is executed when you click on "OK".
Goto End	"Goto End" is only performed if some of the options checked in the "Check needed options" test case are not available. Jumps to the end of the sequence to the "End" test case. The "End" test case is performed and the sequence ends.
TX=RX OC-3 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the "_set_sonet_structure" test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the "_set_sonet_structure" test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.

Table 2-7 "_demooc3.squ" test sequence: test cases used and their functions

Test case, function name	Function
Read Level _read_rx_optical_level	Reads the current optical level and compares it with the user-defined limits.
LOS _check_sonet_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. You can easily change the test case parameters to suit specific test needs. This test case switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by “_set_alarm” and “_set_stimulus” will continue until switched off specifically.
LOF (A1/2) _check_sonet_alarm	See “LOS” test case.
BIP-S (B1) Single _check_sonet_stimulus_response	This test stimulates the error indication sensors of SONET or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site. You can specify an upper/lower limit for the expected result.
BIP-S (B1) Rate _check_sonet_alarm	See “LOS” test case.
AIS Line (K2) _check_sonet_alarm	See “LOS” test case.
BIP-L (B2) Single _check_sonet_stimulus_response	See “BIP-S (B1) Single” test case.
BIP-L (B2) Rate _check_sonet_alarm	See “LOS” test case.
Setup STS-1 List _setup_mscan_trib	Defines a list of tributaries which will be set with the _mscan_set_inc_trib test case. This list can be used In combination with a GOTO to scan several tributaries and carry out any number of additional test cases for each tributary from the defined list.
STS-1 # Increment _mscan_set_inc_trib	Selects the next tributary from the list defined with the _setup_mscan_trib test case. When the end of the list is reached, no new tributary will be set and the test case status is PASS.
AIS Path _check_sonet_alarm	See “LOS” test case.
LOP Path _check_sonet_alarm	See “LOS” test case.
UNEQ Path _check_sonet_alarm	See “LOS” test case.
PLM Path _check_sonet_alarm	See “LOS” test case.
BIP-P (B3) Single _check_sonet_stimulus_response	See “BIP-S (B1) Single” test case.
BIP-P (B3) Rate _check_sonet_alarm	See “LOS” test case.
Goto STS-1 # Increment	Jumps to the “STS-1 # Increment” test case.

Table 2-7 “_demooc3.squ” test sequence: test cases used and their functions

Test case, function name	Function
Setup VT List _setup_mscan_trib	Defines a list of tributaries which will be set with the _mscan_set_inc_trib test case. This list can be used In combination with a GOTO to scan several tributaries and carry out any number of additional test cases for each tributary from the defined list.
VT # Increment _mscan_set_inc_trib	Selects the next tributary from the list defined with the _setup_mscan_trib test case. When the end of the list is reached, no new tributary will be set and the test case status is PASS.
DS1 Transparency _check_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
AIS VT Path _check_sonet_alarm	See “LOS” test case.
LOP VT Path _check_sonet_alarm	See “LOS” test case.
UNEQ VT Path _check_sonet_alarm	See “LOS” test case.
PLM VT Path _check_sonet_alarm	See “LOS” test case.
BIP-VT (V5) Single _check_sonet_stimulus_response	See “BIP-S (B1) Single” test case.
BIP-VT (V5) Rate _check_sonet_alarm	See “LOS” test case.
Goto VT # Increment	Jumps to the “VT # Increment” test case.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-7 “_demooc3.squ” test sequence: test cases used and their functions

2.5.6 _toolkit.squ

This sequence demonstrates most of the non-instrument related test cases for the CATS product, such as:

- Messages to the user
- User data entry
- Show a bitmap or photograph
- Measuring test time
- Timer start
- Wait a fixed amount of time
- Sending any SCPI commands to other instruments
- Beep

Test case, function name	Function
Message _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
User Query _user_io	The user is prompted for a “Yes” or “No” response to a question.
User Editor _user_io	Opens a window for entering a message or comment. The next test case is executed when you click on “OK”.
Show Bitmap _show_image	Displays a bitmap.
SCPI Commands _scpi_io	Sends user-defined SCPI commands to the specified instrument via the open interface. If a query is sent, the response is displayed. No PASS/FAIL result is derived from the contents of the received result string.
Start Stopwatch _stopwatch	Measures elapsed time between first call of this test case (start of stopwatch) and subsequent calls of the same test case.
Delay 5 s _wait	Waits for a specified period.
Get elapsed time _stopwatch	Measures elapsed time between first call of this test case (start of stopwatch) and subsequent calls of the same test case.
Timer Start _wait_until_time	Waits until a specific point in time.
Beep _acoustical_signal	Generates an audible signal (beep).
Enter Device Info _show_info_field	Opens a window for entering data.
End! _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-8 “_toolkit.squ” test sequence: test cases used and their functions

2.6 Sequences for ANSI bit rates

This section describes sequences for ANSI or SONET bit rates supplied with the software.

2.6.1 a_almdyn.squ

This short sequence is an example that demonstrates the insertion of dynamic alarms (N in M) into an STS-1 signal.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX STS-1 VT1.5 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX STS-1 VT1.5 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
Transparency check _check_sonet_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
SET AIS_P N in M _set_sonet_alarm_N_in_M	Inserts specific alarm conditions for a specific number of frames, followed by a specific number frames without alarms. This sequence is repeated periodically until switched off by calling this test case with the parameter “all alarm insertions off”.
Check Trouble _check_trouble	Checks the ANT-20’s event queue for events (alarms, errors, ...).
Turn N in M OFF _set_sonet_alarm_N_in_M	Ends the “SET AIS_P N in M” test case.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-9 “a_almdyn.squ” test sequence: test cases used and their functions

2.6.2 a_atmtra.squ

This sequence demonstrates a transparency check of a number of different ATM VPI and VCI pairs. TX and RX VPI VCI can be identical or different depending on the application.

See “i_atmtra.squ” test sequence (Sec. 2.7.6, Page 1-45).

2.6.3 a_aps.squ

This sequence demonstrates the principle of an automatic APS time measurement:
TX and RX are set up for an STS-1 signal. The APS function is then activated for a duration of 5 seconds with a permitted maximum APS time of 2500 ms. The detection criterion is set to TSE (bit errors). While the test is running, TSEs are inserted for a duration of approx. 2 seconds, which is less than the permitted switch-over time set in this example (a real system will have a different APS time limit). The test therefore should pass if there is a loop between the ANT-20 TX and RX.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
RX=TX OC-12 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Read Level read_rx_optical_level	Reads the current optical level and compares it with the user-defined limits.
Wait for Synch _wait	Waits for a specified period.
Init APS measurement _set_and_start_sonet_APS	Initializes and starts APS measurement. APS results can be checked by calling the test case “_check_APS” subsequently.
Inject TSE 1E-3 _set_sonet_stimulus	Inserts an error ratio or frequency offset until switched off by the test case “TSE insert OFF”.
Wait < 2 s _wait	Waits for a specified period.
TSE insert OFF _set_sonet_stimulus	Ends the “Inject TSE 1E-3” test case.
Check APS time _check_APS	Checks the APS time and compares it with the upper and lower limits.

Table 2-10 “a_aps.squ” test sequence: test cases used and their functions

2.6.4 a_atmqos.squ

This sequence demonstrates “Quality of Service” measurements for a number of VCIs and VPI of a ATM 149 Mbit/s signal embedded within an OC-3 signal. ATM errors and alarms are measured over a user-defined time. HUNC errors can be inserted for demo purposes to see what happens in a “FAIL” condition.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
Set TX Signal Struc. _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
Set RX Signal Struc. _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adapt _set_tx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
RX Adapt _set_rx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
TX HUNC 1E-3 _scpi_io	Sends user-defined SCPI commands to the specified instrument via the open interface. If a query is sent, the response is displayed. No PASS/FAIL result is derived from the contents of the received result string.
ATM STR-HAN _check_atm	Measures quality of service parameters during a long-term, out of service test.
ATM STR-LHR _check_atm	Measures quality of service parameters during a long-term, out of service test.
ATM STR-MUC _check_atm	Measures quality of service parameters during a long-term, out of service test.
ATM STR-ZRH _check_atm	Measures quality of service parameters during a long-term, out of service test.
TX HUNC OFF _scpi_io	Ends the “TX HUNC 1E-3” test case.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-11 “a_atmqos.squ” test sequence: test cases used and their functions

2.6.5 a_atmsca.squ

This sequence demonstrates a scan of a number of different VCIs/VPIs of an ATM 149 Mbit/s signal embedded in an OC-3c. Channel range and gating time can be set, and a preceding test case allows the injection of a specific load into one of the VPI/VCI to be tested.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
SET TX STS-3C SPE AT _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
SET RX STS-3C SPE AT _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
Set TX=Testcell _set_tx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
Set RX=Testcell _set_rx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
Set VCI Load _set_tx_vpi_vci_load	Sets the load (% CBR) of a specific VPI/VCI.
Scan VCIs for Load _scan_vci_for_load	Scans through a range of VCIs (fixed VPI) and determines which VCIs are in use. The load for the used channels is measured.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-12 “a_atmsca.squ” test sequence: test cases used and their functions

2.6.6 a_delay.squ

This sequence demonstrates how to measure “round trip delay” using CATS:
After the signal structure has been set up, a reference measurement must be made first. The delay value determined by this reference measurement can be used for future measurements made using the same ANT-20 at the same bit rate / signal structure. The reference measurement only needs to be made once for each user sequence. The reference value must then be entered manually in the delay test case being used and can be employed in all subsequent measurements.

Test case, function name	Function
TX STS1 DS3 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal. This is just a setup test case without any PASS/FAIL evaluation.
RX STS1 DS3 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 RX signal. This is just a setup test case without any PASS/FAIL evaluation.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Calib Ref delay _check_pattern_delay	Checks the delay of a signal through the network.
Connect DUT _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
check delay _check_pattern_delay	Checks the delay of a signal through the network.

Table 2-13 “a_delay.squ” test sequence: test cases used and their functions

2.6.7 a_ds1.squ

This sequence includes a transparency check and a number of alarm sensor checks for a DS1 signal. It also demonstrates long-term monitoring (check all events) and a check of pulling range and Alarm N in M.

The “a_ds1.squ” test sequence is practically identical to the “_demods1” test sequence (see Sec. 2.5.2, Page 1-12).

2.6.8 a_cj_ds1.squ

This sequence is a demo example which stresses the UUT with pointer jitter and maximum mapping jitter. It sets the critical mapping offset, which leads to the maximum of mapping jitter. To find the critical mapping offset, run sequence “*_mj_*.squ”.

See “i_o_172.squ” test sequence (Sec. 2.7.20, Page 1-60).

2.6.9 a_dwdm.squ

This sequence only works in conjunction with the OSA-155 Optical Spectrum Analyzer, which is controlled from the ANT-20 via a RS 232 or GPIB interface.

The test starts by setting up the DWDM parameters (number of channels, spacing, threshold). This is followed by a measurement of DWDM parameters (level, optical signal/noise ratio, wavelength) for each “color”. The results are compared with user-defined thresholds. The next test sets the OSA-155 to “Filter” mode. This filters out a specific wavelength, which can then be fed into the ANT-20 optical RX. Any ANT-20-specific test can be carried out on the filtered wavelength. In this example, only a BIP error rate check on all STS-3c tributaries of an OC-48 signal is performed.

This sequence uses the CATS loop feature, an outer loop that switches between 8 ‘colors’ of a DWDM signal, and an inner loop that switches between the 16 STS-3c channels of the OC-48 signal.

Test case, function name	Function
Information _show_image	Displays a bitmap.
ANT TX OC-48 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
OSA WDM Parameters _WDM_set_parameter	Sets the general parameters for D-WDM mode.
Check Sel. DWDM Ch. _WDM_check_sel_chnls	Searches for the selected channels of a grid and evaluates the measured results against the defined limits.
DWDM Channel List _WDM_loop_select_list	Defines a list of D-WDM channels to be set with the _WDM_loop_set_chnl test case. This list can be combined with a GOTO command to scan several channels and carry out any number of additional test cases for each channel in the defined list.
ANT RX OC-48 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Wait for Synch. _wait	Waits for a specified period.
Incr. DWDM Ch. _WDM_loop_set_chnl	Sets the next channel in the list defined by the _WDM_loop_select_list test case.
STS-1 Trib. # List _setup_mscan_trib	Defines a list of tributaries to be set with the _mscan_set_inc_trib test case. This list can be combined with a GOTO command to scan several channels and carry out any number of additional test cases for each channel in the defined list.

Table 2-14 “a_dwdm.squ” test sequence: test cases used and their functions

Test case, function name	Function
Incr. STS-1 Trib. # mscan_set_inc_trib	Selects the next tributary in the defined list set with the _setup_mscan_trib test case. When the end of the list is reached, no new tributary will be set and the test case status is PASS.
Check SONET Alarm _check_sonet_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. You can easily change the test case parameters to suit specific test needs. This test case switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by “_set_sonet_alarm” and “_set_sonet_stimulus” will continue until switched off specifically.
Check B3 Errors _check_sonet_stimulus_response	This test stimulates the error indication sensors of SONET or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site. You can specify an upper/lower limit for the expected result.
Check TSE Errors _check_sonet_stimulus_response	See “Check B3 Errors” test case.
Goto Incr. STS-1 Trib. #	Jumps to the “Incr. STS-1 Trib.#” test case.
Goto Incr. DWDM Ch.	Jumps to the “Incr. DWDM Ch” test case.
Set OSA Filter _set_filter_man	Sets the optical filter of the OSA to a defined wavelength/frequency.

Table 2-14 “a_dwdm.squ” test sequence: test cases used and their functions

2.6.10 a_jit_48.squ

This sequence demonstrates how to set up jitter measurements for OC-48.

See “a_jitter.squ” test sequence for principles (Sec. 2.6.12, Page 1-29).

2.6.11 a_oc48.squ

This sequence demonstrates tests for OC-192 with STS-12C or STS-48C substructures, including reading the optical level, monitoring alarms, a transparency check and B2, TSE, and path trace tests.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check OC-48 optic _check_options_one_of	Checks whether at least one of the ANT-20/ANX-920 options listed is present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX=RX OC48 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Wait for sync. _wait	Waits for a specified period.
Read Level _read_rx_optical_level	Reads the current optical level and compares it with the user-defined limits.
Actual Alarm _read_sonet_actual_alarms	Reads the current SONET/PDH alarms at ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval).
Check Response _check_sonet_stimulus	This test stimulates the error indication sensors of SONET or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site. You can specify an upper/lower limit for the expected result.
Set LOF _set_sonet_alarm	Inserts an alarm until switched off by the next test case.
Check Alarm SONET _check_sonet_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. You can easily change the test case parameters to suit specific test needs. This test case switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by “_set_sonet_alarm” and “_set_sonet_stimulus” will continue until switched off specifically.

Table 2-15 “a_oc48.squ” test sequence: test cases used and their functions

Test case, function name	Function
Set LOF OFF _set_sonet_alarm	Inserts an alarm until switched off by the next test case.
Set Trib _set_sonet_tributaries	Sets the tributary/channel number of an ANSI/SONET signal. All subsequent test cases will be done for the tributary/channel defined here.
Check Path Trace _check_sonet_path_trace	Inserts a user defined path trace at port A and reads path trace at port B. A comparison is made between the expected path trace and the received path trace.
Check Transparency _check_sonet_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-15 “a_oc48.squ” test sequence: test cases used and their functions

2.6.12 a_jitter.squ

This sequence demonstrates how to correctly set up a sequence that includes jitter measurements. Please note that a BASIC device and a JITTER device are used in this configuration. Use this sequence (especially the first 6 test cases) as a sample whenever setting up a jitter test sequence (up to OC-12). Measurements include JTF, intrinsic jitter and MTJ.

Test case, function name	Function
Setup RX=TX Signal _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
Setup RX Jitter _setup_RX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
Setup RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Setup TX Jitter _setup_TX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
Setup TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
Setup Jitter Stimul. _setup_jitter_stimulus	Sets up jitter stimulus.
Jitter Measurement _check_jitter	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.

Table 2-16 “a_jitter.squ” test sequence: test cases used and their functions

Test case, function name	Function
Reset Jitter Stimul. _setup_jitter_stimulus	Sets up jitter stimulus.
FTJ _check_jitter_tolerance	Up to 20 user selectable values for jitter amplitude / jitter frequency can be set. A bit error check is performed at each of these points.
MTJ _check_mtj	Up to 10 user selectable values for jitter frequencies can be set. A bit error check is performed at each of these points at the given expected maximum jitter amplitude. The measurement is started using the “mask” jitter amplitude value. If no bit errors and no alarms occur during the measurement period (“mask” value), jitter is increased until an error or alarm is detected. The jitter amplitude value below the value causing an error or alarm is considered the MTJ result. If a bit error or alarm occurred during the first measurement period, jitter is decreased until no error or alarm is found. This value (frequency and amplitude) is considered the MTJ result.
JTF _check_jitter_transfer	A fixed jitter amplitude is set by TX. Jitter frequency list is user selectable (20 values). Jitter amplitude at RX is measured for each frequency point. Reference measurement (Yes/No) is used to eliminate any error caused by the measurement setup. Values are stored for all subsequent runs of this test case. When a sequence is loaded, the reference results are deleted.

Table 2-16 “a_jitter.squ” test sequence: test cases used and their functions

2.6.13 a_list.squ

This sequence demonstrates the use of the “LIST” feature, i.e. performing a number of different tests on a group of tributaries defined within one test case. In this example, a transparency check and alarm sensor checks are performed on all VT1.5s of an STS-3 signal.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Show Bitmap _show_image	Displays a bitmap.
TX STS-3 VT1.5 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX STS-3 VT1.5 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].

Table 2-17 “a_list.squ” test sequence: test cases used and their functions

Test case, function name	Function
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the "_set_sonet_structure" test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Define VT List _setup_mscan_trib	Defines a list of tributaries to be set with the _mscan_set_inc_trib test case. This list can be combined with a GOTO command to scan several channels and carry out any number of additional test cases for each channel in the defined list.
How to built a loop _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on "OK".
Increment VT-Trib _mscan_set_inc_trib	Selects the next tributary from the defined list set with the _setup_mscan_trib test case. When the end of the list is reached, no new tributary is set and the test case status is PASS.
Check Trouble _check_trouble	Checks the ANT-20's event queue for events (alarms, errors, ...).
Transparency VT _check_sonet_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Check AIS-P _check_sonet_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. You can easily change the test case parameters to suit specific test needs. This test case switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by "_set_sonet_alarm" and "_set_sonet_stimulus" will continue until switched off specifically.
Check LOP-P _check_sonet_alarm	See "Check AIS-P" test case.
Check UNEQ-V _check_sonet_alarm	See "Check AIS-P" test case.
Check BIP-V _check_sonet_stimulus_response	This test stimulates the error indication sensors of SONET or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site. You can specify an upper/lower limit for the expected result.
Goto Increment VT-Trib	Jumps to the "Increment VT-Trib" test case.
End _acoustical_signal	Generates an audible signal (beep).

Table 2-17 "a_list.squ" test sequence: test cases used and their functions

2.6.14 a_mj_ds1.squ

This sequence is a demo example for determining the critical mapping offset for an STS-3 / OC-3 embedded STS-1 SPE VT-1.5 Mbit/s signal where mapping jitter is at a maximum.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Test with 2 ANT-20 _user_io	A user-defined message is displayed when this test is run.
TX OC-3 VT1.5 unfr _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX DS1 unfr _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level. Additional parameters are sensitivity and equalizer settings.
TX Set Tributary _set_sonet_tributaries	Sets the tributary/channel number of an ANSI/SONET signal. All subsequent test cases are performed on the tributary/channel defined here.
RX Setup Jitter _setup_RX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
Check Transparency _check_sonet_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Transparency Error _user_io	A user-defined message is displayed if the preceding “Check Transparency” test case determines that there is an error.
Goto The End	“Goto The End” is only performed if the “Check Transparency” test case determines that there is an error. Jumps to the end of the sequence to the “The End” test case. “The End” is executed and the sequence ends.
Set Map.Offs -50 ppm _set_mapp_offset	Sets the mapping offset to -50 ppm.
Check DS1 Jitter _check_jitter	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.
Set Map.Offs 0 ppm _set_mapp_offset	Sets the mapping offset to 0 ppm.
Delay 9s _wait	Waits for a specified period.
Max. Mapping Jitter _find_max_mapping_jitter	Determines the critical mapping offset causing a jitter maximum.
The End _acoustical_signal	Generates an audible signal (beep).

Table 2-18 “a_mj_ds1.squ” test sequence: test cases used and their functions

2.6.15 a_oc3_mi.squ

This sequence demonstrates a test of all the different maintenance interactions at line, path and VT level of an OC-3 signal. It is written in such a way that it can be used with a “real” UUT, i.e. it will return a fail result with just a cable connected between the ANT-20 RX and TX.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all of the listed ANT-20/ANX-920 options are present.
Check laser option _check_options_one_of	Checks whether at least one of the ANT-20/ANX-920 options listed is present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
Enter UUT data _show_info_field	Opens a window for entering data.
TX=RX OC-3 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Connection _show_image	Displays a bitmap.
Read Level _read_rx_optical_level	Reads the current optical level and compares it with the user-defined limits.
LOS _check_sonet_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. You can easily change the test case parameters to suit specific test needs. This test case switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by “_set_sonet_alarm” and “_set_sonet_stimulus” will continue until switched off specifically.
LOF (A1/2) _check_sonet_alarm	See “LOS” test case.

Table 2-19 “a_oc3_mi.squ” test sequence: test cases used and their functions

Test case, function name	Function
BIP-S (B1) Single _check_sonet_stimulus_response	This test stimulates the error indication sensors of SONET or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site. You can specify an upper/lower limit for the expected result.
BIP-S (B1) Rate _check_sonet_alarm	See “LOS” test case.
AIS Line (K2) _check_sonet_alarm	See “LOS” test case.
BIP-L (B2) Single check_sonet_stimulus_response	See “BIP-S (B1) Single” test case.
BIP-L (B2) Rate _check_sonet_alarm	See “LOS” test case.
Set STS-1 TX Trib. _set_sonet_tributaries	Sets the tributary/channel number of an ANSI/SONET signal. All subsequent test cases are performed on the tributary/channel defined here.
Set STS-1 RX Trib. _set_sonet_tributaries	See “Set STS-1 TX Trib” test case.
AIS Path _check_sonet_alarm	See “LOS” test case.
LOP Path _check_sonet_alarm	See “LOS” test case.
UNEQ Path _check_sonet_alarm	See “LOS” test case.
PLM Path _check_sonet_alarm	See “LOS” test case.
BIP-P (B3) Single _check_sonet_stimulus_response	See “BIP-S (B1) Single” test case.
BIP-P (B3) Rate _check_sonet_alarm	See “LOS” test case.
Set VT TX Trib. _set_sonet_tributaries	See “Set STS-1 TX Trib.” test case.
Set VT RX Trib. _set_sonet_tributaries	See “Set STS-1 TX Trib.” test case.
AIS VT Path _check_sonet_alarm	See “LOS” test case.
LOP VT Path _check_sonet_alarm	See “LOS” test case.
UNEQ VT Path _check_sonet_alarm	See “LOS” test case.
PLM VT Path _check_sonet_alarm	See “LOS” test case.

Table 2-19 “a_oc3_mi.squ” test sequence: test cases used and their functions

Test case, function name	Function
BIP-VT (V5) Single _check_sonet_stimulus_response	See “BIP-S (B1) Single” test case.
BIP-VT (V5) Rate _check_sonet_alarm	See “LOS” test case.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-19 “a_oc3_mi.squ” test sequence: test cases used and their functions

2.6.16 a_oc192.squ

This sequence demonstrates tests for OC-192 with STS-12C or STS-48C substructures including reading the optical level, and checking transparency, TSE, B2, and alarms.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
TX=RX OC192 STS48c _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
TX & RX Tributaries _set_sonet_tributaries	Sets the tributary/channel number of an ANSI/SONET signal. All subsequent test cases are performed on the tributary/channel defined here.
Wait for sync. _wait	Waits for a specified period.
Read Level _read_rx_optical_level	Reads the current optical level and compares it with the user-defined limits.
Alarm Monitor _read_sonet_actual_alarms	Reads the current SONET/PDH alarms at ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval).
Check Transparency _check_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.

Table 2-20 “a_oc192.squ” test sequence: test cases used and their functions

Test case, function name	Function
Check TSE _check_sonet_stimulus_response	This test stimulates the error indication sensors of SONET or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site. You can specify an upper/lower limit for the expected result.
Check B2 – B2 _check_sonet_stimulus_response	See “Check TSE” test case.
Check Alarm _check_sonet_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. You can easily change the test case parameters to suit specific test needs. This test case switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by “_set_sonet_alarm” and “_set_sonet_stimulus” will continue until switched off specifically.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-20 “a_oc192.squ” test sequence: test cases used and their functions

2.6.17 a_sts3.squ

This sequence demonstrates a few typical tests for an STS-3 signal, such as overhead analysis, alarm sensors, monitoring and path trace checks.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX STS-3 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX STS-3 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].

Table 2-21 “a_sts3.squ” test sequence: test cases used and their functions

Test case, function name	Function
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the "_set_sonet_structure" test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Connection _show_image	Displays a bitmap.
LOS _check_sonet_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. You can easily change the test case parameters to suit specific test needs. This test case switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by "_set_sonet_alarm" and "_set_sonet_stimulus" will continue until switched off specifically.
LOF (A1/2) _check_sonet_alarm	See "LOS" test case.
BIP-S (B1) Single _check_sonet_stimulus_response	This test stimulates the error indication sensors of SONET or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site. You can specify an upper/lower limit for the expected result.
AIS Line (K2) _check_sonet_alarm	See "LOS" test case.
BIP-L (B2) Single _check_sonet_stimulus_response	See "BIP-S (B1) Single" test case.
Set STS-1 TX Trib _set_sonet_tributaries	Sets the tributary/channel number of an ANSI/SONET signal. All subsequent test cases are performed on the tributary/channel defined here.
Set STS-1 RX Trib _set_sonet_tributaries	See "Set STS-1 TX Trib" test case.
AIS Path _check_sonet_alarm	See "LOS" test case.
LOP Path _check_sonet_alarm	See "LOS" test case.
UNEQ Path _check_sonet_alarm	See "LOS" test case.
PLM Path _check_sonet_alarm	See "LOS" test case.
BIP-P (B3) Single _check_sonet_stimulus_response	See "BIP-S (B1) Single" test case.
Set VT TX Trib. _set_sonet_tributaries	See "Set STS-1 TX Trib" test case.
Set VT RX Trib. _set_sonet_tributaries	See "Set STS-1 TX Trib" test case.
AIS VT Path _check_sonet_alarm	See "LOS" test case.

Table 2-21 "a_sts3.squ" test sequence: test cases used and their functions

Test case, function name	Function
LOP VT Path _check_sonet_alarm	See “LOS” test case.
UNEQ VT Path _check_sonet_alarm	See “LOS” test case.
PLM VT Path _check_sonet_alarm	See “LOS” test case.
BIP-VT (V5) Single _check_sonet_stimulus_response	See “BIP-S (B1) Single” test case.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-21 “a_sts3.squ” test sequence: test cases used and their functions

2.6.18 a_scan.squ

This sequence demonstrates an error and alarm scan of all VT 1.5s embedded in an STS-1 signal. A control function for an electrical test point selector can be easily added to this sequence to create a small “monitoring system”. Please ask the WWG Solution Center for details.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
TX STS1 VTG VT1.5 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX STS1 VTG VT1.5 _set_sonet_structure	Sets the ANSI SONET/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
Set Testpoint Position _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_sonet_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
TX Channel _set_sonet_tributaries	Sets the tributary/channel number of an ANSI/SONET signal. All subsequent test cases are performed on the tributary/channel defined here.
RX Channel _set_sonet_tributaries	See “TX Channel” test case.

Table 2-22 “a_scan.squ” test sequence: test cases used and their functions

Test case, function name	Function
Insert TX Alarm _set_sonet_alarm	Inserts an alarm until switched off by the next test case.
Insert TX Dyn. Alarm _set_sonet_alarm_N_in_M	Inserts specific alarm conditions for a specific number of frames, followed by a specific number of frames without alarms. This sequence is repeated periodically until switched off by calling this test case with the parameter “all alarm insertions off”.
Insert TX Errors _set_sonet_stimulus	Inserts an error ratio or frequency offset until switched off by the next test case. This test case can be combined with the “_wait” test case to define the insertion time. To switch off error insertion, “_set_sonet_stimulus” must be called again with the stimulus set to “Switch off error insertion”.
Moni Alarm _read_sonet_actual_alarms	Reads the current SONET/PDH alarms at ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval).
STS1 VT1.5 Fast Scan _scan_monitor	**“Fast Alarm Scan” mode:** Scans all selected tributaries and displays their current alarm status. **“Error Test” mode:** Scans all selected tributaries and starts an error measurement. All error events and alarms that occur during the gate time will be displayed.
STS1 VT1.5 Error Scan _scan_monitor	See “STS1 VT1.5 Fast Scan” test case.
Good-bye! _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-22 “a_scan.squ” test sequence: test cases used and their functions

2.7 Sequences for ITU-T bit rates

This section describes sequences for ITU-T or SDH bit rates supplied with the software.

2.7.1 i_almdyn.squ

This sequence demonstrates how to insert dynamic alarms. In this example, an STM-1 signal is used and the long-term “Check Trouble” test shows the impact of alarm insertion.

Test case, function name	Function
TX=RX Signal _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Check Transparency _check_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Set LOF-SDH N in M _set_alarm_N_in_M	Inserts specific alarm conditions for a specific number of frames, followed by a specific number of frames without alarms. This sequence is repeated periodically until switched off by calling this test case with the parameter “all alarm insertions off”.
Check Trouble _check_trouble	Checks the ANT-20’s event queue for events (alarms, errors, ...).
N in M Alarm OFF _set_alarm_N_in_M	Ends the “Set LOF-SDH N in M” test case.

Table 2-23 “i_almdyn.squ” test sequence: test cases used and their functions

2.7.2 i_aps.squ

This sequence demonstrates the principle of an automatic APS time measurement:
In this example TX and RX are set up for an STM-1 VC-4 signal. The APS function is then activated for a duration of 5 seconds with a permitted maximum APS time of 2500 ms. The detection criterion is set to TSE (bit errors). While the test is running, TSEs are inserted for a duration of approx. 2 seconds, which is less than the permitted switch-over time set in this example (a real system will have a different APS time limit). The test therefore should pass if there is a loop between the ANT-20 TX and RX.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
RX=TX STM-1 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/ FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Read Level read_rx_optical_level	Reads the current optical level and compares it with the user-defined limits.
Wait for Synch _wait	Waits for a specified period.
Init APS measurement _set_and_start_APS	Initializes and starts APS measurement. APS results can be checked by calling the test case “_check_APS” subsequently.
Inject TSE 1E-3 _set_stimulus	Inserts an error ratio or frequency offset until switched off by the “TSE insert OFF” test case.
Wait < 2 s _wait	Waits for a specified period.
TSE insert OFF _set_stimulus	Ends insertion (test case “Inject TSE 1E-3”).
Check APS time _check_APS	Checks the APS time and compares it with the upper and lower limits.

Table 2-24 “i_aps.squ” test sequence: test cases used and their functions

2.7.3 i_atmdel.squ

This sequence demonstrates how to measure the cell delay variation of an ATM signal, using STM-1 VC 4 ATM. Both automatic mode and fixed mode (with fixed resolution and offset) are used. For details, see the ANT-20 user manual under ATM measurements.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
check ATM option _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
option error _user_io	A user-defined message is displayed if some of the options checked in the “Check ATM option” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto end	“Goto End” is only performed if some of the options checked in the “Check ATM option” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX=RX Signal Struct _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adapt CDV _set_tx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
RX Adapt CDV _set_rx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
TX VPI, VCI Load _set_tx_vpi_vci_load	Sets the load (% CBR) of a specific VPI/VCI.
RX Set VPI, VCI _set_rx_vpi_vci	Sets a specific VPI/VCI.
Check Auto CDV _check_auto_CDV	Measures cell transfer delay and cell delay variation using an automatic algorithm to achieve maximum accuracy while the offset is 0 µs. The “_check_CDV” test case with proper offset and resolution values should be used to obtain maximum accuracy. The measurement starts using the broadest resolution value (335.500 ms). Resolution is then tightened to a minimum of 0.16 µs. The CDV value for the best possible resolution is displayed and evaluated.
CDV Res 0.16 µs _check_CDV	Measures cell transfer delay and cell delay variation of ATM test cells. The measurement is made with fixed resolution and offset values. For maximum accuracy, the “_check_auto_CDV” test case should be run first to determine the expected result range.
CDV Res 1.28 µs _check_CDV	See “CDV Res 0.16 ms” test case.
end _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-25 “i_atmdel.squ” test sequence: test cases used and their functions

2.7.4 i_atmqos.squ

This sequence demonstrates “Quality of Service” measurements for a number of VCs and the VP of a 149 Mbit/s ATM signal embedded within an STM-1 signal.

ATM errors and alarms are measured over a user-defined time period. HUNC errors can be inserted for demo purposes to see what happens under “FAIL” conditions.

Test case, function name	Function
TX STM-1 VC ATM _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX STM-1 VC4 ATM _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adapt _set_tx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
RX Adapt _set_rx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
TX HUNC 1E-3 _scpi_io	Sends user-defined SCPI commands to the specified instrument via the open interface. If a query is sent, the response is displayed. No PASS/FAIL result is derived from the contents of the received result string.
ATM STR-HAN _check_atm	Measures quality of service parameters during a long-term, out of service test.
ATM STR-LHR _check_atm	Measures quality of service parameters during a long-term, out of service test.
ATM STR-MUC _check_atm	Measures quality of service parameters during a long-term, out of service test.
ATM STR-ZRH _check_atm	Measures quality of service parameters during a long-term, out of service test.
TX HUNC OFF _scpi_io	Ends the “TX HUNC 1E-3” test case.

Table 2-26 “i_atmqos.squ” test sequence: test cases used and their functions

2.7.5 i_atmsca.squ

This sequence demonstrates a scan of a number of different VCI/VPI of an ATM 149 Mbit/s signal embedded in an STM-1. Channel range and gate time can be set, and a preceding test case allows the injection of a specific load in to one of the VPI/VCI to be tested.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
option check _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
option error _user_io	A user-defined message is displayed if some of the options checked in the “option check” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto end	“Goto End” is only performed if some of the options checked in the “option check” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX=RX STM-1 ATM _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Set TX=Testcell _set_tx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
Set RX=Testcell _set_rx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
Set VCI Load _set_tx_vpi_vci_load	Sets the load (% CBR) of a specific VPI/VCI.
Scan VCIs for Load _scan_vci_for_load	Scans through a range of VCIs (fixed VPI) and determines which VCIs are in use. The load in the used channels is measured.
end _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-27 “i_atmsca.squ” test sequence: test cases used and their functions

2.7.6 i_atmtra.squ

This sequence demonstrates a transparency check of a number of different ATM VPI and VCI pairs. TX and RX VPI VCI can be identical or different depending on the application.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
check options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
option error _user_io	A user-defined message is displayed if some of the options checked in the “check options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto end	“Goto End” is only performed if some of the options checked in the “check options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX=RX Signal Struc. _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adapt AAL0 _set_tx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
RX Adapt AAL0 _set_rx_atm_adaptation	Sets the ATM adaptation layer or test cell insertion.
Check ATM Transparency _check_vci_vpi_transparency	Checks the transparency of a number of user selectable VPI/VCI. An AAL0 or AAL1 signal must be used for this check. Transparency is checked by transmitting a test pattern, injecting exactly one bit error at the beginning of the bit error test, and then continuing a bit error check for the complete gate time.
end _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-28 “i_atmtra.squ” test sequence: test cases used and their functions

2.7.7 i_c4.squ

This sequence demonstrates a few typical measurements for an STM-1 AU4 signal, including path trace checks, alarm sensor checks, injection and measurement of pointers, DCC/ECC transparency check, overhead analysis and clock pulling range.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX=RX STM-1 C4 UNFRM _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Check Path Trace _check_path_trace	Inserts a user defined path trace at the TX site and reads path trace at the RX. A comparison is made between the expected path trace and the received path trace.
SOH/POH Monitor _read_actual_SOH_POH	Reads the complete SOH and POH of an STM-1 C4, C3 or C12 signal. Readout is done in n samples separated by m seconds of update interval time. The final readout will be displayed in the Sequence Display window and written into the ASCII report file. This test case is intended to give a quick overview of the overhead; it does not have any automatic PASS/FAIL evaluation.
Set C2=00h HP_UNEQ _set_SOH_POH_byte	Sets the contents of one overhead byte.
Check Alarm HP_UNEQ _check_alarm	This test stimulates the alarm sensors and checks the presence and/or absence of the respective incoming alarms.
Monitor Alarm _read_actual_alarms	Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval). This test case does not have any automatic PASS/FAIL evaluation.
Reset POH C2=07h _set_SOH_POH_byte	Sets the contents of one overhead byte.

Table 2-29 “i_c4.squ” test sequence: test cases used and their functions

Test case, function name	Function
Payl.-Transprncy _check_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Clock Pull. Range _check_clock_pulling	Checks the pulling range of a UUT by transmitting a signal with an increasing clock offset and checking for bit errors and alarms during the measurement.
Confirm: No Alarm _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
D1_D3 Transprncy _check_byte_group_transparency	Automatically checks the transparency of the SDH SOH data communication channels D1 to D3 or D4 to D12. The transparency check is done by transmitting and receiving a PRBS, starting a BER measurement, injecting a single bit error immediately after opening the ANT-20 gate, and checking for exactly one bit error to be returned.
Set AU Pointer _set_ptr_action	Injects pointer movements or pointer sequences.
Insert AU Pointer _set_ptr_action	Injects pointer movements or pointer sequences.
Pointer Monitoring _ptr_move	Reads the value of an AU and / or TU pointer at fixed intervals and displays them in a graph.
Cancel AU Pointer _set_ptr_action	Injects pointer movements or pointer sequences.
Set Stimulus _set_stimulus	Inserts an error ratio or frequency offset.
Check All Events _check_trouble	Checks the ANT-20’s event queue for events (alarms, errors, ...).
Check LOF-STM _check_alarm	This test stimulates the alarm sensors and checks the presence and/or absence of the respective incoming alarms.
Set Alarm LOS _set_alarm	Inserts an alarm until switched off by the next test case.
Check Alarm LOS _check_alarm	This test stimulates the alarm sensors and checks the presence and/or absence of the respective incoming alarms.
Check B1 Ratio _check_stimulus_response	This test stimulates the error indication sensors of SDH or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site.
Check N Single _check_stimulus_response	See “Check B1 Ratio” test case.
Check Response _check_stimulus_response	See “Check B1 Ratio” test case.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-29 “i_c4.squ” test sequence: test cases used and their functions

2.7.8 i_c4_bis.squ

This sequence can be used for “Bringing into Service” an STM-1 VC 4 circuit. It includes the check of alarm sensors, long-term error check, pulling range test, transparency check for DCC/ ECC and path trace check.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX STM-1 C4 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX STM-1 C4 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Monitor Alarm _read_actual_alarms	Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval). This test case does not have any automatic PASS/FAIL evaluation.
SOH/POH Monitor _read_actual_SOH_POH	Reads the complete SOH and POH of an STM-1 C4, C3 or C12 signal. Readout is done in n samples separated by m seconds of update interval time. The final readout will be displayed in the Sequence Display window and written into the ASCII report file. This test case is intended to give a quick overview of the overhead; it does not have any automatic PASS/FAIL evaluation.
Check Alarm LOS _check_alarm	This test stimulates the alarm sensors and checks the presence and/or absence of the respective incoming alarms.
Check Alarm LOF-STM _check_alarm	See “Check Alarm LOF-STM” test case.
Check Alarm MS-AIS _check_alarm	See “Check Alarm LOF-STM” test case.

Table 2-30 “i_c4_bis.squ” test sequence: test cases used and their functions

Test case, function name	Function
Check Alarm AU-LOP _check_alarm	See “Check Alarm LOF-STM” test case.
Check Alarm AU-AIS _check_alarm	See “Check Alarm LOF-STM” test case.
Check B3 / HP-REI _check_stimulus_response	This test stimulates the error indication sensors of SDH or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site.
Check Exc. B3 Ratio _check_alarm	See “Check Alarm LOF-STM” test case.
Payl.-Transprncy _check_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
D1_D3_Transprncy _check_byte_group_transparency	Automatically checks the transparency of the SDH SOH data communication channels D1 to D3 or D4 to D12. The transparency check is done by transmitting and receiving a PRBS, starting a BER measurement, injecting a single bit error immediately after opening the ANT-20 gate, and checking for exactly one bit error to be returned.
D4_D12_Transprncy _check_byte_group_transparency	See “D1_D3_Transprncy” test case.
Check Path Trace _check_path_trace	Inserts a user defined path trace at the TX site and reads path trace at the RX. A comparison is made between the expected path trace and the received path trace.
Check All Events _check_trouble	Checks the ANT-20’s event queue for events (alarms, errors, ...).
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-30 “i_c4_bis.squ” test sequence: test cases used and their functions

2.7.9 i_c4_mi.squ

This sequence can be used to check all the maintenance interactions (defects and anomalies) of a STM-1 VC 4 path.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX=RX STM-1 C4 UNFRM _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/ FAIL evaluation.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Transp. cy 140 Mb/s _check_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
LOS _check_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms.
LOF-STM (A1/A2) _check_alarm	See “LOS” test case.
RS-BIP (B1) Single _check_stimulus_response	This test stimulates the error indication sensors of SDH or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site.
RS-BIP (B1) Rate _check_alarm	See “LOS” test case.
MS-AIS (K2) _check_alarm	See “LOS” test case.
MS-RDI (K2) _check_alarm	See “LOS” test case.

Table 2-31 “i_c4_mi.squ” test sequence: test cases used and their functions

Test case, function name	Function
MS-BIP (B2) Single _check_stimulus_response	See “RS-BIP (B1) Single” test case.
MS-BIP (B2) Rate _check_alarm	See “LOS” test case.
MS-REI (M1) Single _check_stimulus_response	See “RS-BIP (B1) Single” test case.
MS-REI (M1) Rate _check_alarm	See “LOS” test case.
AU-AIS _check_alarm	See “LOS” test case.
AU-LOP _check_alarm	See “LOS” test case.
HP-UNEQ (C2) _check_alarm	See “LOS” test case.
HP-TIM (J1) _check_alarm	See “LOS” test case.
HP-RDI (G1) _check_alarm	See “LOS” test case.
HP-PLM (C2) _check_alarm	See “LOS” test case.
HP-BIP (B3) Single _check_stimulus_response	See “RS-BIP (B1) Single” test case.
HP-BIP (B3) Rate _check_alarm	See “LOS” test case.
HP-REI (G1) Single _check_stimulus_response	See “RS-BIP (B1) Single” test case.
HP-REI (G1) Rate _check_alarm	See “LOS” test case.
Monitor Alarm _read_actual_alarms	Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval). This test case does not have any automatic PASS/FAIL evaluation.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-31 “i_c4_mi.squ” test sequence: test cases used and their functions

2.7.10 i_c3.squ

This sequence demonstrates some typical measurements for an STM-1 AU4 VC3 signal, including path trace checks and alarm sensor checks.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options	Checks whether all the ANT-20/ANX-920 options listed are present.
Option Error _user_io	A user-defined message is displayed if some of the options checked in the “Check needed options” test case are not available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if some of the options checked in the “Check needed options” test case are not available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX=RX STM-1 AU4C3 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
High Path Trace _check_path_trace	Inserts a user defined path trace at the TX site and reads path trace at the RX. A comparison is made between the expected path trace and the received path trace.
Low Path Trace _check_path_trace	See “High Path Trace” test case.
set J1 byte _set_SOH_POH_byte	Sets the contents of one overhead byte.
check J1 byte _check_SOH_POH_byte	Reads the contents of one overhead byte and compares them with an expected value. A mask can be set to exclude specific bits from the evaluation.
set C3 J1 byte _set_SOH_POH_byte	Sets the contents of one overhead byte.
check C3 J1 byte _check_SOH_POH_byte	Reads the contents of one overhead byte and compares them with an expected value. A mask can be set to exclude specific bits from the evaluation.

Table 2-32 “i_c3.squ” test sequence: test cases used and their functions

Test case, function name	Function
Check LP-TIM _check_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms.
check Byte E1 _check_byte_group_transparency	Automatically checks the transparency of the SDH SOH data communication channels D1 to D3 or D4 to D12. The transparency check is done by transmitting and receiving a PRBS, starting a BER measurement, injecting a single bit error immediately after opening the ANT-20 gate, and checking for exactly one bit error to be returned.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-32 “i_c3.squ” test sequence: test cases used and their functions

2.7.11 i_delay.squ

This sequence demonstrates how to measure “round trip delay” using CATS:
After the signal structure has been set up, a reference measurement must be made first. The delay value determined by this reference measurement can be used for future measurements made using the same ANT-20 at the same bit rate / signal structure. The reference measurement only needs to be made once for each user sequence. The reference value must then be entered manually in the delay test case being used and can be employed in all subsequent measurements.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
TX=RX STM-1 C12 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Calib Ref delay _check_pattern_delay	Checks the delay of a signal through the network.
Connect DUT _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
check delay _check_pattern_delay	Checks the delay of a signal through the network.

Table 2-33 “i_delay.squ” test sequence: test cases used and their functions

2.7.12 i_dwdm.squ

This sequence only works in conjunction with the OSA-155 Optical Spectrum Analyzer, which is controlled from the ANT-20 via an RS 232 or GPIB interface.

The test starts by setting up the DWDM parameters (number of channels, spacing, threshold). This is followed by a measurement of DWDM parameters (level, optical signal/noise ratio, wavelength) for each “color”. The results are compared with user-defined thresholds. The next test sets the OSA-155 to “Filter” mode. This filters out a specific wavelength, which can then be fed into the ANT-20 optical RX. Any ANT-20-specific test can be carried out on the filtered wavelength. In this example, only a BIP error rate check on all STS-3c tributaries of an OC-48 signal is performed.

This sequence uses the CATS loop feature, an outer loop that switches between 8 ‘colors’ of a DWDM signal, and an inner loop that switches between the 16 STS-3c channels of the OC-48 signal.

See “a_dwdm.squ” test sequence (Sec. 2.6.9, Page 1-26).

2.7.13 i_e1_bis.squ

This sequence is the basis for the line-up of a 2 Mbit/s transparent circuit, including pulling range, G.826 and jitter measurements.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check Jitter Options _check_options_one_of	Checks whether at least one of the ANT-20/ANX-920 options listed is present.
Option Error _user_io	A user-defined message is displayed if none of the options checked in the “Check Jitter Options” test case are available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if none of the options checked in the “Check Jitter Options” test case are available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX 2 Mb/s CRC fp _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX HDB3 PRS11 _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
TX Jitter 2 Mb/s _setup_TX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
RX 2 Mb/s CRC fp _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.

Table 2-34 “i_e1_bis.squ” test sequence: test cases used and their functions

Test case, function name	Function
RX HDB3 PRS11 _set_rx_adaptation2	Sets the signal parameters that are not covered by the "_set_signal_structure" test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
RX Jitter 2 Mb/s _setup_RX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
View Alarm _read_actual_alarms	Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval). This test case does not have any automatic PASS/FAIL evaluation.
Check LOS _check_alarm	This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms.
Continuity Check check_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Inject Jitter _setup_jitter_stimulus	Sets up jitter stimulus.
Jitter Measurement _check_jitter	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.
Jitter OFF _setup_jitter_stimulus	Sets up jitter stimulus.
Jitter Tolerance _check_jitter_tolerance	Up to 20 user selectable values for jitter amplitude / jitter frequency can be set. A bit error check is performed at each of these points.
Pulling Range _check_clock_pulling	Checks the pulling range of a UUT by transmitting a signal with an increasing clock offset and checking for bit errors and alarms during the measurement.
Inject Errors _set_stimulus	Inserts an error ratio or frequency offset until switched off by the test case "Inject OFF".
Start G.826 _set_and_start_g826	Sets the parameters for a G.826 evaluation.
Inject OFF _set_stimulus	Ends the "Inject Errors" test case.
G.826 Results _check_g826	Checks the results of the G.826 evaluation.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on "OK".

Table 2-34 "i_e1_bis.squ" test sequence: test cases used and their functions

2.7.14 i_e1_crc.squ

This sequence is the basis for the line-up of a 2 Mbit/s PCM30CRC terminated circuit, including pulling range, G.826 and jitter measurements, as well as a check of the alarm and error sensors (CRC against E bit, LOF against AIS).

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options_one_of	Checks whether at least one of the ANT-20/ANX-920 options listed is present.
Option Error _user_io	A user-defined message is displayed if none of the options checked in the “Check needed options” test case are available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if none of the options checked in the “Check needed options” test case are available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
User Data _show_info_field	Opens a window for entering data.
TX 2 Mb/s CRC _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX HDB3 PRS11 _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
TX Jitter 2 Mb/s _setup_TX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
RX 2 Mb/s CRC _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX HDB3 PRS11 _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
RX Jitter 2 Mb/s _setup_RX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
View Alarm _read_actual_alarms	Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval). This test case does not have any automatic PASS/FAIL evaluation.
Continuity Check _check_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Check LOS/AIS_64 _check_alarm	This test stimulates the alarm sensors and checks the presence and/or absence of the respective incoming alarms.

Table 2-35 “i_e1_crc.squ” test sequence: test cases used and their functions

Test case, function name	Function
Check AIS_2/AIS_64 _check_alarm	This test stimulates the alarm sensors and checks the presence and/or absence of the respective incoming alarms.
CRC vs. E-Bit _check_stimulus_response	This test stimulates the error indication sensors of SDH or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site.
FAS non-transp. _check_stimulus_response	This test stimulates the error indication sensors of SDH or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site.
Set Sa4-Bit Seq. _set_sa_bit_sequence	Sets the patterns of the Sa bits of a 2 Mbit/s signal.
Check Sa4 Seq. _check_sa_bit_sequence	Compares the bit sequence of one Sa bit received at the RX site with an expected pattern.
Inject Jitter _setup_jitter_stimulus	Sets up jitter stimulus.
Jitter Measurement check_jitter	Up to 20 user selectable values for jitter amplitude / jitter frequency can be set. A bit error check is performed at each of these points.
Jitter Tolerance _check_jitter_tolerance	Up to 20 user selectable values for jitter amplitude / jitter frequency can be set. A bit error check is performed at each of these points.
Pulling Range _check_clock_pulling	Checks the pulling range of a UUT by transmitting a signal with an increasing clock offset and checking for bit errors and alarms during the measurement.
Check Trouble _check_trouble	Checks the ANT-20’s event queue for events (alarms, errors, ...).
Init. G.826 00S _set_and_start_g826	Sets the parameters for a G.826 evaluation.
Get G.826 Result _check_g826	Checks the results of the G.826 evaluation.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-35 “i_e1_crc.squ” test sequence: test cases used and their functions

2.7.15 i_e1_trs.squ

This sequence is the basis for the line-up of a 2 Mbit/s unterminated, transparent circuit, including pulling range, G.826 and jitter measurements.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check needed options _check_options_one_of	Checks whether at least one of the ANT-20/ANX-920 options listed is present.
Option Error _user_io	A user-defined message is displayed if none of the options checked in the “Check needed options” test case are available. The “Goto End” test case is executed when you click on “OK”.
Goto End	“Goto End” is only performed if none of the options checked in the “Check needed options” test case are available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
TX 2 Mb/s unfr. _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX HDB3 PRS11 _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
TX Jitter 2 Mb/s _setup_TX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
RX 2 Mb/s unfr. _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX HDB3 PRS11 _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
RX Jitter 2 Mb/s _setup_RX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
View Alarm _read_actual_alarms	Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval). This test case does not have any automatic PASS/FAIL evaluation.
Continuity Check _check_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Check LOS / AIS_2 _check_alarm	This test stimulates the alarm sensors and checks the presence and/or absence of the respective incoming alarms.
Inject Jitter _setup_jitter_stimulus	Sets up jitter stimulus.

Table 2-36 “i_e1_trs.squ” test sequence: test cases used and their functions

Test case, function name	Function
Jitter Measurement _check_jitter	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.
Jitter Tolerance _check_jitter_tolerance	Up to 20 user selectable values for jitter amplitude / jitter frequency can be set. A bit error check is performed at each of these points.
Pulling Range _check_clock_pulling	Checks the pulling range of a UUT by transmitting a signal with an increasing clock offset and checking for bit errors and alarms during the measurement.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-36 “i_e1_trs.squ” test sequence: test cases used and their functions

2.7.16 i_cj_e1.squ

This sequence is a demo example for stressing the DUT with pointer jitter and maximum mapping jitter. It sets the critical mapping offset which produces maximum mapping jitter. Run the “*_mj_*.squ” sequence to determine the critical mapping offset.

See “i_o_172.squ” test sequence (Sec. 2.7.20, Page 1-60).

2.7.17 i_e1_jit.squ

This sequence demonstrates a jitter measurement for an E1 - 2 Mbit/s signal.

See “a_jitter.squ” test sequence (Sec. 2.6.12, Page 1-29).

2.7.18 i_mj_e1.squ

This sequence is a demo example for determining the critical mapping offset for an E1 - 2 Mbit/s STM-1 AU4 VC12 ASYNC signal where mapping jitter is at a maximum.

See “a_mj_ds1.squ” basic test sequence (Sec. 2.6.14, Page 1-31).

2.7.19 i_list.squ

This sequence demonstrates the use of the “LIST” feature, i.e. performing a number of different tests on a group of tributaries defined within one test case. In this example, a transparency check and alarm sensor checks are performed on all 2 Mbit/s tributaries of an STM-1 C12 signal.

See “a_list.squ” test sequence (Sec. 2.6.13, Page 1-30).

2.7.20 i_o_172.squ

This sequence can be used to assess the combined tributary jitter (mapping jitter and pointer jitter) of a 2 Mbit/s signal (ANT-20 RX) derived from an STM-1 C12 signal (ANT-20 TX), with superimposed tributary offset, pointer sequences and cool-down periods as defined in O.172 / G.783. Similar sequences for different mappings and bit rates can be easily derived from this example for testing other types of UUT.

Test case, function name	Function
TX STM-1 C12 2M unfr _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX 2M unfr _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the "_set_signal_structure" test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the "_set_signal_structure" test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
set-up TX_jitter _setup_TX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
set-up RX jitter _setup_RX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
find_max_mapping_jit _find_max_mapping_jitter	Determines the critical mapping offset causing a jitter maximum.
Set Map.Offs -50 ppm _set_mapp_offset	Sets the mapping to the value which was determined with the "_find_max_mapping_jitter" test case.
check_jitter #0 _check_jitter	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.
cool down #1 _wait	Waits for a specified period.
init jitter meter _check_jitter	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.
Initial Ptr Mov _set_ptr_action	Injects pointer movements or pointer sequences.
init period _wait	Waits for a specified period.
Single alternating _set_ptr_action	Injects pointer movements or pointer sequences.
Ptr Monitor #1 _ptr_move	Reads the values of an AU and/or TU pointer at fixed intervals and displays them in a graph.

Table 2-37 "i_o_172.squ" test sequence: test cases used and their functions

Test case, function name	Function
check_jitter #1 _check_jitter	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.
Cancel Ptr #1 _set_ptr_action	Injects pointer movements or pointer sequences.
cool down #2 _wait	Waits for a specified period.
Regular + double _set_ptr_action	Injects pointer movements or pointer sequences.
check_jitter #2 _check_jitter	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.
Cancel Ptr #2 _set_ptr_action	Injects pointer movements or pointer sequences.
cool down #3 _wait	Waits for a specified period.
Regular + missing _set_ptr_action	Injects pointer movements or pointer sequences.
check_jitter #3 _check_jitter	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.
Cancel Ptr #3 _set_ptr_action	Injects pointer movements or pointer sequences.

Table 2-37 “i_o_172.squ” test sequence: test cases used and their functions

2.7.21 i_g826.squ

This sequence is intended for checking the correct function of the performance monitoring sensors of an STM-1 C4 card. It allows injection of a flexible scenario of anomalies and defects in order to determine the G.826 evaluation response of the UUT. A G.826 analysis is performed at the same time on the ANT-20 itself as a check.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
TX STM-1 C4 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX STM-1 C4 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX_Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].

Table 2-38 “i_g826.squ” test sequence: test cases used and their functions

Test case, function name	Function
RX_Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the "_set_signal_structure" test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Initalize _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on "OK".
Setup, Start G826 _set_and_start_g826	Sets the parameters for a G.826 evaluation.
monitor 20 s _monitor_g826	Monitors the results of the G.826 evaluation.
B3_LOW _set_stimulus	Inserts an error ratio or frequency offset until switched off by the "B3_LOW_OFF" test case.
monitor 2 s _monitor_g826	Monitors the results of the G.826 evaluation.
B3_LOW_OFF _set_stimulus	Ends the "B3_LOW" test case.
monitor 11 s _monitor_g826	Monitors the results of the G.826 evaluation.
B3_HIGH _set_stimulus	Inserts an error ratio or frequency offset until switched off by the "B3_HIGH_OFF" test case.
monitor 8 s _monitor_g826	Monitors the results of the G.826 evaluation.
B3_HIGH_OFF _set_stimulus	Ends the "B3_HIGH" test case.
monitor 12 s _monitor_g826	Monitors the results of the G.826 evaluation.
B3_HIGH_UAS _set_stimulus	Inserts an error ratio or frequency offset until switched off by the "B3_HIGH_UAS_OFF" test case.
monitor 13 s _monitor_g826	Monitors the results of the G.826 evaluation.
B3_HIGH_UAS_OFF _set_stimulus	Ends the "B3_HIGH_UAS" test case.
monitor 14 s _monitor_g826	Monitors the results of the G.826 evaluation.
HP_REI_HIGH _set_stimulus	Inserts HP-REI ratio until switched off by the "HP_REI_H_OFF" test case.
HP_REI_H_OFF _set_stimulus	Ends the "HP_REI_HIGH" test case.
monitor 18 s _monitor_g826	Monitors the results of the G.826 evaluation.

Table 2-38 "i_g826.squ" test sequence: test cases used and their functions

Test case, function name	Function
HP_REI_LOW _set_stimulus	Inserts an error ratio or frequency offset until switched off by the "HP_REI_L_OFF" test case.
monitor 19 s _monitor_g826	Monitors the results of the G.826 evaluation.
HP_REI_L_OFF _set_stimulus	Ends the "HP_REI_LOW" test case.
monitor 60 s _monitor_g826	Monitors the results of the G.826 evaluation.
LOS _set_alarm	Inserts an alarm until switched off by the next test case.
monitor 21 s _monitor_g826	Monitors the results of the G.826 evaluation.
HP-UNEQ _set_alarm	Inserts an alarm until switched off by the next test case.
monitor 23 s _monitor_g826	Monitors the results of the G.826 evaluation.
HP-TIM _set_alarm	Inserts an alarm until switched off by the next test case.
monitor 25 s _monitor_g826	Monitors the results of the G.826 evaluation.
AU-LOP _set_alarm	Inserts an alarm until switched off by the next test case.
monitor 27 s _monitor_g826	Monitors the results of the G.826 evaluation.
AU-AIS _set_alarm	Inserts an alarm until switched off by the next test case.
monitor 29 s _monitor_g826	Monitors the results of the G.826 evaluation.
HP-RDI _set_alarm	Inserts an alarm until switched off by the next test case.
monitor 31 s _monitor_g826	Monitors the results of the G.826 evaluation.
All Alarms OFF _set_alarm	Inserts an alarm until switched off by the next test case.
Results G826 _check_g826	Checks the results of the G.826 evaluation.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on "OK".

Table 2-38 "i_g826.squ" test sequence: test cases used and their functions

2.7.22 i_jitter.squ

This sequence demonstrates how to correctly set up a sequence that includes jitter measurements. Please note that a BASIC device and a JITTER device are used in this configuration. Use this sequence (especially the first 6 test cases) as a sample whenever setting up a jitter test sequence (up to STM-4). Measurements include JTF, intrinsic jitter and MTJ.

See “a_jitter.squ” test sequence (Sec. 2.6.12, Page 1-29).

2.7.23 i_4jitt.squ

This sequence demonstrates a jitter measurement for an STM-4 signal.

See “a_jitter.squ” test sequence (Sec. 2.6.12, Page 1-29).

2.7.24 i_stm4.squ

This sequence demonstrates tests for a STM-4 AU4 VC4 payload access FRAMED PATT E4 140 Mbit/s signal.

See “i_c4.squ” basic test sequence (Sec. 2.7.7, Page 1-46).

2.7.25 i_jit16.squ

This sequence demonstrates the use of the STM-16 jitter module with the CATS software and includes jitter measurements, MTJ, FTJ, and F-MTJ.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Check Jitter Options _check_options_one_of	Checks whether at least one of the ANT-20/ANX-920 options listed is present.
Check STM-16 Options _check_options_one_of	Checks whether at least one of the ANT-20/ANX-920 options listed is present.
Goto End	“Goto End” is only performed if none of the options checked in the “Check Jitter Options” test case or if none of the options checked in the “Check STM-16 Options” test case are available. Jumps to the end of the sequence to the “End” test case. The “End” test case is performed and the sequence ends.
Setup TX-RX Signal _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
Setup RX Jitter _setup_RX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).

Table 2-39 “i_jit16.squ” test sequence: test cases used and their functions

Test case, function name	Function
Setup RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the "_set_signal_structure" test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Setup TX Jitter _setup_TX_jitter	Selects the bit rate for jitter measurements and jitter transfer measurements as well as the bit rate for jitter generation (FTJ, MTJ).
Setup TX Adaptation set_tx_adaptation2	Sets the signal parameters that are not covered by the "_set_signal_structure" test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
Check RX Opt. Level _read_rx_optical_level	Reads the current optical level and compares it with the user-defined limits.
Read Actual Alarms _read_actual_alarms	Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval). This test case does not have any automatic PASS/FAIL evaluation.
Inject Jitter _setup_jitter_stimulus	Sets up jitter stimulus.
Check Jitter _check_jitter	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.
Jitter OFF _setup_jitter_stimulus	Sets up jitter stimulus.
FTJ _check_jitter_tolerance	Jitter measurement, e.g. in the presence of TX stimuli such as offset, jitter, and pointer bursts set by preceding test cases.
MTJ _check_mtj	Up to 10 user selectable values for jitter frequencies can be set. A bit error check is performed at each of these points for the given expected maximum jitter amplitude. The measurement is started using the "mask" jitter amplitude value. If no bit errors and no alarms occur during the measurement period ("mask" value), jitter is increased until an error or alarm is detected. The jitter amplitude value below the value causing an error or alarm is considered the MTJ result. If a bit error or alarm occurred during the first measurement period, jitter is decreased until no error or alarm is found. This value (frequency and amplitude) is considered the MTJ result.
JTF _check_jitter_transfer	A fixed jitter amplitude is set by TX. Jitter frequency list is user selectable (20 values). Jitter amplitude at RX is measured for each frequency point. Reference measurement (Yes/No) is used to eliminate any error caused by the measurement setup. Values are stored for all subsequent runs of this test case. When a sequence is loaded, the reference results are deleted.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on "OK".

Table 2-39 "i_jit16.squ" test sequence: test cases used and their functions

2.7.26 i_stm16.squ

This sequence demonstrates tests for a STM-16 AU4 VC12 ASYNC payload access UNFRAMED E1 2 Mbit/s signal.

See “i_c4.squ” basic test sequence (Sec. 2.7.7, Page 1-46).

2.7.27 i_stm64.squ

This sequence demonstrates tests for STM-64 with VC4-16C or VC4-16C substructures, including reading the optical level, and checking transparency, TSE, B2, and alarms.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
TX/RX STM-64 VC4-16C _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.
Set Trib _set_tributaries	Sets the tributary/channel number of an ITU-T signal. All subsequent test cases are performed on the tributary/channel defined here.
Wait some sec _wait	Waits for a specified period.
Read Level _read_rx_optical_level	Reads the current optical level and compares it with the user-defined limits.
Actual Alarm Monitor _read_actual_alarms	Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval). This test case does not have any automatic PASS/FAIL evaluation.
Check Transparency _check_transparency	Checks the transparency of a tributary or channel defined by a previous signal structure and tributary set-up test case. TX and RX payload must be identical, but the channel number and signal structure may be different.
Check TSE _check_stimulus_response	This test stimulates the error indication sensors of SDH or PDH UUTs. A number of single errors or a fixed error ratio is sent by the TX site. The presence of the expected number of incoming errors or the alarm duration is checked at the RX site.
Check B2 Errors _check_stimulus_response	See “Check TSE” test case.
Check Alarm _check_alarm	This test stimulates the alarm sensors and checks the presence and/or absence of the respective incoming alarms.

Table 2-40 “i_stm64.squ” test sequence: test cases used and their functions

2.7.28 i_pdhsca.squ

This sequence demonstrates an error and alarm scan of selected tributaries of an E4 - 140 Mbit/s PDH signal.

See “a_scan.squ” basic test sequence (Sec. 2.6.18, Page 1-38).

2.7.29 i_scan.squ

This sequence scans defined lists of embedded AU3, TUG3, TUG2, TU11 and TUG12 tributaries of a SDH signal.

See “a_scan.squ” test sequence (Sec. 2.6.18, Page 1-38).

2.7.30 i_wantol.squ

This sequence tests the wander tolerance of a STM-1 circuit.

Please note that the ANT-20 Wander measurement option is needed if you want to check the MTIE value, which is determined after the injection of each wander frequency.

A reference clock of 2.048 MHz must be connected to the ANT-20 Wander Reference Clock input, otherwise an LTI alarm will make the MTIE measurement impossible.

If you only want to generate wander without making measurements, you can also use “Clock from RX” in the TX adaptation test case and offset the TX clock against the received clock. An external clock is recommended otherwise.

Test case, function name	Function
Welcome _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
Cables _show_image	Displays a bitmap.
TX STM1 C12 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX STM1 C12 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
Message _user_io	A user-defined message is displayed when this test is run. The next test case is executed when you click on “OK”.
TX Adaptation _set_tx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level, optical wavelengths etc. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
RX Adaptation _set_rx_adaptation2	Sets the signal parameters that are not covered by the “_set_signal_structure” test case, such as interface settings, PRBS sequence, digital word, output level etc. Additional parameters are sensitivity and equalizer settings.

Table 2-41 “i_wantol.squ” test sequence: test cases used and their functions

Test case, function name	Function
Wander Reference _scpi_io	Sends user-defined SCPI commands to the specified instrument via the open interface. If a query is sent, the response is displayed. No PASS/FAIL result is derived from the contents of the received result string.
Wander Tolerance _check_wander_tolerance	Wander can be applied sequentially at ten different “frequency / UI” points. A bit error test (TSE errors) and alarm check are carried out for each individual test point. Wander is applied as an approximated sine wave by slowly changing the ANT-20 TX signal against an external reference clock signal. A table allows entry of individual test points by specifying the wander amplitude value in [Unit Intervals] peak-peak and the corresponding wander frequency in [Hz]. Due to the very low frequencies used, the number of wander periods for each given frequency is limited to five. The duration field automatically calculates the estimated time for each single test point run. This estimate includes the period count as well as the preceding delay time. The delay time allows the UUT to settle before the next wander stimulus is applied. The signal bit rate is read from the ANT-20 automatically, so it is not necessary to set it with the wander tolerance test case. The “_set_signal_structure” test case should precede the “_check_wander_tolerance” test case to set the desired bit rate.
End _user_io	A user-defined message is displayed when this test is run. The sequence ends when you click on “OK”.

Table 2-41 “i_wantol.squ” test sequence: test cases used and their functions

2.7.31 synch.squ

This sequence can be used to measure the time delay between a stimulus in one specific SDH overhead byte (ANT-20 TX) and the subsequent reaction in another (or the same) overhead byte as received by the ANT-20 RX. This application determines how long it takes to change the synchronization direction over an STM-16 ring by looking at the S1 byte. Please contact the WWG Solution Center if you have similar applications.

Test case, function name	Function
TX STM16-STM1 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal. This is just a set-up test case without any PASS/FAIL evaluation.
RX STM16-STM1 _set_signal_structure	Sets the ITU-T SDH/PDH structure of the ANT-20 RX signal. This is just a set-up test case without any PASS/FAIL evaluation.
Actual Alarm Monitor _read_actual_alarms	Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n = number of readouts, m = update interval). This test case does not have any automatic PASS/FAIL evaluation.
Read Optical Level _read_rx_optical_level	Reads the current optical level and compares it with the user-defined limits.

Table 2-42 “synch.squ” test sequence: test cases used and their functions

Test case, function name	Function
SOH/POH MONITOR 1 _read_actual_SOH_POH	Reads the complete SOH and POH of an STM-1 C4, C3 or C12 signal. Readout is done in n samples separated by m seconds of update interval time. The final readout is displayed in the Sequence Display window and written into the ASCII report file. This test case is intended to give a quick overview of the overhead; it does not have any automatic PASS/FAIL evaluation.
SI=222, Start Capture _scpi_io	Sends user-defined SCPI commands to the specified instrument via the open interface. If a query is sent, the response is displayed. No PASS/FAIL result is derived from the contents of the received result string.
SOH/POH MONITOR 2 _read_actual_SOH_POH	See “SOH/POH MONITOR 1” test case.
Wait x sec _wait	Waits for a specified period.
Set S1 = FF _set_SOH_POH_byte	Sets the contents of one overhead byte.
SOH/POH MONITOR 3 _read_actual_SOH_POH	See “SOH/POH MONITOR 1” test case.
Wait y sec _wait	Waits for a specified period.
Stop Capturing _scpi_io	Ends the “SI = 222, Start Capture” test case.
Read Captured Data _scpi_io	Sends user-defined SCPI commands to the specified instrument via the open interface. If a query is sent, the response is displayed. No PASS/FAIL result is derived from the contents of the received result string.

Table 2-42 “synch.squ” test sequence: test cases used and their functions

Notes:

3 Commissioning

This section contains information about

- System requirements
- The contents of the various CATS Test Sequencer variants

This section also tells you how to

- Install your CATS Test Sequencer
- Start your CATS Test Sequencer
- Insert command line parameters
- Set up a log-in procedure

3.1 System requirements

Possible Configurations

Configurations using Windows 95 or NT together with LabWindows CVI 5.01 have been tested. Configurations using Windows 3.11 together with LabWindows CVI 5.01 have been tested.

Requirements for CATS (BN 3045/01)

- PC with Pentium 133 MHz or above, 16 MB RAM or more
- Windows 95, Windows 98 or Windows NT (Windows 3.11 is still supported)
- National Instruments LabWindows CVI 5.01 run-time engine
- VISA 32 bit library

If GPIB control is required, a GPIB card (GPIB PCMCIA or GPIB PCI from National Instruments) is needed.

Requirements for ANT-20 CATS Test Sequencer (BN 3035/xx.xx) and CATS DWDM (BN 3045/93.43)

The ANT-20 CATS Test Sequencer runs on any ANT-20/ANT-20E or DominoCom. The TCP/IP interface is available for ANT-20E, as the test set runs Windows 95 (Windows 3.11 is still supported).

3.2 Package contents

This section indicates the contents of the various CATS Test Sequencer variants.

3.2.1 CATS (BN 3045/01)

- CATS Test Sequencer Software (32 bit version)
- CATS Test Sequencer handbook

3.2.2 CATS DWDM (BN 3045/93.43)

- CATS Test Sequencer Software, pre-installed or for later installation (software is supplied on five 3½” floppy disks)
- CATS Test Sequencer handbook

3.2.3 ANT-20 CATS Test Sequencer (BN 3035/95.90)

- CATS Test Sequencer Software, pre-installed or for later installation (software is supplied on three 3½” floppy disks)
- CATS Test Sequencer handbook

3.2.4 ANT-20 CATS Test Sequencer with TCP/IP interface (BN 3035/95.91)

- CATS Test Sequencer Software
- PCMCIA Ethernet card, MICROSYSTEM GMBH (SCM) with Driver disk (or an equivalent PCMCIA Ethernet card
- “Libraries and Drivers” floppy disk
- CATS Test Sequencer handbook

3.2.5 ANT-20 CATS Test Sequencer with DDE interface (BN 3035/95.92)

- CATS Test Sequencer Software
- CATS Test Sequencer handbook

3.3 Installation

A number of requirements must be met by your controller PC to enable successful operation of CATS. The following pages indicate the drivers and software packages that are necessary and give hints on how to install them and how to verify their correct function.

However, these descriptions do not replace the instructions supplied with the various hardware and software that needs to be installed. Only a brief description is given of the sequence of steps required for proper CATS Test Sequencer installation.

Wavetek Wandel Goltermann has prepared a set of floppy disks that can be used if the original floppy disks supplied with the various hardware are defective or cannot be located. The original software should always be used if possible. Only use the Wavetek Wandel Goltermann floppy disks in the event of a real emergency.

For licensing reasons, any installations derived from these emergency disks should be erased after a demo, or be replaced by the official version supplied with the hardware item concerned.

3.3.1 CATS (BN 3045/01)

This section describes in detail all the requirements that must be met to prepare your PC for proper installation of the CATS Test Sequencer, followed by instructions on how to install the CATS Test Sequencer software.

The sequence given in this section must be followed exactly when you are installing the software yourself.

Installation procedure

Notice: Before you perform any installation work, check which installation steps (if any) have already been performed on your PC to avoid corruption of existing, working installations.

Installing the GPIB card and “GPIB for Windows” driver

Only needed if control via GPIB is required.

Install the GPIB card (National Instruments PCMCIA-GPIB, AT-GPIB or PCI-GPIB) according to the instructions supplied with the card. The Windows (not the DOS) driver must be installed.

1. Install the GPIB card.
2. Install the “GPIB for Windows” driver.
   After successful installation of the driver software, the Windows Program Manager will contain a group labeled “NI-488.2 GPIB Software”.

You can perform a function check after installation is complete:

1. Check the hardware using the “Hardware Diagnostic Test” program.
2. Check the software using the “Software Diagnostic Test” program.
3. Use “Interactive Control” to ensure that a communication path is established between the controlling PC and the hardware connected to it.

Installing VISA-VXI Plug & Play software

The VISA-VXI Plug & Play Software is part of the GPIB software or the CVI package.

This software is essential if you want to control VXI based test systems or use the V.24 interface from the CATS Test Sequencer. Please use the floppy disks supplied with your VXI controller hardware. Wavetek Wandel Goltermann provides a set of three floppy disks for installing the software for controlling VXI hardware from National Instruments.

1. Install “NI VISA 1.x” by running the setup program from the corresponding floppy disk.
2. Install “VISA Interface for CVI” by running the setup program from the corresponding floppy disk.
3. Install the “VISA Patch” by following the hints given in the file “visa.txt” on the corresponding floppy disk.

Installing the CATS Test Sequencer

⇒ Run “SETUP.EXE” from the disk labeled “ANT-20 CATS Test Sequencer, Executable + Test cases”.

3.3.2 ANT-20 CATS Test Sequencer (BN 3035/95.90) and CATS DWDM (BN 3045/93.43)

The ANT-20 CATS Test Sequencer is normally supplied pre-installed when ordered together with your ANT-20 test set. If you purchased it separately, follow the steps given below to install it.

Installation procedure

Determining the ANT-20 software version

⇒ Check the software version of your ANT-20 by starting the ANT-20 user interface and looking up the respective entry under “HELP”, “ABOUT”.

Preparing to install

If an older version or a test version of the CATS Test Sequencer is already installed, some preparation is needed before you can install the latest version. In most cases, there will be a test version installed on your ANT-20.

Alternative 1: A previous version of the CATS Test Sequencer is installed on your ANT-20.

1. Save any personal sequences (files ending with “.squ” that were not provided by Wavetek Wandel Goltermann) that you want to use again by copying them from the “.\SEQUENCE” directory to a temporary location.
2. Save any reports (files ending with “.rpt”) that you want to keep by copying them from the “.\REPORT” directory to a temporary location.
3. Delete the contents of the directory “.\WGTX_SRC” (including subdirectories) from your ANT-20.

Alternative 2: A test version of the CATS Test Sequencer is installed on your ANT-20.

⇒ Delete the contents of the directory “C:\CVI\WGTX_SRC” (including subdirectories) from your ANT-20.

Installing the CATS Test Sequencer

Note: Do not change the directory proposed by the setup program.
With the CATS DWDM (BN 3045/93.43) package, installation of the “NI VISA” software starts automatically when installation of the CATS Test Sequencer is completed. Use the default parameters for the installation of the “NI VISA” software.

1. Run “SETUP.EXE” from the disk labeled “ANT-20 CATS Test Sequencer, Executable + Testcases”.
2. Follow the instructions displayed on the screen by the installation program.

3.3.3 ANT-20 CATS Test Sequencer with TCP/IP interface (BN 3035/95.91)

Installation is described using the PC (workstation) as client and ANT-20 as server configuration as an example.

3.3.3.1 Installation on the ANT-20 (server)

Installation procedure

Installing the CATS Test Sequencer

To install the ANT-20 CATS Test Sequencer with TCP/IP interface, follow the installation procedure described for the ANT-20 CATS Test Sequencer (BN 3035/95.90).
See Sec. 3.3.2, Page 1-75.

Installing the PCMCIA Ethernet card

The CATS Test Sequencer with TCP/IP interface package includes a MICROSYSTEM GmbH (SCM) PCMCIA Ethernet card. This card must be installed for the CATS Test Sequencer software to work.

The card can be inserted when the ANT-20 is switched on or switched off.

1. Plug the card into the ANT-20’s PCMCIA slot.
2. Connect the device to your LAN.
3. If your ANT-20 was switched off, switch it on now.
   The automatic hardware detection feature of Windows 95 will recognize that a new hardware component has been added.
   The “Driver Device Wizard” panel pops up.
4. Click the “Next” button.
   Windows will now look for a driver file.
   Windows will not locate the file, so select the “Other Location” button.
5. Select the browse button.
6. Insert the driver disk supplied with the SCM card.
7. Select floppy disk “A” and the “Win95” folder.
8. Confirm with “OK”.
9. Confirm with “OK” in the “Other Location” window.
10. Select the “Finish” button in the “Driver Device Wizard” panel.
    Windows will complete the installation.
    After this, the “Network” panel will pop up.

The following components must be installed for the Test Sequencer to work properly:

- Client for MS Network
- SCM Ethernet Combo Card
- TCP/IP Protocol

Follow the instructions below to install any missing components:

Installing missing components

1. Click the “Add” button.
2. Select the type of component you want to install.
3. Select the “Add” button.
   A window showing a detailed list of all the available components opens.
4. Select a component.
5. Confirm with OK.

Uninstalling components

If you have installed the wrong component by mistake, you can uninstall it as follows:

1. Select the component in the Configuration panel.
2. Click the “Remove” button.

Setting the properties of a component

Various properties need to be set for the installed components. For example, the TCP/IP Protocol properties are:

- IP Address and Subnet Mask
- Gateway Address
- DNS Configuration

Notice: Please ask your Network Administrator for detailed information.

To set the properties, proceed as follows:

1. Select the component (under Control Panel / Network) and then click the “Properties” button.
   – or –
   Double click on the component.
2. Make the settings required.
3. Confirm with “OK”.

Completing installation

After you have made the settings, Windows will prompt you whether to restart Windows or not.

⇒ Restart Windows.

3.3.3.2 Installation on the PC / workstation (client)

Installation procedure

Creating the client application

You must create your own client application on your PC / workstation. The C source code for the example client (file “Tcpclnt.c”) may be useful as a starting point.

✓ Operating system: Windows 95, 98, NT

1. Copy the following files from the directory “.\dde_clnt” on your ANT-20 to your PC / workstation:
   – Example client “tcpcl32.exe”
   – C source file “Tcpclnt.c“
   – H file “Tcpclnt.h“
   – UIR file “Tcpclnt.uir“
2. Create your own client application.
3. If the PCMCIA card and the CATS Test Sequencer are already installed on your ANT-20, you can activate TCP/IP control (see Sec. 3.5.3, Page 1-80).

3.3.4 ANT-20 CATS Test Sequencer with DDE interface (BN 3035/95.92)

To install the ANT-20 CATS Test Sequencer with DDE interface, follow the installation procedure described for the ANT-20 CATS Test Sequencer (BN 3035/95.90). See Sec. 3.3.2, Page 1-75.

3.4 Cabling, network connection

This section contains information on connecting CATS BN 3045/xx and on the network connection for the ANT-20 CATS Test Sequencer with TCP/IP interface (BN 3035/95.91).

3.4.1 CATS (BN 3045/01)

⇒ Connect the devices to be controlled to your PC using an IEEE488.2 or V.24 cable.

3.4.2 CATS DWDM (BN 3045/93.43)

⇒ Connect the ANT-20 to your OSA-155 using an IEEE488.2 or V.24 cable.

3.4.3 ANT-20 CATS Test Sequencer with TCP/IP interface (BN 3035/95.91)

1. Connect the ANT-20 to the network.
2. Connect the client PC (workstation) to the network.

3.5 Starting the CATS Test Sequencer

This section tells you how to start the Test Sequencer software.

3.5.1 CATS (BN 3045/01)

1. Switch on your PC.
2. Locate the program group “ANT-20 CATS Test Sequencer” and double-click on the “CATS Test Sequencer” icon.
   The default sequence “_dem_c12.squ” will load.
   The “CATS Test Sequencer” window opens.

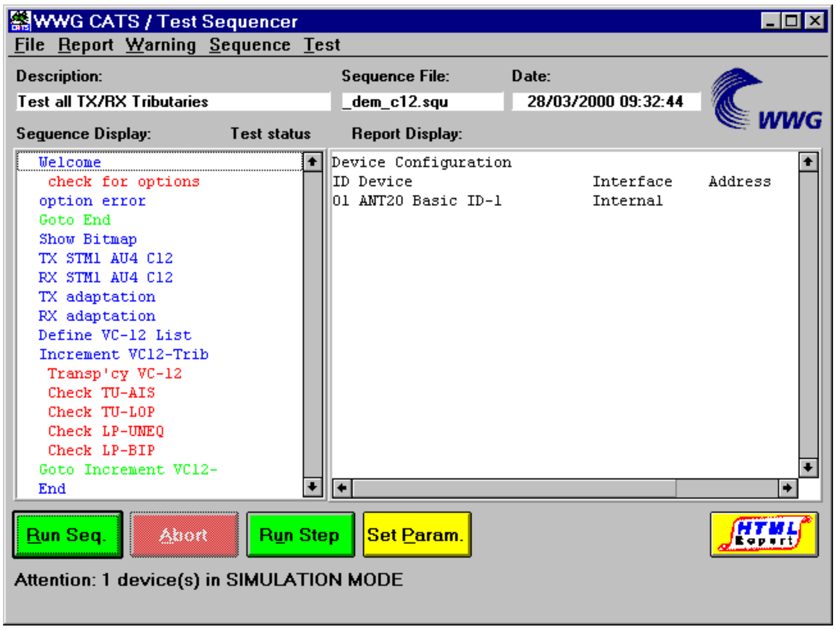


Fig. 3-1 “CATS Test Sequencer” window

3.5.2 ANT-20 CATS Test Sequencer (BN 3035/95.90) and CATS DWDM (BN 3035/93.43)

Your ANT-20 will usually start in “ANT-20 mode”, which means that you will see the ANT-20 world map “... for better communications” welcome screen, followed by a number of virtual instruments popping up. This mode is for interactive operation of the ANT-20. You must exit from this mode before starting the ANT-20 CATS Test Sequencer.

Tip: If you want to run CATS Test Sequencer without loading the ANT-20 interactive user interface on boot-up, delete the ANT-20 icon from the start-up group.

1. Switch on your ANT-20.
2. If your ANT-20 has booted up in “ANT-20” mode, close the ANT-20 user menu.
3. Locate the program group “ANT-20 CATS Test Sequencer” and double-click on the “CATS Test Sequencer” icon.
   The default sequence “_dem_c12.squ” will load.
   The “CATS Test Sequencer” window opens (see Fig. 3-1).

3.5.3 ANT-20 CATS Test Sequencer with TCP/IP interface (BN 3035/95.91)

Example for a configuration with PC as client and ANT-20 as server.

Starting the CATS Test Sequencer on the ANT-20

Tip: If you want to run CATS Test Sequencer without loading the ANT-20 interactive user interface on boot-up, delete the ANT-20 icon from the start-up group.

1. Switch on your ANT-20.
2. If your ANT-20 has booted up in “ANT-20” mode, close the ANT-20 user menu.
3. Locate the program group “ANT-20 CATS Test Sequencer” and double-click on the “CATS Test Sequencer” icon.
   The default sequence “_dem_c12.squ” will load.
   The “CATS Test Sequencer” window opens in “Standard” mode.

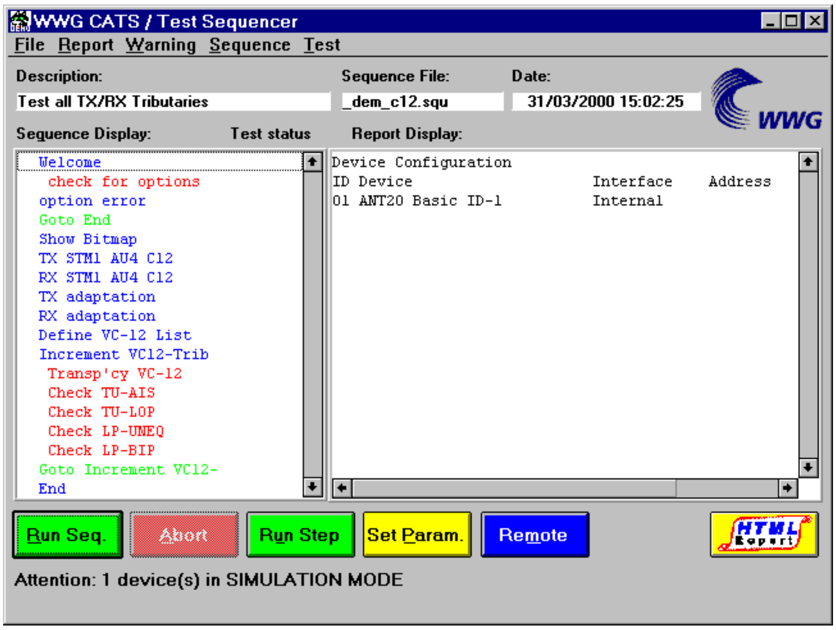


Fig. 3-2 “CATS Test Sequencer” window in “Standard” mode

Note: The following steps can be ignored if you want to run the CATS Test Sequencer in "Standard" mode. "Standard" mode operation is described in Sec. 4.1, Page 1-87.

Switching to "Remote" mode

The CATS Test Sequencer on the ANT-20 must be switched to "Remote" mode before you can start the client software on the PC. In "Remote" mode, the ANT-20 CATS Test Sequencer acts as a server waiting to be called from a client.

Note: If you set the command line option "-tcp", the CATS Test Sequencer will start directly in "Remote" mode.

⇒ Click on the "Remote" button.
The CATS Test Sequencer switches to "Remote" mode.
The "TCP/IP Remote Interface" window is displayed in the "CATS Test Sequencer" window over the button bar.

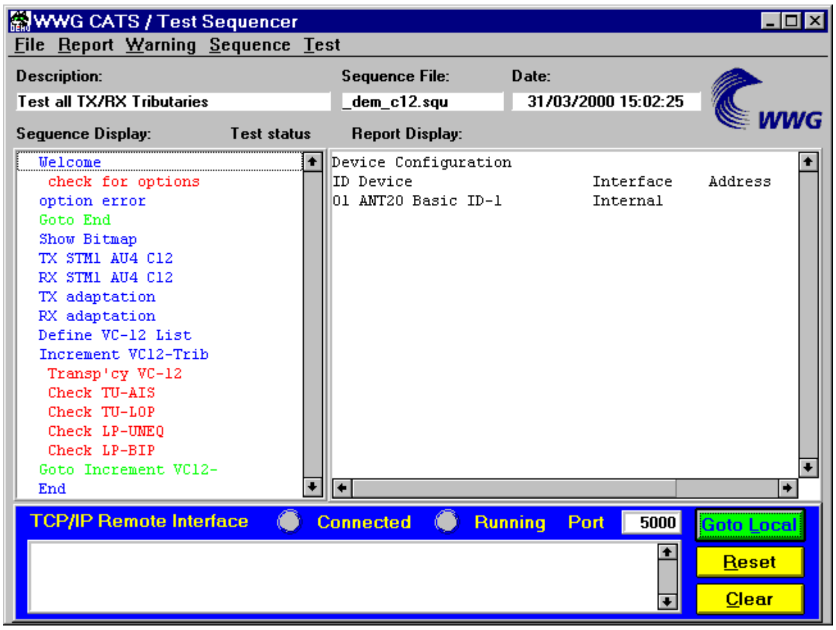


Fig. 3-3 "CATS Test Sequencer" window in "Remote" mode

The "TCP/IP Remote Interface" window displays the commands that are sent from the client software and prevents use of the CATS Test Sequencer button bar in "Remote" mode.

Note: Click the "Goto Local" button to switch the CATS Test Sequencer back to "Standard" mode. The "Goto Local" button is disabled if the connection between the client and the server is active.

Starting the CATS Test Sequencer client software on the PC

✓ The CATS Test Sequencer on the ANT-20 is in “Remote” mode.

1. Switch on your PC.
2. Double-click on the “TCP/IP client” icon.
   The “Server Name?” window opens.
3. Enter the name of the computer running the CATS Test Sequencer (server) software in the “Server Name?” window.
   The “Port Number?” window opens.

Note: The port number to be entered in this window is displayed in the “TCP/IP Remote Interface” window of the CATS Test Sequencer (server) software to the right next to “Port” (see Fig. 3-3, Page 3-81).

4. Enter the port number of the CATS Test Sequencer (server) software in the “Port Number?” window.
   The “TCP client” window opens.

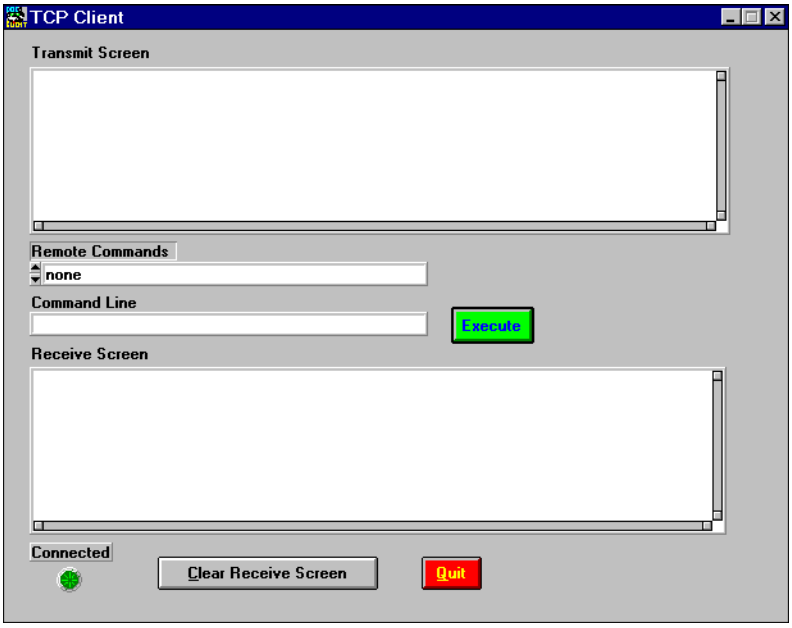


Fig. 3-4 “TCP client” window

The green “Connected” indicator on the PC and ANT-20 shows that the ANT-20 CATS Test Sequencer is ready to receive and execute commands.

Sec. 4.2, Page 1-90 describes how to operate the CATS Test Sequencer from the “Client” window.

3.5.4 ANT-20 CATS Test Sequencer with DDE interface (BN 3035/95.92)

Starting the CATS Test Sequencer

Tip: If you want to run CATS Test Sequencer without loading the ANT-20 interactive user interface on boot-up, delete the ANT-20 icon from the start-up group.

1. Switch on your ANT-20.
2. If your ANT-20 has booted up in “ANT-20” mode, close the ANT-20 user menu.
3. Locate the program group “ANT-20 CATS Test Sequencer” and double-click on the “CATS Test Sequencer” icon.
   The default sequence “_dem_c12.squ” will load.
   The “CATS Test Sequencer” window opens in “Standard” mode.

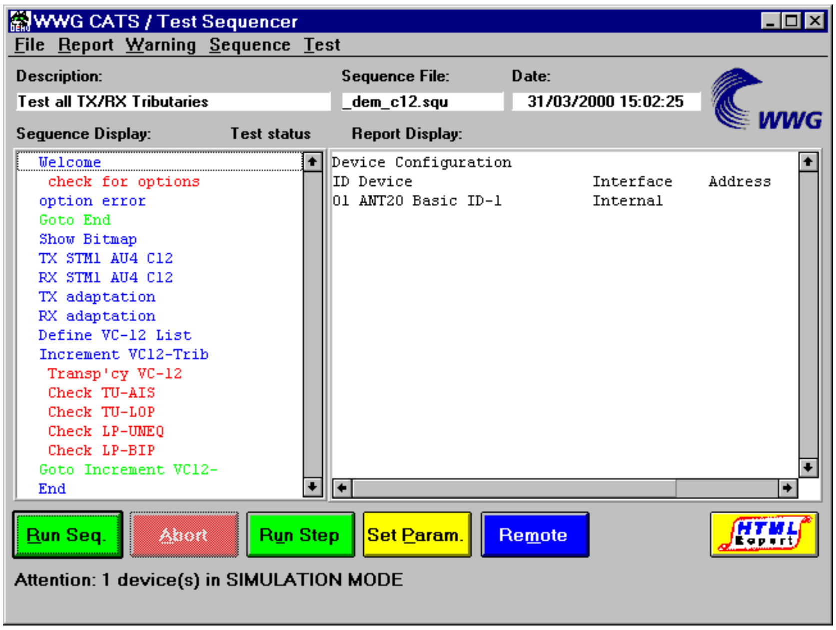


Fig. 3-5 “CATS Test Sequencer” window in “Standard” mode

Note: The following steps can be ignored if you want to run the CATS Test Sequencer in “Standard” mode. “Standard” mode operation is described in Sec. 4.1, Page 1-87.

Switching to “Remote” mode

The CATS Test Sequencer must be switched to “Remote” mode before you can start the client software. In “Remote” mode, the software acts as a server waiting to be called from a client.

Note: If you set the command line option “-dde”, the CATS Test Sequencer will start directly in “Remote” mode.

⇒ Click on the “Remote” button (see Fig. 3-5).
The CATS Test Sequencer switches to “Remote” mode.
The “DDE Remote Interface” window is displayed in the “CATS Test Sequencer” window over the button bar.

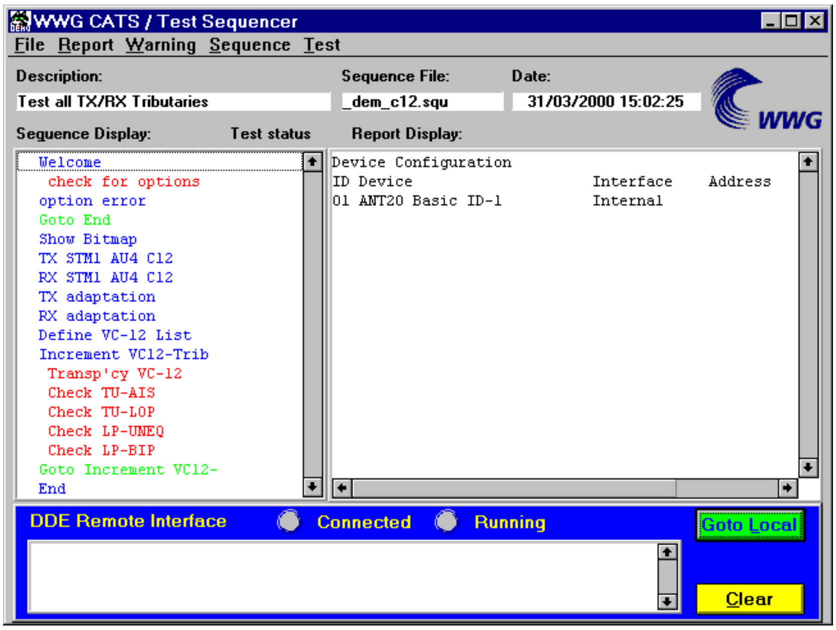


Fig. 3-6 “CATS Test Sequencer” window in “Remote” mode

The “DDE Remote Interface” window displays the commands that are sent from the client software and prevents use of the CATS Test Sequencer button bar in “Remote” mode.

Note: Click the “Goto Local” button to switch the CATS Test Sequencer back to “Standard” mode. The “Goto Local” button is disabled if the connection between the client and the server is active.

Starting the CATS Test Sequencer client software

✓ The CATS Test Sequencer is in “Remote” mode.

⇒ Locate the program group “ANT-20 CATS Test Sequencer” and double-click on the “DDE client” icon.
The “DDE client” window opens.

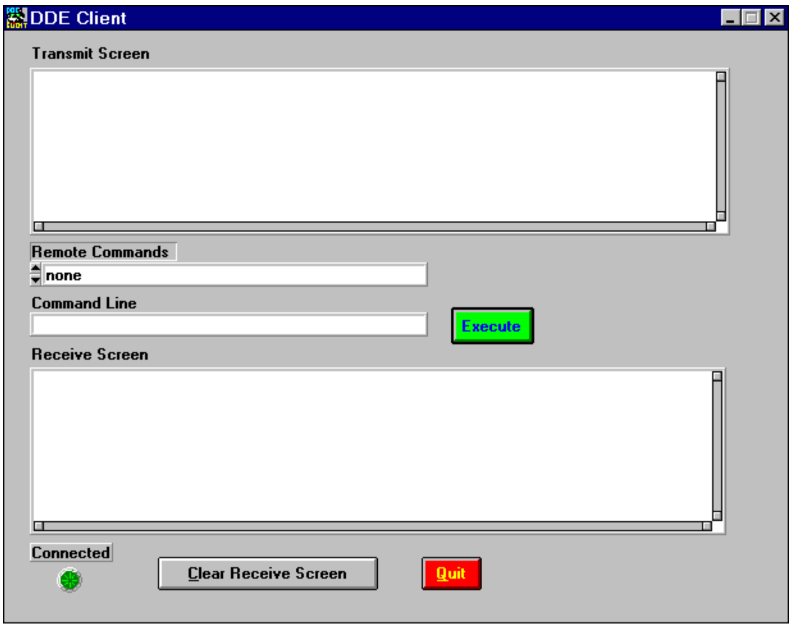


Fig. 3-7 “DDE client” window

The green “Connected” indicator in both windows shows that the ANT-20 CATS Test Sequencer is ready to receive and execute commands.

Sec. 4.2, Page 1-90 describes how to operate the CATS Test Sequencer from the “Client” window.

3.6 Command line parameters

Command line parameters allow adaptation of the CATS Test Sequencer to your specific needs, thereby simplifying operation.

Command line parameter	Effect
-autoconf	Opening the CATS Test Sequencer with “-autoconf” suppresses the Configuration window, i.e. you do not have the opportunity to change the device configuration which was defined by the sequence developer.
-autologin	No “log-in” window will appear. Log-in mode will be set to “developer”. This is the default setting from software version 3.5 onwards.
-batch <sequence>	“-batch” has the same effect as “-autologin -seq ./sequence/xyz.squ -autoconf -run -exit “. The option allows you to create applications (e.g. for installation tests) that can be started simply by double-clicking on an icon. The operator does not need to know anything about how the CATS Test Sequencer operates.
-dde	The DDE server will be started (if DDE version BN 3035/95.92 was purchased).
-exit	Opening the CATS Test Sequencer with “-exit” causes it to close down automatically after it has executed the sequence currently loaded. This option is only valid together with “-run”.
-run	The “-run” option causes the CATS Test Sequencer to automatically start the sequence currently loaded as soon as the Configuration window has been closed. The sequence will be executed once in “Single Pass” mode (see “-exit”).
-seq <sequence>	the CATS Test Sequencer will load the sequence defined by the parameter “sequence”.
-tcp	The TCP/IP server will be started (if TCP/IP version BN 3035/95.91 was purchased).
-login	If this command line parameter is set, the CATS Test Sequencer will prompt you to enter a log-in name and password.

Table 3-1 Command line parameters

3.7 Log-in and password

Previous releases of CATS used a log-in procedure requiring you to enter a user name and a password. **No log-in procedure is required by default with the current software version**. However, a log-in procedure can be set up by means of a command line parameter.

There are two levels of password with different default passwords and access rights:

Password level	Default password	Access rights
Developer	developer	• Defining new sequences or editing existing sequences • The “Run Test” button is provided to let you execute a sequence step by step. This is useful for debugging sequences.
Tester or technician	technician or tech	• **No** sequence editing • Parameter editing
User	no password needed	• **No** sequence or parameter editing • Running sequences (Run Seq.)

Table 3-2 Log-in and password

Setting a log-in procedure

⇒ Set the command line parameter “-login” using the “Properties” command of the “Windows Program Manager”.

4 Getting started – Basic operation

This section tells you how to

- Load test sequences
- Adjust settings
- Start and stop measurements

4.1 CATS (BN 3045.xx), ANT-20 CATS Test Sequencer (Standard mode)

The following information applies to:

- CATS (BN 3045/01)
- CATS DWDM (BN 3045/93.43)
- ANT-20 CATS Test Sequencer (BN 3035/95.90)
- ANT-20 CATS Test Sequencer (BN 3035/95.91, BN 3035/95.92) in “Standard” mode

Operation of the ANT-20 CATS Test Sequencer (BN 3035/95.91, BN 3035/95.92) in “Remote” mode from the “Client” window is described in Sec. 4.2, Page 1-90.

4.1.1 Loading a test sequence

The first time that the CATS Test Sequencer is started, the default test sequence “_demo_c12.squ” opens and is displayed in the “Sequence Display”.

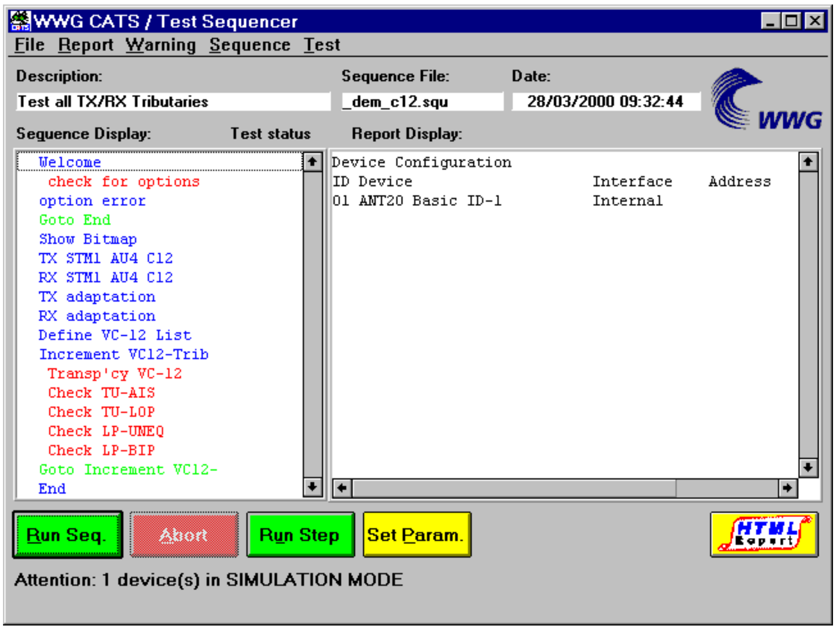


Fig. 4-1 Loaded test sequence displayed in the “Sequence Display” area of the “CATS Test Sequencer” window

Each time the Test Sequencer is started subsequently, the last test sequence saved before switching off will be loaded automatically.

If other test sequences are available, they can be selected as follows:

1. Select “Open” in the “File” menu.
   The “Load Test Sequence” window opens.

2. If necessary, select the directory ".\Sequence".
3. Select the test sequence required and click the [Load] button to load it.
   The "CATS Test Sequencer" window shows the name of the current test sequence in the "Sequence File" box.
   All the test cases that make up the test sequence are listed in the "Sequence Display".

Tip: A list of the last test sequences opened is also displayed below the "Open" command in the "File" menu. Test sequences can also be opened using these menu items.

4.1.2 Setting the test case parameters

Every test case has two main parts:

- A test algorithm for collecting the data
- A user interface for adjusting the test parameters

Editing of the test algorithm is not envisaged, but the measurement parameters can be adjusted very easily with the aid of the user interface. Such adjustments can be saved as specific configurations of the sequence.

Test sequence (editing a sequence) see Sec. 5, Page 1-95.

Adjusting the test parameters

1. Highlight the required test case in the "Sequence Display".
2. Open the test case user interface by clicking the [Set Param.] button or by double clicking the test case.
3. Adjust the settings for each individual test case.

Note: The user interfaces and parameters for all test cases are described in detail in subsequent tab index sections of this documentation.

Saving the settings

If the same or a similar measurement is to be performed several times, it is a good idea to save the settings under a user-defined name.

1. Select "Save as" in the "File" menu.
   The "Save Test Sequence" window opens.
2. Enter the destination directory and a user-defined name.

4.1.3 Starting a test sequence

✓ The required test sequence is activated.

⇒ Click the "Run Seq.“ button.
The test sequence runs according to the conditions set.

4.1.4 Stopping a test sequence

Normally, a test sequence stops automatically according to the specified maximum run time or after a certain number of runs have been performed.

The test sequence can also be stopped manually at any time.

Stopping the test sequence

1. Click the [Abort] button.
2. Confirm that you want to abort the test sequence.
   The test sequence will stop.

4.1.5 Performing a single test

This option allows you to execute a single test from the sequence that is currently loaded.

1. Click on the required test case in the “Sequence Display” area of the “CATS Test Sequencer” window.
   The test case is highlighted in black.
2. Click on the [Run Step] button.
   The selected test case will be executed.

4.1.6 Opening the HTML report

The HTML report window displays the report for the currently loaded sequence in HTML format.

⇒ Click on the [HTML Report] button.

4.2 ANT-20 CATS Test Sequencer (BN 3035/95.91 and .92), Remote mode

The following sections describe remote control of the ANT-20 CATS Test Sequencer using the TCP/IP interface (BN 3035/95.91) and the DDE mechanism (BN3035/95.92). The two packages are basically similar as regards operation and transfer of parameters, etc. These basic principles are described once. Any specific differences between the variants are clearly stated.

4.2.1 Loading a test sequence

✓ Client software has been started (“Client” window is open).
✓ The “Connected” LED is on.

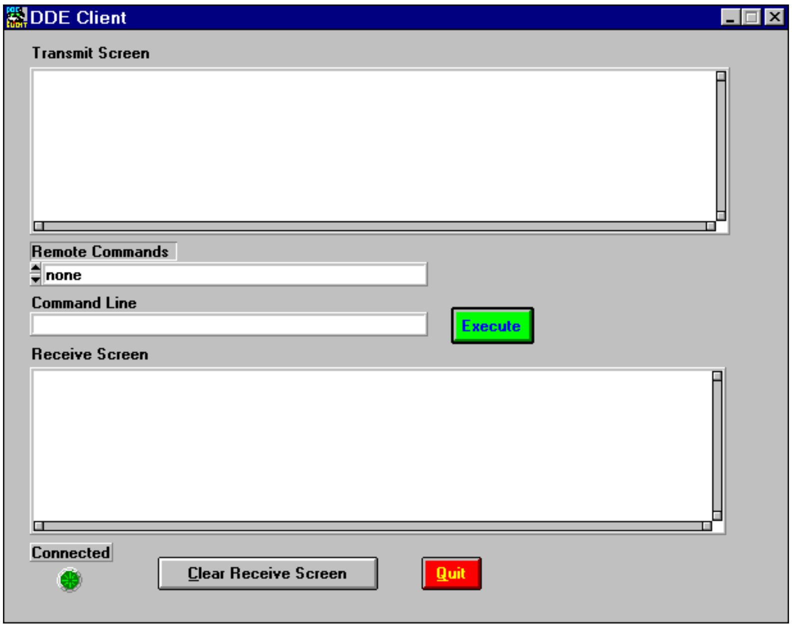


Fig. 4-2 “DDE client” window

1. Use the arrow buttons to select the command “:load_seq:[Drive:Path\FileName.squ]” in the “Remote Commands” box.
   The command line shows the start of the command, i.e. “:load_seq:”

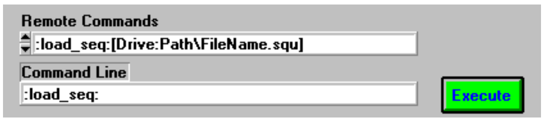


Fig. 4-3 Section of the “DDE client” window

2. Add the drive letter, relative path (i.e. relative to the installation directory) and the name of the sequence to be loaded to the command in the command line.
   The command line will then look something like this:

Fig. 4-4 Section of the “DDE client” window

3. Click on “Execute”.
   The selected sequence will be loaded.

A “Result String” is displayed in the Receive screen.

4.2.2 Setting the test case parameters

✓ The required sequence has been loaded.

1. Select “:set_param:[TestNumber:Parameterstring]” in the “Remote Commands” box using the arrow buttons.
   The command line will show the start of the command, i.e. “:set_Param:”.

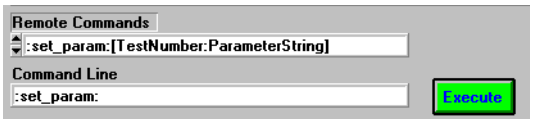


Fig. 4-5 Section of the “DDE client” window

2. Add the test number (position of the test in the sequence) and the “Parameter String” to the command in the command line.
   The command line will then look something like this:

Fig. 4-6 Section of the “DDE client” window

3. Click on “Execute”.
   The parameter will be set.
   A “Result String” is displayed in the Receive screen.

4.2.3 Setting the Report mode of the loaded sequence

The Report mode determines whether the current report overwrites the existing report in the report file or if it is to be appended to the previous report (also see Fig. 6.2.9.3, Page 6-122).

The existing report is overwritten by default.

✓ The required sequence has been loaded.

1. Select “:set_option:report_mode:[0;1]” in the “Remote Commands” box using the arrow buttons.
   The command line will show the start of the command, i.e.“:set_option:report_mode:”.

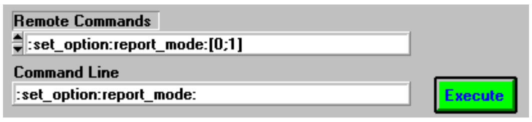


Fig. 4-7 Section of the “DDE client” window

2. Add “0” (i.e “Append”) or “1” (i.e. “Overwrite”) to the command line.
   The command line will then look something like this:

Fig. 4-8 Section of the “DDE client” window

3. Click on “Execute”.

The report mode will be set.
A “Result String” is displayed in the Receive screen.

4.2.4 Defining a report file and setting the file report mode

The messages displayed in the report display are saved in the report file. The reports are saved in a file named “Sequence name.rpt” by default.

The Report mode determines whether the current report overwrites the existing report in the report file or if it is to be appended to the previous report.

1. Select “:set_report_file:[Path\FileMode]” in the “Remote Commands” box using the arrow buttons.
   The command line will show the start of the command, i.e. “:set_report_file:”.

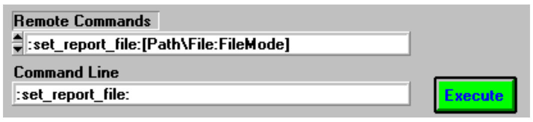


Fig. 4-9 Section of the “DDE client” window

2. Add the path, report file name and report mode to the command line
   (“0” = Overwrite or “1” = Append).
   The command line will then look something like this:

Fig. 4-10 Section of the “DDE client” window

3. Click on “Execute”.
   The Report mode for the report file will be set.
   A “Result String” is displayed in the Receive screen.

4.2.5 Starting a test sequence

✓ The required sequence has been loaded.

1. Select “:single_pass” in the “Remote Commands” box using the arrow buttons.
   The command line will show the (complete) command, i.e. “:single_pass”.

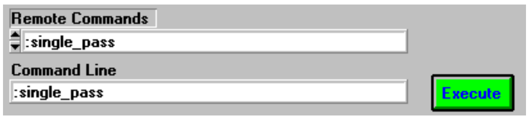


Fig. 4-11 Section of the “DDE client” window

2. Click on “Execute”.
   The sequence will be executed.
   A “Result String” is displayed in the Receive screen when the sequence has ended.

4.2.6 Performing a single test

This option allows you to perform a single test from the currently loaded test sequence.

1. Select ":exe_test:[Test Number in Sequence]" in the "Remote Commands" box using the arrow buttons.
   The command line will show the start of the command, i.e. ":exe_test:".

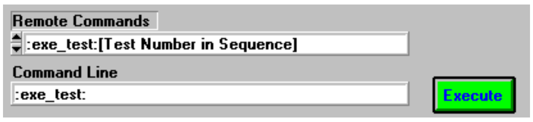


Fig. 4-12 Section of the "DDE client" window

2. Add the test number (position of the test in the sequence).
   The command line will then look something like this:

Fig. 4-13 Section of the "DDE client" window

3. Click on "Execute".
   The selected test will be executed.
   A "Result String" is displayed in the Receive screen when the test has ended.

Notes:

5 Editing test sequences

This section tells you how to

- modify existing test sequences
- create new test sequences

5.1 Sequence editing

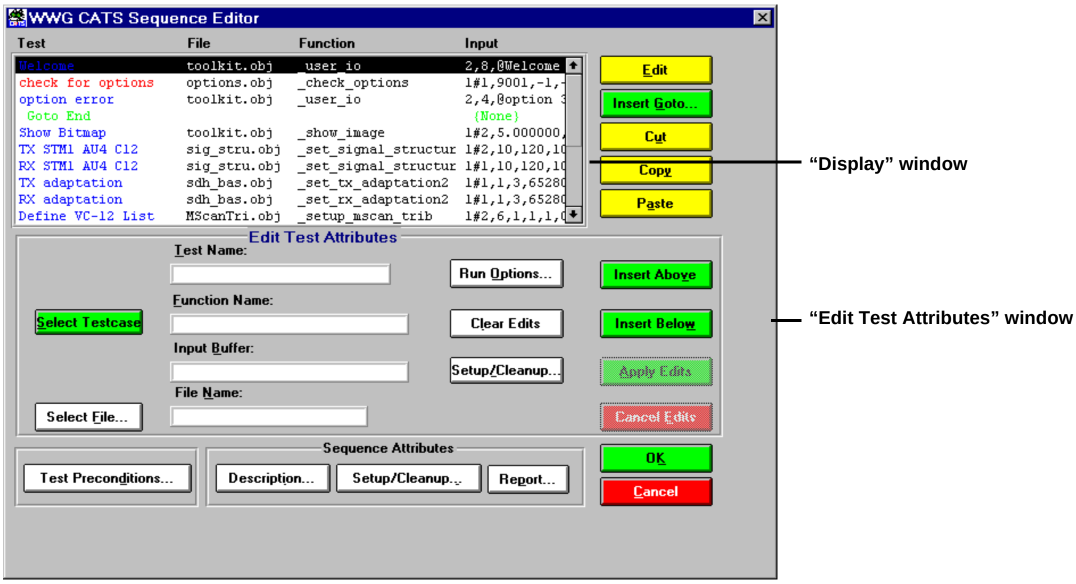


Fig. 5-1 “Sequence Editor” window

Note: All parts of the “Sequence Editor” window and their functions are described in detail in the function overview (see Sec. 6, Page 1-101).

5.1.1 Opening the Sequence Editor

✓ The “CATS Test Sequencer” window is open.

⇒ Select “Edit Sequence” in the “Sequence” menu.
The “Sequence Editor” window opens.
The loaded sequence is displayed in the same order as in the “Sequence” display of the CATS Test Sequencer window.

Note: You can display an overview of all the CATS Test Sequencer functions by clicking the [Select Testcase] button in the “Sequence Editor” window (see Sec. 6.2.7.1, Page 1-114).

5.1.2 Inserting a new test case into a sequence

Note: The Cut, Copy and Paste functions may be used to simplify building your own sequences.
Each test case in a sequence must have a unique name.
If you click the [Cancel] button in the Sequence Editor, any changes you made will be lost and you will return to the “CATS Test Sequencer” window.

There are three ways to insert a new test case into a sequence.

Selecting the test case from a list

✓ The “Sequence Editor” window is open.

1. Mark the position where the new test case is to be inserted in the “Display” window.
2. Click the [Select Testcase] button.
   The “Testcase List” window opens.

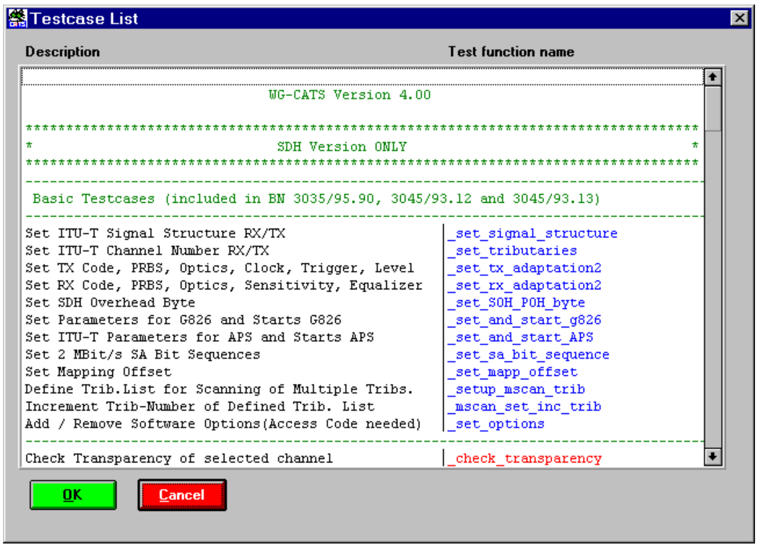


Fig. 5-2 “Testcase List” window

3. Mark the test case required in the list of all the available test cases.
4. Click the [Insert Above] or [Insert Below] button in the “Sequence Editor” window.
   The new test case is shown in the “Display” window.
5. Click [OK] to confirm.
   The “Sequence Editor” window closes and the “CATS Test Sequencer” window is active.
6. Save the changes you made in the test sequence using the “File” / “Save” or “File” / “Save as” menu commands.

Entering the name of the required test case

✓ The “Sequence Editor” window is open.

1. Enter the user-defined test case name, function name and file name in the “Edit Test Attribute” window.
2. Click the [Run options ...] button.
   The “Test Run Options” window opens.
3. Define the Test Run Options and click the [OK] button to confirm.
4. Mark the position where the new test case is to be inserted in the “Display” window.
5. Click the [Insert Above] or [Insert Below] button.
   The new test case is shown in the “Display” window.

6. Click [OK] to confirm.
   The “Sequence Editor” window closes and the “CATS Test Sequencer” window is active.
7. Save the changes you made in the test sequence using the “File” / “Save” or “File” / “Save as” menu commands.

Entering a new test case in edit mode

✓ The “Sequence Editor” window is open.

1. Mark the position where the new test case is to be inserted in the “Display” window.
2. Click the [Insert Above] or [Insert Below] button.
   A new line is inserted in the “Display” window.
3. Mark the empty line.
4. Click the [Edit] button.
   The “Edit Test Attribute” window is active. All other parts of the “Sequence Editor” window are disabled.
5. Fill in the user-defined test case name, the function name and the file name.
6. Click the [Run options ...] button.
   The “Test Run Options” window opens.
7. Define the Test Run Options and click the [OK] button to confirm.
8. Press the [Apply Edits] button.
   The entries will be applied.
   The new test case is shown in the “Display” window.
9. Click [OK] to confirm.
   The “Sequence Editor” window closes and the “CATS Test Sequencer” window is active.
10. Save the changes you made in the test sequence using the “File” / “Save” or “File” / “Save as” menu commands.

5.1.3 Creating a new test sequence

Keep the following points in mind if you create sequences using the “Sequence Editor” window.

- A sequence is a collection of data that describes the flow of test execution.
- The main component of a sequence is a test case.
- A test case is a single execution step in the testing process.
- A test case executes a function or sub-sequence that performs the required test operation.

5.1.3.1 Saving an existing test sequence under a new name

Saving a test sequence

✓ The “CATS Test Sequencer” window is open.

1. Select “Open” from the “File” menu.
2. Select the test sequence that is to be used as the basis for the new test sequence.
3. Click [OK] to confirm.
   The selected test sequence is shown in the “Sequence Display”.
4. Save the opened sequence under a new name using the “File” / “Save as” menu command.
5. Start the “Sequence Editor” window using the “Sequence” / “Edit Sequence” menu command.

Deleting test cases from a sequence

1. Highlight the test case that is to be deleted from the test sequence.
2. Click on the [Cut] button.
   The highlighted test case is deleted from the test sequence.

Adding test cases to a test sequence

Inserting a new test case into a sequence: See chapter 5.1.2, page 5-79.

5.1.4 Editing a test case entry

Note: If you click the [Cancel] button in the Sequence Editor, any changes you made will be lost and you will return to the “CATS Test Sequencer” window.

✓ The “Sequence Editor” window is open.

1. Highlight the test case that is to be edited.
2. Click the [Edit] button.
   The “Edit Test Attribute” window is activated. All other parts of the “Sequence Editor” window are disabled.
3. Make changes if required.
4. Click the [Apply Edits] button.
   The edited parameters will be set.
5. Click [OK] to confirm.
   The “Sequence Editor” window closes and the “CATS Test Sequencer” window is activated.
6. Save the changes you made in the test sequence using the “File” / “Save” or “File” / “Save as” menu commands.

5.1.5 Saving a test sequence

The test sequence must be saved under a user-defined name when all the changes have been completed.

1. Select “Save as” from the “File” menu.
   The “Save Test Sequence” window opens.
2. Enter the destination directory and user-defined name.
3. Click “OK“ to confirm.

5.2 Device configuration

✓ The “Test Sequencer” window is open.

1. Select the “Sequence”, “Device Configuration - Interfaces, Adr.” menu.
   The “Device Configuration” window opens.
   The sequence last used is stored in the file “lastseq.seq”. Leave the parameters as they are for the first attempt.

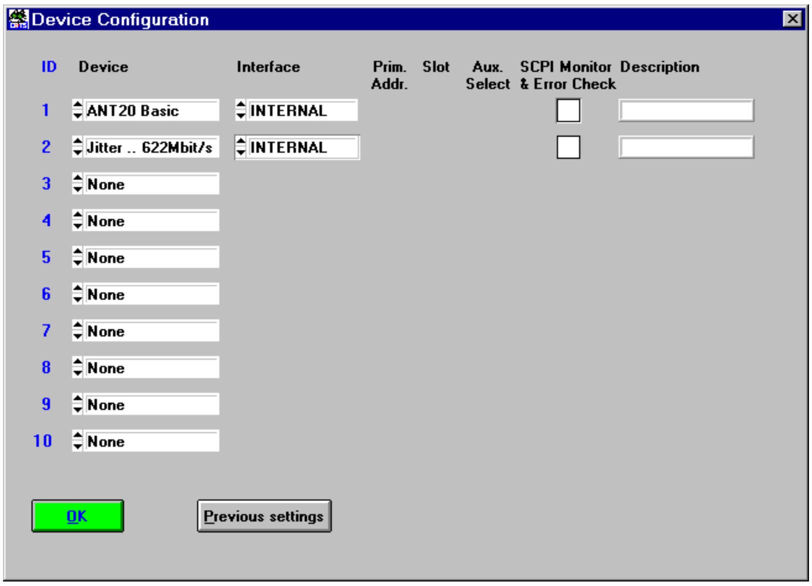


Fig. 5-3 “Device Configuration” window

2. Click [OK] to open the devices requested.
   If the open procedure fails, a message will appear indicating that you may start a simulation.

Note: The appearance of the query “Start Device Simulation?” must be considered as an error when you are making “real” measurements. The message should only appear when a demonstration is performed with no measurement hardware connected to the controller PC.

CATS (BN 3035/x.x) only

If the query “Start Device Simulation ANT-20?” appears, proceed as follows:

1. Click on the “Yes” button.
2. Close down the ANT-20 CATS Test Sequencer.
3. Restart the ANT-20 CATS Test Sequencer.
4. Make sure that the “INTERNAL” interface is selected for each device.
5. If device configuration fails again, please contact your local WWG sales organisation.

Notes:

6 GUI function overview

The windows, controls and functions of all variants of the CATS Test Sequencer software are described in this section.

6.1 CATS Test Sequencer window

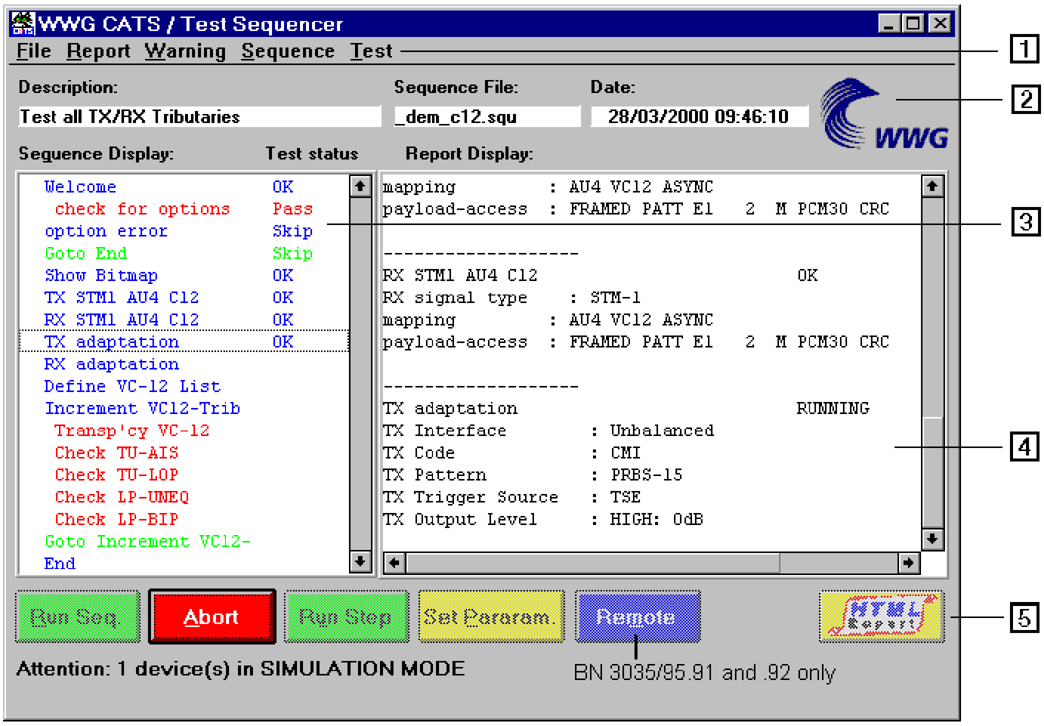


Fig. 6-1 “CATS Test Sequencer” window

[1]	Menu bar • File, see Sec. 6.1.1, Page 1-102 • Report, see Sec. 6.1.2, Page 1-104 • Warning, see Sec. 6.1.3, Page 1-105 • Sequence, see Sec. 6.2, Page 1-111 • Test, see Sec. 6.1.4, Page 1-106
[2]	Information line • Description, see Sec. 6.1.5, Page 1-106 • Sequence File, see Sec. 6.1.6, Page 1-106 • Date, see Sec. 6.1.7, Page 1-106
[3]	Sequence Display, see Sec. 6.1.8, Page 1-107
[4]	Report Display, see Sec. 6.1.9, Page 1-107
[5]	Buttons, see Sec. 6.1.10, Page 1-108 to Sec. 6.1.13, Page 1-108

6.1.1 File

The File menu provides functions for viewing the log-in level, for loading and saving sequences and for exiting the CATS Test Sequencer.

6.1.1.1 Login

Menu: File - Login

Fig. 6-2 “Login” window

Meaning: Enter the login name and password.
The login name can be used to assign access rights. The password allows authorized users access.

6.1.1.2 New

Menu: File - New

Meaning: Open new test sequence.
The new test sequence is empty and must be filled in completely.
It is usually more practical to modify an existing test sequence.

6.1.1.3 Open

Menu: File - Open

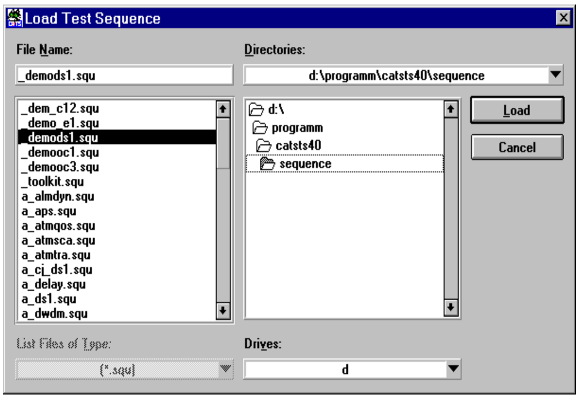


Fig. 6-3 “Load Test Sequence” window

Meaning: Open an existing test sequence.
The file name extension *.squ is set as default.

6.1.1.4 Save

Menu: File - Save

Meaning: Saves the current settings for the test sequence under the existing name.

6.1.1.5 Save as

Menu: File - Save as

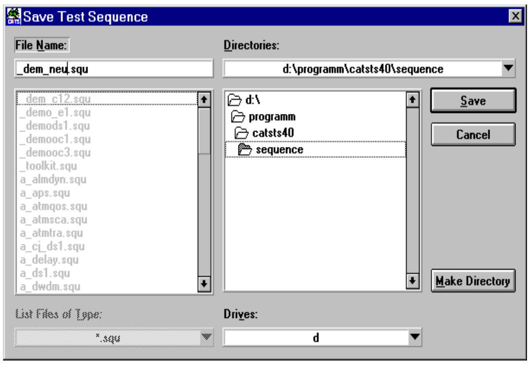


Fig. 6-4 “Save Test Sequence” window

Meaning: Saves the current settings for the test sequence under a different name with file name extension *.squ.

6.1.1.6 About

Menu: File - About

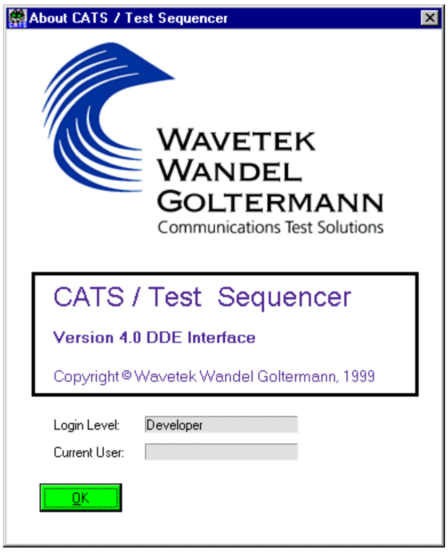


Fig. 6-5 “About CATS / Test Sequencer” window

Meaning: Displays the current version of the CATS Test Sequencer software and the current log-in level.

6.1.1.7 Exit

Menu: File - Exit

Meaning: Exits from the CATS Test Sequencer.

6.1.2 Report

The Report menu provides functions for changing the report mode and for viewing, printing out and clearing the report display.

6.1.2.1 Mode

Menu: Report - Mode

Meaning: The report file always contains all the information about the test sequence. Selecting a report mode determines the amount of detail shown in the test results in the “Report Display”.
See Sec. 6.1.9, Page 1-107.

Display All	All test results are displayed and stored in full detail.
Failures Only	Only failed test cases are displayed in detail, passed test cases are shown with the “PASS” indication only.
Condensed	Only “PASS” or “FAIL” indication is shown for each test case.

6.1.2.2 View

Menu: Report - View

Meaning: “Report Display” is updated and jumps back to the start of the report.
See Sec. 6.1.9, Page 1-107.

6.1.2.3 View HTML

Menu: Report - View HTML

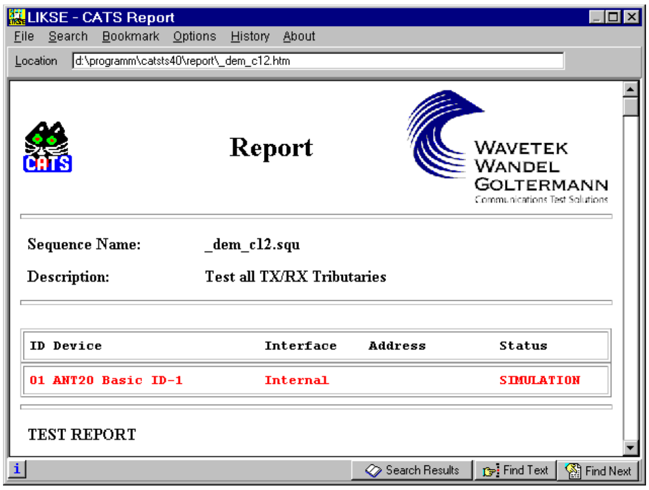


Fig. 6-6 “Report” window

Meaning: Display report in HTML format.

6.1.2.4 Print

Menu: Report - Print

Meaning: Print “Report Display” as ASCII file.
See Sec. 6.1.9, Page 1-107.

6.1.2.5 Clear

Menu: Report - Clear

Meaning: Clear test results from “Report Display”.
See Sec. 6.1.9, Page 1-107.

6.1.3 Warning

Menu: Report - Warning

Meaning: Switch specifying how the platform software should deal with warnings occurring during test execution.

Suppress	Warning is not displayed.
Write_to_Report	Warning is written into the report.
Prompt_Operator	Warning is signaled by a pop-up window that must be acknowledged. Default setting.

6.1.4 Test

Menu: Report - Test

Meaning: Switch forcing a specific test status for the selected test.

Normal	Test case generates “Pass”, “Fail” and “Abort” statements.
Forced to Pass	Test case only generates “Pass” statements.
Forced to Fail	Test case only generates “Fail” statements.
Forced to Skip	Test case is skipped when the sequence is run.

6.1.5 Description

Fig. 6-7 “Description:” display area of the “CATS Test Sequencer” window

Meaning: User defined description of test sequence.
Defined in the “Test Description” window.
See Sec. 6.2.9.1, Page 1-120.

6.1.6 Sequence File

Fig. 6-8 “Sequence File:” display area of the “CATS Test Sequencer” window

Meaning: Indicates the name of the currently loaded test sequence.
See Sec. 6.1.1.3, Page 1-102.

6.1.7 Date

Fig. 6-9 “Date:” display area of the “CATS Test Sequencer” window

Meaning: Displays current date and exact time to the nearest second.
The contents of this display appear as a timestamp in the designation for the submenu where the measurement data are stored.

6.1.8 Sequence Display

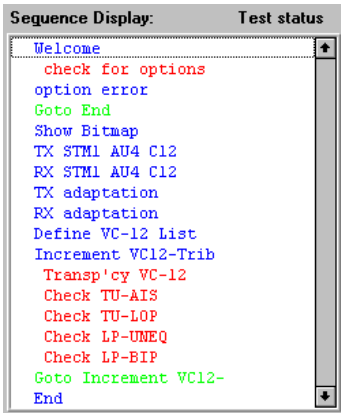


Fig. 6-10 “Sequence Display:” area of the “CATS Test Sequencer” window

Meaning: The Sequence Display lists all the test cases for the current test sequence and their status after the last run of the sequence. Double clicking on a test case or using the [Set Param.] button opens the windows where the test case parameters for the measurement are set.

Note: Test cases are usually shown in the test sequence under user-defined names. The “Sequence Editor” window gives information about the functions represented by the names.
See Sec. 6.2, Page 1-111.

6.1.9 Report Display

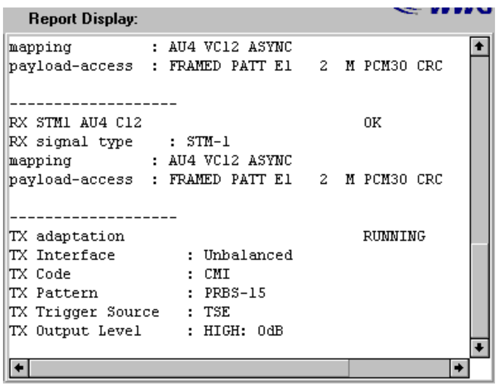


Fig. 6-11 “Report Display:” area of the “CATS Test Sequencer” window

Meaning: Shows the results of the last test.

Note: The test protocol is stored as an ASCII file. The name may be modified (or looked up) using the sequence editor “Report” selection (see Sec. 6.2.9.3, Page 1-122).
The default directory for reports is “.\REPORT*.RPT”. HTML files are saved in the same directory with file name extension *.HTML.

6.1.10 Run Seq. button

Fig. 6-12 [Run Seq] button in the “CATS Test Sequencer” window

Meaning: Starts the test sequence.

6.1.11 Abort button

Fig. 6-13 [Abort] button in the “CATS Test Sequencer” window

Meaning: Cancels the test sequence.

6.1.12 Run Step button

Fig. 6-14 [Run Step] button in the “CATS Test Sequencer” window

Meaning: Starts an individual test case for test purposes.

6.1.13 Set Param. button

Fig. 6-15 [Set Param.] button in the “CATS Test Sequencer” window

Meaning: Opens the window for setting the parameters of the selected test case.

6.1.14 Remote button (BN 3035/95.91 and .92 only)

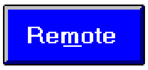


Fig. 6-16 [Remote] button in the “CATS Test Sequencer” window

Meaning: Switches the CATS Test Sequencer to “Remote” mode.
The “Remote Interface” window is displayed over the button bar in the “CATS Test Sequencer” window.

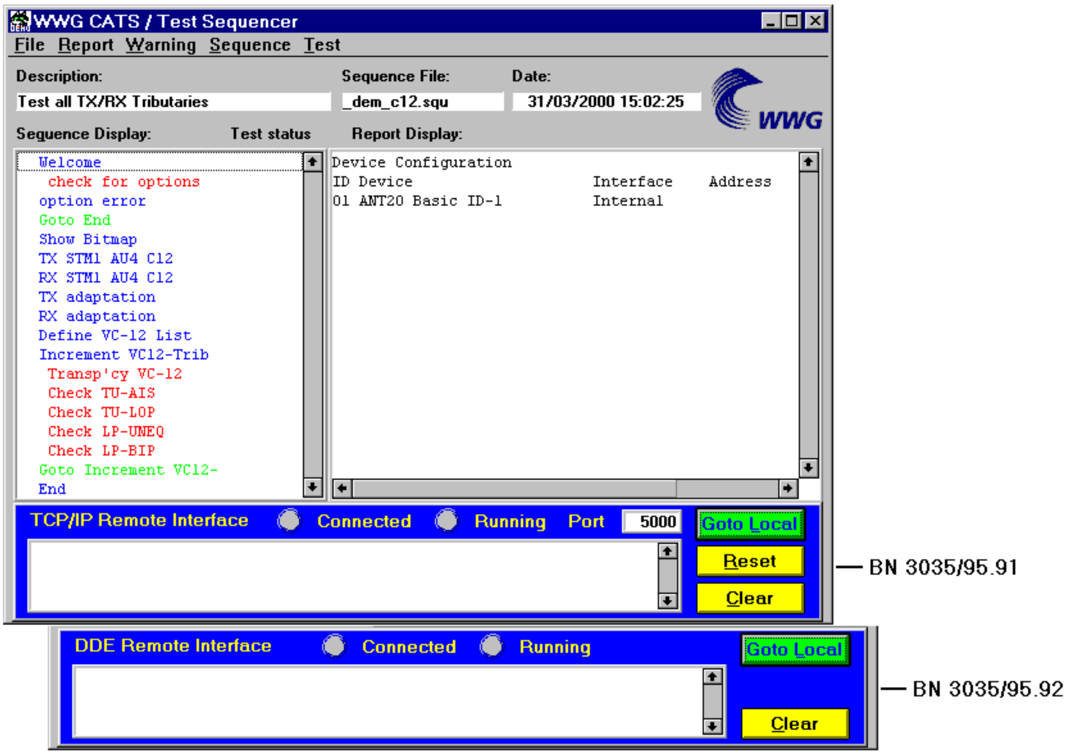


Fig. 6-17 “CATS Test Sequencer” window with “Remote Interface” window displayed over button bar

6.1.14.1 Connected LED

Meaning: Indicates when there is a connection between the CATS Test Sequencer (server) software and the client software.

6.1.14.2 Running LED

Meaning: Indicates when an action is being performed by the CATS Test Sequencer (server) software (e.g. loading a sequence, executing a sequence).

6.1.14.3 Port display box (BN 3035/95.91 only)

Meaning: Displays the port number of the CATS Test Sequencer (server) software.

6.1.14.4 Transmit screen

Fig. 6-18 “Transmit screen” area of the “Remote Interface” window

Meaning: Indicates the commands transmitted by the client software.

6.1.14.5 Goto Local button

Fig. 6-19 [Goto Local] button in the “Remote Interface” window

Note: The [Goto Local] button is deactivated when the connection between the Test Sequencer (server) software and the client software is active.

Meaning: Switches the CATS Test Sequencer to “Standard” mode.

6.1.14.6 Reset button (BN 3035/95.91 only)

Fig. 6-20 [Reset] button in the “(TCP) Remote Interface” window

Meaning: Terminates the TCP/IP connection.

6.1.14.7 Clear button

Fig. 6-21 [Clear] button in the “Remote Interface” window

Meaning: Clears the contents of the Transmit screen.

6.2 Sequence Editor window

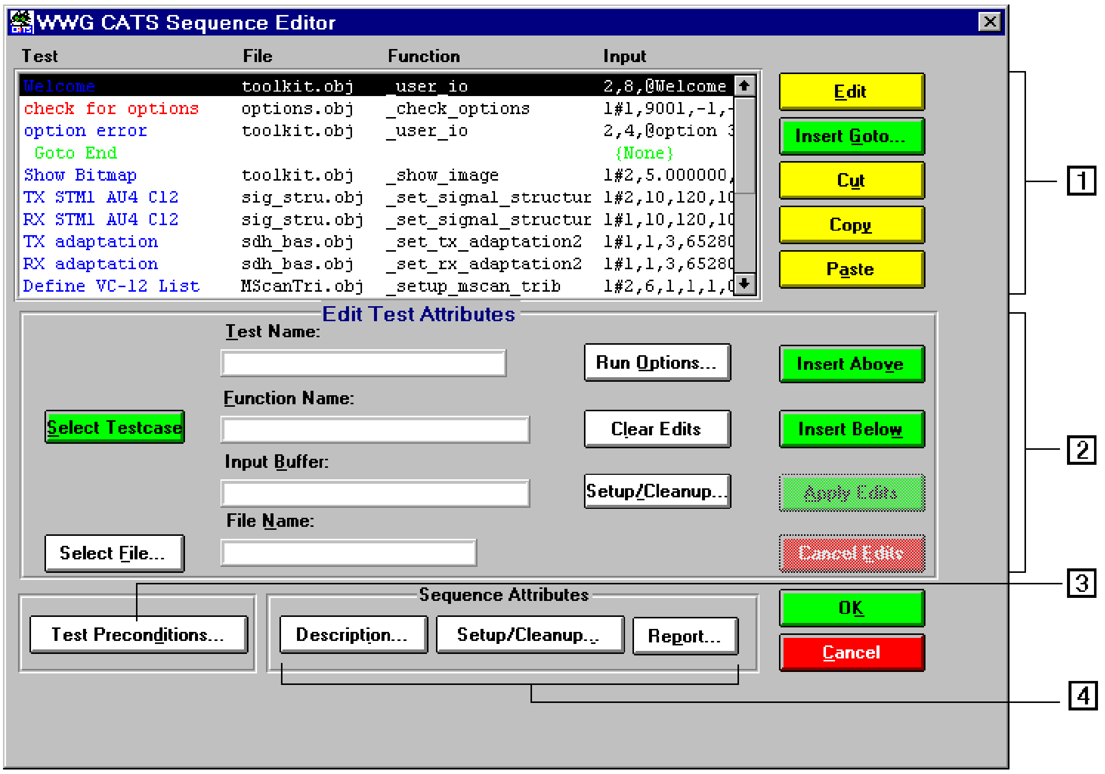


Fig. 6-22 “Sequence Editor” window

[1]	Sequence content • “Display” window, see Sec. 6.2.1, Page 1-111 • Buttons, see Sec. 6.2.2, Page 1-112 to Sec. 6.2.6, Page 1-113
[2]	Edit Test Attributes, see Sec. 6.2.7, Page 1-113
[3]	Test Preconditions, see Sec. 6.2.8, Page 1-118
[4]	Sequence Attributes, see Sec. 6.2.9, Page 1-120

6.2.1 Display window

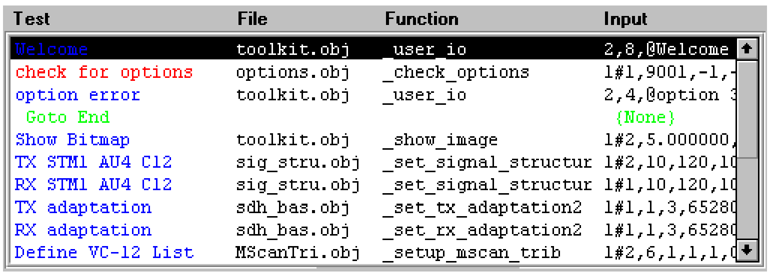


Fig. 6-23 “Display window” area of the “Sequence Editor” window

Meaning: Overview of test sequence contents.

Test	User-defined name for the test case in the test sequence
File	Name of file containing the test case
Function	Function name of predefined test case
Input	Test case parameters

6.2.2 Edit button

Fig. 6-24 [Edit] button in the “Sequence Editor” window

Meaning: The attributes of the test case selected in the “Display” window are transferred to the “Edit Test Attribute” window and can be edited there.
See Sec. 6.2.7, Page 1-113.

6.2.3 Insert Goto button

Fig. 6-25 [Insert Goto] button in the “Sequence Editor” window

Meaning: Inserts a Goto command above or below the test case selected in the “Display” window or replaces the selected test case by the Goto command.
Opens the “Insert Goto” window.

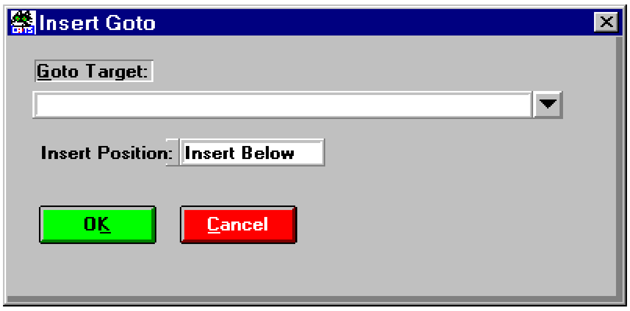


Fig. 6-26 “Insert Goto” window

Goto Target	Enter the target for the Goto command
Insert Position	Select position of Goto command • Replace: Replace the selected test case • Above: Insert above selected test case. • Below: Insert below selected test case.

6.2.4 Cut button

Fig. 6-27 [Cut] button in the “Sequence Editor” window

Meaning: Copies the highlighted test case in the “Display” window to the clipboard and deletes the test case from the sequence.

6.2.5 Copy button

Fig. 6-28 [Copy] button in the “Sequence Editor” window

Meaning: Copies the highlighted test case in the “Display” window to the clipboard without deleting the test case from the sequence.

6.2.6 Paste button

Fig. 6-29 [Paste] button in the “Sequence Editor” window

Meaning: Pastes the test case from the clipboard into the “Display” window. The test case is inserted below the test case that is currently selected.

6.2.7 Edit Test Attributes

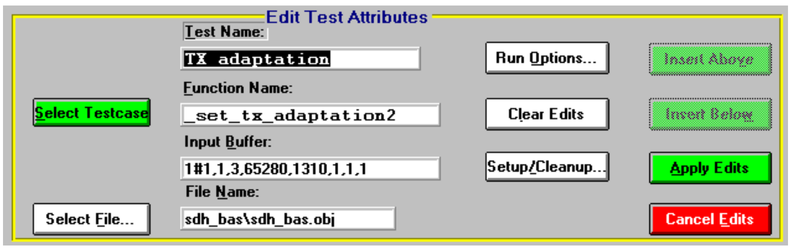


Fig. 6-30 “Edit Test Attributes” area of the “Sequence Editor” window

Meaning: Edit attributes of the selected test case.

Note: Each test case in a given sequence must have a unique name.

Test Name	User-defined test case name. Type any ASCII string in the Test Name dialog box. This name should describe the test case function, as it is the name used in the Report Display of the “CATS Test Sequencer” window.
Function Name	Type the name of the C function into the Function Name dialog box. This name is the C function of the corresponding C object file.
Input Buffer	The Input Buffer dialog box displays a string that is added into the test case function. The content and meaning of the string depends on the test case function.
File Name	Type the relative path name of the file that contains the test case. If you use a relative path, it is defined relative to the Test Executive project or executable, not your sequence file.

6.2.7.1 Select Testcase button

Fig. 6-31 [Select Testcase] button in the “Sequence Editor” window

Meaning: Opens the “Testcase List” window showing all available test cases. All the test cases listed in the file “textlist.txt” are shown in the “Testcase List” window.

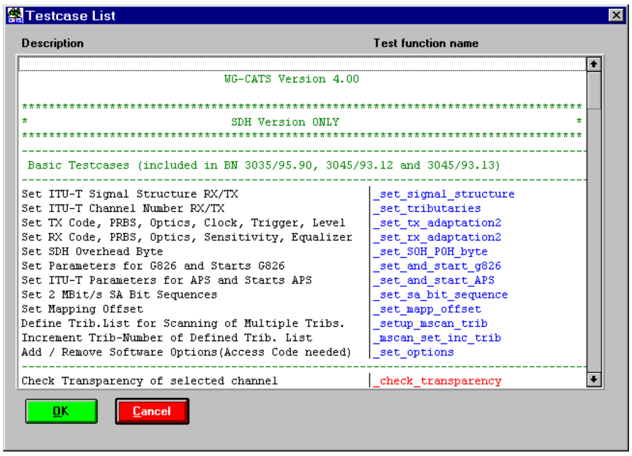


Fig. 6-32 “Testcase List” window

Meaning: Lists all available CATS Test Sequencer test cases. A test case can be selected and its test attributes will appear in the Test Attributes dialog boxes.

6.2.7.2 Select File ... button

Fig. 6-33 [Select File] button in the “Sequence Editor” window

Meaning: Opens the “Select Object File” window.

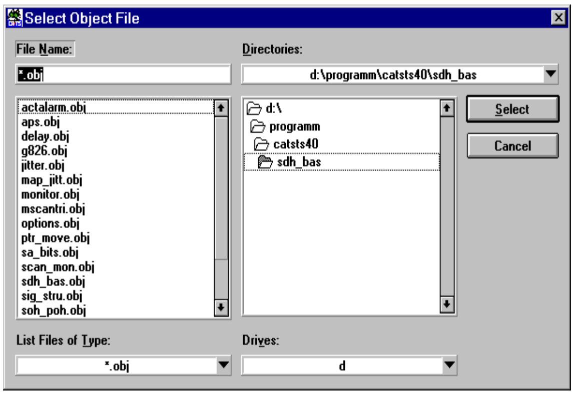


Fig. 6-34 “Select Object File” window

Meaning: Select *.obj file in which the test sequence is to be saved.

6.2.7.3 Run Options button

Fig. 6-35 [Run Options] button in the “Sequence Editor” window

Meaning: Specify the Run mode.
Opens the “Test Run Options” window.

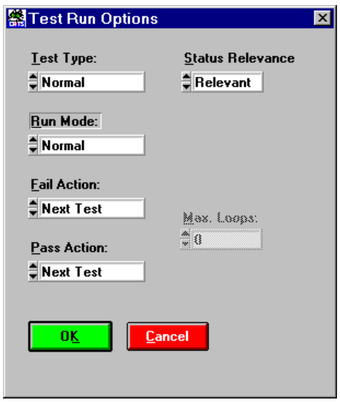


Fig. 6-36 “Test Run Option” window

Test Type	• Normal • Loop Control
Run Mode	• Normal • Skip • Force to Fail • Force to Pass
Fail Action	• Next Test • Loop • Stop
Pass Action	• Next Test • Loop • Stop
Status relevance	• Relevant: Indicator color red Pass and Fail are evaluated • Don‘t care: Indicator color blue Pass and Fail not evaluated
Max. Loops	• Enter maximum number loops

6.2.7.4 Clear Edits button

Fig. 6-37 [Clear Edits] button in the “Sequence Editor” window

Meaning: Clears the changes made in the Test Attributes.

Notice: If you have selected a test case and you press the [Clear Edits] button and then the [Apply Edits] button, the selected test case will be deleted.

6.2.7.5 Setup/Cleanup button

Fig. 6-38 [Setup/Cleanup] button in the “Sequence Editor” window

Meaning: Opens the “Test Setup/Cleanup Routines” window for an individual test case. These settings do not normally have to be changed.

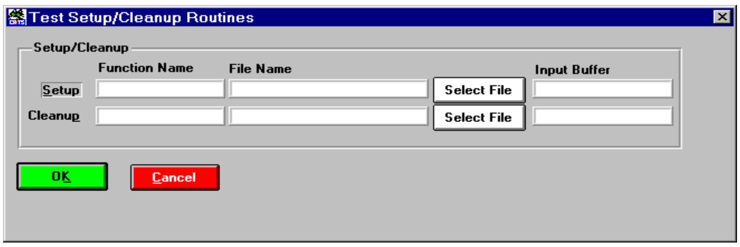


Fig. 6-39 “Test Setup/Cleanup Routines” window

Setup	A setup function is executed before a test case. • Function Name Enter the name of the setup function. • File Name Enter the name of the file that contains the function, or click on the [Select File] button to open the “File dialog” window. • Buffer
Cleanup	A cleanup function is executed after a test case. • Function Name Enter the name of the cleanup function. • File Name: Enter the name of the file that contains the function, or click on the [Select File] button to open the “File dialog” window. • Buffer

6.2.7.6 Insert Above button

Fig. 6-40 [Insert Above] button in the “Sequence Editor” window

Meaning: Inserts an edited test case into the list of test cases in the “Display” window above the selected test case.
See Sec. 6.2.1, Page 1-111.

Note: The [Insert Above] button is only available if you have selected a test case from the Select TC List or if the test case was created by inserting the attributes into the Test Attributes. If you have selected the test case with a double click in the “Display” window or by pressing the [Edit] button, the [Insert Above] button is grayed out.

6.2.7.7 Insert Below button

Fig. 6-41 [Insert Below] button in the “Sequence Editor” window

Meaning: Inserts an edited test case into the list of test cases in the “Display” window below the selected test case; see Sec. 6.2.1, Page 1-111.

Note: The [Insert Below] button is only available if you have selected a test case from the Select TC List or if the test case was created by inserting the attributes into the Test Attributes. If you have selected the test case with a double click in the “Display” window or by pressing the [Edit] button, the [Insert Below] button is grayed out.

6.2.7.8 Apply Edits buttons

Fig. 6-42 [Apply Edits] button in the “Sequence Editor” window

Meaning: Applies the edits to an edited test case.

6.2.7.9 Cancel Edits button

Fig. 6-43 [Cancel Edits] button in the “Sequence Editor” window

Meaning: Exits edit mode for the selected test case without applying any edits.

6.2.8 Test Preconditions button

Fig. 6-44 [Test Preconditions] button in the “Sequence Editor” window

Meaning: Opens the “Precondition Editor” window.

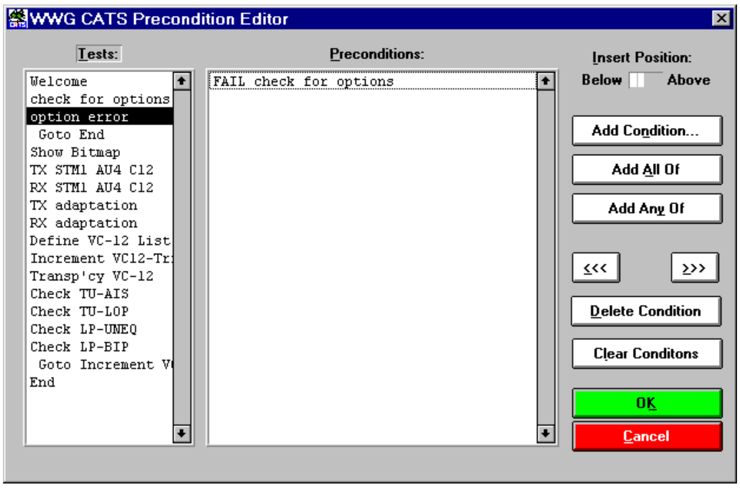


Fig. 6-45 “Preconditions Editor” window

Meaning: The preconditions for a test case specify which other tests must PASS or FAIL before this particular test case is executed.

Tests	The test case names appear in the “Test list” window.
Preconditions	The “Preconditions Editor” window shows the precondition tests. These are tests on which execution of the test case you select in the “Test list” window depends. You can make various settings to control the flow of the sequence here.

6.2.8.1 Insert Position Switch

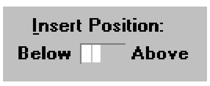


Fig. 6-46 [Insert Position] switch in the “Precondition Editor” window

Meaning: The switch determines whether new preconditions are inserted before or after the current precondition.

6.2.8.2 Add Conditions ... button

Fig. 6-47 [Add Condition ...] button in the “Precondition Editor” window

Meaning: Opens the “Add Condition” window.

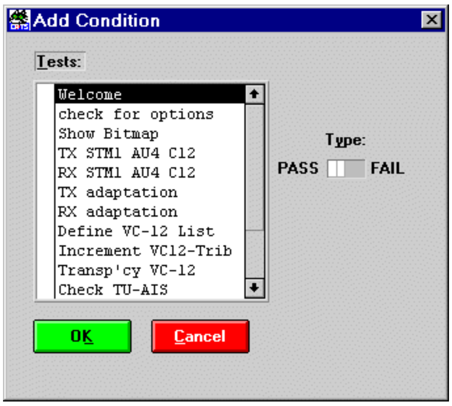


Fig. 6-48 “Add Condition” window

Tests	Shows the precondition test cases that are available.
Type Switch	The setting of the Type switch (PASS or FAIL) determines whether the selected precondition test case must PASS or FAIL.

6.2.8.3 Add All Of button

Fig. 6-49 [Add All Of] button, part of the “Precondition Editor” window

Meaning: Starts a block of preconditions which all have to be true.

6.2.8.4 Add Any Of button

Fig. 6-50 [Add Any Of] button in the “Precondition Editor” window

Meaning: Starts a block of preconditions, any of which must be true.

6.2.8.5 Move to the Left and Move to the Right buttons

Fig. 6-51 [Move to the Left] and [Move to the Right] buttons

Meaning: Adjusts the position of the selected precondition.

6.2.8.6 Delete Condition button

Fig. 6-52 [Delete Condition] button in the “Precondition Editor” window

Meaning: Deletes the selected precondition.

6.2.8.7 Clear Conditions button

Fig. 6-53 [Clear Conditions] button in the “Precondition Editor” window

Meaning: Clears all the preconditions for the test case selected in the “Tests” list box.

6.2.9 Sequence Attributes

Meaning: Global settings for the test sequence can be made using the [Description ...], [Setup/Cleanup ...] and [Report ...] buttons.

6.2.9.1 Description button

Fig. 6-54 [Description] button in the “Sequence Editor” window

Meaning: Opens the “Test Description” window.

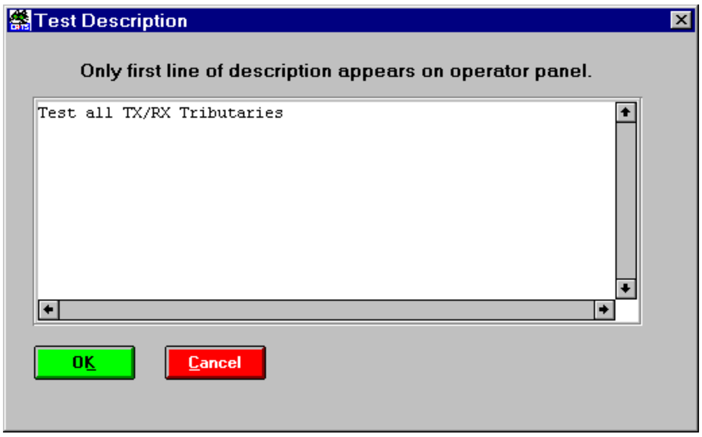


Fig. 6-55 “Test Description” window

Meaning: Insert a sequence description or edit the description of an existing sequence. The first line of this description will appear in the description text box in the CATS Test Sequencer window.

6.2.9.2 Setup/Cleanup button

Fig. 6-56 [Setup/Cleanup] button in the “Sequence Editor” window

Meaning: Opens the “Sequence Setup/Cleanup Routines” window.

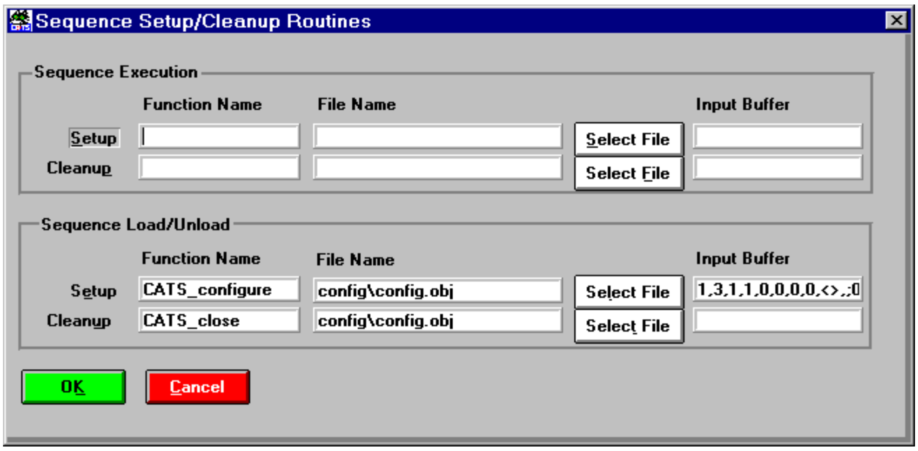


Fig. 6-57 “Sequence Setup/Cleanup Routines” window

Meaning: In the CATS Test Sequencer environment, Setup/Cleanup is used to open and close all connected devices when starting or finishing a sequence. For normal operation these settings should never be changed.
When starting a sequence from scratch with the “File - New” menu, make sure that you enter the Setup/Cleanup functions that are used with sequences prepared by WWG.

Setup function	_CATS_configure
Cleanup function	_CATS_close

6.2.9.3 Report button

Fig. 6-58 [Report] button in the “Sequence Editor” window

Meaning: Opens the “Set Default Report File” window.

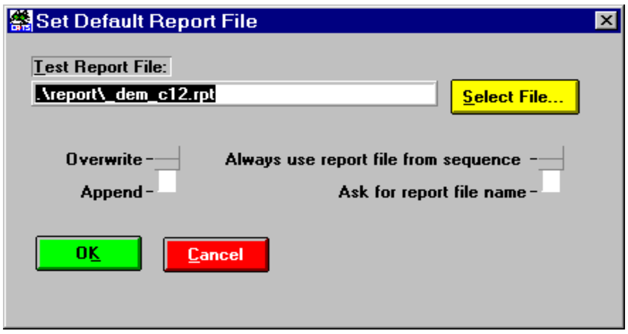


Fig. 6-59 “Set Default Report File” window

Test Report File	Select a report file. The default directory for reports is .\REPORT. The default file name extension is *.rpt.
Switch Overwrite - Append	• Overwrite: The current report always overwrites the previous report. • Append: The current report is appended to the previous report.
Switch Always use ... - Ask for Report file name	• Always use report files from sequence: The default file name for the report file is the name of the sequence file. The report file is overwritten automatically when the test sequence is closed and then re-opened. • Ask for report file name: A name for the report file is requested by default whenever a test sequence is opened.

6.3 Device Configuration window

Opened using the “Sequence>Device Configuration - Interface Adr.” command.

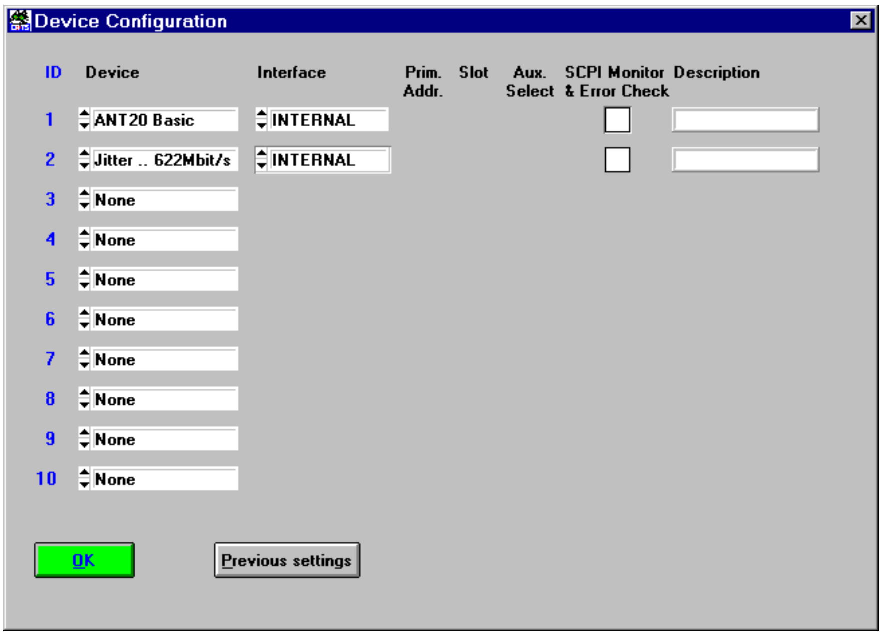


Fig. 6-60 “Device Configuration” window

Note: Monitor&Error Check should always be ON when creating or modifying test sequences, as it allows the sequence developer to monitor all warnings and error messages output by the instrument firmware (debug mode). Setup errors will not be displayed if it is switched off.
Monitor&Error Check should be switched OFF when running a fully tested, debugged sequence to increase speed and avoid “flicker” with certain test cases.

ID	Used to identify the logical device.
Device	Name of the instrument.
Interface	Selects the interface to be opened. • INTERNAL: Communication via the internal port • GPIB: Communication via IEEE 488.2 • GPIB-VXI: Communication via IEEE 488.2 (VXI) • RS-232 COM1: Communication via V.24 (COM1) • RS-232 COM2: Communication via V.24 (COM2)
Prim. Addr.	Primary address of the device.
Slot	Slot number of the VXI card (if CATS Test Sequencer is used in a VXI environment).

Aux. Select	Switch to select OMS module (if CATS Test Sequencer is used with OMS or OVX).
SCPI Monitor&Error Check	This check box enables monitor mode. When activated, all internal SCPI commands transmitted between the CATS Test Sequencer and the measurement hardware, including warnings and error messages, will be monitored and written into a display window.
Description	Optional text field to describe the instrument ID.

6.4 Client window (BN 3035/95.91 and .92 only)

The “Client” window is part of the client software for the

- ANT-20 CATS Test Sequencer with TCP/IP interface (BN 3035/95.91)
- ANT-20 CATS Test Sequencer with DDE interface (BN 3035/95.92)

The “Client” window is similar for both CATS Test Sequencer variants. This section describes the “DDE Client” window as an example.

The “Client” window is opened by double-clicking on the “DDE Client” icon (or “TCP/IP Client” icon).

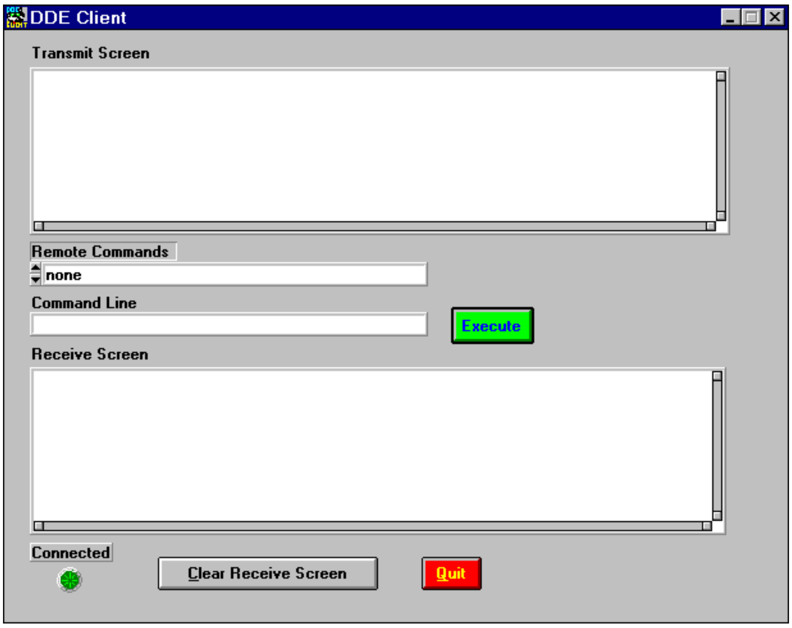


Fig. 6-61 “(DDE) Client” window

Meaning: Operating window for the client software. With the aid of the “Client” window, commands are sent to the ANT-20 CATS Test Sequencer (server) software (e.g. “Load sequence”, “Start sequence”, etc.) and results are received from the ANT-20 CATS Test Sequencer (server) software.

6.4.1 Transmit Screen

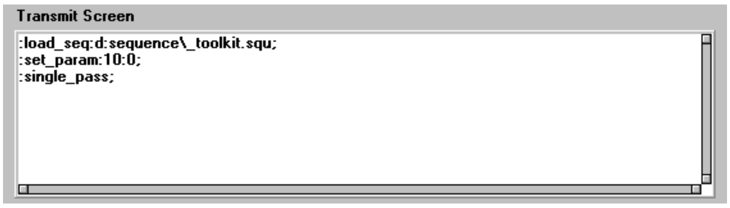


Fig. 6-62 Transmit Screen area of the “Client” window

Meaning: Indicates the commands sent by the client software.

6.4.2 Remote Commands area with Execute button

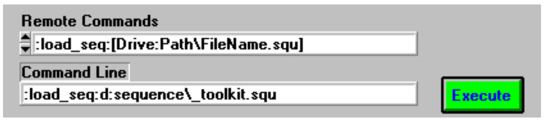


Fig. 6-63 “Remote Commands” area with [Execute] button; part of the “Client” window

Meaning: The commands to be transmitted to the ANT-20 CATS Test Sequencer (server) software are entered and transmitted here.
Also see Sec. 4.2, Page 1-90.

6.4.2.1 Remote Commands

Meaning: You can select possible commands using the [arrow] buttons.
The commands selected with the [arrow] buttons are displayed in the Command Line and must be completed using the keyboard.

6.4.2.2 Command Line

Meaning: The commands selected from the “Remote Commands” list are displayed here.
The selected commands must be completed using the keyboard (e.g. by typing in the drive letter, path, and sequence name, see below).

Note: You can also type in the entire command using the keyboard.

Possible commands in the Command Line

The following is a list of the implemented commands. A colon “:” is used to separate the parameters of the command. Each command must terminate with a semicolon “;”.

load_seq		
Loads a specific sequence.		
Command syntax		:load_seq:*Drive*:*Path\FileName*.squ;
	Drive	Drive where the file to be loaded is located
	Path\FileName	Relative (i.e. relative to the installation directory) path of the sequence file to be loaded\Name of sequence file The sequence file name extension is always “.squ”.
Example		:load_seq:d:sequence_toolkit.squ; Loads the “_toolkit” sequence from directory “.\sequence” on drive “d:”.

Table 6-1 “load_seq” command

set_param		
Sets the parameters of a specific test.		
Command syntax		:set_param:*TestNumber*:*ParameterString*;
	TestNumber	Position of the test case in the sequence.
	ParameterString	Contains the parameter values to be set. The description of this string is part of the test case documentation.
Example		:set_param:10:3; Sets the parameter of test case 10 (i.e. beep) to value 3 (forced) for the loaded test case (_toolkit).

Table 6-2 “set_param” command

set_option		
Sets the report mode to a defined value.		
Command syntax		:_set_option:report_mode:*Value*;
	Value	Value of report_mode. Possible values are “0” = Overwrite or “1” = Append.
Example		:set_option:report_mode:0; Sets report mode for the loaded sequence to overwrite.

Table 6-3 “set_option” command

set_report_file		
Defines the report file name and mode.		
Command syntax		:set_report_file:*File*:*FileMode*;
	File	Relative (i.e. relative to the installation directory) path to the report file.
	FileMode	Overwrite = “0” or Append = “1”
Example		:set_report_file:report_toolkit.rpt:1; The current report will be appended to the previous report in the report file “_toolkit.rpt”.

Table 6-4 “set_report_file” command

single_pass	
Executes each test in the sequence loaded.	
Command syntax	:single_pass;

Table 6-5 “single_pass” command

exe_test		
Starts and executes a test.		
Command syntax		:exe_test:*Number*;
	Number	Position of the test case in the sequence
Example		:exe_test:2; Executes the second test case in the sequence currently loaded.

Table 6-6 “exe_test” command

6.4.2.3 Execute button

Fig. 6-64 [Execute] button in the “Client” window

Meaning: Transmits the command in the command line to the ANT-20 CATS Test Sequencer (server) software.

6.4.3 Receive Screen

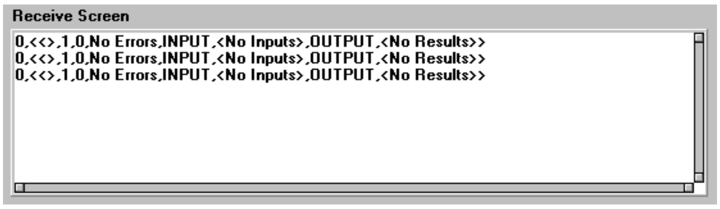


Fig. 6-65 Receive screen area of the “Client” window

Meaning: The result strings are displayed here.

Result string

Syntax		%d,<<%s>,%d,%d,%s,INPUT,<%s>,OUTPUT,<%s>>
Values		Call-Status,<<ReturnString>,ReturnValue,ErrCode, ErrMessage, INPUT,<InputString>,OUTPUT,<OutputString>>
	Call-Status	"0" = OK "7001" = Error in command composition "7002" = Syntax error in command line or bad command "7003" = Missing or wrong parameters in command line "other" = Unknown error
	ReturnString	Name of the function called
	ReturnValue	"1" = PASS "0" = FAIL "4" = ABORT
	ErrCode	"0" = OK "!= 0" ==> See remote operation manual of the instrument
	ErrMessage	Error description
	InputString	See test case documentation
	OutputString	See test case documentation
	Caution	The whole string between the outer brackets "<>" is be replaced by "No Results", if the test case called produces no results.

Table 6-7 Result string

6.4.4 Connected LED

Meaning: Indicates when there is a connection between the CATS Test Sequencer (server) software and the client software.

6.4.5 Clear Receive Screen button

Fig. 6-66 [Clear Receive Screen] button in the "Client" window

Meaning: Clears the contents of the Receive screen.

6.4.6 Quit button

Fig. 6-67 [Quit] button in the "Client" window

Meaning: Terminates the client software.
The "Client" window closes.

Notes:

7 Appendix: Frequently asked questions

This section provides the answers to some frequently asked questions.

7.1 Difference between BN 3035/xx.xx and BN 3045/xx.xx

Question

What is the difference between the ANT-20 CATS Test Sequencer and CATS?

Answer

The ANT-20 CATS Test Sequencer (BN 3035/95.90) is a special run-time version of the CATS family of products. It runs on the ANT-20 itself and allows you to create your own sequence of test cases based on test cases predefined by WWG. The CATS Test Sequencer is ideal if you want to simply and automatically perform the standard tests that can be done using an ANT-20. Typical applications are simple tests in R&D, manufacturing and installation.

The ANT-20 CATS Test Sequencer includes all the CATS test cases that can be performed using an ANT-20 alone. It does not provide for the integration of additional instruments, or for the integration of additional, non-WWG test cases.

No additional hardware or software is needed to run the ANT-20 CATS Test Sequencer. In other words, you do not need a PC, GPIB card for the ANT-20 or PC, or the LabWindows driver (since this is already included). Most importantly, you do not need any training or expertise in programming the remote control interface of the ANT-20. You have the full benefit of a tried and tested solution at a fraction of the cost of a test system.

CATS (BN 3045/x.x) is a family of test automation tools and applications, consisting of:

- “CATS Test Executive Run-Time Package” BN 3045/01 (this is the EMPTY framework that allows you to run a sequence of test cases) plus a selection of test case packages:
- Transmission (3045/93.12)
- Optics (BN 3045/93.41)
- Test Point Selection (BN 3045/93.20)
- Analog, etc.

The LabWindows CVI development environment is required in order to work with these packages and to create your own test cases using LabWindows. This can be purchased from WWG using order number BN 3045/93.01 if you do not have it already.

7.2 Version of LW CVI

Question

Which version of LW CVI is needed to work with the CATS Test Sequencer?

Answer

As of May 2000, version 5.01 of LW CVI is required.

7.3 Difference between the CATS Test Sequencer demo and full versions

Question

What is the difference between the demo and full versions of the CATS Test Sequencer?

Answer

The demo version allows you to perform all the functions available from the full version of the CATS Test Sequencer except saving a sequence to a file. In other words, you can load sequences, run sequences, and change the order or the parameters of test cases, etc. All the measurements performed using the demo version are real measurements. Only the “Save” and “Save as” buttons are disabled (grayed out) in the demo version.

7.4 Operating the ANT-20 user interface and ANT-20 CATS Test Sequencer together

Question

Can I run the ANT-20 user interface and the ANT-20 CATS Test Sequencer or CATS Test Sequencer together?

Answer

No. Both address the same hardware, so there would be a conflict between the two. If you are running the ANT-20 user interface the background, ANT-20 CATS Test Sequencer will only run in simulation mode.

7.5 CATS Run-Time Package and Transmission test case package

Question

If I purchase the “CATS Run-Time Package” and the “Transmission” test case package, can I develop an application that will run on the ANT-20?

Answer

Yes, but the “CATS Test Sequencer” option of the ANT-20 must be enabled. You may order one “CATS Test Sequencer” option for a specific ANT-20 free of charge when ordering the “Transmission” test case package. Please state the serial number of the ANT-20 for which the “CATS Test Sequencer” option (3035/95.90) is required.

7.6 Combining the CATS Test Sequencer with control of an external test point selector

Question

If I want to combine the CATS Test Sequencer with control of an external test point selector, which software packages do I need?

Answer

You need the “CATS Run-Time Package”, the “LW CVI development environment”, the “Transmission” test case package and the “Test Point Selection” test case package. Unless you are an expert user of LabWindows, you should contact WWG D1 SC for a customized proposal that will include all of the above plus an integrated solution for your exact requirements (type of test point selector, monitoring test cases etc.). The ANT-20 CATS Test Sequencer is not suitable, as it does not provide for integration of other instruments (i.e. the test point selector).

7.7 Developing sequences with CATS and running them on different ANT-20s

Question

I want to develop test cases and sequences using the CATS packages (3045/x.x) and to run these sequences on different ANT-20s. What do I need in addition to CATS?

Answer

You can order one “CATS Test Sequencer” option 3035/95.90 for one ANT-20 of your choice free of charge. Each additional ANT-20 must be equipped with a separate, full-price “CATS Test Sequencer 3035/95.90”option (3035/95.90).

7.8 Using the ANT-20 to run CATS (3045/x.x)

Question

Can I use the ANT-20 itself to run CATS (3045/x.x)?

Answer

Yes, but you need a release code for option 3035/95.90. This is provided free of charge for one specific ANT-20 when you purchase the “CATS Test Sequencer” option, 3035/95.90.

7.9 Using ANT-20 to control itself and an OSA-155 for DWDM testing

Question

Can I use ANT-20 to control itself and an OSA-155 for DWDM testing?

Answer

Yes, you need the CATS 3045/93.43 DWDM package and the NI “Lab Windows CVI” development environment. You must also have the “ANT-20 CATS Test Sequencer” option. This is provided free of charge when purchasing the “CATS Test Sequencer” option, BN 3035/95.90.

7.10 Combining CATS Test Sequencer 3035/95.90 with other test case packages

Question

Can I combine the “CATS Test Sequencer” 3035/95.90 with other test case packages, e.g. optics packages, if I want to control optical instruments of the OMS-100 family from the ANT-20?

Answer

No. The “CATS Test Sequencer” 3035/95.90 is a special all-in-one, low-cost member of the CATS family that does provide for integration of other test instruments or your own test case developments. If your application uses more than just one ANT-20, or you want to develop test cases in addition to those provided by WWG, you must purchase the full 3045/x.x packages.

7.11 Combining CATS Test Sequencer 3035/95.90 with Test Point Selector 3045/93.20

Question

Can I combine the “CATS Test Sequencer” 3035/95.90 with the “Test Point Selector” 3045/93.20 package if I want to control a Pickering Test Point Selector from an ANT-20?

Answer

No. The “CATS Test Sequencer” 3035/95.90 is a special all-in-one, low-cost member of the CATS family that does provide for integration of other test instruments or your own test case developments. If your application uses more than just one ANT-20, or you want to develop test cases in addition to those provided by WWG, you must purchase the full 3045/x.x packages.

7.12 Test case packages with C source code

Question

When do I need the test case packages with C source code?

Answer

The C source code of the test cases is only needed if you want to use WWG test cases as a basis for your own development, e.g. for editing minor things in test algorithms. If you only want to use the test cases created by WWG and those that you have produced “from scratch”, e.g. when integrating third party instruments, you do not need the WWG source code.

7.13 Start Device Simulation ...? message

Question

When I start the ANT-20 CATS Test Sequencer, I get the message “Start Device Simulation ...?”. What is the reason for this, and what can I do to avoid this message?

Answer

One reason could be that an old version of the file WGWSDRVW.DLL is present in the directory C:\CVI\WGTX_SRC. This old version cannot be used with versions 5.0 and above of the ANT-20 firmware and should therefore be deleted or renamed. Make sure that a file named WGWSDRVW.DLL is present in the C:\WINDOWS\SYSTEM directory of your ANT-20.

Another reason could be that you are running the ANT-20 user interface at the same time. Exit from the ANT-20 user interface, remove the ANT-20 icon from the startup group, and try again after restarting Windows.

It is also possible that you are trying to load a sequence that needs an option that is not available on your particular ANT-20. For example, you will get the message “Start Device Simulation ANX-930” when loading the sequence WAND_TOL.SQU on an ANT-20 that is not equipped with the Jitter/Wander option. In such cases, ignore the message by clicking “YES”. All the tests in the sequence will run properly except those tests requiring the jitter option, which will run in simulation mode only.

The message “Start device simulation ...?” should not normally be displayed by your ANT-20. If it is displayed, determine the exact cause and remedy the situation as described above. Try again after restarting Windows.

7.14 CATS Test Sequencer demo – Loading sequences that differ from the default sequence

Question

How can I load sequences that differ from the default sequence once I have started the CATS Test Sequencer demo?

Answer

Just select FILE NEW and then FILE LOAD. A selection of available demo files will be displayed.

7.15 Configuration Window showing two or more entries for ANT-20

Question

When entering the CATS Test Sequencer, the “Configuration Window” displayed shows two or more entries for ANT-20, even though I have only one internal ANT-20. Why?

Answer

The TX and RX and the Jitter TX and Jitter RX of the ANT-20 are handled as separate logical devices. This assures compatibility with tests that you might run in a VXI environment or a with a combination of 2 or more ANT-20s. When using internal ANT-20 mode, make sure that all the devices that you will need in your sequences are properly selected in the configuration window. For example, to do a jitter transfer function test, you will need all 4 of the devices mentioned above.

7.16 Configuration Window shows SCPI Monitor & Error Check checkbox

Question

When entering the CATS Test Sequencer, the “Configuration Window” displayed shows a checkbox labeled “SCPI Monitor & Error Check”. What is this for?

Answer

If you check this box, all internal commands to and from the measurement hardware and the CATS Test Sequencer or CATS user interface are shown in a special window (Standard_IO). This is useful for debugging sequences, but it also slows down execution of the tests. You should switch off the error check once you have properly debugged a new sequence.

WWG-CATS Testcase Library

PDH/SDH/SONET Basic Testcases

PDH/SDH/SONET Basic Testcases

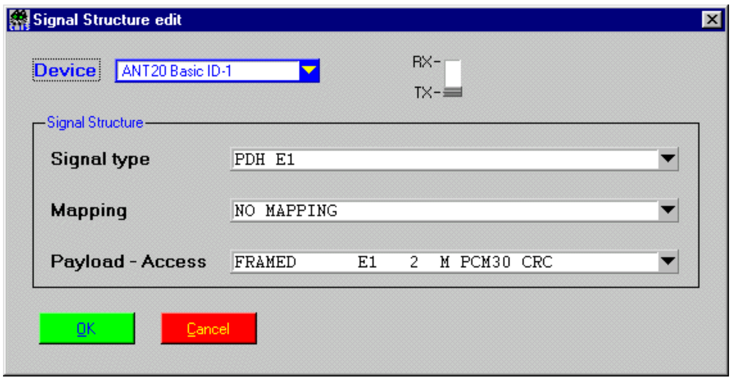
Notes:

1 Set ITU-T Signal Structure RX/TX

Function Name

```
_set_signal_structure
            .\sdh_bas\sig_stru.obj
```

User Interface



Description

Purpose: Sets the ITU-T SDH/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a setup testcase without any pass/fail evaluation. As a rule, "_set_signal_structure" has to precede all such testcases that do error tests, readout functions etc.
In all Wavetek Wandel Goltermann provided test sequences this testcase or its SONET version is used for the initial instrument setup for both RX and TX.

Caution: Always use this testcase before attempting to run any other testcase. This is especially true for the "Run Test" single step mode. In general, none of the Wavetek Wandel Goltermann testcases will assume a specific signal structure setting, and hence using any of the WG testcases without prior use of the "_set_signal_structure" function can lead to errors or misleading results!
Please keep in mind that ANT-20 is reset to a 2Mb/s unframed signal structure whenever a new sequence is loaded.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
RX AU4 12 12       CONTINUE
    RX signal type : STM-1
    mapping        : AU4 TU12 C12 ASYNC
    payload-access : FRAMED      E1   2  M PCM30 CRC
```

Applications

–

Parameters

Device Id: Integer, 1 .. 10

TX/RX: Enumeration
RX = 1, TX = 2

ANSI/SONET signal types
(see 2.2 Set SONET Signal Structure RX/TX)

ITU signal types
GUI text, define name=enum or define value
STM-16, STM16=13
STM-16 C Contignous, SIG_TYPE_STM_16_C_CONT=734
STM-4, STM4=11
STM-4 C Contiguous , SIG_TYPE_STM_4_C_CONT=730
STM-4 C Virtual , SIG_TYPE_STM_4_C_VIRT=731
STM-1 , STM1=10
PDH E4 , PDH_140M=33
STM-0 , SIG_TYPE_STM_0=701
PDH E3 , PDH_34M32
PDH E2 , PDH_8M=31
PDH E1 , PDH_2M=30

Mapping: Enumeration
GUI text, define name=enum or define value
NO MAPPING, NO_MAPPING=0
AU4 VC4,AU4_C4=100
AU4 VC4 STM-16C, SIG_MAP_AU4_C4_STM_16C=842
AU4 VC4 STM-4C, SIG_MAP_AU4_C4_STM_16_4C=843
AU4 VC3, AU4_C3=101
AU4 VC3 DS3 C-PARITY, SIG_MAP_AU4_C3_DS3_CPAR=830
AU4 VC3 DS3 M13, SIG_MAP_AU4_C3_DS3_M13=831
AU4 VC12 ASYNC, AU4_C12_ASYNC=120
AU4 VC12 BYTE SYNC, AU4_C12_FLOAT_BYTE=121
AU4 VC11 TU12 ASYNC, AU4_TU12_C11_ASYNC=130
AU4 VC11 TU12 BYTE SYNC, AU4_TU12_C11_FLOAT_BYTE=131
AU4 VC11 TU11 ASYNC, AU4_TU11_C11_ASYNC=140
AU4 VC11 TU11 BYTE SYNC, AU4_TU11_C11_FLOAT_BYTE=141
AU3 VC3, AU3_C3=201
AU3 VC3 DS3 C-PARITY, SIG_MAP_AU3_C3_DS3_CPAR=832
AU3 VC3 DS3 M13, SIG_MAP_AU3_C3_DS3_M13=833
AU3 VC11 TU12 ASYNC, AU3_TU12_C11_ASYNC=230
AU3 VC11 TU12 BYTE SYNC, AU3_TU12_C11_FLOAT_BYTE=231
AU3 VC11 TU11 ASYNC, AU3_TU11_C11_ASYNC=240
AU3 VC11 TU11 BYTE SYNC, AU3_TU11_C11_FLOAT_BYTE=241
AU3 VC12 ASYNC, AU3_C12_ASYNC=220
AU3 VC12 BYTE SYNC, AU3_C12_FLOAT_BYTE=221

Access: Enumeration
GUI text, define Name=enum or define value
ATM 622M, SIG_PAY_ATM_622M=942
ATM 149 M , ATM_149M=307
FRAMED PATT E4 140 M, FRM_PATT_140M=106
UNFRAMED E4 140 M , UNFRM_140M=4
ATM E4 140 M , ATM_140M=303
ATM 51,3 M , ATM_51M =308
FRAMED PATT DS3 44,7M , SIG_PAY_FRM_PATT_44M=911
UNFRAMED DS3 44,7M , UNFRM_44M=12
ATM DS3 44,7M , ATM_44M =305
ATM PLCP DS3 44,7M , SIG_PAY_ATM_PLCP_44M =931
FRAMED PATT E3 34 M, FRM_PATT_34M=105
UNFRAMED E3 34 M , UNFRM_34M=3
ATM E3 34 M , ATM_34M=302
FRAMED PATT E2 8 M , FRM_PATT_8M=104
FRAMED PATT E1 2 M PCM30 CRC, FRM_PATT_2M_PCM_30_CRC=100
FRAMED PATT E1 2 M PCM30 , FRM_PATT_2M_PCM_30=101
FRAMED E1 2 M PCM30 CRC , PCM_30_CRC=200
FRAMED E1 2 M PCM30 , PCM_30=201
FRAMED PATT E1 2 M PCM31 CRC , FRM_PATT_2M_PCM_31_CRC=102
FRAMED PATT E1 2 M PCM31 , FRM_PATT_2M_PCM_31=103
FRAMED E1 2 M PCM31 CRC , PCM_31_CRC=202
FRAMED E1 2 M PCM31 , PCM_31=203
UNFRAMED E1 2 M , UNFRM_2M=1
ATM E1 2 M, ATM_2M =301
FRAMED PATT DS1 1,5M SF , FRM_PATT_1_5M_SUP=120
FRAMED DS1 1,5M SF , FRM_1_5M_SUP=210
FRAMED PATT DS1 1,5M ESF , FRM_PATT_1_5M_EXT_SUP=121
FRAMED DS1 1,5M ESF , FRM_1_5M_EXT_SUP=211
UNFRAMED DS1 1,5M , UNFRM_1_5M=10
ATM DS1 1,5M , ATM_1_5M=304
PATTERN BULK , PATTERN_BULK=300

Remote Control of Testcase

Input: %d#%d,%d,%d,%d
Device Id,TX/RX,Bitrate,Mapping,Access

Output: No Results

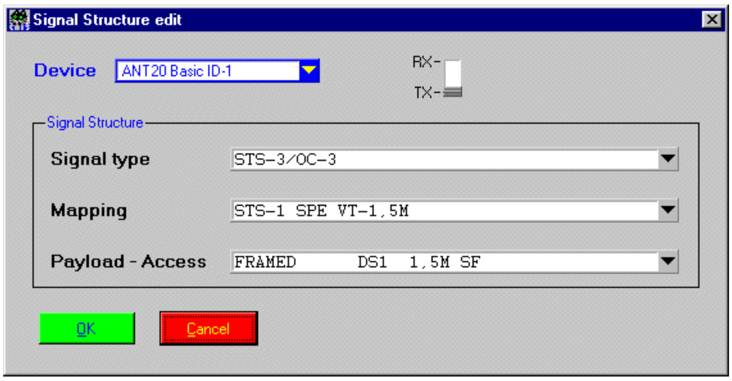
Notes:

2 Set SONET Signal Structure RX/TX

Function Name

```
_set_sonet_structure
            .\sdh_bas\sig_stru.obj
```

User Interface



Description

Purpose: Sets the ANSI SONET/PDH structure of the ANT-20 TX signal or the ANT-20 RX signal. This is just a setup testcase without any pass/fail evaluation. As a rule, “_set_sonet_structure” has to preced all such testcases that do error tests, readout functions etc. In all WG provided test sequences, this testcase or its SDH version is used for the initial instrument setup for both RX and TX.

Caution: Always use this testcase before attempting to run any other testcase. This is especially true for the “Run Test” single step mode. In general, none of the WG testcases will assume a specific signal structure setting, and hence using any of the WG testcases without prior use of the “_set_signal_structure” function can lead to errors or misleading results!
Please keep in mind that ANT-20 is reset to a 2Mb/s unframed signal structure whenever a new sequence is loaded.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
TX STS3 STS-1 VT 1,CONTINUE
    TX signal type : STS-3
    mapping        : STS-1 SPE VT-1,5M
    payload-access : FRAMED PATT DS1  1,5M SF
```

Applications

–

Parameters

Device Id: Integer, 1 .. 10

TX/RX: Enumeration
RX = 1, TX = 2

Bitrate: Enumeration

ANSI/SONET signal types

GUI text, define name=enum or define value
OC-48, STS48 =25
OC-48 C Contignous, SIG_TYPE_STS_48_C_CONT=736
OC-12, STS12=23
OC-12 C Contiguous , SIG_TYPE_STS_12_C_CONT =732
OC-12 C Virtual, SIG_TYPE_STS_12_C_VIRT=733
STS-3/OC-3, STS3=21
STS-1/OC-1, STS1=20
PDH DS3 C-PARITY, PDH_44M= 42
PDH DS3 M13, SIG_TYPE_PDH_44M_M13=721
PDH DS2, SIG_TYPE_PDH_6_3M=718
PDH DS1, PDH_1_5M=40

ITU signal types
(see 2.1 Set ITU-T Signal Structure RX/TX)

Mapping: Enumeration
GUI text, define name=enum or define value
NO MAPPING, NO_MAPPING=0
STS-48C SPE, SIG_MAP_AU4_C4_STS_48C=844
STS-12C SPE (only for OC48C), SIG_MAP_AU4_C4_STS_48_12C=845
STS-12C SPE,SIG_MAP_AU4_C4_STS_12C=840
STS-3C SPE ATM 149M, SONET_STS_3C_ATM_149M=256
STS-3C SPE DS4NA(E4), SONET_STS_3C_DS4NA=255
STS-1 SPE ATM 51,3M, SONET_STS_1_SPE_ATM_51M=257
STS-1 SPE DS3 C-PARITY, SONET_STS_1_SPE_44M=253
STS-1 SPE DS3 M13, SIG_MAP_SONET_STS_1_SPE_44M_M13=822
STS-1 SPE VT-6,3M, SIG_MAP_SONET_STS_1_VT_6_3M=820
STS-1 SPE VT-2 M, SONET_STS_1_VTG_VT_2M =251
STS-1 SPE VT-1,5M, SONET_STS_1_VTG_VT_1_5M=250

Access: Enumeration
GUI text, define Name=enum or define value
ATM 622M, SIG_PAY_ATM_622M=942
ATM 149 M , ATM_149M=307
FRAMED PATT E4 140 M, FRM_PATT_140M=106
UNFRAMED E4 140 M , UNFRM_140M=4
ATM E4 140 M , ATM_140M=303
ATM 51,3 M , ATM_51M =308
FRAMED PATT DS3 44,7M , SIG_PAY_FRM_PATT_44M=911
UNFRAMED DS3 44,7M , UNFRM_44M=12
ATM DS3 44,7M , ATM_44M =305
ATM PLCP DS3 44,7M , SIG_PAY_ATM_PLCP_44M =931
FRAMED PATT DS2 6,3M , SIG_PAY_FRM_PATT_6_3M=907
UNFRAMED DS2 6,3M , SIG_PAY_UNFRM_6_3M=921
FRAMED PATT E1 2 M PCM30 CRC, FRM_PATT_2M_PCM_30_CRC=100
FRAMED PATT E1 2 M PCM30 , FRM_PATT_2M_PCM_30=101
FRAMED E1 2 M PCM30 CRC , PCM_30_CRC=200
FRAMED E1 2 M PCM30 , PCM_30=201
FRAMED PATT E1 2 M PCM31 CRC , FRM_PATT_2M_PCM_31_CRC=102
FRAMED PATT E1 2 M PCM31 , FRM_PATT_2M_PCM_31=103
FRAMED E1 2 M PCM31 CRC , PCM_31_CRC=202
FRAMED E1 2 M PCM31 , PCM_31=203
UNFRAMED E1 2 M , UNFRM_2M=1
ATM E1 2 M, ATM_2M =301
FRAMED PATT DS1 1,5M SF , FRM_PATT_1_5M_SUP=120
FRAMED DS1 1,5M SF , FRM_1_5M_SUP=210
FRAMED PATT DS1 1,5M ESF , FRM_PATT_1_5M_EXT_SUP=121
FRAMED DS1 1,5M ESF , FRM_1_5M_EXT_SUP=211
UNFRAMED DS1 1,5M , UNFRM_1_5M=10
ATM DS1 1,5M , ATM_1_5M=304
PATTERN BULK , PATTERN_BULK=300

Remote Control of Testcase

Input: %d#%d,%d,%d,%d
Device Id,TX/RX,Bitrate,Mapping,Access

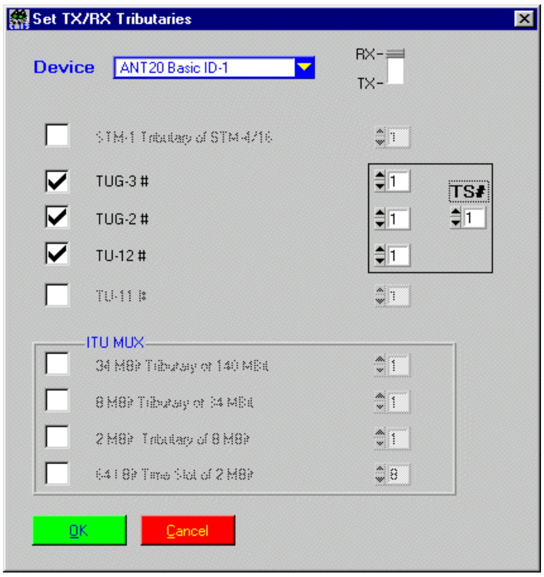
Output: No Results

3 Set ITU-T Channel Number RX/TX

Function Name

```
_set_tributaries
            .\sdh_bas\sdh_bas.obj
```

User Interface



Description

Purpose: Sets the tributary/channel number of an ITU-T signal. All subsequent testcases will be done for the tributary/channel defined here.

Caution: There are a few testcases where the tributaries/channels are defined internally. Example: In the testcase "_check_C12_transparency", a list of STM-1 C12 channels can be defined. In this case, the settings from "_set_tributaries" are overridden.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
TX C12 Trib. 1-1-1 CONTINUE
    TX: TUG3       : 1,
        TUG2       : 1,
        TU12       : 1,
        TS         : 1
```

Applications

–

Parameters

Device Id:	Integer, 1 .. 10
TX/RX:	Enumeration RX = 1, TX =2
TUG3:	Integer, -1 .. 3
TUG2:	Integer, -1 .. 7
TUG12:	Integer, -1 .. 3
Tributary	Integer, -1 .. 63 (calculated from TUG3, TUG2, TU12)
TU11:	Integer, -1 .. 4
Time Slot 64K:	Integer, -1 .. 10
Trib. 2M:	Integer, -1 .. 4
Trib. 8M:	Integer, -1 .. 4
Trib. 34M:	Integer, -1 .. 4
Trib. STM-1:	Integer, -1 .. 16

-1 ==> Not applicable

Remote Control of Testcase

Input: %d#%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d
Device Id,TX/RX,TUG3,TUG2,TUG12,Tributary,Time Slot 64K,
Trib. 2M,Trib. 8M,Trib. 34M,Trib. STM-1,TU11

Output: No Results

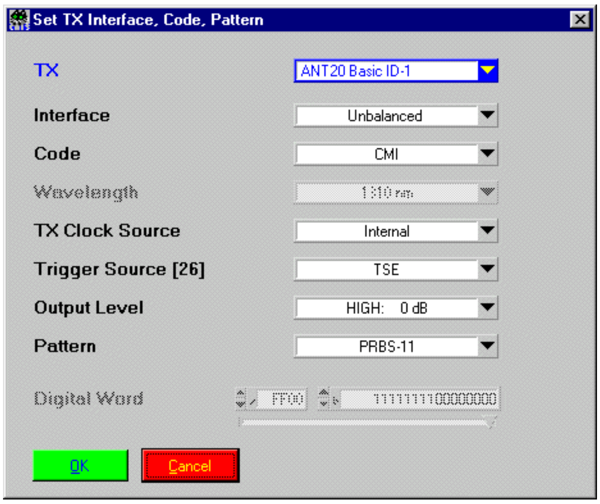
Notes:

4 Set TX Code, PRBS, Optics Clock, Trigger, Level

Function Name

```
_set_tx_adaptation2
            .\sdh_bas\sdh_bas.obj
```

User Interface



Description

Purpose: Sets those signal parameters like interface settings, PRBS sequence, Digital Word, output level, optical wavelengths (for STM-1, STM-4 and STM-16) etc. that are not covered by the "_set_signal_structure" testcase. It is also possible to set the clock source for the TX signal as well as the trigger source for the output [26].
This testcase is a replacement for the testcase "_set_tx_adaptation", which will still be part of the CATS "Transmission testcases" package for compatibility reasons. In this new testcase, the entry field for SCPI commands is no longer existing, because the testcase "_scpi_io" serves as a full and better replacement for this function.

Caution: For new sequence developments, only "_set_tx_adaptation2" should be used.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
TX Adaptation2        CONTINUE
    TX Interface      : UNBALANCED 75 OHM
    TX Code           : CMI
    TX Pattern        : PRBS-23
    TX Clock Source   : internal
    TX Trigger Source : TSE
    TX Output Level   : HIGH: 0dB
```

Applications

–

Parameters

TX Id: Integer, 1 .. 10

Interface: Enumeration
UNBALANCED 75 OHM = 1, BALANCED 120 OHM = 2, OPTIC STM1/4 = 3, OPTIC STM16 = 4.

Code: Enumeration
CMI = 1, NRZ = 2 (optical only), HDB3 = 3.

Pattern: Enumeration
PRBS-11 = 2, PRBS-15 = 3, PRBS-20 = 8, PRBS-23 = 4, QRSS-20 = 7, PRBS-31= 6, PRBS-11 Inverted = 12, PRBS-15 Inverted = 13, PRBS-20 Inverted = 18, PRBS-23 Inverted = 14, PRBS-31 Inverted = 16
Digital Word = 707.

Digital Word: Unsigned long, 0 .. FFFF (Hex)

Wavelength: Enumeration
1310nm = 1310, 1550nm = 1550.

Clock Source: Enumeration
Internal = 1, External data 2Mbit/s = 2, External clock 2MHz = 3, External data 1.5Mbit/s = 4, External clock 1.5MHz = 5, From RX = 6.

Trigger Source: Enumeration
TSE = 1, Reference clock 2MHz = 2, TX frame trigger (SDH) = 3, TX pattern trigger = 4.

Output Level: Enumeration
HIGH (0dB) = 1, LOW (-13,8dB) = 2, DSX (450ft) = 3.

Remote Control of Testcase

Input: %d#%d,%d,%d,%ld,%d,%d,%d,%d
TX Id,Interface,Code,Pattern,Digital Word,Wavelength,Clock Source, Trigger Source,Output Level.

Output: No Results

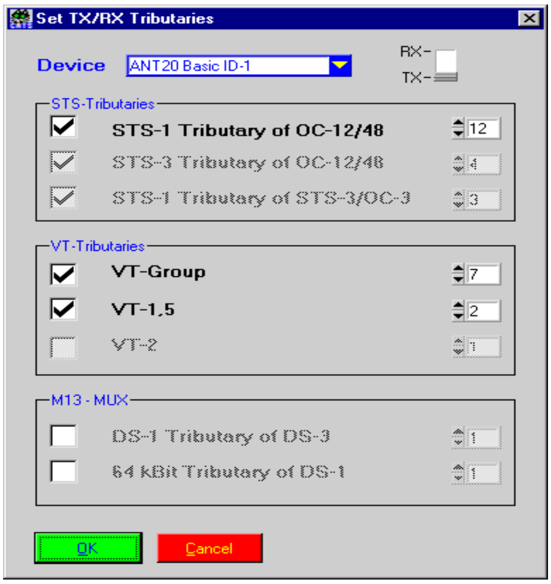
Notes:

5 Set SONET Channel Number RX/TX

Function Name

```
_set_sonet_tributaries
             .\sdh_bas\sdh_bas.obj
```

User Interface



Description

Purpose: Sets the tributary/channel number of an ANSI/SONET signal. All subsequent testcases will be done for the tributary/channel defined here.

Caution: There are a few testcases where the tributaries/channels are defined internally. Example: In the testcase "_check_VT1_5_transparency" (planned for the next version), a list of STS-1 VT1_5 channels can be defined. In this case, the settings from "_set_sonet_tributaries" are overridden.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
TX VTG VT            CONTINUE
    TX: STS-1        : 3,
        VT-Group     : 7,
        VT-1,5       : 4,
```

Applications

–

Parameters

Device Id:	Integer, 1 .. 10
TX/RX:	Enumeration RX = 1, TX =2
Trib. STS-3:	Integer, -1 .. 3
Trib. STS-1:	Integer, -1 .. 7
VTG:	Integer, -1 .. 3
VT 1.5M	Integer, -1 .. 63 (calculated from TUG3, TUG2, TU12)
VT 2M:	Integer, -1 .. 4
Trib. DS-3:	Integer, -1 .. 10
Trib. DS-2:	Integer, -1 .. 4
Trib. DS-1:	Integer, -1 .. 4
Time Slot 64K:	Integer, -1 .. 4

-1 ==> Not applicable

Remote Control of Testcase

Input: %d#%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d
Device Id,TX/RX,Trib. STS-3,Trib. STS-1,VTG,VT 1.5M,VT 2M,Trib. DS-3, Trib. DS-2,Trib. DS-1,Time Slot 64K

Output: No Results

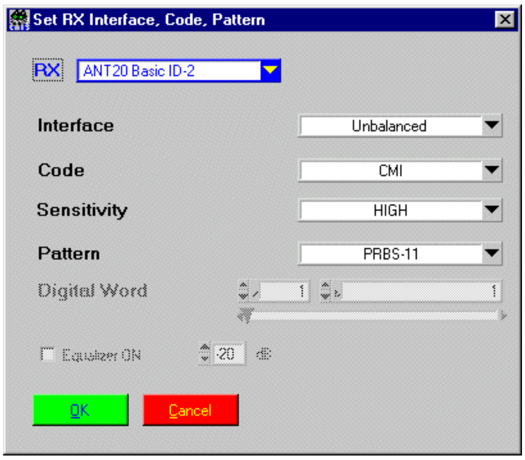
Notes:

6 Set RX Code, PRBS, Optics, Sensitivity, Equalizer

Function Name

```
_set_rx_adaptation2
            .\sdh_bas\sdh_bas.obj
```

User Interface



Description

Purpose: Sets those signal parameters like interface settings, PRBS sequence, Digital Word, output level etc. that are not covered by the "_set_signal_structure" testcase. Additional parameters are sensitivity and equalizer settings. The equalizer can be switched "ON" if sensitivity is set to "PMP". Equalizer level can be set from -15 to -26dB.
This testcase is a replacement for the testcase "_set_rx_adaptation", which will still be part of the CATS "Transmission testcases" package for compatibility reasons. In this new testcase, the entry field for SCPI commands is no longer existing, because the testcase "_scpi_io" serves as a full and better replacement for this function.

Caution: For new sequence developments, only "_set_rx_adaptation2" should be used.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
RX Adaptation2        CONTINUE
    RX Interface      : UNBALANCED 75 OHM
    RX Code           : CMI
    RX Pattern        : PRBS-23
    RX Sensitivity    : ITU-T (HIGH)
```

Applications

–

Parameters

RX Id: Integer, 1 .. 10

Interface: Enumeration
UNBALANCED 75 OHM = 1, BALANCED 120 OHM = 2, OPTIC STM1/4 = 3, OPTIC STM16 = 4.

Code: Enumeration
CMI = 1, NRZ = 2 (optical only), HDB3 = 3.

Pattern: Enumeration
PRBS-11 = 2, PRBS-15 = 3, PRBS-20 = 8, PRBS-23 = 4, QRSS-20 = 7, PRBS-31 = 6, PRBS-11 Inverted = 12, PRBS-15 Inverted = 13, PRBS-20 Inverted = 18, PRBS-23 Inverted = 14, PRBS-31 Inverted = 16,
Traffic = 40,
Digital Word = 707.

Digital Word: Unsigned long, 0 .. FFFF (Hex)

Sensitivity: Enumeration
1310nm = 1310, 1550nm = 1550.

Equalizer: Enumeration
OFF = 0, ON = 1.

Equalizer Lev.: Integer, -26 .. -15 [dB]

Remote Control of Testcase

Input: %d#%d,%d,%d,%ld,%d,%d,%d
TX Id,Interface,Code,Pattern,Digital Word,Sensitivity,Equalizer,Equalizer Lev..

Output: No Results

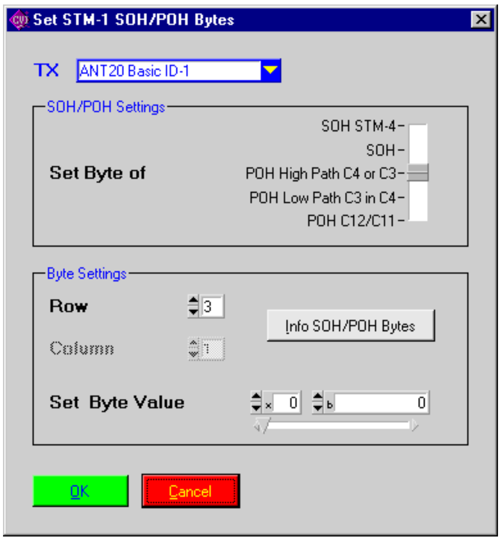
Notes:

7 Set SDH Overhead Byte

Function Name

```
_set_SOH_POH_byte
            .\sdh_bas\sdh_bas.obj
```

User Interface



Description

Purpose: Sets contents of one overhead byte.

Caution: Do not use to set parity bytes (B1, B2, B3).

STM-4: Column counting is from 1 to 36 (info not valid).

STM-16: Use “SOH” setting to set overhead of tributary #1. Setting other tributary #’s Overhead bytes is not possible with this testcase.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Set C2=00h HP_UNEQ  CONTINUE
    Set C4 POH byte (row 3) to HEX 00
```

Applications

Set specific overhead bytes permanently, e.g. to create alarms.

Parameters

TX Id:	Integer, 1 .. 10
Selector:	Enumeration SOH = 1, C4 POH = 2, C3 POH = 3, C12/C11 POH = 4
Row:	Integer, 1 .. 9
Column:	Integer, 1 .. 36
ByteValue:	Hex, 00 .. FF

Remote Control of Testcase

Input:	%d#%d,%d,%02X,%d TX Id,Row,Column,Byte Value,Selector
Output:	No Results

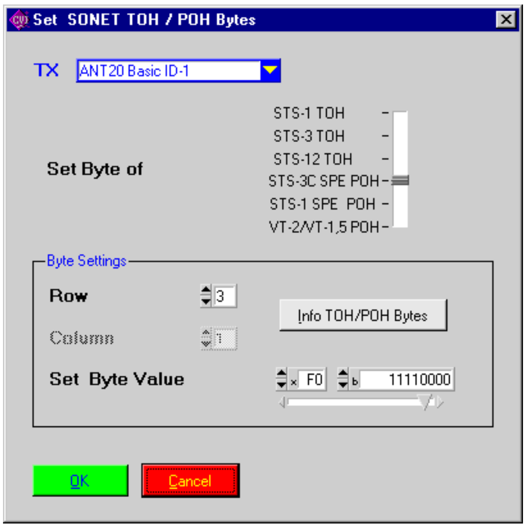
Notes:

8 Set SONET Overhead Byte

Function Name

```
_set_TOH_POH_byte
            .\sdh_bas\sonetbas.obj
```

User Interface



Description

Purpose: Sets contents of one overhead byte.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Set C2=00h UNEQ_P   CONTINUE
    Set STS-3 C POH byte (row 3) to HEX 00
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
Selector:	Enumeration TOH = 1, STS-3C SPE POH = 2, STS-1 SPE POH = 3, VT POH = 4,
Row:	Integer, 1 .. 9
Column:	Integer, 1 .. 36
ByteValue:	Hex, 00 .. FF

Remote Control of Testcase

Input:	%d#%d,%d,%02X,%d TX Id,Row,Column,Byte Value,Selector
Output:	No Results

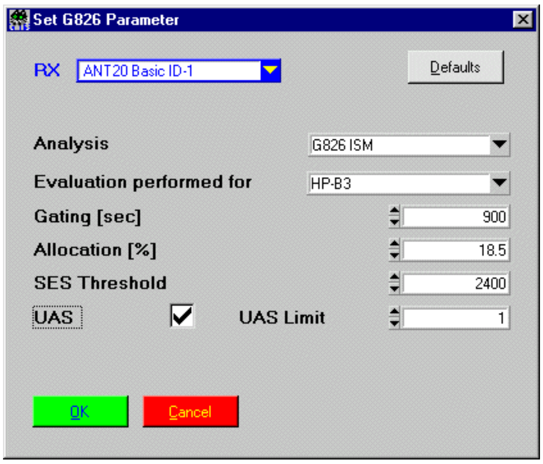
Notes:

9 Set Parameters for G.826 and Start G.826

Function Name

```
_set_and_start_g826
             .\sdh_bas\g826.obj
```

User Interface



Description

Purpose: Sets the parameters of a G.826 evaluation.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup G826       PASS
    AnalysisType             : G826 ISM
    Evaluation performed for : PDH 2MBit/s - CRC
    Gating                   : 200 sec
    Allocation               : 18.5 %
    SES Threshold            : 2400
    UAS Limit                : OFF
```

Applications

See sequence “i_g826.squ”.

Parameters

RX Id:	Integer, 1 .. 10
Analysis:	Enumeration G.826 Evaluation Off = 0, G.826 ISM = 1, G.826 OOS = 2
Evaluation:	Enumeration RSOH B1 = 0, MSOH B2 Summary = 1, HP B3 = 2, LP BIP 2/8 = 3, PDH 140Mbit/s = 4, PDH 34Mbit/s = 5, PDH 8Mbit/s = 6, PDH 2Mbit/s FAS = 7, PDH 2Mbit/s CRC = 8, G.832 PDH 140Mbit/s = 9, G.832 PDH 34Mbit/s = 10, G.832 PDH 140Mbit/s EM = 11, G.832 PDH 34Mbit/s EM = 12
Gating:	Double, 1 .. 1000000000 [s]
Allocation:	Double, 0.0 .. 100.0 [%]
SES Threshold:	Unsigned Integer, 1 .. 47563
UAS Flag:	Enumeration ON = 1, OFF = 0
UAS Limit:	Unsigned Integer, 0 .. 100000

Remote Control of Testcase

Input:	%d#%d,%d,%lf,%lf,%d,%d,%d RX Id,Analysis,Evaluation,Gating,Allocation,SES Threshold,USA Flag, UAS Limit
Output:	No Results

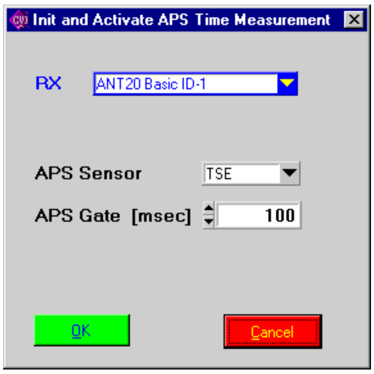
Notes:

10 Set ITU-T Parameters for APS and Start APS

Function Name

```
_set_and_start_APS
            .\sdh_bas\aps.obj
```

User Interface



Description

Purpose: Initializes APS measurement and starts it. APS results can be checked by calling the testcase “_check_APS” afterwards.

Caution: For APS measurement ANT-20 Software (firmware) > 6.0 is required

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
init APS          PASS
```

Applications

See sequence “i_aps.squ or a_aps.squ”.

Parameters

RX Id: Integer, 1 .. 10 corresponds to device name, interface address of the configuration window

APS mode: Integer
1=TSE (Bit Errors)
2=MS-AIS
3=AU-AIS
4=TU-AIS

APS gate: double 100..5000 msec

Remote Control of Testcase

Input: %d#%d,%lf
RX Id,APS_mode,APS_gate

Output: No Results

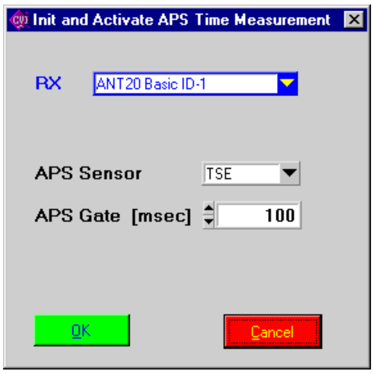
Notes:

11 Set SONET Parameters for APS and Start APS

Function Name

```
_set_and_start_sonet_APS
            .\sdh_bas\aps.obj
```

User Interface



Description

Purpose: Initializes APS measurement and starts it. APS results can be checked by calling the testcase "_check_APS" afterwards.

Caution: For APS measurement ANT-20 Software (firmware) > 6.0 is required

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
init APS          PASS
```

Applications

See sequence “i_aps.squ or a_aps.squ”.

Parameters

RX Id: Integer, 1 .. 10 corresponds to device name, interface address of the configuration window

APS mode: Integer
1=TSE (Bit Errors)
5=AIS-L
6=AIS-P
7=AIS-V

APS gate: double 100..5000 msec

Remote Control of Testcase

Input: %d#%d,%lf
RX Id,APS_mode,APS_gate

Output: No Results

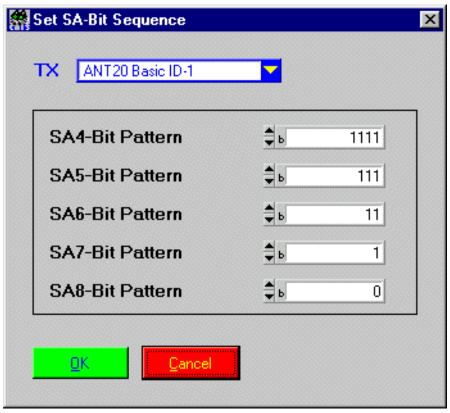
Notes:

12 Set 2 MBit/s CRC Sa Bit Sequences

Function Name

```
_set_sa_bit_sequence
            .\sdh_bas\sa_bits.obj
```

User Interface



Description

Purpose: Sets the patterns of Sa-Bits of 2Mbit/s signal.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
set SA bit seq      PASS
    SA4-Bit Pattern: 00001111
    SA5-Bit Pattern: 00000111
    SA6-Bit Pattern: 00000011
    SA7-Bit Pattern: 00000001
    SA8-Bit Pattern: 00000000
```

Applications

See sequence “i_2Mbit.squ”.

Parameters

TX Id:	Integer, 1 .. 10
Sa-Bit 4 Seq.:	Integer, 0 .. 255
Sa-Bit 5 Seq.:	Integer, 0 .. 255
Sa-Bit 6 Seq.:	Integer, 0 .. 255
Sa-Bit 7Seq.:	Integer, 0 .. 255
Sa-Bit 8 Seq.:	Integer, 0 .. 255

Remote Control of Testcase

Input:	%d#%d,%d,%d,%d,%d TX Id,Sa-Bit 4 Seq,Sa-Bit 5 Seq.,Sa-Bit 6 Seq.,Sa-Bit 7 Seq.,Sa-Bit 8 Seq.
Output:	No Results

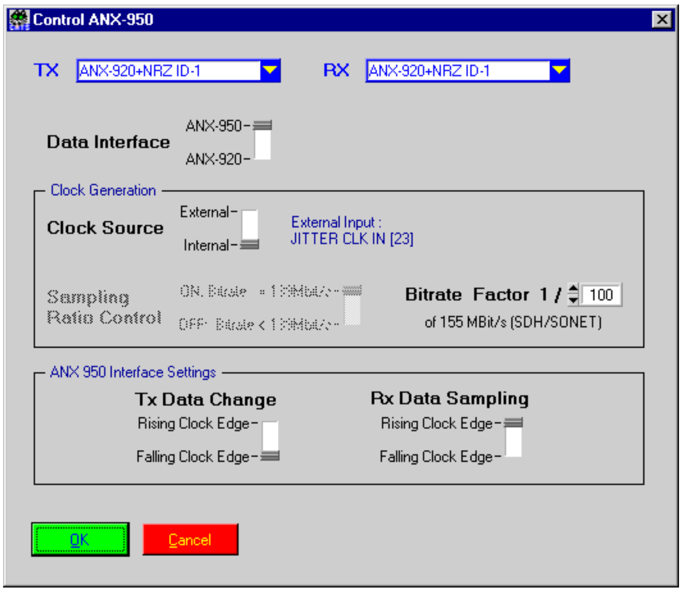
Notes:

13 Setup ANX950 Module (VXI module ANX-950 only)

Function Name

```
_control_NRZ
                .\sdh_bas\nrz.obj
```

User Interface



Description

Purpose: Controls the NRZ module ANX-950 and chooses the Clock Generation.

Caution: While using an external clock pay attention to the sampling ratio control. It must be turned off if the signal is below 139MBit/s, otherwise it will come to unpredictable results.
On using the internal clock be aware, that the dividing of the bitrate will only work if the signal is routed over the ANX-950. While routing the divided signal over the ANX-920 there will be an alarm state.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup NRZ Module PASS
   Interface             : ANX-950
   Clock Source          : internal
   Bitrate Factor        : 1 / 50
   Sampling Ratio Control: OFF: Bitrate < 139 Mbit/s

   ANX-950 Interface Settings:
     Tx Data Change      : Rising Clock Edge
     Tx Data Sampling    : Falling Clock Edge
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
Interface:	Enumeration ANX-920 = 0, ANX-950 = 1
Sam. R. Ctrl:	Enumeration ON Bitrate >=139MBit/s = 1, OFF Bitrate < 139MBit/s = 0
Bitrate Factor:	Integer, 1 .. 255
Tx Change:	Enumeration Rising Edge = 0, Falling Edge = 1
Rx Sampling:	Enumeration Falling Edge = 0, Rising Edge = 1

Remote Control of Testcase

Input:	%d,%d#%d,%d,%d,%d,%d,%d TX Id,RX Id,Interface,Data Change,Bitrate Factor,Sampling,Sampling Ratio Control,Clock Source
Output:	No Results

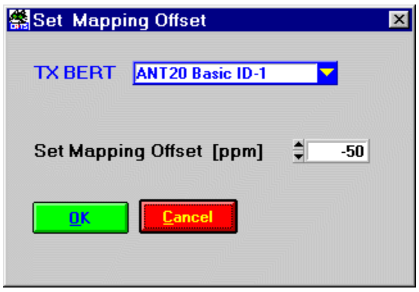
Notes:

14 Set Mapping Offset

Function Name

```
_set_mapp_offset
               .\sdh_bas\mapp_jit.obj
```

User Interface



Description

Purpose: Sets the mapping to the value which was found with the testcase "_find_max_mapping_jitter".

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
set mapping offset:    -50 ppm
```

Applications

See the following sequences in the Operating Manual:

- i_cj_e1.squ
- a_cj_ds1.squ

Parameters

tx_id	Integer 1..10
critical mapping offset	-100 .. 100 ppm

Remote Control of Testcase

Input:	"%d#%d" tx_BERT_Id,mapping_offset
Output:	–

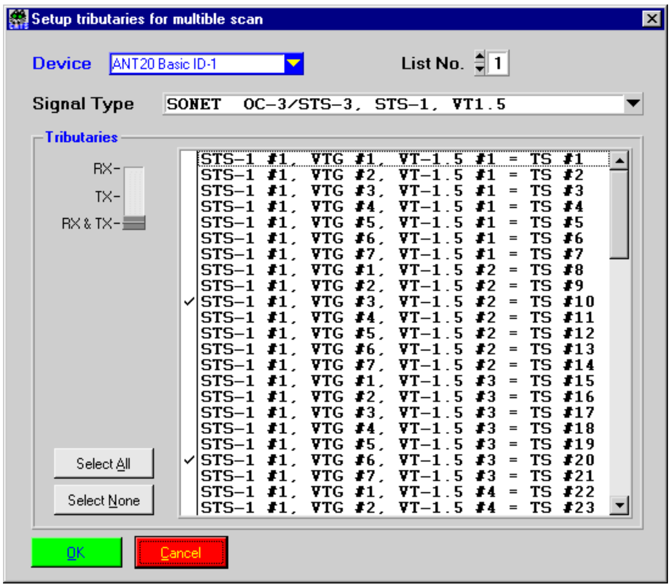
Notes:

15 Define Trib. List for Scanning Multiple Tribs.

Function Name

```
_setup_mscan_trib
            .\sdh_bas\mscantri.obj
```

User Interface



Description

Purpose: Defines a list of tributaries which will be set with the _mscan_set_inc_trib testcase. In combination with a GOTO this list can be used to scan several tributaries and carry out any number of additional testcases for each tributary form the defined list. (See application part for further information, or see example sequences i_List.squ or a_List.squ)

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Define VT List   PASS
    Setup tributaries list #1
    Device        : ANT20 Basic ID-1
    Signal Type   : SONET  OC-3/STS-3, STS-1, VT1.5
    Select #6
```

Applications

The combination of the testcases _setup_mscan_trib, _mscan_set_inc_trib and a GOTO allows a loop of measuring testcases over several tributaries.

One example is to check all or some tributaries included in a STM-1 AU4 VC12 signal.

The mechanism for loop control is described in the following testcase “Increment Trib-Number of defined Trib. List” (_mscan_set_inc_trib)

examples sequences: i_List.squ or a_List.squ

Parameters

Id: Integer, 1 .. 10

RX/TX: Enumeration
RX = 0, TX = 1, TX&RX = 2

Signal Type: Enumeration
SDH STM-16 = 26
SDH STM-4 = 25
SDH STM-1, AU4, TU12, VC12 = 6
SDH STM-1, AU4, TU12, VC11 = 8
SDH STM-1, AU4, TU11, VC11 = 9
SDH STM-1, AU3, TU12, VC12 = 7
SDH STM-1, AU3, TU12, VC11 = 10
SDH STM-1, AU3, TU11, VC11 = 11
SDH STM-1, AU4/3, C3 DS3 = 19
SDH STM-1, AU4/3, C3 E3 = 20
SONET OC-48 = 28
SONET OC-12 = 27
SONET OC-3/STS-3 = 22
SONET OC-3/STS-3, STS-1, VT1.5 = 14
SONET OC-3/STS-3, STS-1, VT2 = 15
SONET OC-1/STS-1, VT1.5 = 12
SONET OC-1/STS-1, VT2 = 13
PDH DS3 - 45M = 23
PDH E4 - 140 M = 18
PDH E3 - 34M = 17
PDH E2 - 8M = 16

List No.: Integer, 1 .. 8

Sel. Trib. #1: Enumeration
1 = Selected, 0 = Not Selected

...

Sel. Trib. #86: Enumeration
1 = Selected, 0 = Not Selected

Remote Control of Testcase

Input: %d#%d,%d,%d,%d, .. %d
Id,RX/TX,Signal Type,List Number,Sel. Trib. #1, ..,Sel. Trib. #86

Output: No Results

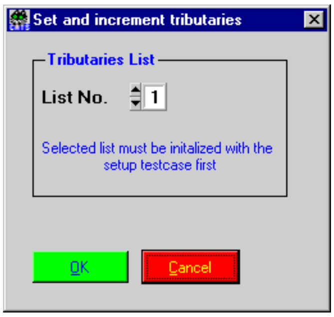
Notes:

16 Increment Trib-Number of defined Trib.-List

Function Name

```
_mscan_set_inc_trib
            .\sdh_bas\mscantri.obj
```

User Interface



Description

Purpose:	Selects next tributary from defined list which is set with the _setup_mscan_trib testcase. If the end of the list is reached, then no new tributary will be set and the testcase is PASS. (See applications part for further information)
Caution:	The _setup_mscan_trib testcase must run before.

PASS / FAIL Conditions

PASS:	End of list is reached.
FAIL:	New tributary selected.

Test Report

```
Increment VT-Trib       FAIL
    Set tributary from list #1
    Set RX STS-1 #1 trib.: VTG=3, VT=2;  TS=10
    Set TX STS-1 #1 trib.: VTG=3, VT=2;  TS=10
```

Applications

Loop control is done in the following way:

_setup_mscan_trib		definition of list
	_mscan_set_inc_trib	Increment tributary entry of defined list
	_test1	Testcase (loop body)
	_test2	
	_test_n	
GOTO **_mscan_set_inc_trib**		as long as entries are in the define list until the end of the list is reached

_further testcases, which are not inside the loop

To enable a conditional break of the loop a precondition for the GOTO statement is defined. The precondition allows the execution of the GOTO statement as long as the return value of the testcase “_mscan_set_inc_trib” is FAIL otherwise the GOTO statement is skipped.
For more information about preconditions refer to chapter 1.4.3.4.

The ‘_mscan_set_inc_trib’ testcase is FAIL until the end of the list is NOT reached.
When the end of the list is reached ‘_mscan_set_inc_trib’ testcase will PASS and therefore the GOTO will be skipped. Now the testcase after the GOTO are executed.

Please have a look at the example sequences ‘i_list.squ or a_list.squ’.

Parameters

Id: Integer, 1 .. 10

Remote Control of Testcase

Input: %d#
Id

Output: No Results

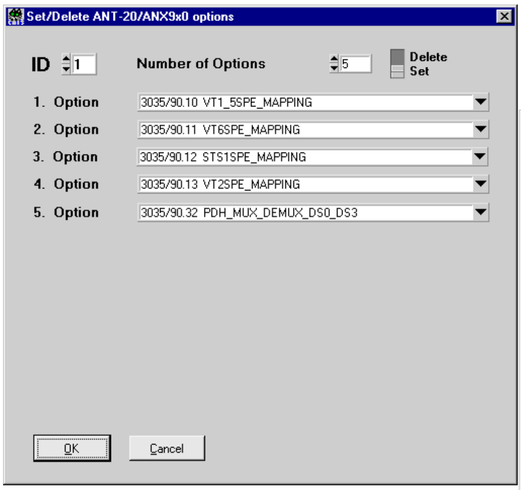
Notes:

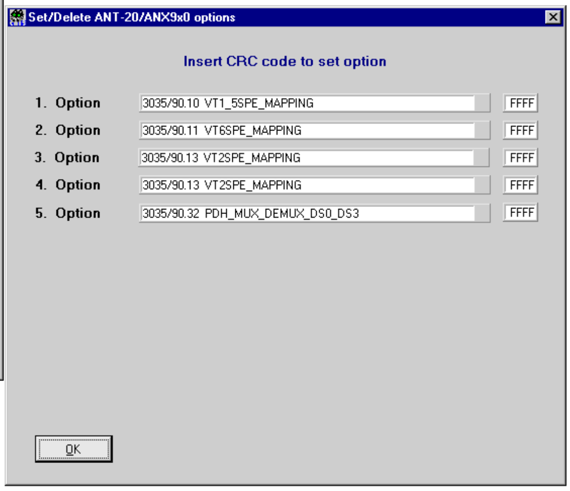
17 Add / Remove Software Options

Function Name

```
_set_options
             .\sdh_bas\options.obj
```

User Interface





Description

Purpose: Set or delets options of the ANT-20/ANX-9x0.

Caution: Should not be used in remote mode.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Set Opt                                   PASS
    Set/Delete Options
    new options :
    3035/90.10  VT1_5SPE_MAPPING        : FFFF
    3035/90.11  VT6SPE_MAPPING          : FFFF
    3035/90.12  STS1SPE_MAPPING         : FFFF
    3035/90.32  PDH_MUX_DEMUX_DS0_DS3   : FFFF
    switch the ANT20/ANX9x0 off/on
    to initialize the options
```

Applications

–

Parameters

Id: Integer, 1 .. 10

Option Cnt.: Integer, 1 .. 12

Delete Option: Enumeration
Set Options=0, Delete Options=1

Option: Enumeration (up to 12)
3035/90.01 C12_MAPPING = 9001, 3035/90.02 C31_MAPPING = 9002,
3035/90.03 C4_MAPPING = 9003, 3035/90.04 C11_MAPPING = 9004,
3035/90.05 C32_MAPPING = 9005,
3035/90.10 VT1_5SPE_MAPPING = 9010,
3035/90.11 VT6SPE_MAPPING = 9011,
3035/90.12 STS1SPE_MAPPING = 9012,
3035/90.13 VT2SPE_MAPPING = 9012, 3035/90.20 DROP_INSERT = 9020,
3035/90.21 MIXED_MAPPING = 9021,
3035/90.22 LOCONC_VC11_VC12 = 9022,
3035/90.30 PDH_MUX_DEMUX_64K_140M = 9030,
3035/90.31 PDH_DEMUX_64K_140M = 9031,
3035/90.32 PDH_MUX_DEMUX_DS0_DS3 = 9032,
3035/90.33 BERT_2_8_34_140M = 9033,
3035/90.34 BERT_1M5_6M3_45M = 9034,
3035/90.40 OPTIC_OC1_1310NM = 9040,
3035/90.41 OPTIC_OC1_1550NM = 9041,
3035/90.42 OPTIC_OC1_1310_1550NM = 9042,
3035/90.43 OPTIC_STM1_1310NM = 9043,
3035/90.44 OPTIC_STM1_1550NM = 9044,
3035/90.45 OPTIC_STM1_1310_1550NM = 9045,
3035/90.46 OPTIC_STM1_4_1310NM = 9046,
3035/90.47 OPTIC_STM1_4_1550NM = 9047,
3035/90.48 OPTIC_STM1_4_1310_1550NM = 9048,
3035/90.50 OPTIC_STM1_16_1310NM = 9051,
3035/90.51 OPTIC_STM1_16_1550NM = 9050,
3035/90.59 OPTIC_STM1_16_1310_1550NM = 9059,
3035/90.70 ATM_MODUL = 9070,
3035/90.71 STS1_CELL_MAPPING = 9071,
3035/90.72 E4_140MB_G804 = 9072, 3035/90.73 DS3_45MB_G804 = 9073,
3035/90.74 E3_34MB_G804 = 9074, 3035/90.75 E1_2MB_G804 = 9075,
3035/90.76 DS1_1P5MB_G804 = 9076,
3035/90.77 VC3_CELL_MAPPING = 9077,
3035/90.90 SDH_SONET_COMP1 = 9090,
3035/90.91 SDH_SONET_COMP2 = 9091,
3035/90.93 SDH_SONET_COMP3 = 9093,
3035/91.01 V24_REMOTE = 9101, 3035/92.10 IEC_REMOTE = 9210

3035/90.60 TX_JITTER_UNTIL_STM1 = 9060,
3035/90.61 TX_JITTER_UNTIL_STM4 = 9061,
3035/90.65 RX_JITTER_UNTIL_STM1 = 9065,
3035/90.66 RX_JITTER_UNTIL_STM4 = 9066,
3035/90.67 TX_RX_WANDER_ANALYZER = 9067

Remote Control of Testcase

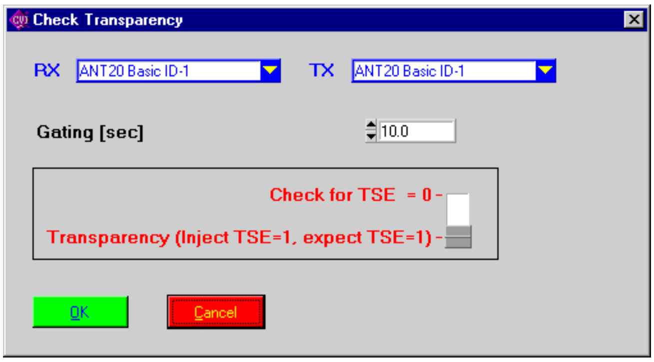
Input:	%d#%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d Id,Option Cnt,Delete Option,Option 1, .. ,Option 12
Output:	No Results

18 Check Transparency of Selected Tributary

Function Name

```
_check_transparency
            .\sdh_bas\sdh_bas.obj
```

User Interface



Description

Purpose: Checks the transparency of a tributary or channel that was defined by a previous signal structure and tributary setup testcase. TX and RX payload have to be identical, whereas the channel number and signal structure may be different.
Transparency (inject one TSE, expect one TSE) A PRBS is transmitted, then a BER measurement is started. One single bit error is injected right after opening the ANT-20 gating, and exactly one bit error is expected to come back.
Check for no TSE (TSE = 0) A PRBS is transmitted, then a BER measurement is started and no TSE (Bit error) is expected to come back.
To get more confidence into the transparency of a given channel, a gating up to 9999999 s can be set, however no more bit errors will be inserted after the first one. Number of bit errors and alarm time will be given as a result.

Caution: It is not advisable to use this testcase for monitoring applications. Special testcases are available for monitoring; these give a much more detailed view of the alarm and error situation during the test (see “check_bit_error”, “_check_alarm”).

PASS / FAIL Conditions

PASS: **Transparency (inject one TSE, expect one TSE)** Number of received TSE (bit errors) = 1 and alarm time = 0 seconds.
Check for no TSE (TSE = 0) Number of received TSE (bit errors) = 0 and alarm time = 0 seconds.

FAIL: **Transparency (inject one TSE, expect one TSE)** Number of received TSE (bit errors) <>1 or alarm time <> 0 seconds.
Check for no TSE (TSE = 0) Number of received TSE (bit errors) <> 0 or alarm time <> 0 seconds.

Test Report

```
Payl.-Transprncy        FAIL
   gating = 2.0 sec
   Transparency Check   : FAIL
   reason:
   LOF-2M               97.07.22 09:02:51.6 -0.2 [sec]
   LOF-2M               97.07.22 09:02:52.6 -0.2 [sec]
    ALARM SECONDS = 0.6
```

Applications

Test the transparency of a 64 k timeslot in a 2 Mb/s stream by transmitting and receiving a 2 Mb/s signal with a test signal in channel number 1.
Test the transparency of a 2 Mb/s unframed signal through an SDH ADM by transmitting a 2 Mb/s unframed signal and receiving an STM-1 C12 with unframed 2 Mb/s at tributary # x.

see i_List.squ or a_List.squ

Parameters

TX Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
Gating:	Double, 1.0 .. 99.0 [s]
Mode:	1= inject one TSE, 2= check for TSE = 0

Remote Control of Testcase

Input:	%d,%d#%lf,%d RX Id,TX Id,Gating,Mode
Output:	%d,%d,%d,%d Valid?,bit errors,Valid?,alarm sec. count Valid?: Valid = 1, Not valid = 0

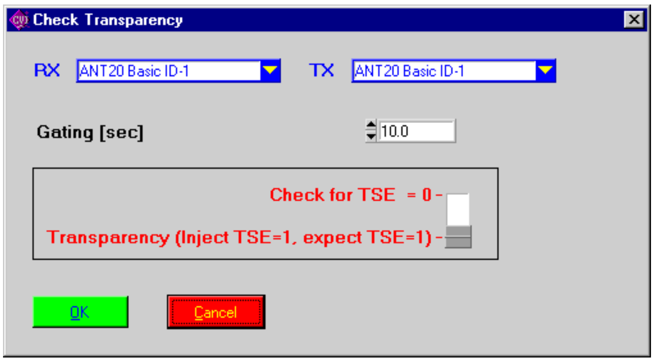
Notes:

19 Check Transparency of Selected SONET Tributary

Function Name

```
_check_sonet_transparency
            .\sdh_bas\sdh_bas.obj
```

User Interface



Description

Purpose: Checks the transparency of a tributary or channel that was defined by a previous signal structure and tributary setup testcase. TX and RX payload have to be identical, whereas the channel number and signal structure may be different.
Transparency (inject one TSE, expect one TSE) A PRBS is transmitted, then a BER measurement is started. One single bit error is injected right after opening the ANT-20 gating, and exactly one bit error is expected to come back.
Check for no TSE (TSE = 0) A PRBS is transmitted, then a BER measurement is started and no TSE (Bit error) is expected to come back.
To get more confidence into the transparency of a given channel, a gating up to 9999999 s can be set, however no more bit errors will be inserted after the first one. Number of bit errors and alarm time will be given as a result.

Caution: It is not advisable to use this testcase for monitoring applications. Special testcases are available for monitoring; these give a much more detailed view of the alarm and error situation during the test (see “check_bit_error”, “_check_alarm”).

PASS / FAIL Conditions

PASS: **Transparency (inject one TSE, expect one TSE)** Number of received TSE (bit errors) = 1 and alarm time = 0 seconds.
Check for no TSE (TSE = 0) Number of received TSE (bit errors) = 0 and alarm time = 0 seconds.

FAIL: **Transparency (inject one TSE, expect one TSE)** Number of received TSE (bit errors) <>1 or alarm time <> 0 seconds.
Check for no TSE (TSE = 0) Number of received TSE (bit errors) <> 0 or alarm time <> 0 seconds.

Test Report

```
Payl.-Transprncy        PASS
    gating = 2.0 sec
    Transparency Check  : PASS
```

Applications

see a_STS1.squ or i_C12.squ

Parameters

TX Id: Integer, 1 .. 10

RX Id: Integer, 1 .. 10

Gating: Double, 1.0 .. 99.0 [s]

Mode: 1= inject one TSE, 2= check for TSE = 0

Remote Control of Testcase

Input: %d,%d#%lf,%d
RX Id,TX Id,Gating, Mode

Output: %d,%d,%d,%d
Valid?,bit errors,Valid?,alarm sec. count
Valid?: Valid = 1, Not valid = 0

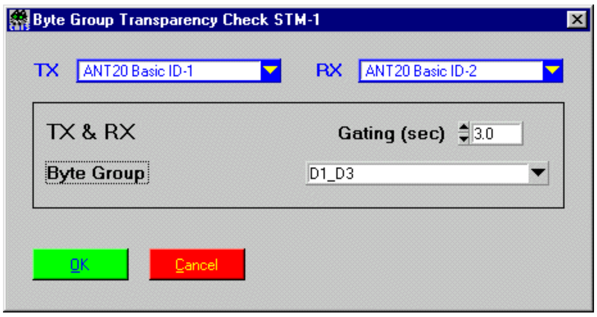
Notes:

20 Check Transparency of SDH DCCs, E1, F1, K1/K2

Function Name

```
_check_byte_group_transparency
            .\sdh_bas\sdh_bas.obj
```

User Interface



Description

Purpose: Automatically checks the transparency of the SDH SOH data communication channels D1...D3 or D4...D12. The transparency check is done by transmitting and receiving a PRBS, starting a BER measurement, injecting one single bit error right after opening the ANT-20 gating, and checking for exactly one bit error to come back.
To get more confidence into the transparency of the DCCs, a gating up to 9999 s can be set, however no more bit errors will be inserted after the first one.
Number of biterrors and alarm time will be given as a result. Testcase is PASS if number of received bit-errors = 1 and alarm time = 0 s; else FAIL.

Caution: –

PASS / FAIL Conditions

PASS: Number of received bit errors = 1 and alarm time = 0 sec.

FAIL: Number of received bit errors <>1 or alarm time > 0 sec.

Test Report

```
D1_D3 Transprncy        FAIL
   byte group D1-D3 not transparent: FAIL
   reason:
   Bit error count = 3336
```

Applications

–

Parameters

TX Id: Integer, 1 .. 10

RX Id: Integer, 1 .. 10

Selector: Enumeration
D1_D3 = 1, D4_D12 = 2

Gating: Double, 1.00 .. 99.00 [sec]

Remote Control of Testcase

Input: %d,%d#%d,%lf
RX Id,TX Id,Selector,Gating

Output: %d,%d,%d,%d
Valid?,bit errors,Valid?,alarm sec. count
Valid?: Valid = 1, Not valid = 0

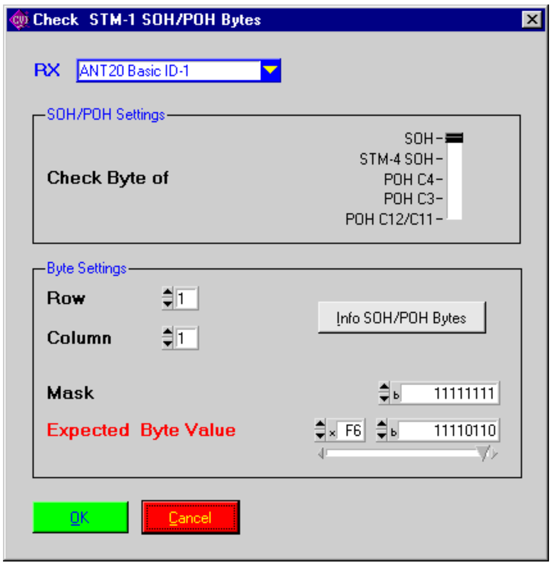
Notes:

21 Check Overhead Byte Against Expected Value

Function Name

```
_check_SOH_POH_byte
            .\sdh_bas\sdh_bas.obj
```

User Interface



Description

Purpose: Reads contents of one overhead byte and compares against an expected value. A mask can be set to exclude specific bits from the evaluation.

Caution: Leading 0's are not displayed in the mask.

PASS / FAIL Conditions

PASS: Masked byte value read matches the expected value.

FAIL: Masked byte value read does not match the expected value.

Test Report

```
Check C2=00h HP_UNE PASS
   C4 POH byte (row 3): measured value matches
```

Applications

–

Parameters

RX Id:	Integer, 1 .. 10
Selector:	Enumeration SOH = 1, C4 POH = 2, C3 POH = 3, C12/C11 POH = 4
Row:	Integer, 1 .. 9
Column:	Integer, 1 .. 36
Mask:	Hex, 00 .. FF A “1” ==> this byte posititon will be evaluated A “0” ==> this byte position will not be considered for evaluation
Exp. Value:	Hex, 00 .. FF

Remote Control of Testcase

Input:	%d#%d,%d,%d,%02X,%02X RX Id,Selector,Row,Column,Exp. Value,Mask
Output:	%s Read Byte Value (as HEX value)

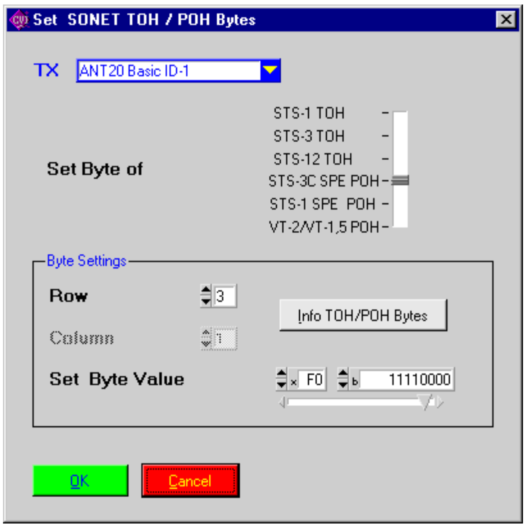
Notes:

22 Check SONET Overhead Byte Against Expected Value

Function Name

```
_check_sonet_TOH_POH_byte
            .\sdh_bas\sonetbas.obj
```

User Interface



Description

Purpose:	Reads contents of one overhead byte and compares against an expected value. A mask can be set to exclude specific bits from the evaluation.
Caution:	Leading 0's are not displayed in the mask.

PASS / FAIL Conditions

PASS:	Masked byte value read matches the expected value.
FAIL:	Masked byte value read does not match the expected value.

Test Report

```
Check C2=00h UNEQ_P PASS
    STS-3 C POH byte (row 3): measured value matches
```

Applications

–

Parameters

RX Id:	Integer, 1 .. 10
Selector:	Enumeration TOH = 1, STS-3C SPE POH = 2, STS-1 SPE POH = 3, VT POH = 4 STS-12 TOH = 5, STS-3 TOH = 6
Row:	Integer, 1 .. 9
Column:	Integer, 1 .. 36
Mask:	Hex, 00 .. FF A “1” ==> this byte posititon will be evaluated A “0” ==> this byte position will not be considered for evaluation
Exp. Value:	Hex, 00 .. FF

Remote Control of Testcase

Input:	%d#%d,%d,%d,%02X,%02X RX Id,Selector,Row,Column,Exp. Value,Mask
Output:	%s Read Byte Value (as HEX value)

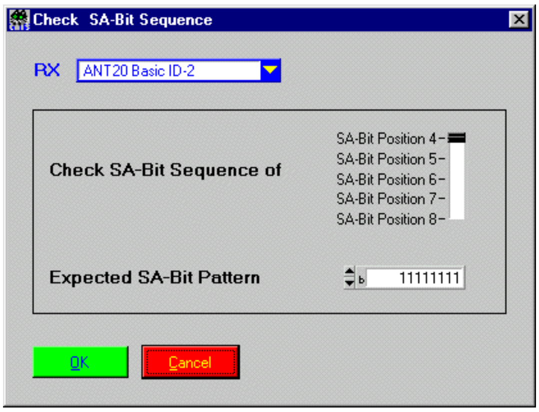
Notes:

23 Check 2 MBit/s Sa-Bit Sequence

Function Name

```
_check_sa_bit_sequence
            .\sdh_bas\sa_bits.obj
```

User Interface



Description

Purpose: Compares the bit sequence of one Sa-Bit received on RX site with an expected pattern.

Caution: –

PASS / FAIL Conditions

PASS: Expected pattern matches received bit sequence.

FAIL: Expected pattern does not match received bit sequence or results are not valid (due to alarms or invalid signal structure).

Test Report

```
check SA4 bit seq   PASS
    SA4-Bit Pattern  : 00001111
```

Applications

Check the transparency of a 2Mbit/s link for Sa-control information. See sequence “i_2Mbit.squ”.

Parameters

RX Id:	Integer, 1 .. 10
Sa-Bit Pos.:	Integer, 4 .. 8
Sa-Bit Seq.:	Integer, 0 .. 255

Remote Control of Testcase

Input:	%d#,%d,%d RX Id,Sa-Bit Pos.,Sa-Bit Seq.
Output:	%d Received Sa-Bit Sequence

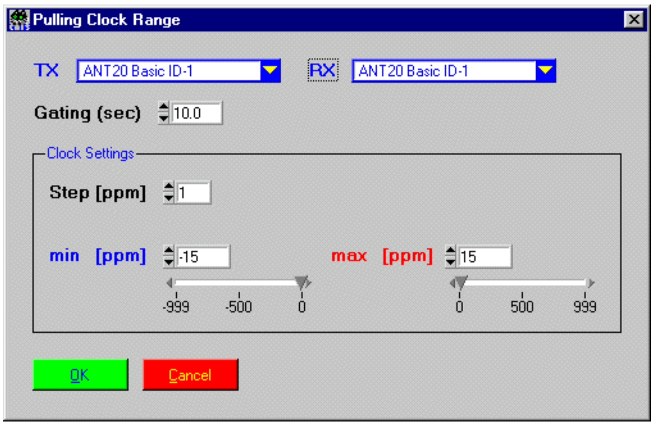
Notes:

24 Check Clock Pulling Range (BER vs. Offset)

Function Name

```
_check_clock_pulling_range
            .\sdh_bas\sdh_bas.obj
```

User Interface



Description

Purpose: Checks the pulling range of a Unit Under Test by transmitting a signal with an increasing clock offset and checking for bit errors and alarms during the measurement.
The algorithm starts at 0 ppm and continues at (step) ppm, (1*step) ppm, (2*step) ppm etc until (n*step) is higher than the maximum expected pulling range. A similar algorithm then takes place in negative direction.

Caution: To save time, you should select a step value that is slightly higher than the expected range, e.g. select step = 16 ppm if +/- 15 ppm is considered. In this case, only three measurements will take place (at -16 ppm, 0 ppm and +16 ppm).

PASS / FAIL Conditions

PASS: Neither bit errors nor alarms are found for any of the measurements.

FAIL: Bit errors or alarms are found for one of the measurements.

Test Report

```
Clock Pull. Range   FAIL

   step = 7 ppm, gating = 1.0 sec

   Expected pulling range:  -12 ppm .. 0 ppm

   Measured pulling range:    0 ppm .. 0 ppm
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
Min. Offset:	Integer, -999 .. 0 [ppm]
Max. Offset:	Integer, 0 .. 999 [ppm]
Step Width:	Integer, 0 .. 20 [ppm]
Gating:	Double, 1.0 .. 99.0 [sec]

Remote Control of Testcase

Input:	%d,%d#%d,%d,%d,%lf RX Id,TX Id,Min. Offset,Max. Offset,Step Width,Gating
Output:	%d,%d Measured Min. Offset [ppm],Measured Max. Offset [ppm]

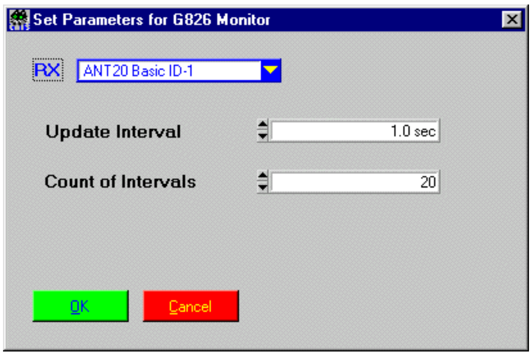
Notes:

25 Get Current G.826 Results (after G.826 Start)

Function Name

```
_monitor_g826
             .\sdh_bas\g826.obj
```

User Interface



Description

Purpose: Monitors the results of the G.826 evaluation.

Caution: –

PASS / FAIL Conditions

PASS: The user quits the testcase using the “Abort - Test PASS” button.

FAIL: The user quits the testcase using the “Abort - Test FAIL” button.

Test Report

```
Monitor of G826     PASS
```

Applications

See sequence “i_g826.squ”.

Parameters

RX Id: Integer, 1 .. 10

Update Int.: Enumeration
1 sec = 1.00, 2 sec = 2.00, 5 sec = 5.00, 10 sec = 10.00, 1 minute = 60.00

Update Cnt: Unsigned Integer, 1 .. 4294967295

Remote Control of Testcase

Input: %d#%lf,%ld
RX Id,Update Int.,Update Cnt.

Output: No Results.

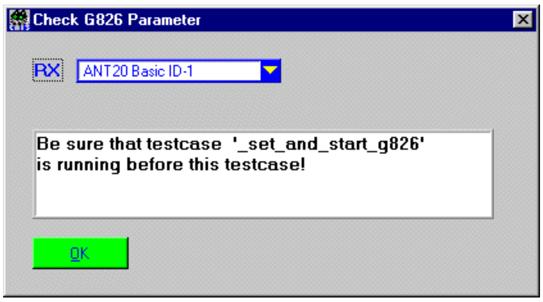
Notes:

26 Get Final G.826 Results (after G.826 Start)

Function Name

```
_check_g826
            .\sdh_bas\g826.obj
```

User Interface



Description

Purpose: Checks the results of the G.826 evaluation.

Caution: –

PASS / FAIL Conditions

PASS: The user quits the testcase using the “Abort - Test PASS” button, or status FarEnd and Status NearEnd are “OK”.

FAIL: The user quits the testcase using the “Abort - Test FAIL” button, or status FarEnd or Status NearEnd are “not OK”.

Test Report

```
check g826           PASS
   G826 Test, Gating: 200 sec, Allocation: 18.5 %
   Near End: PDH 2MBit/s - CRC - Far End: E-BIT
   Quality Near End       : rejected = FAIL
   EFS                    : 91
   EFS  Ratio             : 97.85 %
   EB                     : 0
   ES                     : 2
   ES  Ratio              : 2.15 %
   SES                    : 2
   SES Ratio              : 2.15 %
   BBE                    : 0
   BBE Ratio              : 0.00 %
   UAS                    : 0
   Quality Far End        : accepted = PASS
   EFS                    : 91
   EFS Ratio              : 100.00 %
   EB                     : 0
   ES                     : 0
   ES  Ratio              : 0.00 %
   SES                    : 0
   SES Ratio              : 0.00 %
   BBE                    : 0
   BBE Ratio              : 0.00 %
```

Applications

See sequence “i_g826.squ”.

Parameters

RX Id: Integer, 1 .. 10

Remote Control of Testcase

Input: %d#
RX Id

Output: %d,%d,{%s,%lf]
Status NearEnd,Status FarEnd,[result_id_string, result_value]

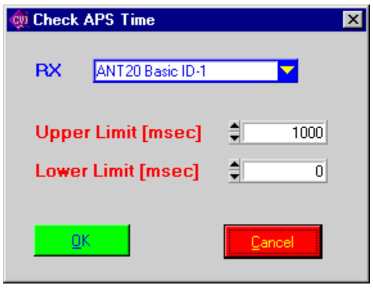
Notes:

27 Check APS Result (SDH and SONET)

Function Name

```
_check_APS
            .\sdh_bas\aps.obj
```

User Interface



Description

Purpose: Checks the APS time and compares it with the upper and lower limits.

Caution: For APS measurement ANT-20 Software (firmware) > 6.0 is required. Before you can check the APS time you have to initialize and start tge APS testcase "_set_and_start_APS" or "_set_and_start_sonet_APS"

PASS / FAIL Conditions

PASS: If the APS time result is within the user defined limits.

FAIL: APS result is not valid or beyond the user defined limits.

Test Report

```
check APS         PASS
    APS time [msec]  : 100
```

Applications

See sequence “i_aps.squ or a_aps.squ”.

Parameters

RX Id: Integer, 1 .. 10 corresponds to device name, interface adr .. of configuration window

Upper limit: *double* 0 .. 1 000 000 msec

Lower limit: *double* 0 .. 1 000 000 msec

Remote Control of Testcase

Input: %d#%lf,%lf
RX Id,Upper_limit,Lower Limit

Output: No Results.

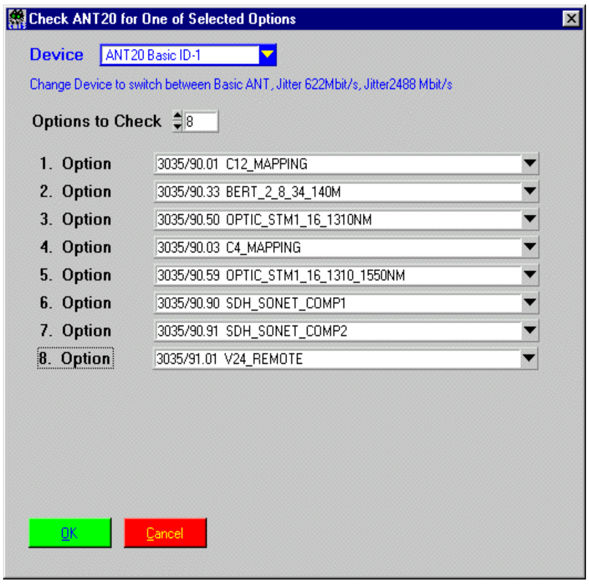
Notes:

28 Check Presence of a Group of Options (n of n)

Function Name

```
_check_options
            .\sdh_bas\options.obj
```

User Interface



Description

Purpose: Checks whether all of the listed ANT-20/ANX-920 options are present.

Caution: –

PASS / FAIL Conditions

PASS: All listed options are present.

FAIL: At least one of the listed options is missing.

Test Report

```
All Options         FAIL
   Option 3035/90.01 C12_MAPPING              available
   Option 3035/90.30 PDH_MUX_DEMUX_64K_140M  available
   Option 3035/90.43 OPTIC_STM1_1310NM       not available
   Option 3035/90.41 OPTIC_OC1_1550NM        not available
   Option 3035/90.31 PDH_DEMUX_64K_140M      not available
   Option 3035/91.01 V24_REMOTE              available
   Option 3035/92.10 IEC_REMOTE              available
```

Applications

Check for the presence of a specific set of options before starting a testcase that needs those options.

Parameters

Id: Integer, 1 .. 10

Option Cnt.: Integer, 1 .. 12

Option: Enumeration (up to 12)
3035/90.01 C12_MAPPING = 9001, 3035/90.02 C31_MAPPING = 9002,
3035/90.03 C4_MAPPING = 9003, 3035/90.04 C11_MAPPING = 9004,
3035/90.05 C32_MAPPING = 9005,
3035/90.10 VT1_5SPE_MAPPING = 9010,
3035/90.11 VT6SPE_MAPPING = 9011,
3035/90.12 STS1SPE_MAPPING = 9012,
3035/90.13 VT2SPE_MAPPING = 9012, 3035/90.20 DROP_INSERT = 9020,
3035/90.21 MIXED_MAPPING = 9021,
3035/90.22 LOCONC_VC11_VC12 = 9022,
3035/90.30 PDH_MUX_DEMUX_64K_140M = 9030,
3035/90.31 PDH_DEMUX_64K_140M = 9031,
3035/90.32 PDH_MUX_DEMUX_DS0_DS3 = 9032,
3035/90.33 BERT_2_8_34_140M = 9033,
3035/90.34 BERT_1M5_6M3_45M = 9034,
3035/90.43 OPTIC_STM1_1310NM = 9043,
3035/90.44 OPTIC_STM1_1550NM = 9044,
3035/90.45 OPTIC_STM1_1310_1550NM = 9045,
3035/90.46 OPTIC_STM1_4_1310NM = 9046,
3035/90.47 OPTIC_STM1_4_1550NM = 9047,
3035/90.48 OPTIC_STM1_4_1310_1550NM = 9048,
3035/90.50 OPTIC_STM1_16_1550NM = 9050,
3035/90.51 OPTIC_STM1_16_1310NM = 9051,
3035/90.53 OPTIC_STM1_16_1550NM NEW = 9053,
3035/90.54 OPTIC_STM1_16_1310NM NEW = 9054,
3035/90.59 OPTIC_STM1_16_1310_1550NM = 9059,
3035/90.70 ATM_MODUL = 9070, 3035/90.71 STS1_CELL_MAPPING = 9071,
3035/90.72 E4_140MB_G804 = 9072, 3035/90.73 DS3_45MB_G804 = 9073,
3035/90.74 E3_34MB_G804 = 9074, 3035/90.75 E1_2MB_G804 = 9075,
3035/90.76 DS1_1P5MB_G804 = 9076,
3035/90.77 VC3_CELL_MAPPING = 9077,
3035/90.90 OC-12C/STM-4C BER= 9090,
3035/90.91 OC-12C/STM-4C ATM= 9091,
3035/90.92 OC-12C/STM-4C Virtual Conc. = 9092,
3035/91.01 V24_REMOTE = 9101, 3035/92.10 IEC_REMOTE = 9210

3035/90.60 TX_JITTER_UNTIL_STM1 = 9060,
3035/90.61 TX_JITTER_UNTIL_STM4 = 9061,
3035/90.65 RX_JITTER_UNTIL_STM1 = 9065,
3035/90.66 RX_JITTER_UNTIL_STM4 = 9066,
3035/90.67 TX_RX_WANDER_ANALYZER = 9067

Remote Control of Testcase

Input: %d#%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d
Id,Option Cnt,Option 1, .. ,Option 12

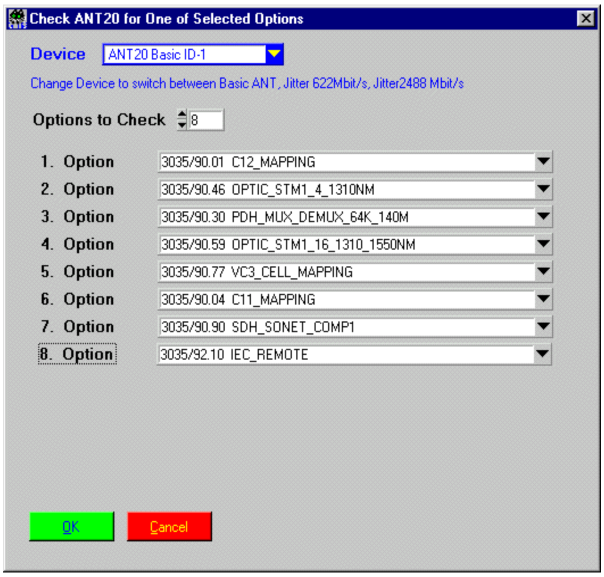
Output: %d{,%d,%d}
Option Cnt.,Option,Available?
Available?: Available = 1, Not available = 0

29 Check Presence of at Least One Option (1 of n)

Function Name

```
_check_options_one_of
            .\sdh_bas\option.obj
```

User Interface



Description

Purpose: Checks whether at least one option out of a group of possible ANT-20/ANX-920 options is present.

Caution: –

PASS / FAIL Conditions

PASS: At least one option is present.

FAIL: None of the listed options is available.

Test Report

```
One of Options       FAIL
    Option 3035/90.40   OPTIC_OC1_1310NM           not available
    Option 3035/90.41   OPTIC_OC1_1550NM           not available
```

Applications

Check for the presence of a specific set of options before starting a testcase that needs those options.

Parameters

Id: Integer, 1 .. 10

Option Cnt.: Integer, 1 .. 12

Option: Enumeration (up to 12)
3035/90.01 C12_MAPPING = 9001, 3035/90.02 C31_MAPPING = 9002,
3035/90.03 C4_MAPPING = 9003, 3035/90.04 C11_MAPPING = 9004,
3035/90.05 C32_MAPPING = 9005,
3035/90.10 VT1_5SPE_MAPPING = 9010,
3035/90.11 VT6SPE_MAPPING = 9011,
3035/90.12 STS1SPE_MAPPING = 9012,
3035/90.13 VT2SPE_MAPPING = 9012, 3035/90.20 DROP_INSERT = 9020,
3035/90.21 MIXED_MAPPING = 9021,
3035/90.22 LOCONC_VC11_VC12 = 9022,
3035/90.30 PDH_MUX_DEMUX_64K_140M = 9030,
3035/90.31 PDH_DEMUX_64K_140M = 9031,
3035/90.32 PDH_MUX_DEMUX_DS0_DS3 = 9032,
3035/90.33 BERT_2_8_34_140M = 9033,
3035/90.34 BERT_1M5_6M3_45M = 9034,
3035/90.40 OPTIC_OC1_1310NM = 9040,
3035/90.41 OPTIC_OC1_1550NM = 9041,
3035/90.42 OPTIC_OC1_1310_1550NM = 9042,
3035/90.43 OPTIC_STM1_1310NM = 9043,
3035/90.44 OPTIC_STM1_1550NM = 9044,
3035/90.45 OPTIC_STM1_1310_1550NM = 9045,
3035/90.46 OPTIC_STM1_4_1310NM = 9046,
3035/90.47 OPTIC_STM1_4_1550NM = 9047,
3035/90.48 OPTIC_STM1_4_1310_1550NM = 9048,
3035/90.50 OPTIC_STM1_16_1550NM = 9050,
3035/90.51 OPTIC_STM1_16_1310NM = 9051,
3035/90.53 OPTIC_STM1_16_1550NM NEW = 9053,
3035/90.54 OPTIC_STM1_16_1310NM NEW = 9054,
3035/90.59 OPTIC_STM1_16_1310_1550NM = 9059,
3035/90.70 ATM_MODUL = 9070, 3035/90.71 STS1_CELL_MAPPING = 9071,
3035/90.72 E4_140MB_G804 = 9072, 3035/90.73 DS3_45MB_G804 = 9073,
3035/90.74 E3_34MB_G804 = 9074, 3035/90.75 E1_2MB_G804 = 9075,
3035/90.76 DS1_1P5MB_G804 = 9076,
3035/90.77 VC3_CELL_MAPPING = 9077,
3035/90.90 OC-12C/STM-4C BER = 9090,
3035/90.91 OC-12C/STM-4C ATM = 9091,
3035/90.92 OC-12C/STM-4C Virtual Conc.= 9092,
3035/91.01 V24_REMOTE = 9101, 3035/92.10 IEC_REMOTE = 9210

3035/90.60 TX_JITTER_UNTIL_STM1 = 9060,
3035/90.61 TX_JITTER_UNTIL_STM4 = 9061,
3035/90.65 RX_JITTER_UNTIL_STM1 = 9065,
3035/90.66 RX_JITTER_UNTIL_STM4 = 9066,
3035/90.67 TX_RX_WANDER_ANALYZER = 9067

Remote Control of Testcase

Input: %d#%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d
Id,Option Cnt,Option 1, .. ,Option 12

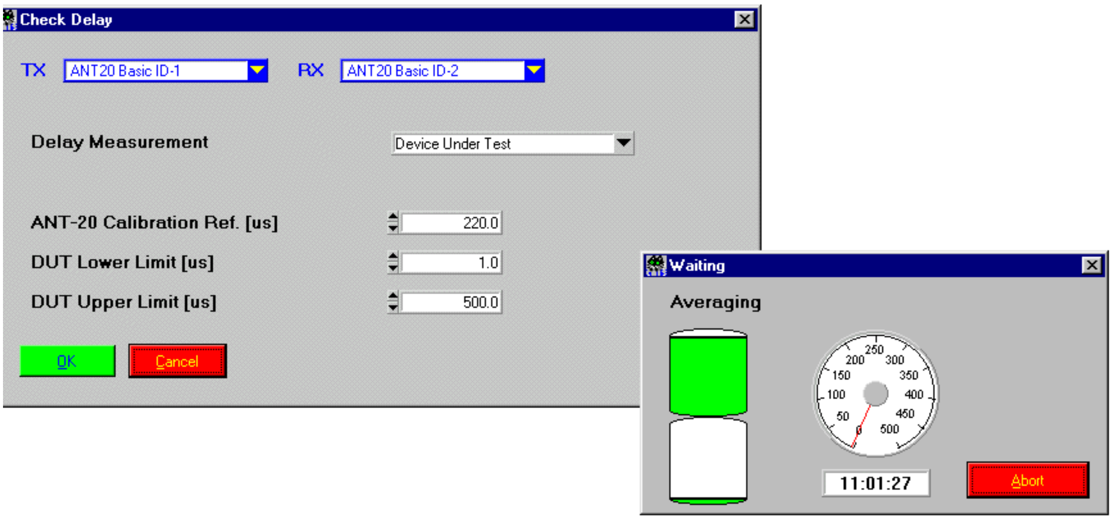
Output: %d{,%d,%d}
Option Cnt.,Option,Available?
Available?: Available = 1, Not available = 0

30 Check Round-Trip Pattern Delay

Function Name

```
_check_pattern_delay
             .\sdh_bas\delay.obj
```

User Interface



Description

Purpose: Check the delay of a signal through the network.

Caution: Always use this testcase with ANT-20 TX/RX loop cable to determine calibration reference. To carry out calibration measurement, select "Delay Measurement = Get ANT-20 Calibration Ref.". Signal setup for calibration measurement must be identical to the one used in "real" test.

PASS / FAIL Conditions

PASS: Delay result is between limits.

FAIL: Delay result not valid or out of limits.

Test Report

```
check delay                   FAIL
     Delay [us]               :   5441 > upper limit = 500
```

Applications

See sequence “i_c12.squ”.

Parameters

TX Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
Mode:	Enumeration Calibration =1, DUT = 2
Reference:	Double, 0.0 .. 10000000000.0 [us]
Upper Limit:	Double, -100000000000.0 .. 100000000000.0 [us]
Lower Limit:	Double, -100000000000.0 .. 100000000000.0 [us]

Remote Control of Testcase

Input:	%d,%d#%d,%lf,%lf,%lf TX Id,RX Id,Mode,Reference,Upper Limit,Lower Limit
Output:	%d,%g Valid?,Delay Valid: Valid = 1,Not valid = 0

Notes:

WWG-CATS Testcase Library

Monitoring Testcases & Scanner Testcases (for Pickering Scanner)

3

Monitoring Testcases & Scanner Testcases (for Pickering Scanner)

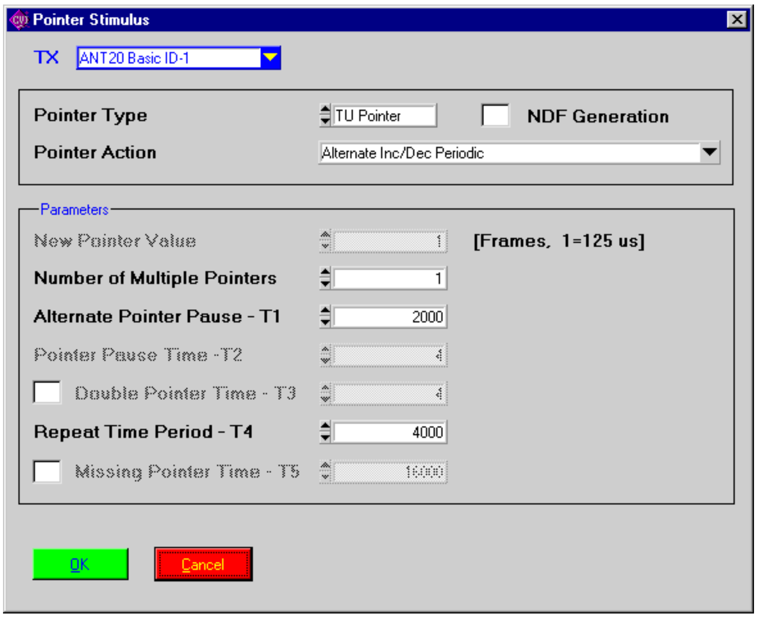
Notes:

1 Set SDH Pointer Actions and Sequences

Function Name

```
_set_ptr_action
            .\sdh_bas\ptr_move.obj
```

User Interface



Description

Purpose: Injects pointer movements or pointer sequences.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Initial Ptr Mov          PASS
    TU Pointer - Alternate Inc/Dec Periodic
    T1 = 2000

Regular + missing        PASS
    TU Pointer - Increment Periodic
    N = 40 T2 = 1500 T4 = 60000
```

Applications

see i_O_172.squ

Parameters

TX Id:	Integer, 1 .. 10
Pointer:	Enumeration TU Pointer = 1, AU Pointer = 2
Pointer Action:	Enumeration *#define PTR_CANCEL_ACTION -1* *#define PTR_NEW_PTR_VALUE 1* *#define PTR_INC 2* *#define PTR_DEC 3* *#define PTR_INC_PERIODIC 4* *#define PTR_DEC_PERIODIC 5* *#define PTR_ALTERNATE_INC_DEC 6* *#define PTR_ALTERNATE_INC_DEC_PERIODIC 7* *#define PTR_87_3_INC 8* *#define PTR_87_3_DEC 9* *#define PTR_87_3_INC_PERIODIC 10* *#define PTR_87_3_DEC_PERIODIC 11* *#define PTR_43_44_INC 12* *#define PTR_43_44_DEC 13* *#define PTR_43_44_INC_PERIODIC 14* *#define PTR_43_44_DEC_PERIODIC 15* *#define PTR_86_4_INC 16* *#define PTR_86_4_DEC 17* *#define PTR_86_4_INC_PERIODIC 18* *#define PTR_86_4_DEC_PERIODIC 19*
Pointer Value:	Unsigned Integer, 0 .. 782
ALternate Pause T1:	Unsigned Integer, 2 .. 4800000 [frms]
Pause T2:	Unsigned Integer, 2 .. 80000 [frms]
Repeat Time T4:	Unsigned Integer, 2 .. 4800000 [frms]
N-Multiple:	Unsigned Integer, 1.. 2000 [frms]
t3 Flag:	Unsigned Integer, ON=1, OFF=0
Double Pointer Time T3:	Unsigned Integer, 1.. 8000 [frms]
t5Flag:	Unsigned Integer, ON=1, OFF=0
Missing Pointer Time T5:	Unsigned Integer, 1.. 4800 000 [frms]
NDF Flag:	Unsigned Integer, ON=1, OFF=0

Remote Control of Testcase

Input: %d#%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d”
tx_id
pointer_type
pointer_action,pointer_value,time_t1,time_t2,time_t4,n_multiple,
t3_flag,time_t3,t5_flag,time_t5,ndf_flag

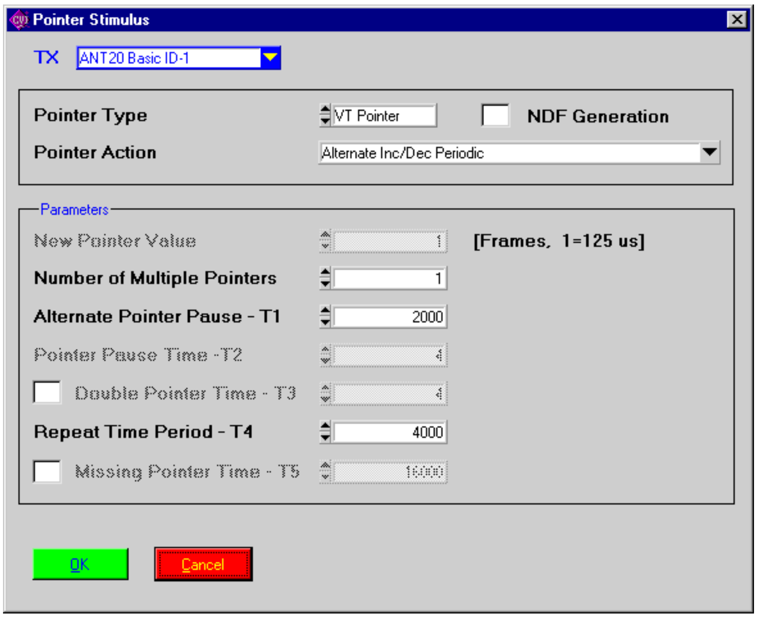
Output: No Results

2 Set SONET Pointer Actions and Sequences

Function Name

```
_set_sonet_ptr_action
            .\sdh_bas\ptr_move.obj
```

User Interface



Description

Purpose: Injects pointer actions.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Initial Ptr Mov        PASS
   VT Pointer - Alternate Inc/Dec Periodic
   T1 = 2000

Regular + missing      PASS
   VT Pointer - Increment Periodic
   N = 40 T2 = 1500 T4 = 60000
```

Applications

see i_O_172.squ

Parameters

TX Id:	Integer, 1 .. 10
Pointer:	Enumeration VT Pointer = 1, STS Pointer = 2
Pointer Action:	Enumeration *#define PTR_CANCEL_ACTION -1* *#define PTR_NEW_PTR_VALUE 1* *#define PTR_INC 2* *#define PTR_DEC 3* *#define PTR_INC_PERIODIC 4* *#define PTR_DEC_PERIODIC 5* *#define PTR_ALTERNATE_INC_DEC 6* *#define PTR_ALTERNATE_INC_DEC_PERIODIC 7* *#define PTR_87_3_INC 8* *#define PTR_87_3_DEC 9* *#define PTR_87_3_INC_PERIODIC 10* *#define PTR_87_3_DEC_PERIODIC 11* *#define PTR_43_44_INC 12* *#define PTR_43_44_DEC 13* *#define PTR_43_44_INC_PERIODIC 14* *#define PTR_43_44_DEC_PERIODIC 15* *#define PTR_86_4_INC 16* *#define PTR_86_4_DEC 17* *#define PTR_86_4_INC_PERIODIC 18* *#define PTR_86_4_DEC_PERIODIC 19*
Pointer Value:	Unsigned Integer, 0 .. 782
Alternate Pause T1:	Unsigned Integer, 2 .. 4800000 [frms]
Pause T2:	Unsigned Integer, 2 .. 80000 [frms]
Repeat Time T4:	Unsigned Integer, 2 .. 4800000 [frms]
N-Multiple:	Unsigned Integer, 1.. 2000 [frms]
t3 Flag:	Unsigned Integer, ON=1, OFF=0
Double Pointer Time T3:	Unsigned Integer, 1.. 8000 [frms]
t5Flag:	Unsigned Integer, ON=1, OFF=0
Missing Pointer Time T5:	Unsigned Integer, 1.. 4800 000 [frms]
NDF Flag:	Unsigned Integer, ON=1, OFF=0

Remote Control of Testcase

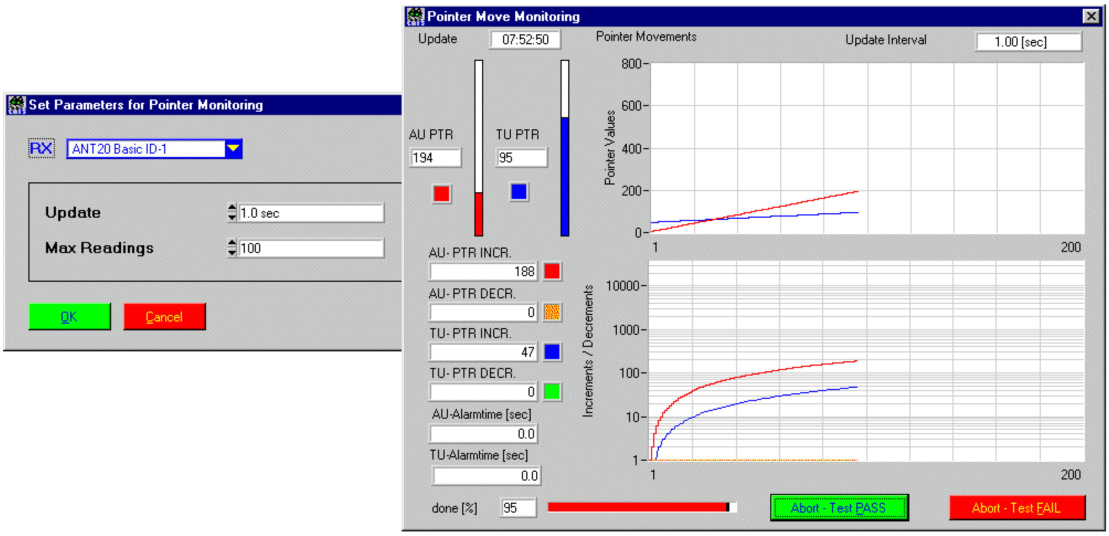
Input:	%d#%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d” tx_id pointer_type pointer_action,pointer_value,time_t1,time_t2,time_t4,n_multiple, t3_flag,time_t3,t5_flag,time_t5,ndf_flag
Output:	No Results

3 View SDH Pointer Movements vs. Time

Function Name

```
_ptr_move
            .\sdh_bas\ptr_move.obj
```

User Interface



Description

Purpose: Reads the value of an AU and/or TU pointer in fixed intervals and displays them in a graph.

Caution: No mesurements are done during the time gap in between sampling intervals, i.e. pointer movements that occur between two samples may get by undetected.

PASS / FAIL Conditions

PASS: User quits test case using the “Abort - Test PASS” button.

FAIL: User quits test case using the “Abort - Test FAIL” button.

Test Report

```
Pointer Monitoring  PASS
```

Applications

–

Parameters

RX Id: Integer, 1 .. 10

Update Int.: Enumeration
1 sec = 1.00, 10 sec = 10.00, 1 minute = 60.00, 5 minutes = 300.00,
15 minutes = 900.00, 1 hour = 3600.00

Update Cnt.: Unsigned Integer, 1 .. 100000000

Remote Control of Testcase

Input: %d#%lf,#--#,%ld
RX Id,Update Int.,Update Cnt.

Output: No Results

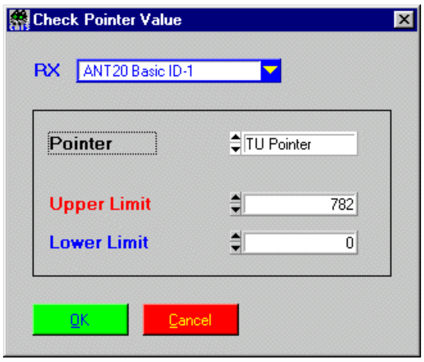
Notes:

4 Check Received SDH Pointer Against Expected Value

Function Name

```
_check_ptr_value
            .\sdh_bas\ptr_move.obj
```

User Interface



Description

Purpose: Compares a pointer value with given limits.

Caution: –

PASS / FAIL Conditions

PASS: Pointer value is between limits.

FAIL: Pointer value exceeds given limits.

Test Report

```
Check TU Pointer PASS
   TU Pointer - Value : 3
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
Pointer:	Enumeration TU Pointer = 1, AU Pointer = 2
Upper Limit:	Integer, 0 .. 782
Lower Limit:	Integer, 0 .. 782

Remote Control of Testcase

Input:	%d#%d,%d,%d RX Id,Pointer,Upper Limit,Lower Limit
Output:	%d,%d Valid?,Pointer Value Valid?: Valid = 1, Not valid = 0

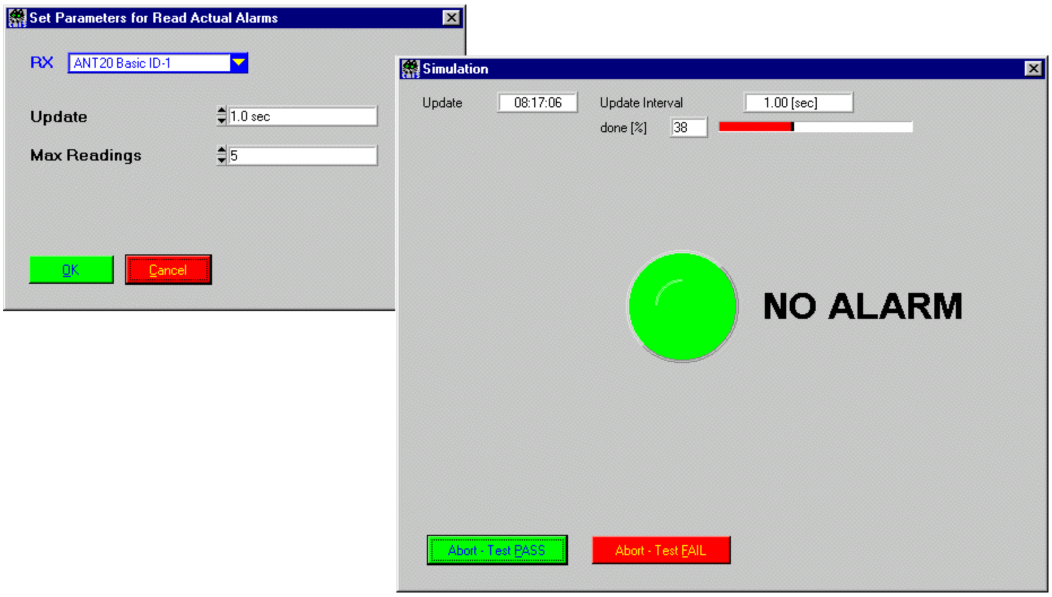
Notes:

5 Show Current Alarm Situation

Function Name

```
_read_actual_alarms
            .\sdh_bas\actalarm.obj
```

User Interface



Description

Purpose: Reads the current SDH/PDH alarms at the ANT-20 RX once or n times every m seconds (n=number of readouts, m=update interval). This testcase is meant to give a quick overview of the current alarm situation and is especially useful for debugging test sequences. It does not give any indication regarding the alarm situation in between alarm samples. The alarm readout value together with a timestamp is stored into a user-defined file.
This testcase doesn't have any automatic PASS/FAIL evaluation.

Caution: Do not use this testcase for surveillance measurements or monitoring applications! Neither errors nor alarms that occur during the gap between two samples will be recorded. For monitoring applications, special testcases are available that give a complete picture of the quality of a received signal (see "_check_bit_error", "_check_atm", etc.)

PASS / FAIL Conditions

PASS: The user quits the testcase using the "Abort - Test PASS" button.

FAIL: The user quits the testcase using the "Abort - Test FAIL" button.

Test Report

```
Monitor Alarm    PASS
```

Applications

–

Parameters

RX Id:	Integer, 1 .. 10
Update Int.:	Enumeration 1 sec = 1.00, 2 sec = 2.00, 5 sec = 5.00, 10 sec = 10.00, 1 minute = 60.00
Update Cnt.:	Unsigned Integer, 1 .. 100000000

Remote Control of Testcase

Input:	%d#%lf,#--#,%ld RX Id,Update Int.,Update Cnt.
Output:	%d,{%s,} Alarm Count,{Alarm Strings,}

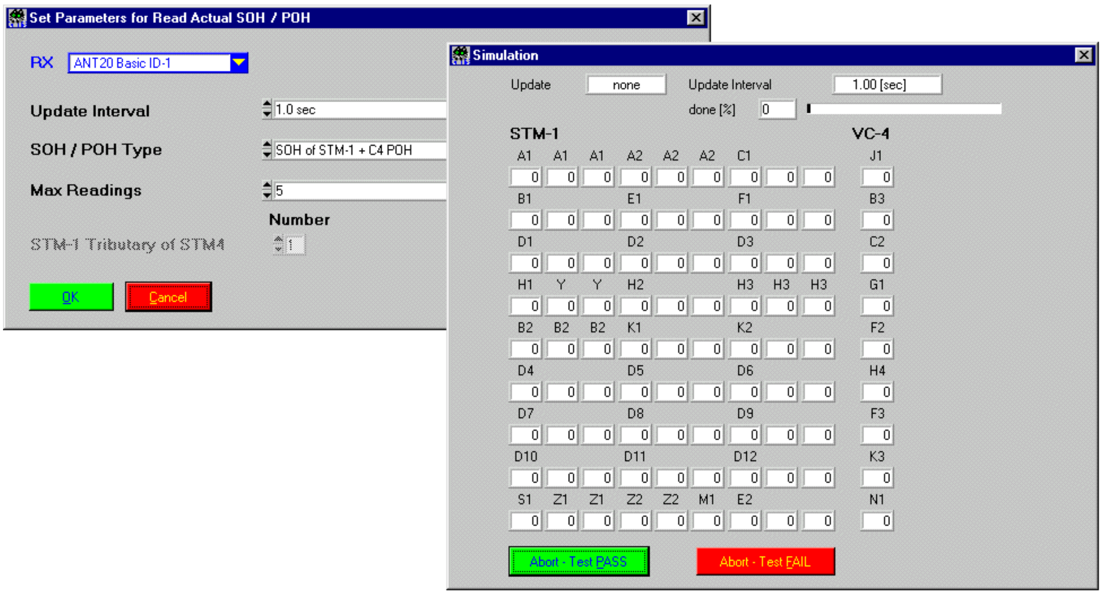
Notes:

6 Show Current SDH Overhead

Function Name

```
_read_actual_SOH_POH
            .\sdh_bas\soh_poh.obj
```

User Interface



Description

Purpose: Reads the complete SOH and POH of an STM-1 C4, C3 or C12 signal. Readout is done in n samples separated by m seconds of update interval time. The final readout will be displayed in the Sequence Display window and written into the ASCII report file.
This testcase is mainly meant to give a quick overview of the overhead; it doesn't have any automatic pass/fail evaluation.

Caution: Please be sure to set up the test for the mapping that you have actually set using the signal structure setup. TX/RX signal structure is NOT set automatically by this testcase, the relevant entries in the user interface only specify the number of bytes that have to be read in and displayed!

STM-4: A preceding testcase has to be used to set a tributary number different from #1.

STM-16: Only STM-1 #1 overhead is shown.

PASS / FAIL Conditions

PASS: The user quits the testcase using the "Abort - Test PASS" button.

FAIL: The user quits the testcase using the "Abort - Test FAIL" button.

Test Report

```
SOH/POH Monitor  PASS
    SOH of STM-1
                    POH C4   C12/C11
    F6,F6,F6,28,28,28,01,AA,AA,  50,   04,
    94,00,00,00,00,00,00,00,00,  29,   45,
    00,00,00,00,00,00,00,00,00,  02,   00,
    68,9B,9B,00,FF,FF,00,00,00,  00,   00,
    EE,2B,2B,00,00,00,00,00,00,  00,
    00,00,00,00,00,00,00,00,00,  FC,
    00,00,00,00,00,00,00,00,00,  00,
    00,00,00,00,00,00,00,00,00,  00,
    00,00,00,00,00,00,00,00,00,  00,
```

Applications

Get a quick information about current overhead bytes.

Parameters

RX Id:	Integer, 1 .. 10
Type:	Enumeration SOH of STM-1 + C4 POH + C12 POH = 100, SOH of STM-1 + C4 POH + C3 POH = 101, SOH of STM-1 + C4 POH= 102, SOH of STM-1 Tributary of STM-4 + C4 POH + C12 POH = 110, SOH of STM-1 Tributary of STM-4 + C4 POH + C3 POH = 111, SOH of STM-1 Tributary of STM-4 + C4 POH= 112, SOH of STM-1 Tributary of STM-16 + C4 POH + C12 POH = 113, SOH of STM-1 Tributary of STM-16 + C4 POH + C3 POH = 114, SOH of STM-1 Tributary of STM-16 + C4 POH= 115,
Update Int.:	Enumeration 1 sec = 1.00, 2 sec = 2.00, 5 sec = 5.00, 10 sec = 10.00, 1 minute = 60.00
Update Cnt.:	Unsigned Integer, 1 .. 100000000
STM-1 Trib.:	Integer, 1 .. 4

Remote Control of Testcase

Input:	%d#%lf,%d,%ld,%d RX Id,Update Int.,Type,Update Cnt.,STM-1 Trib.
Output:	%d,%s{,%d}{,%d}{,%d}{,%d} Valid?,Alarm String,SOH (81 or 324 bytes),C4 POH,C3 POH,C12 POH

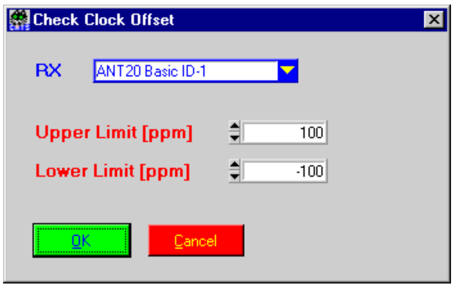
Notes:

7 Read Frequency Offset

Function Name

```
_read_rx_offset
            .\sdh_bas\monitor.obj
```

User Interface



Description

Purpose: Reads the actual clock offset and compares it against the defined limits.

Caution: This is a shnapshot result.

PASS / FAIL Conditions

PASS: If the clock offset result is within the limit.

FAIL: If the clock offset result is not within the limit or result is not valid.

Test Report

```
read_rx_offset   PASS
    freq. offset [ppm]  : 0
```

Applications

–

Parameters

RX Id:	Integer, 1 .. 10
Upper Limit:	double in ppm
Lower Limit:	double in ppm

Remote Control of Testcase

Input:	%d#%lf,%lf *rx_id, upper limit, lower limit*
Output:	%d,%d Valid?, Clock Offset[ppm] Valid?: Valid = 1, Not valid = 0

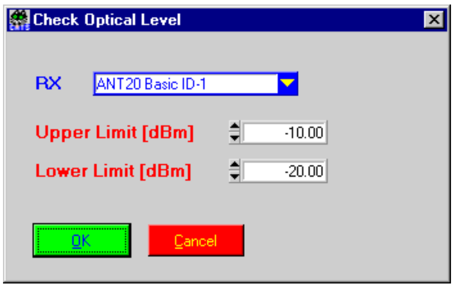
Notes:

8 Read Optical Level

Function Name

```
_read_rx_optical_level
            .\sdh_bas\monitor.obj
```

User Interface



Description

Purpose: Reads the actual optical level and compares it against the user-defined limits.

Caution: This is a shnapshot result.

PASS / FAIL Conditions

PASS: If the optical level is within the limit.

FAIL: If the optical level is not within the limit or result is not valid.

Test Report

```
read_rx_optical_level PASS
    freq. offset [dBm]  : -6
```

Applications

–

Parameters

RX Id: Integer, 1 .. 10

Upper Limit: double [dBm]

Lower Limit: double [dBm]

Remote Control of Testcase

Input: %d#%lf,%lf
rx_id,upper limit,lower limit

Output: %d,%d
Valid?,Optical Level[dBm]
Valid?: Valid = 1,Not valid = 0

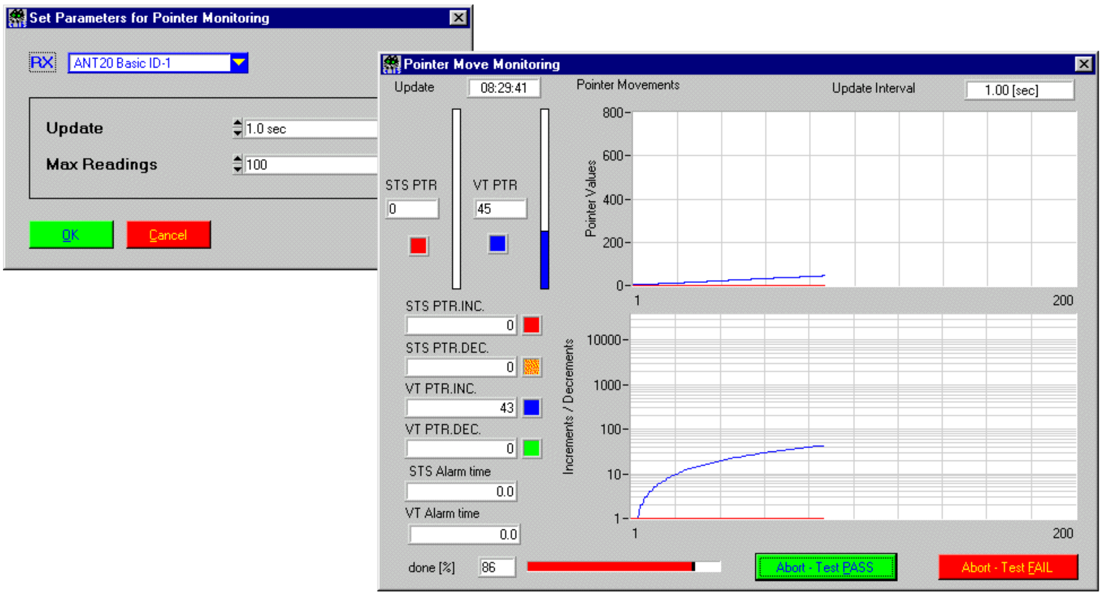
Notes:

9 View SONET Pointer Movements vs. Time

Function Name

```
_ptr_sonet_move
            c:\ ... \sdh_bas\ptr_move.obj
```

User Interface



Description

Purpose: Reads the value of an STS and/or VT pointer in fixed intervals and displays them in a graph.

Caution: No mesurements are done during the time gap in between sampling intervals, i.e. pointer movements that occur between two samples may get by undetected.

PASS / FAIL Conditions

PASS: User quits test case using the “Abort - Test PASS” button.

FAIL: User quits test case using the “Abort - Test FAIL” button.

Test Report

```
Pointer Monitoring  PASS
```

Applications

–

Parameters

RX Id: Integer, 1 .. 10

Update Int.: Enumeration
1 sec = 1.00, 10 sec = 10.00, 1 minute = 60.00, 5 minutes = 300.00,
15 minutes = 900.00, 1 hour = 3600.00

Update Cnt.: Unsigned Integer, 1 .. 100000000

Remote Control of Testcase

Input: %d#%lf,#--#,%ld
RX Id,Update Int.,Update Cnt.

Output: No Results.

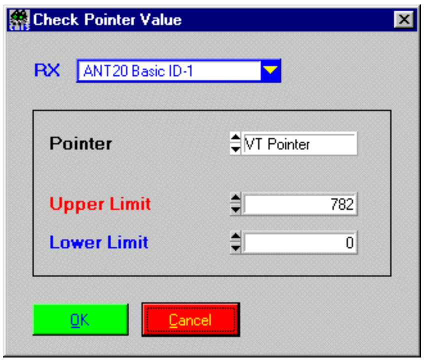
Notes:

10 Check Received SONET Pointer Against Expected Value

Function Name

```
_check_sonet_ptr_value
            .\sdh_bas\ptr_move.obj
```

User Interface



Description

Purpose: Compares a pointer value with given limits.

Caution: –

PASS / FAIL Conditions

PASS: Pointer value is between limits.

FAIL: Pointer value exceeds given limits.

Test Report

```
Check VT Pointer    PASS
   TU Pointer - Value : 3
```

Applications

–

Parameters

RX Id:	Integer, 1 .. 10
Pointer:	Enumeration TU Pointer = 1, AU Pointer = 2
Upper Limit:	Integer, 0 .. 782
Lower Limit:	Integer, 0 .. 782

Remote Control of Testcase

Input:	%d#%d,%d,%d RX Id,Pointer,Upper Limit,Lower Limit
Output:	%d,%d Valid?,Pointer Value Valid?: Valid = 1, Not valid = 0

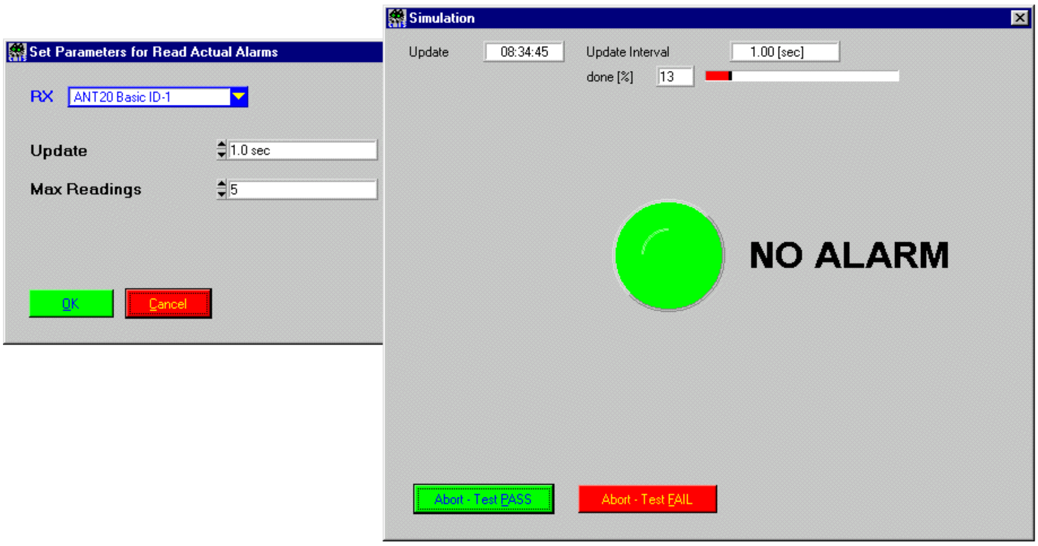
Notes:

11 Show Current SONET Alarm Situation

Function Name

```
_read_sonet_actual_alarms
            .\sdh_bas\actalarm.obj
```

User Interface



Description

Purpose: Reads the current SONET/PDH alarms at the ANT-20 RX once or n times every m seconds (n=number of readouts, m=update interval). This testcase is meant to give a quick overview of the current alarm situation and is especially useful for debugging test sequences. It does not give any indication regarding the alarm situation in between alarm samples. The alarm readout value together with a timestamp is stored into a user-defined file.
This testcase doesn't have any automatic PASS/FAIL evaluation.

Caution: Do not use this testcase for surveillance measurements or monitoring applications! Neither errors nor alarms that occur during the gap between two samples will be recorded. For monitoring applications, special testcases are available that give a complete picture of the quality of a received signal (see "_check_bit_error", "_check_atm", etc.).

PASS / FAIL Conditions

PASS: The user quits the testcase using the "Abort - Test PASS" button.

FAIL: The user quits the testcase using the "Abort - Test FAIL" button.

Test Report

```
Monitor Alarm       FAIL
```

Applications

–

Parameters

RX Id:	Integer, 1 .. 10
Update Int.:	Enumeration 1 sec = 1.00, 2 sec = 2.00, 5 sec = 5.00, 10 sec = 10.00, 1 minute = 60.00
Update Cnt.:	Unsigned Integer, 1 .. 100000000

Remote Control of Testcase

Input:	%d#%lf,#--#,%ld RX Id,Update Int.,Update Cnt.
Output:	%d,{%s,} Alarm Count,{Alarm Strings,}

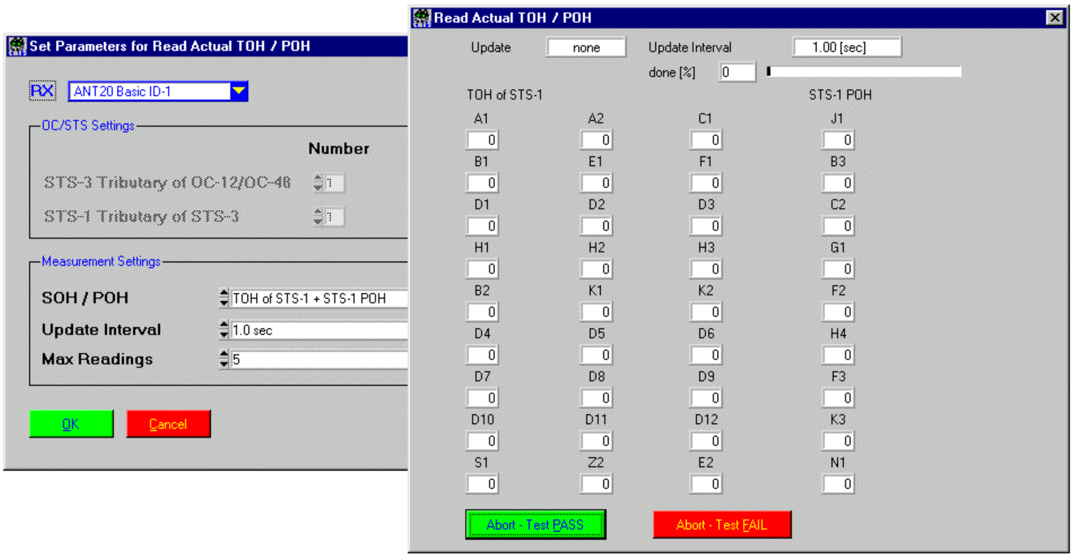
Notes:

12 Show Current SONET Overhead

Function Name

```
_read_actual_TOH_POH
            .\sdh_bas\tohpohso.obj
```

User Interface



Description

Purpose: Reads the complete SOH and TOH of an STS-1 or STS-3 signal. Readout is done in n samples separated by m seconds of update interval time. The final readout will be displayed in the Sequence Display window and written into the ASCII report file.
This testcase is mainly meant to give a quick overview of the overhead; it doesn't have any automatic pass/fail evaluation.

Caution: Please be sure to set up the test for the mapping that you have actually set using the signal structure setup. TX/RX signal structure is NOT set automatically by this testcase, the relevant entries in the user interface only specify the number of bytes that have to be read in and displayed!

PASS / FAIL Conditions

PASS: The user quits the testcase using the “Abort - Test PASS” button.

FAIL: The user quits the testcase using the “Abort - Test FAIL” button.

Test Report

```
TOH/POH Monitor    FAIL
    TOH of STS-1
                   POH STS-1, VT
    F6,28,01,      00,   84,
    68,00,00,      EE,   00,
    00,00,00,      02,   00,
    60,00,00,      00,   00,
    2F,00,00,      00,
    00,00,00,      FC,
    00,00,00,      00,
    00,00,00,      00,
    00,00,00,      00,
```

Applications

Get a quick information about current overhead bytes.

Parameters

RX Id: Integer, 1 .. 10

Type: Enumeration
TOH of STS-1 + STS-1 POH = 10,
TOH of STS-1 + STS-1 POH + VT POH = 11,
TOH of STS-3/OC-3 + STS-1 POH = 12,
TOH of STS-3/OC-3 + STS-1 POH + VT POH = 13,
TOH of STS-3/OC-3 + STS-3C POH = 14,
TOH of STS-12/OC-12 + STS-1 POH = 15,
TOH of STS-12/OC-12 + STS-1 POH + VT POH = 16,
TOH of STS-12/OC-12 + STS-3C POH = 17,
TOH of STS-48 + STS-1 POH = 18,
TOH of STS-48 + STS-1 POH + VT POH = 19,
TOH of STS-48 + STS-3C POH = 20

Update Int.: Enumeration
1 sec = 1.00, 2 sec = 2.00, 5 sec = 5.00, 10 sec = 10.00, 1 minute = 60.00

Update Cnt.: Unsigned Integer, 1 .. 100000000

STS-3 Trib.: Integer, 1 .. 16

STS-1 Trib.: Integer, 1 .. 3

Remote Control of Testcase

Input: %d#%lf,%d,%ld,%d,%d
RX Id,Update Int.,Type,Update Cnt.,STS-3 Trib.,STS-1 Trib.

Output: %d,%s{,%d}{,%d}{,%d}{,%d}
Valid?,Alarm String,TOH (81 or 324 bytes),STS-3C POH,STS-1 POH,VT POH

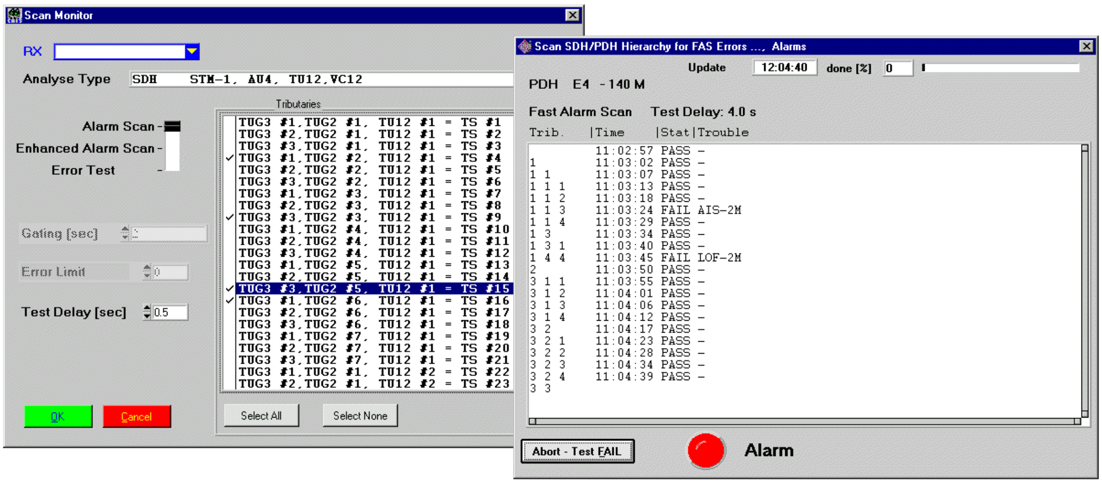
Notes:

13 Scan PDH/SDH Hierachies (TSE, BIP, FAS, Alarms)

Function Name

```
_scan_monitor
             .\sdh_bas\scan_mon.obj
```

User Interface



Description

Purpose: Mode “Fast Alarm Scan”:
Scans all selected VC12 tributaries and displays their current alarm situation. Alarms of STM-1 and VC4 as well as alarms on PDH 2Mbit/s level (2M-LOF, 2M-AIS, ...) are shown.

Mode “Error Test”:
Scans all selected VC12 tributaries and starts an error measurement. All error events as well as alarms that have occurred during the gating time will be displayed. For the VC12 hierarchy, only those errors and alarms related to the selected tributary will be shown.

Caution: As this is an in-service measurement TSE (test sequence errors = bit errors) and LSS alarm (loss of sequence synchronisation = no pattern) are not taken into consideration. This testcase does not set the RX signal structure itself (the testcase “_set_signal_structure” has to be used before). This also affects 2Mbit/s CRC or without CRC. If CRC errors/alarms have to be considered, set the RX signal structure to “PCM 30 CRC” otherwise use “PCM 30”.

PASS / FAIL Conditions

PASS: Mode “Fast Alarm Scan”:
No alarms found during scan.

Mode “Error Test”:
No alarms and errors found during error testing.

FAIL: Alarm or errors during scan or measurement.

Test Report

```
ALA MON - AU4 12 12 FAIL
   SDH    STM-1, AU4 .. TU12,VC12   - E1  - 2   MBits
   Fast Alarm Scan     test delay: 0.5 sec
   Trib.    |Time     |Stat|Trouble
   1,1,1= 1 16:45:36 PASS -
   2,1,1= 2 16:45:37 PASS -
   3,1,1= 3 16:45:38 PASS -
   1,2,1= 4 16:45:39 PASS -
   2,2,1= 5 16:45:40 PASS -
   3,2,1= 6 16:45:41 PASS -
   1,3,1= 7 16:45:42 PASS -
   2,3,1= 8 16:45:43 PASS -
   3,3,1= 9 16:45:44 PASS -
   1,4,1=10 16:45:45 PASS -
   2,4,1=11 16:45:46 PASS -
   3,4,1=12 16:45:47 PASS -
   1,5,1=13 16:45:48 PASS -
   2,5,1=14 16:45:49 PASS -
   3,5,1=15 16:45:50 PASS -
   1,6,1=16 16:45:51 PASS -
   2,6,1=17 16:45:52 PASS -
   3,6,1=18 16:45:53 PASS -
   1,7,1=19 16:45:54 PASS -
   2,7,1=20 16:45:55 PASS -
   3,7,1=21 16:45:56 PASS -
   1,1,2=22 16:45:57 PASS -
   2,1,2=23 16:45:58 FAIL TU-AIS
   3,1,2=24 16:45:59 PASS -
   1,2,2=25 16:46:00 PASS -
```

Applications

Pro-active monitoring and trouble shooting.

Parameters

RX Id:	Integer, 1 .. 10
Signal Type:	Enumeration STM-1 AU4 TU12 E1 = 6 (more Signal Types will be available in the future))
Op. Mode:	Enumeration Fast alarm scan = 1, Error Test = 2
Trib. Buffer:	String Contains up to 63 flags (comma separated) indicating whether a tributary should be tested or not. The contents off the buffer depends on the selected Signal Type.
Gating:	Double, 1 .. 10000000 [s]
Error Limit:	Double, 0 .. 4E+19
Test Delay:	Double, 0.0 .. 100.0 [s]

Remote Control of Testcase

Input:	%d#%lf,%d,%lf,%lf,%d,%s RX Id,Gating,Signal Type,Error Limit,Test Delay,Op. Mode,Trib. Buffer
Trib. Buffer:	{%d,} Flag Flag: channel selected = 1, channel not selected =0 Number of flags depends on Signal Type
Output:	0.{,%d,%d,%d,%d,%s,%s} T1,T2,T3,PassFail,Alarm String,Error String
T1:	STS1, E3, DS3, DS1, TUG3, AU3, 34M (depends on Signal Type), “0” ==> not relevant
T2:	TUG2,VTG, 8M (depends on Signal Type), “0” ==> not relevant
T3:	VT2, VT1.5, TU12, TU11, 2M, 1.5M (depends on Signal Type), “0” ==> not relevant

If mode “Fast Alarm Scan” is selected, there will be no Error String at the end of the output.

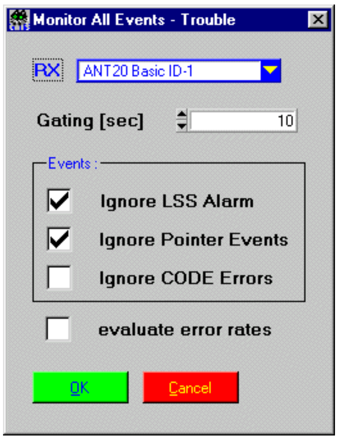
Notes:

14 Check All Events (TSE, BIP, FAS, Alarms ...)

Function Name

```
_check_trouble
            .\sdh_bas\trouble.obj
```

User Interface



Description

Purpose: Checks the ANT-20's event queue for events (alarms, errors, ...).

Caution: –

PASS / FAIL Conditions

PASS: No events detected.

FAIL: Event detected.

Test Report

```
Check All Events FAIL
   Gating = 30 s, Ignore LSS Alarm, Ignore Pointer Events
   Time         Trouble
   09:55:52     EBIT = 3
```

Applications

See sequence “i_c12.squ”.

Parameters

RX Id:	Integer, 1 .. 10
Gating:	Double, 1 .. 10000000 [s]
LSS Flag:	Enumeration Ignore LSS = 1, Don’t ignore LSS = 0
PTR Flag:	Enumeration Ignore Pointer = 1, Don’t ignore Pointer = 0

Remote Control of Testcase

Input:	%d#%lf,%d,%d RX Id,gating,LSS Flag,PTR Flag
Output:	%d,%d,%d,%.1lf,{%d,%.3e,} gAbortFlag, final_res_cnt, alarm_ON_cnt, sum_duration, ids[i], results[i]

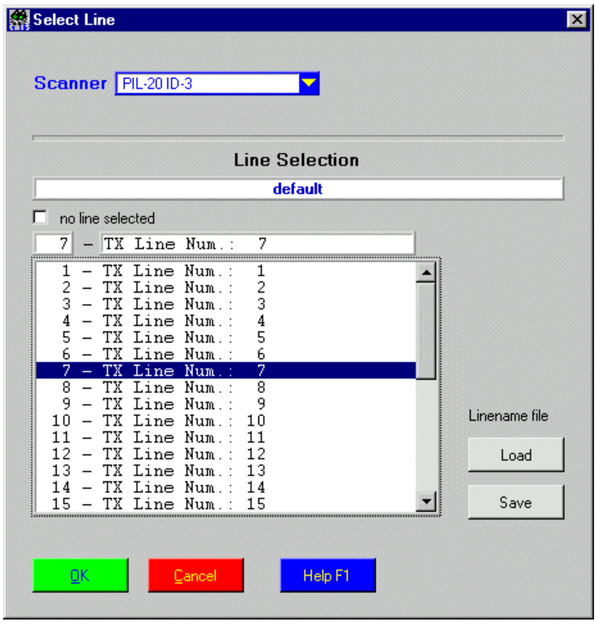
Notes:

15 Set Switch Position of Testpoint Selector

Function Name

```
_scan_select_channel
             .\scanner\scan.obj
```

User Interface



Description

Purpose: Switches a selected line with the attched scanner.

Caution: Only for Pickering Testpoint Selector.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
line 1          DC
   select line number   1
   HEL_TAM
   ->sub_channel: 1 on bank 0
```

Applications

–

Parameters

Scanner Id:	Integer, 1 .. 10
Line Id:	Integer, 1 .. xx
Line Descr.:	String
Descr. File:	String

xx depending on used Testpoint Selector (16,32,64,96).

Remote Control of Testcase

Input: %d#%d,#%s#,0,#%s#,0
Scanner Id,Line Id,Line Descr.,Descr. File

Output: No Results

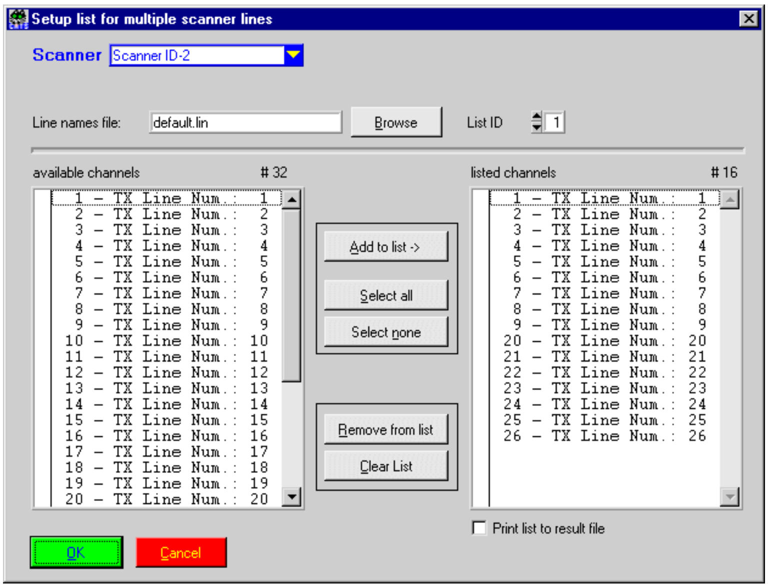
Notes:

16 Setup Channel List for Testpoint Selector

Function Name

```
_setup_mscan_ch
            .\scanner\mscan.obj
```

User Interface



Description

Purpose: Setup a list of channels witch will be set with the _msacn_set_inc_ch testcase. In combination with a GOTO this list can be used to scan several channels. (See application part for further information)

Caution: Only for Pickering Testpoint Selector.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Define channel-list   PASS
    setup  16 lines
    1 - TX Line Num.: 1
    2 - TX Line Num.: 2
    3 - TX Line Num.: 3
    4 - TX Line Num.: 4
    5 - TX Line Num.: 5
    6 - TX Line Num.: 6
    7 - TX Line Num.: 7
    8 - TX Line Num.: 8
    9 - TX Line Num.: 9
    20 - TX Line Num.:20
    21 - TX Line Num.:21
    22 - TX Line Num.:22
    23 - TX Line Num.:23
    24 - TX Line Num.:24
    25 - TX Line Num.:25
    26 - TX Line Num.:26
```

Applications

The combination of the testcases _setup_mscan_ch, _mscan_set_inc_ch and a GOTO allows a loop of measuring testcases over several channels.

One example is to check all or some channels connected to the testpoint selector.

Such a loop consist of three parts.

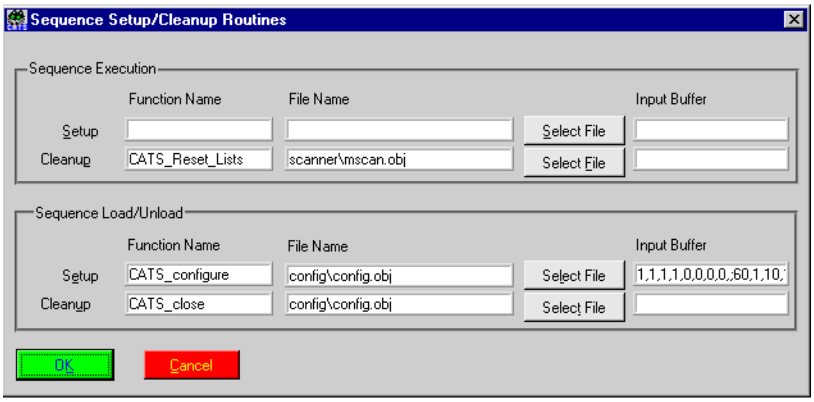
The first part is the loop setup. It is used to select the channels, which will be set during the loop execution. The setup is done by the _setup_mscan_ch testcase.

The second part is the loop frame. The frame consists of the channel selection (_mscan_set_inc_ch) and a GOTO statement which closes the loop to the channel selection. For the GOTO there must be set a precondition, otherwise the loop will never end. This precondition only enables the GOTO if the channel selection itself is FAIL. The reason for this is that the channel selection testcase is set to PASS if the end of the list is reached and FAIL if there are still some channels to select. Therefore the loop exit if the end of the list is reached which means that the channel selection testcase is PASS.

The third part is the loop body. It is enclosed by the loop frame. The body can include all available test for the selected channels.

See the selector.squ for a example of such a loop. For further information about the setting of preconditions and GOTO’s see the chapter 1.4 of this manual.

The example sequence also demonstrates the use of a setup/cleanup setting of the sequence. The CATS_reset_lists removes the list from memory and should therefore included in all sequences with the channel loop feature. (See figure below)



Parameters

Id:	Integer, 1 .. 10
Block-Id:	Integer, 1 .. 4
Sync. Switch:	Enumeration OFF = 0, ON = 1
Sel. Ch. #:	Integer, 1.. max. available channels
Ch. List-Id:	Integer, 1 .. 10
Print report:	Enumeration OFF = 0, ON = 1
Linename file:	String Filename
Selected Ch.1:	Integer, 1 .. max. available channels
....	
Selected Ch.x:	Integer, 1 .. max. available channels

Remote Control of Testcase

Input: %d#%d,%d,%d,%d,%d,#%s#,%d .. %d
Id,Block-Id,Synchronous Switching,Selected Channels Number, Channel List-Id,Print List to Report,Linename File,Selected Channel1, ..,Selected Channelx

Output: No Results

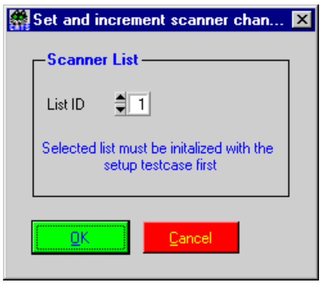
x : Selected channels number

17 Set & Increment Number of Channel List

Function Name

```
_mscan_set_inc_ch
            .\scanner\mscan.obj
```

User Interface



Description

Purpose: Selects next channel from list witch is set with the _setup_mscan_trib testcase. If the end of the list is reached the ‘no new channel’ will be set.

Caution: Only for Pickering Testpoint Selector.
The _setup_mscan_trib tescase must be executed in front.

PASS / FAIL Conditions

PASS: End of list is reached.

FAIL: New channel selected.

Test Report

```
Set channel from listFAIL
    set:   1 - TX Line Num.:  1
```

Applications

-

Parameters

Id:	Integer, 1 .. 10

Remote Control of Testcase

Input:	%d# Id
Output:	No Results

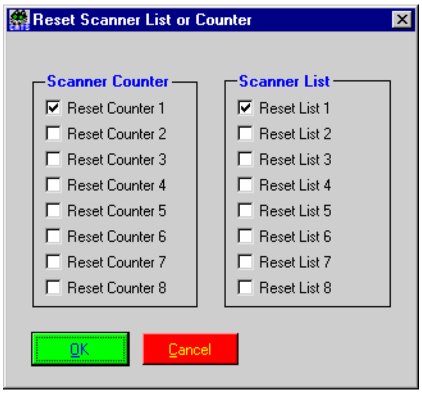
Notes:

18 Reset Channel List or Counter

Function Name

```
_mscan_reset_list
            .\scanner\mscan.obj
```

User Interface



Description

Purpose: Reset a list or counter of a channel list.

Caution: Only for Pickering Testpoint Selector.
The _setup_mscan_ch testcase must be executed in front.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Reset              PASS
    Reset scanner counter #1
    Clear scanner list #1
```

Applications

-

Parameters

List 1:	Enumeration ON = 1, OFF = 0
...	
List 8:	Enumeration ON = 1, OFF = 0
Counter 1:	Enumeration ON = 1, OFF = 0
...	
Counter 8:	Enumeration ON = 1, OFF = 0

Remote Control of Testcase

Input:	%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d, List 1,List 2,List 3,List 4,List 5,List 6,List 7,List 8, Counter 1,Counter 2,Counter 3,Counter 4,Counter 5,Counter 6,Counter 7,Counter 8
Output:	No Results

Notes:

WWG-CATS Testcase Library

Alarm Sensor Testcases

Alarm Sensor Testcases

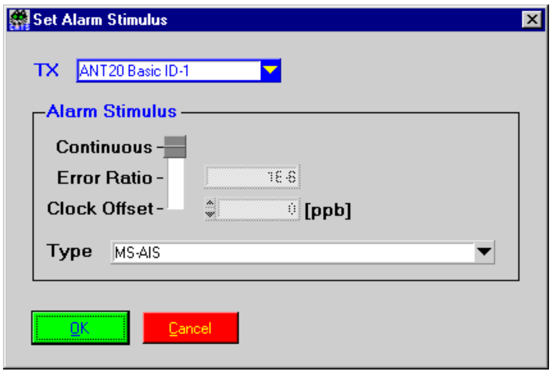
Notes:

1 Insert ITU-T Alarm

Function Name

```
_set_alarm
          .\al_sens\al_sens.obj
```

User Interface



Description

Purpose: Inserts an alarm until switched off by next testcase. To define alarm time, this testcase can be combined with the “_wait” testcase. To switch off an alarm, “_set_alarm” has to be called again with the stimulus set to “Switch off alarm insertion”.

Caution: —

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Set  Alarm                        PASS
     LOS inserted
```

Applications

G.826 stimulus. An alarm scenario can be defined using this testcase together with the “_wait” and “_set_stimulus” testcases.

Parameters

TX Id:	Integer, 1 .. 10
Stimulus:	Enumeration No Alarm Stimulus = 0, All Alarm Insertions OFF = 199, All Error Insertions OFF = 99, Clock Offset = 0, LOS = 700, LOF-STM1 = 718, RS-TIM = 740, MS-AIS = 706, MS-RDI = 707, AU-LOP = 701, AU-AIS = 704, TU-AIS = 705,TU-LOP = 702, TU-LOM = 703, HP-PLM = 741, HP-RDI = 708, HP-RDI-C = 748, HP-RDI-P = 746, HP-RDI-S = 747, HP-TIM = 719, HP-UNEQ = 710, LP-PLM = 742, LP-RDI = 709, LP-RDI-C = 751, LP-RDI-P = 749, LP-RDI-S = 750, LP-TIM = 720, LP-UNEQ = 711, LP-RFI = 743, AIS-DS1 = 712, AIS-2M = 713, AIS-8M = 714, AIS-34M = 715, AIS-DS3 = 717, AIS-140M = 716, LOF-DS1 = 723, LOF-2M = 724, LOF-8M = 725, LOF-34M = 726, LOF-DS3 = 727, LOF-140M = 728, RDI-DS1 = 729, RDI-2M = 730, RDI-8M =731, RDI-34M = 732, RDI-DS3 = 733, RDI-140M = 734, IDEL-DS3 = 735 TSE (Bit) Error Ratio = 200, CRC-4 Error Ratio = 204, CRC-6 Error Ratio = 244, FAS-1,5M (FE-DS1) Error Ratio = 232, FAS-2M (FE-E1) Error Ratio = 206, FAS-8M (FE-E2) Error Ratio = 208, FAS-34M (FE-E3) Error Ratio = 210, FE-DS3 (FAS-45M) Error Ratio = 242, FE-140M (FAS-E4) Error Ratio = 212, E-Bit Error Ratio = 214, B1-(MS-BIP) Error Ratio = 216, B2 Error Ratio = 218, B3-(HP BIP) Error Ratio = 220, MS-REI Error Ratio = 222, HP-REI Error Ratio = 224, LP-REI Error Ratio = 226, LP-BIP Error Ratio = 228
Ratio:	Double, 1E-12 .. 1E-2
Offset:	Double, -500000 .. 500000 [ppb]

Remote Control of Testcase

Input:	%d#%d,0,%lf TX Id,Stimulus,Not used,Ratio / Offset
Output:	No Results

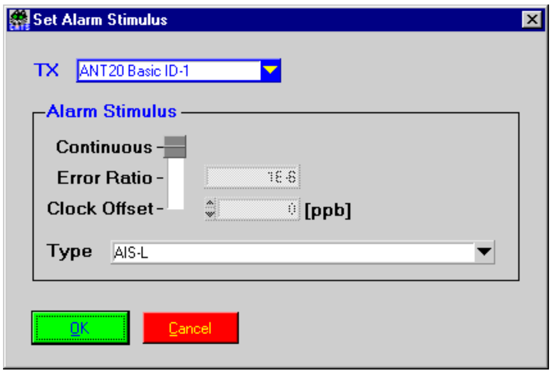
Notes:

2 Insert SONET Alarm

Function Name

```
_set_sonet_alarm
               .\al_sens\al_sens.obj
```

User Interface



Description

Purpose: Inserts an alarm until switched off by next testcase. To define alarm time, this testcase can be combined with the “_wait” testcase. To switch off an alarm, “_set_alarm” has to be called again with the stimulus set to “Switch off alarm insertion”.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Set  Alarm                    PASS
     LOS inserted
```

Applications

Performance monitoring, G.826 stimulus. To generate an alarm scenario for testing performance monitoring sensors.

Parameters

TX Id: Integer, 1 .. 10

Stimulus: Enumeration
No Alarm Stimulus = 0, All Alarm Insertions OFF = 199,
All Error Insertions OFF = 99, Clock Offset = 0, LOS = 700, LOF-STS = 718,
AIS-L = 805, RDI-L = 806, TIM-L = 820, AIS-P = 803, LOP-P = 800,
PLM-P = 821, RDI-P = 807, RDI-C-P = 826, RDI-P-P = 824, RDI-S-P = 825,
TIM-P = 811, UNEQ-P = 809, PDI-P = 744, AIS-V = 804, LOP-V = 801,
LOM-V = 802, PLM-V = 822, RDI-V = 808, RDI-C-V = 829, RDI-P-V = 827,
RDI-S-V = 828, RFI-V = 823, TIM-V = 812, UNEQ-V = 810, PDI-V = 745,
AIS-DS1 = 712, AIS-2M = 713, AIS-DS3 = 717, LOF-DS1 = 732,
LOF-2M = 724, LOF-DS3 = 727, RDI-DS1 = 729, RDI-2M = 730,
RDI-DS3 = 733, IDLE-DS3 = 735
TSE (Bit) Error Ratio = 200, CRC-4 Error Ratio = 204,
CRC-6 Error Ratio = 244, FE-DS1 (FAS-1,5M) Error Ratio = 232,
FE-E1 (FAS-2M) Error Ratio = 206, FE-DS3 (FAS-45M) Error Ratio = 242,
E-Bit Error Ratio = 214, DS3 C-Parity Ratio = 236, DS3 Parity Ratio = 238,
DS3 MFAS-45M Error Ratio = 240,
B1-BIP Error Ratio = 216, B2 Error Ratio = 218, B3-BIP Error Ratio = 220,
REI-L Error Ratio = 222, REI-P Error Ratio = 224, REI-V Error Ratio = 226,
BIP-V Error Ratio = 228

Ratio: Double, 1E-12 .. 1E-2

Offset: Double, -500000 .. 500000 [ppb]

Remote Control of Testcase

Input: %d#%d,0,%lf
TX Id,Stimulus,not used,Ratio / Offset

Output: No Results

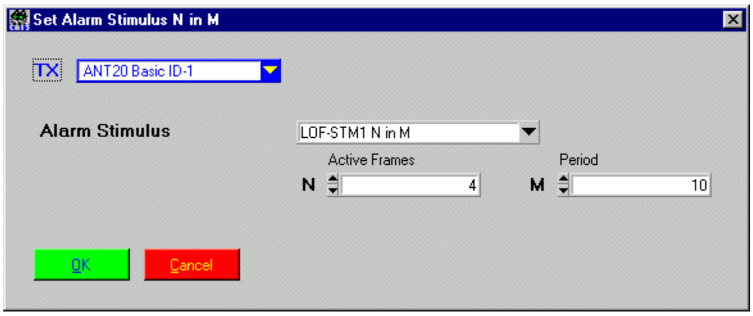
Notes:

3 Insert Dynamic ITU-T Alarm

Function Name

```
_set_alarm_N_in_M
              .\al_sens\al_dyna.obj
```

User Interface



Description

Purpose: Inserts specific alarm conditions for a specific number of frames, followed by a specific number of non-alarm frames. This sequence is repeated periodically until switched off by calling this testcase with the parameter “all alarm insertions off”.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Set Alarm N in M                PASS
     N: 7 in M: 10 inserted
```

Applications

–

Parameters

TX Id: Integer, 1 .. 10

Stimulus: Enumeration
No Stimulus = 0, All Alarm Insertions OFF = 199, LOF-STM1 N in M = 8, AU-LOP N in M = 10, TU-LOP N in M = 18, TU-LOM N in M = 17, AU-AIS N in M = 7, TU-AIS N in M = 19, MS-AIS N in M = 6, MS-RDI N in M = 9, HP-RDI N in M = 11, LP-RDI N in M = 20, HP-RDI-EP N in M = 82, HP-RDI-ES N in M = 83, HP-RDI-EC N in M = 84, LP-RDI-EP N in M = 85, LP-RDI-ES N in M = 86, LP-RDI-EC N in M = 87, LP-RFI N in M = 81, HP-UNEQ N in M = 12, LP-UNEQ N in M = 21, LOF-DS1 2 in 4 = 65, LOF-DS1 2 in 5 = 65, LOF-DS1 2 in 6 = 65, LOF-2M N in M = 27, LOF-8M N in M = 26, LOF-34M N in M = 25, LOF-DS3 2 in 2 = 69, LOF-DS3 2 in 3 = 69, LOF-DS3 3 in 3 = 69, LOF-DS3 3 in 15 = 69, LOF-DS3 2 in 16 = 69, LOF-DS3 3 in 17 = 69, LOF-140M N in M = 24, RDI-2M N in M = 33, RDI-8M N in M = 32, RDI-34M N in M = 31, RDI-140M N in M = 22, UNEQ-140M N in M = 97, UNEQ-140M N in M = 98, ATM LCD N in M = 99, ATM AIS_V_P N in M = 100, ATM RDI_V_P N in M = 101, ATM AIS_V_C N in M = 102, ATM RDI_V_C N in M = 103, ATM AIS_V_2 N in M = 104, ATM RDI_V_2 N in M = 105, PLCP LOF N in M = 106, PLCP RAI N in M = 107

N: Integer, 1 .. 4800000

M: Integer, 1 .. 4800000

Remote Control of Testcase

Input: %d#%d,0,%ld,%ld
TX Id,Stimulus,N,M

Output: No Results

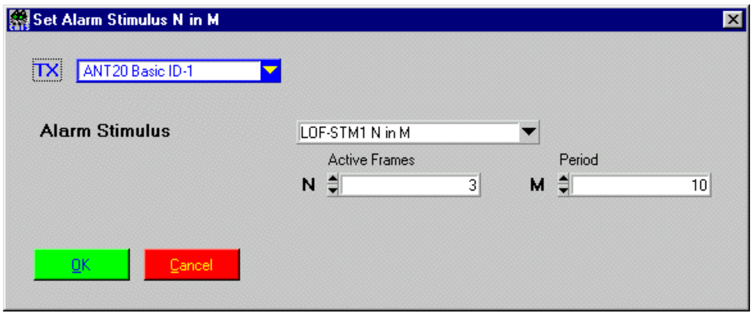
Notes:

4 Insert Dynamic SONET Alarm

Function Name

```
_set_sonet_alarm_N_in_M
               .\al_sens\al_dyna.obj
```

User Interface



Description

Purpose: Inserts specific alarm conditions for a specific number of frames, followed by a specific number of non-alarm frames. This sequence is repeated periodically until switched off by calling this testcase with the parameter “all alarm insertions off”.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Set Alarm N in M                PASS
      N: 7 in M: 10 inserted
```

Applications

–

Parameters

TX Id: Integer, 1 .. 10

Stimulus: Enumeration
No Stimulus = 0, All Alarm Insertions OFF = 199, LOF-STS N in M = 40, LOP-P N in M = 45, LOP-V N in M = 46, LOM-V N in M = 59, AIS-P N in M = 42, AIS-V N in M = 44, AIS-L N in M = 43, RDI-L N in M = 49, RDI-P N in M = 50, RDI-V N in M = 51, UNEQ-P N in M = 54, UNEQ-V N in M = 55, RDI-EPP N in M = 91, RDI-EPS N in M = 92, RDI-EPC N in M = 93, RDI-EVP N in M = 94, RDI-EVS N in M = 95, RDI-EVC N in M = 96, RFI-V N in M = 60, LOF-DS1 2 in 4 = 65, LOF-DS1 2 in 5 = 65, LOF-DS1 2 in 6 = 65, LOF-2M N in M = 27, LOF-DS3 2 in 2 = 69, LOF-DS3 2 in 3 = 69, LOF-DS3 3 in 3 = 69, LOF-DS3 3 in 15 = 69, LOF-DS3 2 in 16 = 69, LOF-DS3 3 in 17 = 69, LOF-140M N in M = 24, RDI-2M N in M = 33, RDI-140M N in M = 22, UNEQ-140M N in M = 98, ATM LCD N in M = 99, ATM AIS_V_P N in M = 100, ATM RDI_V_P N in M = 101, ATM AIS_V_C N in M = 102, ATM RDI_V_C N in M = 103, ATM AIS_V_2 N in M = 104, ATM RDI_V_2 N in M = 105, PLCP LOF N in M = 106, PLCP RAI N in M = 107

N: Integer, 1 .. 4800000

M: Integer, 1 .. 4800000

Remote Control of Testcase

Input: %d#%d,0,%ld,%ld
TX Id,Stimulus,N,M

Output: No Results

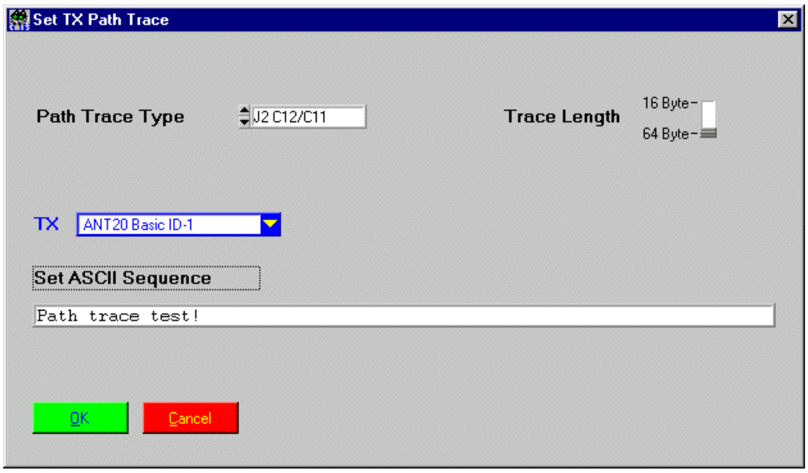
Notes:

5 Insert SDH Path Trace

Function Name

```
_set_path_trace
                c:\ ... \al_sens\al_sens.obj
```

User Interface



Description

Purpose: Inserts a user defined path trace.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Set Path Trace                    PASS
     J2 C12/C11:
     TX setting  <Path trace test!>
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
Type:	Enumeration J0 = 1, J1 C4 = 2, J1 C3 = 3, J2 C12/C11 = 4
Length:	Enumeration 16 Byte = 15, 64 Byte = 64
Inserted P.T.:	String

Remote Control of Testcase

Input:	%d#%d,%s,%d TX Id,Type,Inserted P.T.,Length
Output:	No Results

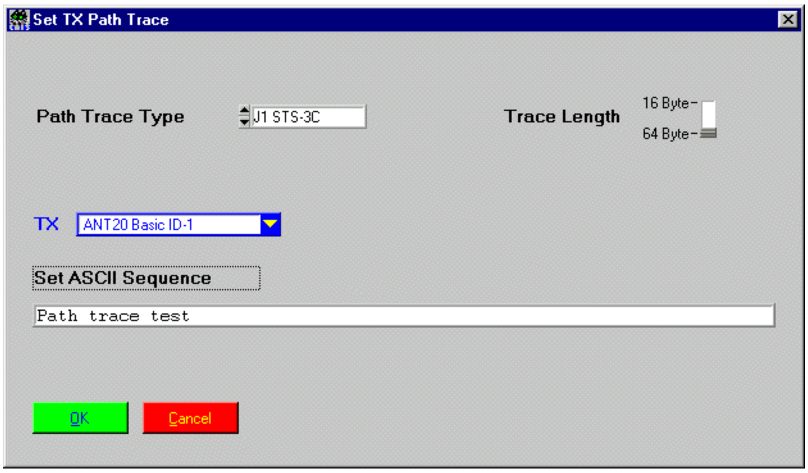
Notes:

6 Insert SONET Path Trace

Function Name

```
_set_sonet_path_trace
               c:\ ... \al_sens\al_sens.obj
```

User Interface



Description

Purpose: Inserts a user defined path trace.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Set Path Trace                    PASS
     VT Path:
     TX setting  <Path trace test!>
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
Type:	Enumeration J0 = 1, J1 STS-3C = 2, J1 STS-1 = 3, J2 VT Path = 4
Length:	Enumeration 16 Byte = 15, 64 Byte = 64
Inserted P.T.:	String

Remote Control of Testcase

Input:	%d#%d,%s,%d TX Id,Type,Inserted P.T.,Length
Output:	No Results

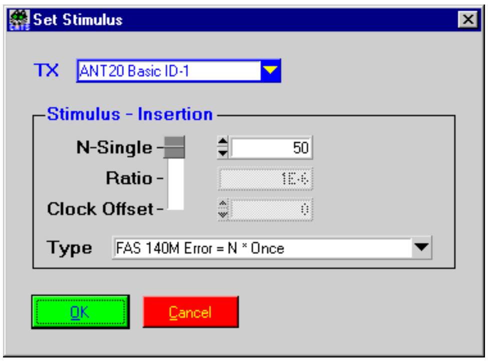
Notes:

7 Insert ITU-T Errors or Offset Stimulus

Function Name

```
_set_stimulus
            c:\ ... \al_sens\al_sens.obj
```

User Interface



Description

Purpose: Inserts an error ratio or frequency offset until switched off by the next testcase. To define insertion time, this testcase can be combined with the “_wait” testcase. To switch off error insertion, “_set_stimulus” has to be called again with the stimulus set to “Switch off error insertion”.

Caution: Up to 9 errors can be inserted within a 10 sec time frame.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Set Stimulus                PASS
```

Applications

G.826 performance monitor stimulus, wander generation.

Parameters

TX Id: Integer, 1 .. 10

Stimulus: Enumeration
No Stimulus = -1, All Alarm Insertions OFF = 199,
All Error Insertions OFF = 99, Clock Offset = 0, N * Single Code Error = 102,
N * Single B1-(MS-BIP) Error = 114, N * Single B2 Error = 116,
N * Single B3-(HP-BIP) Error = 118, N * Single MS-REI Error = 120,
N * Single HP-REI Error = 122, N * Single LP-REI Error = 124,
N * Single LP-BIP Error = 126, N * Single FE-DS1= 130,
N * Single FAS 2M Error = 104, N * Single FAS 8M Error = 106,
N * Single FAS 34M Error = 108, N * Single FE-DS3 Error = 140,
N * Single FAS 140M Error = 110, N * Single E-Bit Error = 112,
N * Single TSE (Bit) Error = 100, N * Single CRC-6 Error = 142,
N * Single CRC-4 Error = 144, TSE (Bit) Error Ratio = 200,
CRC-4 Error Ratio = 204, CRC-6 Error Ratio = 244,
FAS-1,5M (FE-DS1) Error Ratio = 232, FAS-2M (FE-E1) Error Ratio = 206,
FAS-8M (FE-E2) Error Ratio = 208, FAS-34M (FE-E3) Error Ratio = 210,
FE-DS3 (FAS-45M) Error Ratio = 242,FE-140M (FAS-E4) Error Ratio = 212,
E-Bit Error Ratio = 214, B1-(MS-BIP) Error Ratio = 216, B2 Error Ratio = 218,
B3-(HP BIP) Error Ratio = 220, MS-REI Error Ratio = 222,
HP-REI Error Ratio = 224, LP-REI Error Ratio = 226, LP-BIP Error Ratio = 228

N Single: Integer, 1 .. 100

Ratio: Double, 1E-12 .. 1E-2

Offset: Double, -500000 .. 500000 [ppb]

Remote Control of Testcase

Input: %d#%d,0,%lf,%d
TX Id,Stimulus,not used,Ratio / Offset,N Single

Output: No Results

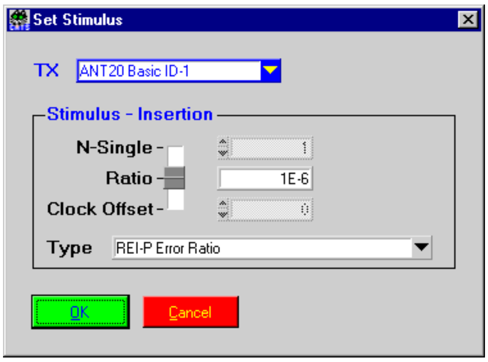
Notes:

8 Insert SONET Errors or Offset Stimulus

Function Name

```
_set_sonet_stimulus
              c:\ ... \al_sens\al_sens.obj
```

User Interface



Description

Purpose:	Inserts an error ratio or frequency offset until switched off by the next testcase. To define insertion time, this testcase can be combined with the "_wait" testcase. To switch off error insertion, "_set_sonet_stimulus" has to be called again with the stimulus set to "Switch off error insertion".
Caution:	Up to 9 errors can be inserted within a 10 sec time frame.

PASS / FAIL Conditions

PASS:	Not applicable.
FAIL:	Not applicable.

Test Report

```
Set Stimulus                        PASS
     FE-DS1 Error = N * Single(N = 1) inserted
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
Stimulus:	Enumeration No Stimulus = -1, All Alarm Insertions OFF = 199, All Error Insertions OFF = 99, Clock Offset = 0, N * Single BPV (Code) Error = 102, N * Single B1 Error = 114, N * Single B2 Error = 116, N * Single B3 Error = 118, N * Single REI-L Error = 120, N * Single REI-P Error = 122, N * Single REI-V Error = 124, N * Single BIP-V Error = 126, N * Single DS3 C-Parity Error = 134, N * Single DS3 PARITY Error = 136, N * Single DS3 MFAS-45M Error = 138, N * Single FE-E4 Error = 110, N * Single FE-DS3 Error = 140, N * Single FE-E1 Error = 104, N * Single FE-DS1 Error = 130, N * Single CRC-4 Error = 144, N * Single CRC-6 Error = 142, N * Single TSE (Bit) Error = 100, TSE (Bit) Error Ratio = 200, CRC-4 Error Ratio = 204, CRC-6 Error Ratio = 244, FE-DS1 (FAS-1,5M) Error Ratio = 232, FE-E1 (FAS-2M) Error Ratio = 206, FE-DS3 (FAS-45M) Error Ratio = 242, E-Bit Error Ratio = 214, DS3 C-Parity Ratio = 236, DS3 Parity Ratio = 238, DS3 MFAS-45M Error Ratio = 240, B1-BIP Error Ratio = 216, B2 Error Ratio = 218, B3-BIP Error Ratio = 220, REI-L Error Ratio = 222, REI-P Error Ratio = 224, REI-V Error Ratio = 226, BIP-V Error Ratio = 228
N Single:	Integer, 1 .. 100
Ratio:	Double, 1E-12 .. 1E-2
Offset:	Double, -500000 .. 500000 [ppb]

Remote Control of Testcase

Input:	%d#%d,0,%lf,%d TX Id,Stimulus,not usedRatio / Offset,N Single
Output:	

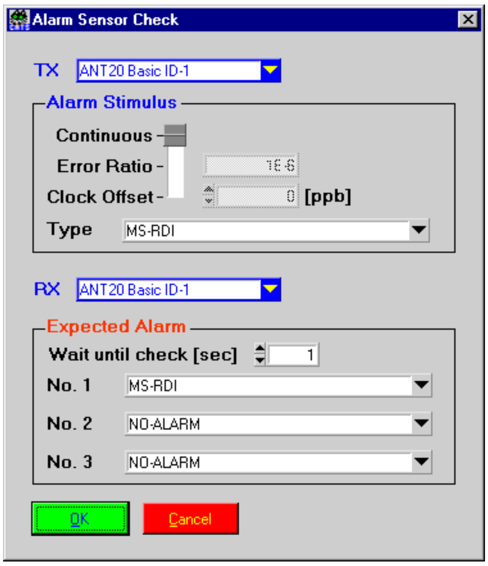
Notes:

9 Insert ITU-T Alarm or Errors and Check Alarms

Function Name

```
_check_alarm
            .\al_sens\al_sens.obj
```

User Interface



Description

Purpose: This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. The testcase parameters may be easily changed by the customer to suit specific test needs. This testcase always switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by “_set_alarm” and “_set_stimulus” testcases will prevail until switched off expressedly.

Caution: Always fill “Expected Alarms” from the top. It is illegal to set 1st expected alarm to “None” and set 2nd and/or 3rd to anything different from none.

PASS / FAIL Conditions

PASS: The observed alarm situation matches with the expected alarms.

FAIL: The observed alarm situation does not match the expectations, that means at least one of the expected alarms could not be observed.

Test Report

```
Check Alarm        PASS
    LOF-2M inserted

                   expected     measured
    LOF-2M            ON           ON
```

Applications

–

Parameters

TX Id: Integer, 1 .. 10

RX Id: Integer, 1 .. 10

Stimulus: Enumeration
No Alarm Stimulus = 0, All Alarm Insertions OFF = 199,
All Error Insertions OFF = 99, Clock Offset = 0, LOS = 700, LOF-STM1 = 718,
RS-TIM = 740, MS-AIS = 706, MS-RDI = 707, AU-LOP = 701, AU-AIS = 704,
TU-AIS = 705,TU-LOP = 702, TU-LOM = 703, HP-PLM = 741, HP-RDI = 708,
HP-RDI-C = 748, HP-RDI-P = 746, HP-RDI-S = 747, HP-TIM = 719,
HP-UNEQ = 710, LP-PLM = 742, LP-RDI = 709, LP-RDI-C = 751,
LP-RDI-P = 749, LP-RDI-S = 750, LP-TIM = 720, LP-UNEQ = 711,
LP-RFI = 743, AIS-DS1 = 712, AIS-2M = 713, AIS-8M = 714, AIS-34M = 715,
AIS-DS3 = 717, AIS-140M = 716, LOF-DS1 = 723, LOF-2M = 724,
LOF-8M = 725, LOF-34M = 726, LOF-DS3 = 727, LOF-140M = 728,
RDI-DS1 = 729, RDI-2M = 730, RDI-8M =731, RDI-34M = 732,
RDI-DS3 = 733, RDI-140M = 734, IDEL-DS3 = 735,
TSE (Bit) Error Ratio = 200, CRC-4 Error Ratio = 204,
CRC-6 Error Ratio = 244, FAS-1,5M (FE-DS1) Error Ratio = 232,
FAS-2M (FE-E1) Error Ratio = 206, FAS-8M (FE-E2) Error Ratio = 208,
FAS-34M (FE-E3) Error Ratio = 210,
FE-DS3 (FAS-45M) Error Ratio = 242, FE-140M (FAS-E4) Error Ratio = 212,
E-Bit Error Ratio = 214, B1-(MS-BIP) Error Ratio = 216, B2 Error Ratio = 218,
B3-(HP BIP) Error Ratio = 220, MS-REI Error Ratio = 222,
HP-REI Error Ratio = 224, LP-REI Error Ratio = 226, LP-BIP Error Ratio = 228

Ratio: Double, 1E-12 .. 1E-2

Offset: Double, -500000 .. 500000 [ppb]

Delay: Double, 0 .. 99 [s]

Exp. Alarm: Enumeration
No Alarm = 0, LOS = 1, LTI = 73, LOS-SEQ-SYNC = 73, LOF-STM1 = 8,
OOF-STM1 = 34, AU-AIS = 7, MS-AIS = 6, TU-AIS = 19, AU-LOP = 10,
TU-LOP = 18, AU-NDF = 74, TU-NDF = 75, MS-RDI = 9, HP-RDI = 11,
LP-RDI = 20, HP-PLM = 76, LP-PLM = 77, HP-UNEQ = 12, LP-UNEQ = 21,
RS-TIM = 78, HP-TIM = 79, LP-TIM = 80, TU-LOM = 17, LP-RFI = 81,
HP-RDI-P = 82, HP-RDI-S = 83, HP-RDI-C = 84, LP-RDI-P = 85,
LP-RDI-S = 86, LP-RDI-C = 87, HP-PDI = 88, LP-PDI = 89, AIS-2M = 30,
LOF-2M = 27, RDI-2M = 33, AIS-8M = 29, LOF-8M = 26, RDI-8M = 32,
AIS-34M = 28, LOF-34M = 25, RDI-34M = 31, AIS-140M = 5, LOF-140M = 24,
RDI-140M = 22, AIS-64K = 90, AIS-DS1 = 63, OOF-DS1 = 64, LOF-DS1 = 65,
YELLOW-DS1 = 66, AIS-DS3 = 67, OOF-DS3 = 68, LOF-DS3 = 69,
YELLOW-DS3 = 70, IDLE-A-DS3 = 71, AIC-DS3 = 72

Remote Control of Testcase

Input: %d,%d#%d,0,%lf,%d,%d,%d,%lf
RX Id,TX Id,Stimulus,not used,Ratio / Offset,Exp. Alarm 1,Exp. Alarm 2, Exp. Alarm 3,Delay

Output: {%s %d,}
{AlarmString AlarmFlag,}
AlarmFlag: ON = 1, OFF = 0

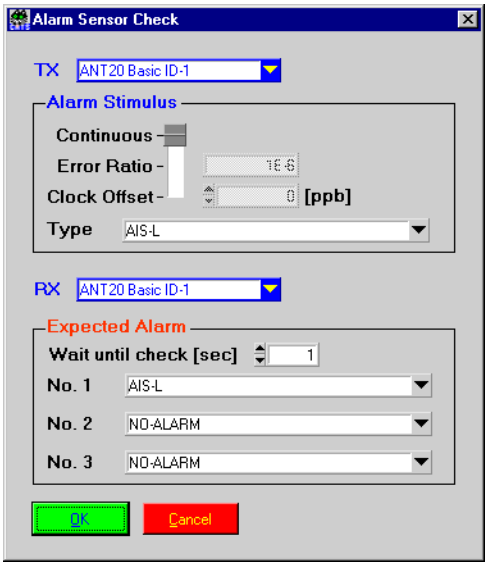
Number of value pairs depends on the number of expected alarms.

10 Insert SONET Alarm or Errors and Check Alarms

Function Name

```
_check_sonet_alarm
               .\al_sens\al_sens.obj
```

User Interface



Description

Purpose: This test stimulates alarm sensors and checks the presence and/or absence of the respective incoming alarms. The testcase parameters may be easily changed by the customer to suit specific test needs. This testcase always switches off the inserted alarm or error when finished. In contrast, alarms or other stimuli inserted by "_set_alarm" and "_set_stimulus" will prevail until switched off expressedly.

Caution: Always fill "Expected Alarms" from the top. It is illegal to set 1st expected alarm to "None" and set 2nd and/or 3rd to anything different from none.

PASS / FAIL Conditions

PASS: The observed alarm situation matches with the expected alarms.

FAIL: The observed alarm situation does not match the expectations, that means at least one of the expected alarms could not be observed.

Test Report

```
Check Alarm        PASS
     LOF-DS1 inserted

                   expected      measured
     LOF-DS1           ON            ON
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
Stimulus:	Enumeration No Alarm Stimulus = 0, All Alarm Insertions OFF = 199, All Error Insertions OFF = 99, Clock Offset = 0, LOS = 700, LOF-STS = 718, AIS-L = 805, RDI-L = 806, TIM-L = 820, AIS-P = 803, LOP-P = 800, PLM-P = 821, RDI-P = 807, RDI-C-P = 826, RDI-P-P = 824, RDI-S-P = 825, TIM-P = 811, UNEQ-P = 809, PDI-P = 744, AIS-V = 804, LOP-V = 801, LOM-V = 802, PLM-V = 822, RDI-V = 808, RDI-C-V = 829, RDI-P-V = 827, RDI-S-V = 828, RFI-V = 823, TIM-V = 812, UNEQ-V = 810, PDI-V = 745, AIS-DS1 = 712, AIS-2M = 713, AIS-DS3 = 717, LOF-DS1 = 732, LOF-2M = 724, LOF-DS3 = 727, RDI-DS1 = 729, RDI-2M = 730, RDI-DS3 = 733, IDLE-DS3 = 735, TSE (Bit) Error Ratio = 200, CRC-4 Error Ratio = 204, CRC-6 Error Ratio = 244, FE-DS1 (FAS-1,5M) Error Ratio = 232, FE-E1 (FAS-2M) Error Ratio = 206, FE-DS3 (FAS-45M) Error Ratio = 242, E-Bit Error Ratio = 214, DS3 C-Parity Ratio = 236, DS3 Parity Ratio = 238, DS3 MFAS-45M Error Ratio = 240, B1-BIP Error Ratio = 216, B2 Error Ratio = 218, B3-BIP Error Ratio = 220, REI-L Error Ratio = 222, REI-P Error Ratio = 224, REI-V Error Ratio = 226, BIP-V Error Ratio = 228
Ratio:	Double, 1E-12 .. 1E-2
Offset:	Double, -500000 .. 500000 [ppb]
Delay:	Double, 0 .. 99 [s]
Exp. Alarm:	Enumeration No Alarm = 0, No Power = 23, LOS = 1, LTI = 73, LOS-SEQ-SYNC = 73, LOF-STS = 40, OOF = 41, AIS-P = 42, AIS-L = 43, AIS-V = 44, LOP-P = 45, LOP-V = 46, NDF-P = 47, NDF-V = 48, RDI-L = 49, RDI-P = 50, RDI-V = 51, PLM-P = 52, PLM-V = 53, UNEQ-P = 54, UNEQ-V = 55, TIM-L = 56, TIM-P = 57, TIM-V = 58, LOM-V = 59, RFI-V = 60, RDI-P-P = 91, RDI-S-P = 92, RDI-C-P = 93, RDI-P-VT = 94, RDI-S-VT = 95, RDI-C-VT = 96, AIS-2M = 30, LOF-2M = 27, RDI-2M = 33, AIS-64K = 90, AIS-DS1 = 63, OOF-DS1 = 64, LOF-DS1 = 65, RDI-DS1 = 66, AIS-DS3 = 67, OOF-DS3 = 68, LOF-DS3 = 69, RDI-DS3 = 70, IDLE-A-DS3 = 71, AIS-DS3 = 72

Remote Control of Testcase

Input:	%d,%d#%d,0,%lf,%d,%d,%d,%lf RX Id,TX Id,Stimulus,not used,Ratio / Offset,Exp. Alarm 1,Exp. Alarm 2, Exp. Alarm 3,Delay
Output:	{%s %d,} {AlarmString AlarmFlag,} AlarmFlag: ON = 1, OFF = 0 Number of value pairs depends on the number of expected alarms.

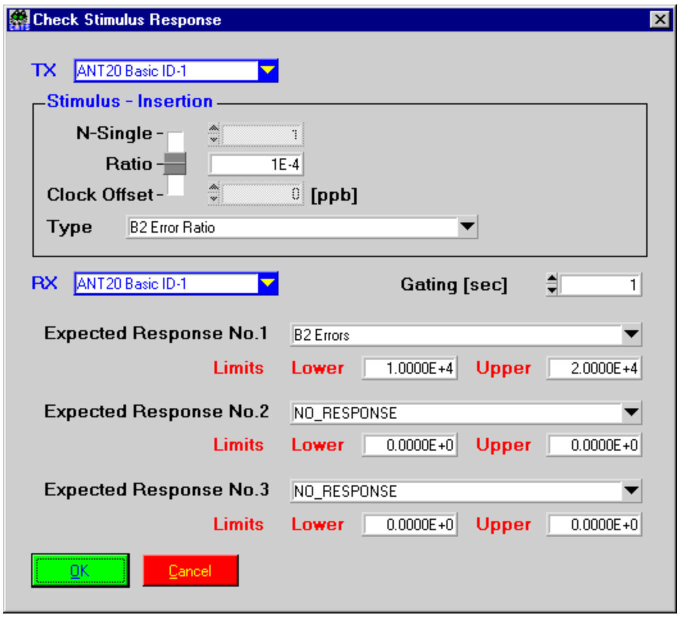
Notes:

11 Insert ITU-T Errors or Offsets and Check Response

Function Name

```
_check_stimulus_response
               .\al_sens\al_sens.obj
```

User Interface



Description

Purpose: This test stimulates the error indication sensors of SDH or PDH UUTs. A number of single errors or a fixed error ratio is sent at TX site. At RX, presence of expected number of incoming errors or alarm duration is checked. Upper/ lower limit for expected result may be set by user.
This testcase may be used as a long term BERT test by selecting error stimulus “None”.

Caution: Up to 9 single errors can be inserted within a 10 sec time frame.

PASS / FAIL Conditions

PASS: Test case is assumed PASS if number of received errors is within the user-defined limits.

FAIL: Results are not valid or beyond the user-defined limits.

Test Report

```
Check Ratio                     PASS
     FAS 2M Error Ratio = 0.0001 inserted
     FAS 2M Error        : 28
```

Applications

–

Parameters

TX Id: Integer, 1 .. 10

RX Id: Integer, 1 .. 10

Stimulus: Enumeration
No Stimulus = -1, All Alarm Insertions OFF = 199,
All Error Insertions OFF = 99, Clock Offset = 0, N * Single Code Error = 102,
N * Single B1-(MS-BIP) Error = 114, N * Single B2 Error = 116,
N * Single B3-(HP-BIP) Error = 118, N * Single MS-REI Error = 120,
N * Single HP-REI Error = 122, N * Single LP-REI Error = 124,
N * Single LP-BIP Error = 126, N * Single FE-DS1= 130,
N * Single FAS 2M Error = 104, N * Single FAS 8M Error = 106,
N * Single FAS 34M Error = 108, N * Single FE-DS3 Error = 140,
N * Single FAS 140M Error = 110, N * Single E-Bit Error = 112,
N * Single TSE (Bit) Error = 100, N * Single CRC-6 Error = 142,
N * Single CRC-4 Error = 144, TSE (Bit) Error Ratio = 200,
CRC-4 Error Ratio = 204, CRC-6 Error Ratio = 244,
FAS-1,5M (FE-DS1) Error Ratio = 232, FAS-2M (FE-E1) Error Ratio = 206,
FAS-8M (FE-E2) Error Ratio = 208, FAS-34M (FE-E3) Error Ratio = 210,
FE-DS3 (FAS-45M) Error Ratio = 242,FE-140M (FAS-E4) Error Ratio = 212,
E-Bit Error Ratio = 214, B1-(MS-BIP) Error Ratio = 216, B2 Error Ratio = 218,
B3-(HP BIP) Error Ratio = 220, MS-REI Error Ratio = 222,
HP-REI Error Ratio = 224, LP-REI Error Ratio = 226, LP-BIP Error Ratio = 228

N Single: Integer, 1 .. 100

Ratio: Double, 1E-12 .. 1E-2

Offset: Double, -500000 .. 500000 [ppb]

Gating: Double, 1 .. 1000000000 [s]

Response (3 triples)

Response: Enumeration
No Response = -1, Alarm time [ms] = 6, TSE-Bit Errors = 0, Code Errors = 2,
CRC-4 Errors = 44, FAS 2M Error = 29, FAS 8M Errors = 32,
FAS 34M Errors = 35, FAS 140M Errors = 38, E-Bit Errors = 47,
B1-(MS BIP) Errors = 9, B2 Errors = 11, B3-(HP-BIP) Errors = 13,
HP-REI Errors = 15, LP-REI Errors = 17, MS-REI Errors = 19,
LP-BIP Errors = 21

Upper Limit: Double, 0.0000E+0 .. 4.2950E+22

Lower Limit: Double, 0.0000E+0 .. 4.2950E+22

Remote Control of Testcase

Input:

%d,%d#%d,0,%lf,%lf,%d,%d,%lf,%lf,%d,%lf,%lf,%d,%lf,%lf
RX Id,TX Id,Stimulus,not used,Gating,Ratio / Offset,N Single,
Response 1,Upper Limit 1,Lower Limit 1,
Response 2,Upper Limit 2,Lower Limit 2,
Response 3,Upper Limit 3,Lower Limit 3

Output:

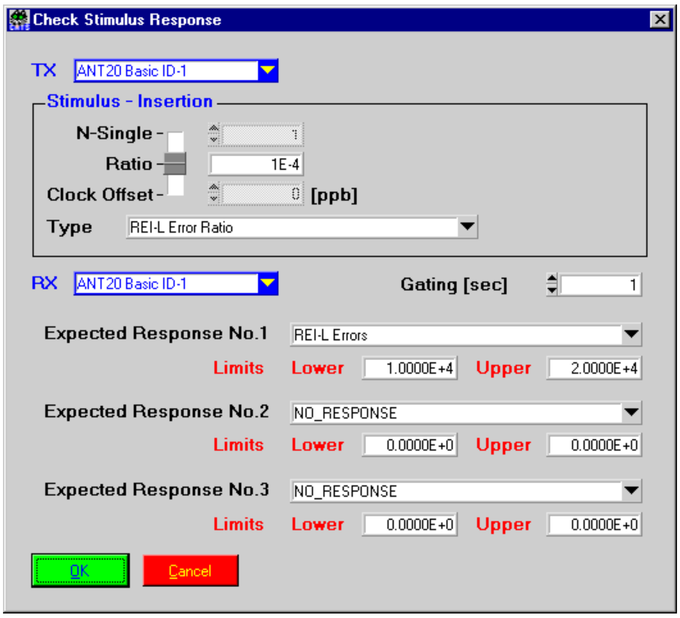
%d,%lf,%d,%lf,%d,%lf
Valid1?,Result1,Valid2?,Result2,Valid3?,Result3
Valid?: Valid = 1, Not valid = 0
Result = -1 ==> No result

12 Insert SONET Errors or Offsets and Check Response

Function Name

```
_check_sonet_stimulus_response
              c:\ ... \al_sens\al_sens.obj
```

User Interface



Description

Purpose: This test stimulates the error indication sensors of SONET or PDH UUTs. A number of single errors or a fixed error ratio is sent at TX site. At RX, presence of expected number of incoming errors or alarm duration is checked. Upper/ lower limit for expected result may be set by user.
This testcase may be used as a long term BERT test by selecting error stimulus “None”.

Caution: Up to 9 single errors can be inserted within a 10 sec time frame.

PASS / FAIL Conditions

PASS: Test case is assumed PASS if number of received errors is within the user-defined limits.

FAIL: Results are not valid or beyond the user-defined limits.

Test Report

```
Check Offset                PASS
    Clock Offset                  = 15 [PPB] inserted
    Alarm time [ms]          : 0
```

Applications

–

Parameters

TX Id: Integer, 1 .. 10

RX Id: Integer, 1 .. 10

Stimulus: Enumeration
No Stimulus = -1, All Alarm Insertions OFF = 199,
All Error Insertions OFF = 99, Clock Offset = 0,
N * Single BPV (Code) Error = 102, N * Single B1 Error = 114,
N * Single B2 Error = 116, N * Single B3 Error = 118,
N * Single REI-L Error = 120, N * Single REI-P Error = 122,
N * Single REI-V Error = 124, N * Single BIP-V Error = 126,
N * Single DS3 C-Parity Error = 134, N * Single DS3 PARITY Error = 136,
N * Single DS3 MFAS-45M Error = 138, N * Single FE-E4 Error = 110,
N * Single FE-DS3 Error = 140, N * Single FE-E1 Error = 104,
N * Single FE-DS1 Error = 130, N * Single CRC-4 Error = 144,
N * Single CRC-6 Error = 142, N * Single TSE (Bit) Error = 100,
TSE (Bit) Error Ratio = 200, CRC-4 Error Ratio = 204,
CRC-6 Error Ratio = 244, FE-DS1 (FAS-1,5M) Error Ratio = 232,
FE-E1 (FAS-2M) Error Ratio = 206, FE-DS3 (FAS-45M) Error Ratio = 242,
E-Bit Error Ratio = 214, DS3 C-Parity Ratio = 236, DS3 Parity Ratio = 238,
DS3 MFAS-45M Error Ratio = 240, B1-BIP Error Ratio = 216,
B2 Error Ratio = 218, B3-BIP Error Ratio = 220, REI-L Error Ratio = 222,
REI-P Error Ratio = 224, REI-V Error Ratio = 226, BIP-V Error Ratio = 228

N Single: Integer, 1 .. 100

Ratio: Double, 1E-12 .. 1E-2

Offset: Double, -500000 .. 500000 [ppb]

Gating: Double, 1 .. 1000000000 [s]

Response (3 triples)

Response: Enumeration
No Response= -1, Alarm time [ms] = 6, TSE-Bit Errors = 0,
BPV-Code Errors = 2, B1 Errors = 9, B2 Errors = 11, B3 Errors = 13,
CRC-DS1 Errors = 92, FE-DS1 Errors = 90, FE-DS3 Errors = 106,
DS3 MFAS-45M Errors = 108, DS3 C-45M Parity Errors = 102,
DS3-45M Parity Errors = 100, HP-REI Errors = 15, LP-REI Errors = 17,
M1-REI Errors = 19, LP-BIP Errors = 21

Upper Limit: Double, 0.0000E+0 .. 4.2950E+22

Lower Limit: Double, 0.0000E+0 .. 4.2950E+22

Remote Control of Testcase

Input:

%d,%d#%d,0,%lf,%lf,%d,%d,%lf,%lf,%d,%lf,%lf,%d,%lf,%lf
RX Id,TX Id,Stimulus,not used,Gating,Ratio / Offset,N Single,
Response 1,Upper Limit 1,Lower Limit 1,
Response 2,Upper Limit 2,Lower Limit 2,
Response 3,Upper Limit 3,Lower Limit 3

Output:

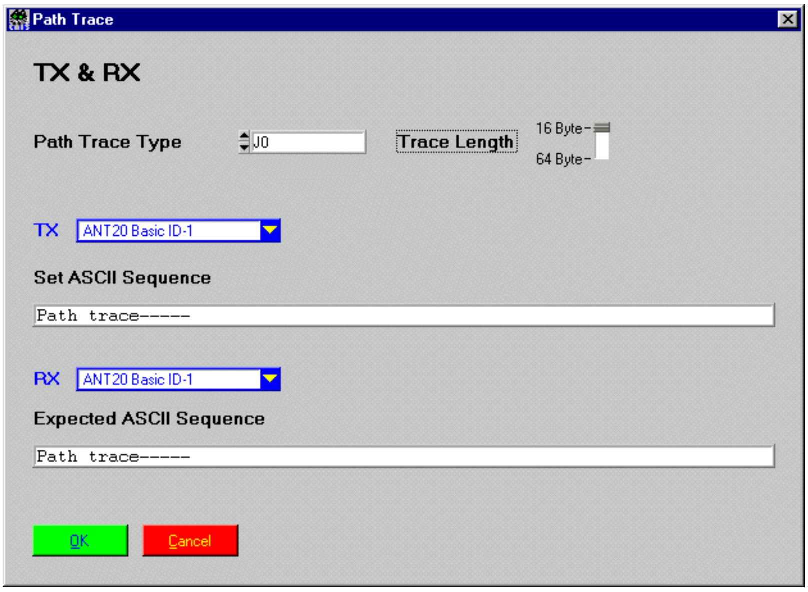
%d,%lf,%d,%lf,%d,%lf
Valid1?,Result1,Valid2?,Result2,Valid3?,Result3
Valid?: Valid = 1, Not valid = 0
Result = -1 ==> No result

13 Check SDH Path Trace against expected Value

Function Name

```
_check_path_trace
              .\al_sens\al_sens.obj
```

User Interface



Description

Purpose: Inserts a user defined path trace at the TX site and reads path trace at the RX. A comparison is made between the expected path trace and the received path trace.

Caution: This testcase does not work in 1/100 bitrate mode (refers to VXI only).

PASS / FAIL Conditions

PASS: Test case is considered PASS if expected path trace equals received path trace.

FAIL: Received path trace does not match with expected path trace or is not valid (due to alarms).

Test Report

```
Check Path Trace          PASS
    J0:
    TX setting  <Path trace----->
    RX measured <Path trace----->
    RX expected <Path trace----->
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
Type:	Enumeration J0 = 1, J1 C4 = 2, J1 C3 = 3, J2 C12/C11 = 4
Length:	Enumeration 16 Byte = 15, 64 Byte = 64
Inserted P.T.:	String
Exp. P.T.:	String

Remote Control of Testcase

Input:	%d,%d#%d,%s,%s,%d RX Id,TX Id,Type,Inserted P.T.,Exp. P.T.,Length
Output:	%d,%s Validity,Received P.T.

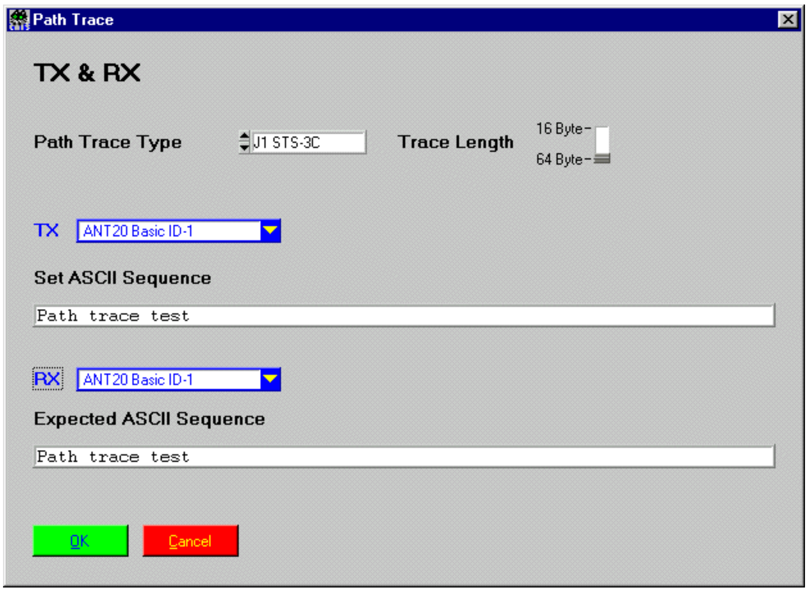
Notes:

14 Check SONET Path Trace against expected Value

Function Name

```
_check_sonet_path_trace
               .\al_sens\al_sens.obj
```

User Interface



Description

Purpose: Inserts a user defined path trace at port A and reads path trace at port B. A comparison is made between the expected path trace and the received path trace.

Caution: This testcase does not work in 1/100 bitrate mode (refers to VXI only).

PASS / FAIL Conditions

PASS: Test case is considered PASS if expected path trace equals received path trace.

FAIL: Received path trace does not match with expected path trace or is not valid (due to alarms).

Test Report

```
Check Path Trace                PASS
     J1 STS-3C:
     TX setting  <Path trace----->
     RX measured <Path trace----->
     RX expected <Path trace----->
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
Type:	Enumeration J0 = 1, J1 STS-3C = 2, J1 STS-1 = 3, J2 VT Path = 4
Length:	Enumeration 16 Byte = 15, 64 Byte = 64
Inserted P.T.:	String
Exp. P.T.:	String

Remote Control of Testcase

Input:	%d,%d#%d,%s,%s,%d RX Id,TX Id,Type,Inserted P.T.,Exp. P.T.,Length
Output:	%d,%s Validity,Received P.T.

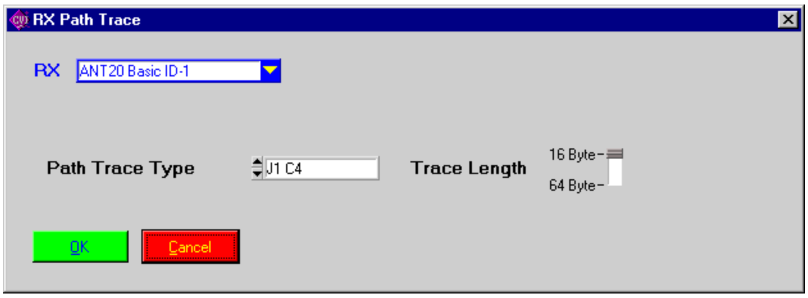
Notes:

15 Read SDH Path Trace

Function Name

```
_read_rx_path_trace
              .\al_sens\al_sens.obj
```

User Interface



Description

Purpose: Inserts a user defined path trace at port A and reads path trace at port B. A comparison is made between the expected path trace and the received path trace.

Caution: This testcase does not work in 1/100 bitrate mode (refers to VXI only).

PASS / FAIL Conditions

PASS: Test case is considered PASS if it is possible to read the path trace.

FAIL: Received path trace is not valid (due to alarms).

Test Report

```
Check Path Trace                PASS
     J1 STS-3C:
     TX setting  <Path trace----->
     RX measured <Path trace----->
     RX expected <Path trace----->
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
Type:	Enumeration J0 = 1, J1 STS-3C = 2, J1 STS-1 = 3, J2 VT Path = 4
Length:	Enumeration 16 Byte = 15, 64 Byte = 64
Inserted P.T.:	String
Exp. P.T.:	String

Remote Control of Testcase

Input:	%d,%d#%d,%s,%s,%d RX Id,TX Id,Type,Inserted P.T.,Exp. P.T.,Length
Output:	%d,%s Validity,Received P.T.

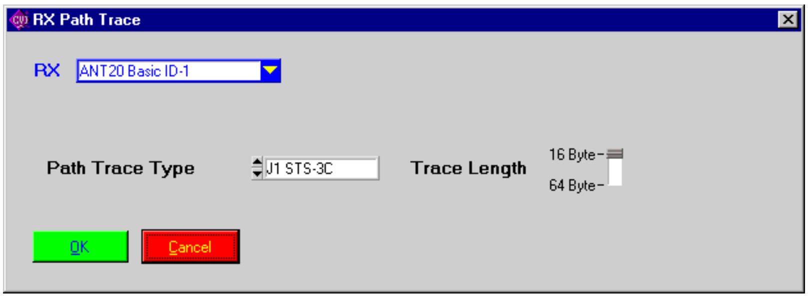
Notes:

16 Read SONET Path Trace

Function Name

```
_read_rx_sonet_path_trace
               .\al_sens\al_sens.obj
```

User Interface



Description

Purpose: Inserts a user defined path trace at port A and reads path trace at port B. A comparison is made between the expected path trace and the received path trace.

Caution: This testcase does not work in 1/100 bitrate mode (refers to VXI only).

PASS / FAIL Conditions

PASS: Test case is considered PASS if it is possible to read the path trace.

FAIL: Received path trace is not valid (due to alarms).

Test Report

```
Check Path Trace                PASS
     J1 STS-3C:
     TX setting  <Path trace----->
     RX measured <Path trace----->
     RX expected <Path trace----->
```

Applications

–

Parameters

TX Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
Type:	Enumeration J0 = 1, J1 STS-3C = 2, J1 STS-1 = 3, J2 VT Path = 4
Length:	Enumeration 16 Byte = 15, 64 Byte = 64
Inserted P.T.:	String
Exp. P.T.:	String

Remote Control of Testcase

Input:	%d,%d#%d,%s,%s,%d RX Id,TX Id,Type,Inserted P.T.,Exp. P.T.,Length
Output:	%d,%s Validity,Received P.T.

Notes:

WWG-CATS Testcase Library

ATM Testcases

5

ATM Testcases

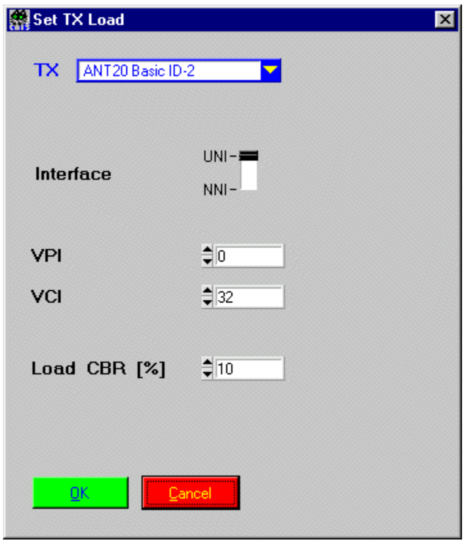
Notes:

1 Set TX VPI/VCI, UNI/NNI and Load

Function Name

```
_set_tx_vpi_vci_load
            .\atm\atm_bas.obj
```

User Interface



Description

Purpose: Sets the load (% CBR) of a specific VPI/VCI.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
TX VPI,VCI Load       CONTINUE
    VPI: 0 VCI: 32 Load :  10.00 %
```

Applications

Can be used in stress tests for ATM switches when a fixed CBR load is required.

See sequence “atmdelay.squ”.

Parameters

TX Id:	Integer, 1 .. 10
TX Interface:	Enumeration UNI = 1, NNI = 2.
TX VPI:	Integer, 0 .. 4095
TX VCI:	Integer, 0 .. 65535
CBR Load:	Double, 0.01 .. 100.00 [%]

Remote Control of Testcase

Input:	%d#%d,%d,%d,%lf TX Id,TX Interface,TX VPI,TX VCI,CBR Load
Output:	No Results

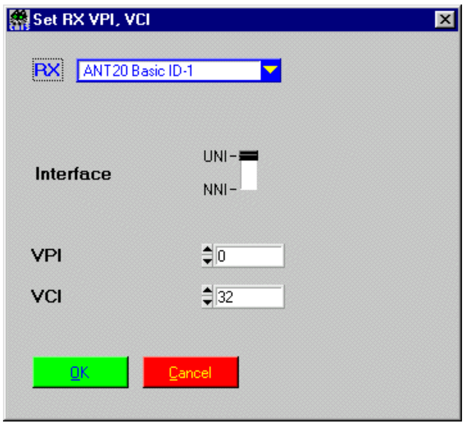
Notes:

2 SET RX VPI/VCI & UNI/NNI

Function Name

```
_set_rx_vpi_vci
             .\atm\atm_bas.obj
```

User Interface



Description

Purpose: Sets a specific VPI/VCI.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
RX Set VPI, VCI     CONTINUE
    VPI: 0 VCI: 32
```

Applications

See sequence “atmdelay.squ”.

Parameters

RX Id:	Integer, 1 .. 10
RX Interface:	Enumeration UNI = 1, NNI = 2.
RX VPI:	Integer, 0 .. 4095
RX VCI:	Integer, 0 .. 65535

Remote Control of Testcase

Input:	%d#%d,%d,%d,%lf RX Id,RX Interface,RX VPI,RX VCI
Output:	No Results

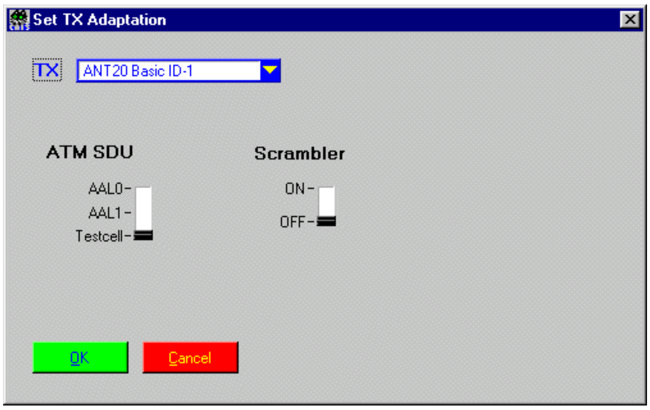
Notes:

3 Set TX AAL & Scrambler

Function Name

```
_set_tx_atm_adaptation
            .\atm\atm_bas.obj
```

User Interface



Description

Purpose: Sets the ATM adaptation layer or test cell insertion.

Caution: Only use this testcase after having set signal structure to an ATM signal.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
TX Adapt AAL0          CONTINUE
    TX SDU Type:       Testcell
    TX Scrambler:      ON
```

Applications

See sequence “atmdelay.squ”.

Parameters

TX Id:	Integer, 1 .. 10
SDU:	Enumeration AAL0 = 1, AAL1 = 2, Testcell = 3.
Scrambler:	Enumeration ON = 1, OFF = 2.

Remote Control of Testcase

Input:	%d#%d,%d TX Id,SDU,Scrambler
Output:	No Results

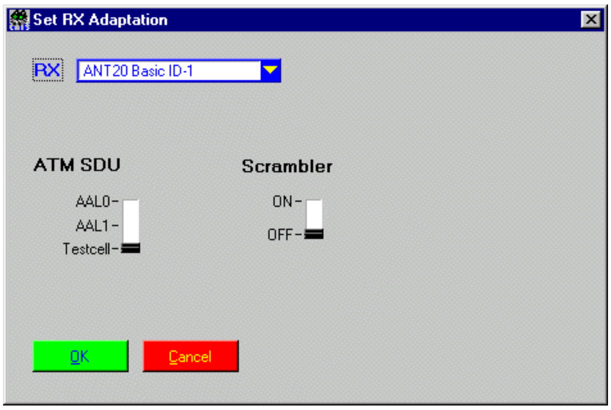
Notes:

4 Set RX AAL & Scrambler

Function Name

```
_set_rx_atm_adaptation
             .\atm\atm_bas.obj
```

User Interface



Description

Purpose: Sets the ATM adaptation layer or test cell insertion.

Caution: Only use this testcase after having set signal structure to an ATM signal.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
TX Adapt AAL0         CONTINUE
    TX SDU Type:      Testcell
    TX Scrambler:     ON
```

Applications

See sequence “atm_delay.squ”.

Parameters

RX Id:	Integer, 1 .. 10
SDU:	Enumeration AAL0 = 1, AAL1 = 2, Testcell = 3.
Scrambler:	Enumeration ON = 1, OFF = 2.

Remote Control of Testcase

Input:	%d#%d,%d RX Id,SDU,Scrambler
Output:	No Results

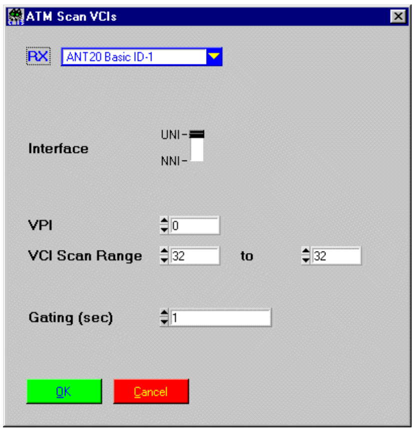
Notes:

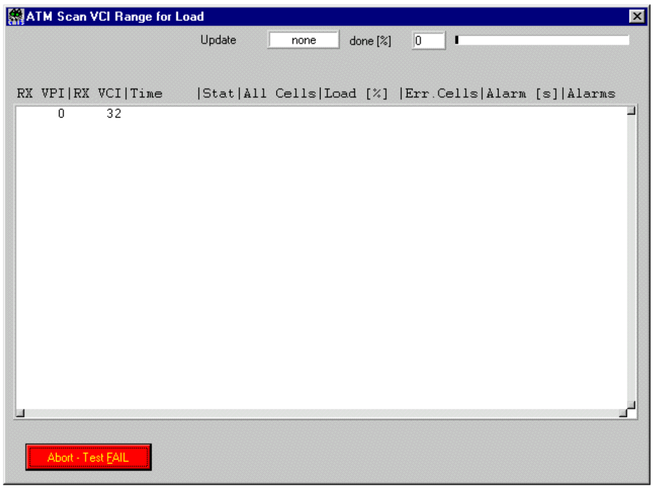
5 Scan VCI and Show Load

Function Name

```
_scan_vci_for_load
             .\atm\atm_bas.obj
```

User Interface





Description

Purpose: Scans through a range of VCI's (fixed VPI) and finds out which VCI's are in use. The load for used channels is measured.

Caution: –

PASS / FAIL Conditions

PASS: No alarms during gating.

FAIL: Alarms during gating.

Test Report

```
Scan VCIs for Load     PASS
   Interface:          UNI
   VPI: 0 VCI Scan from: 32 to: 36
   Gating [s]:                10.000
   RX     |RX     |Time    |Stat  |All Cells|Load  |Err.Cells|Alarm|Alarms
   VPI    VCI                              [%]                [s]
   0      32     17:34:56PASS   3.53e+06  **     0          0.0
   0      33     17:35:08FAIL   3.14e+06  12.00  0          1.1   LOS
   0      34     17:35:20FAIL   2.19e+06  **     0          3.8   LOS
   0      35     17:35:32PASS   3.53e+06  **     0          0.0
   0      36     17:35:44PASS   3.53e+06  **     0          0.0
```

Applications

Get a quick overview about channel usage and load at an ATM port.

See sequence “atm_scan.squ”.

Parameters

RX Id:	Integer, 1 .. 10
RX Interface:	Enumeration UNI = 1, NNI = 2.
RX VPI:	Integer, 0 .. 4095
From VCI:	Integer, 0 .. 65535
To VCI:	Integer, 0 .. 65535
Gating:	Double, 1.. 10 000 000 [sec]

Remote Control of Testcase

Input:	%d#%d,%d,%d,%d,%lf RX Id,RX Interface,RX VPI,From VCI,To VCI,Gating
Output:	%d{,%d,%d,%s,%s,%lf,%lf,%lf,%lf,%s} Number of VCI's,{RX VPI,RX VCI,Time,Result Status,All Cells, Filtered Cells Ratio,Errored Cells,Alarm Time Errored Cells,Alarms}.

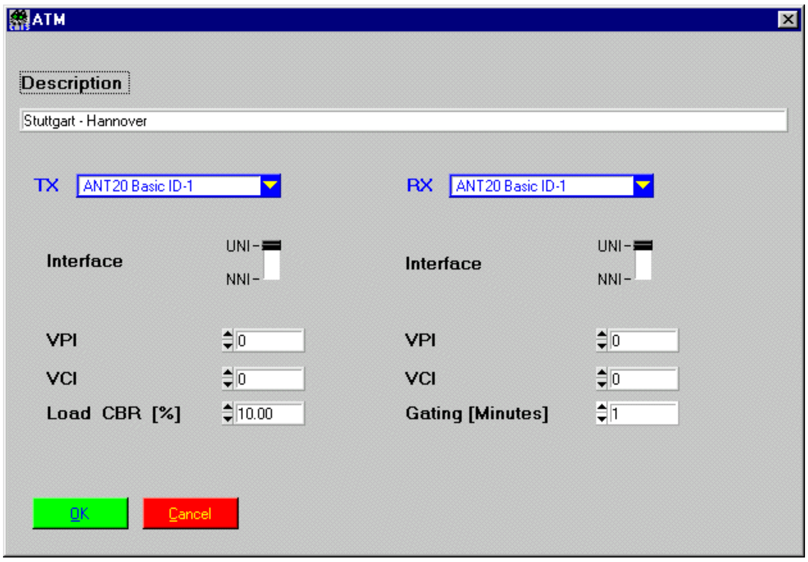
Notes:

6 ATM QoS Check (SONET or SDH or PDH)

Function Name

```
_check_atm
            .\atm\atm.obj
```

User Interface



Description

Purpose: Measures quality of service parameters during a longterm, out of service test.

Caution: –

PASS / FAIL Conditions

PASS: No alarms during gating.

FAIL: Alarms during gating.

Test Report

```
ATM STR-HAN          FAIL
   B1 Errors                   :0
   B2 Errors                   :0
   B3 Errors                   :0
   Lost Cells                  :0
   Alarm Time Lost Cell[ms]    :0
   Misinserted Cells           :0
   Alarm Time Misins. Cells[ms]:0
   Errored Cells               :0
   Errored Cell Ratio          :0
   Alarm Time Errored Cell [ms]:55072
   HCOR Errors                 :0
   Alarm Time HCOR Error [ms]  :0
   HUNC Errors                 :21192
   HUNC Error Ratio            :0.001
   Alarm Time HUNC Error [ms]  :0
   All Cells                   :2.11925e+07
   Load Cell Ratio [1/100 %]   :0.15
```

Applications

Longterm error check for a particular ATM link (out of service). A number of these testcases may be combined in a test sequence to test many different VPI/VCI’s in a row.

See sequence “atm.squ”.

Parameters

Description:	String
TX Id:	Integer, 1 .. 10
TX Interface:	Enumeration UNI = 1, NNI = 2.
TX VPI:	Integer, 0 .. 4095
TX VCI:	Integer, 0 .. 65535
TX Load:	Double, 0.01 .. 100.00 [%]
RX Id:	Integer, 1 .. 10
RX Interface:	Enumeration UNI = 1, NNI = 2.
RX VPI:	Integer, 0 .. 4095
RX VCI:	Integer, 0 .. 65535
Gating:	Double, 1.. 129600 [min]

Remote Control of Testcase

Input: %d,%d#%d,%d,%d,%d,%d,%d,%d,%d,#%s#
TX Id,RX Id,TX Interface,TX VCI,TX VPI,TX Load,RX Interface,RX VCI,RX VPI,Gating,Description

Output: %s,%s,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g,%g
Date,Time,TSE Errors,TSE Error Ratio,B1 Errors,B1 Error Ratio,
B2 Errors,B2 Error Ratio,B3 Errors,B3 Error Ratio,
Lost Cells, Lost Cell Ratio,Alarm Time Lost Cells [ms],
Misinserted Cells,Misinserted Cell Ratio,Alarm Time Misinserted Cells [ms],
Errored Cells,Errored Cells Ratio,Alarm Time Errored Cells,
HCOR Errors,HCOR Error Ratio,Alarm Time HCOR Errors [ms],
HUNC Errors,HUNC Error Ratio,Alarm Time HUNC Errors [ms],
All Cells,Filtered Cells Ratio [1/100 %],Load Cell Ratio [1/100 %].

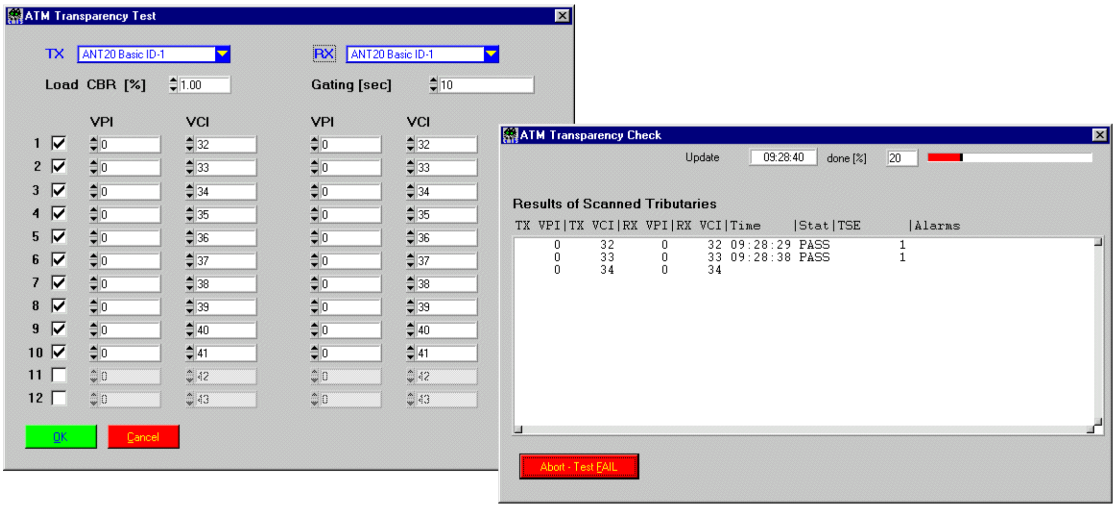
Notes:

7 Check Transparency of a List of VCI/VPI

Function Name

```
_check_vci_vpi_transparency
             .\atm\atmtrans.obj
```

User Interface



Description

Purpose: Checks the transparency of a number of user selectable VPI/VCI. For this check, an AAL0 or AAL1 signal has to be used.
Transparency is checked by transmitting a test pattern, injecting exactly one bit error at the beginning of the bit error test, and then continuing a bit error check for the complete gating time.

Caution: –

PASS / FAIL Conditions

PASS: No alarm and exactly one bit error during gating.

FAIL: More than one or no bit error, or any alarm during gating.

Test Report

```
Check ATM Transparency   FAIL
    TX VPI |TX VCI |RX VPI |RX VCI|Time      |Stat |TSE       |Alarms
    0      32      0       32     17:04:21  PASS  1
    0      36      0       36     17:04:29  PASS  1
    0      37      0       37     17:04:37  FAIL  1          LOS
    0      40      0       40     17:04:46  PASS  1
```

Applications

Check the right setup of an ATM switch by testing its ability to switch VPI/VCI's to the selected destinations.

See sequence “atmtrans.squ”.

Parameters

TX Id:	Integer, 1 .. 10
CBR Load:	Double, 0.01 .. 100.00 [%]
RX Id:	Integer, 1 .. 10
Gating:	Double, 1 .. 10 000 000 [sec]
Selector:	Flag Selected = 1, Not Selected = 0.
TX VPI:	Integer, 1 .. 4095
TX VCI:	Integer, 1 .. 65535
RX VPI:	Integer, 1 .. 4095
RX VCI:	Integer, 1 .. 65535

Remote Control of Testcase

Input:	%d,%d#%lf,%lf,#{,%1d,%4d,%5d,%4d,%5d}12# TX Id,RX Id,Gating,CBR Load{Selector,TX VPI,TX VCI,RX VPI,RX VCI}
Output:	%d,%d{,%d,%d,%d,%d,%s,%d,%lf,%s} Number of Channels,{TX VPI,TX VCI,RX VPI,RX VCI,Time,Result Status, TSE Errors,Alarms}.

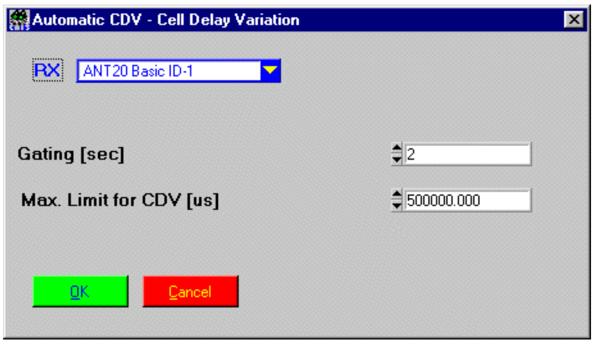
Notes:

8 Check Cell Delay Variation w/Fixed Resolution

Function Name

```
_check_auto_CDV
            .\atm\celdelay.obj
```

User Interface



Description

Purpose: Measures cell transfer delay and cell delay variation using an automatic algorithm to achieve maximum accuracy while the offset is 0 us.
To achieve even better accuracy, "_check_CDV" testcase with proper offset and resolution values should be used.
Measurement is started using the maximum resolution value (335.500 ms). Resolution is then decreased to a minimum of 0.16 us. The CDV value for the best possible resolution is displayed and evaluated.

Caution: Testcase relies on a constant, time invariant behaviour of the UUT regarding CDV.
The ATM payload must be set to 0.191 testcells.
See testcase "Set RX AAL & Scrambler" (_set_rx_atm_adaptation).

PASS / FAIL Conditions

PASS: Cell delay variation is below limit.

FAIL: Alarms during gating or cell delay variation beyond limit.

Test Report

```
Check Auto CDV              PASS
    Resolution Range:             0.16 us
    Gating [s]:                   1.000
    Max. Limit for CDV [us]:      5.600
    Cell Delay Variation   [us] =       1.60
    Mean Cell Delay        [us] =       3.98
    Min. Cell Delay        [us] =       3.36
    Max. Cell Delay        [us] =       4.96
    All Cells                   =       353207
    Errored Cells               =       0
    Alarmtime Errored Cells[s]  =       0
```

Applications

This testcase is used for a quick, automatic check to find out the mean value of a Cell Transfer Delay. The result from this testcase can be used to setup the “_check_CDV” testcase with best fitting offset and resolution parameters.

See sequence “atmdelay.squ”.

Parameters

RX Id: Integer, 1 .. 10

Gating: Double, 1 .. 10 000 000 [sec]

CDV Limit: Double, 0.00 .. 10000000.00 [us]

Remote Control of Testcase

Input: %d#%lf,%lf
RX Id,Gating,Limit

Output: %lf,%lf,%lf,%lf,%lf,%lf,%lf,%lf
Max. Delay Limit,CDV [us],Mean Cell Delay [us],Min. Cell Delay [us],
Max. Cell Delay [us],All Cells,Errored Cells,Alarmtime Errored Cells [s].

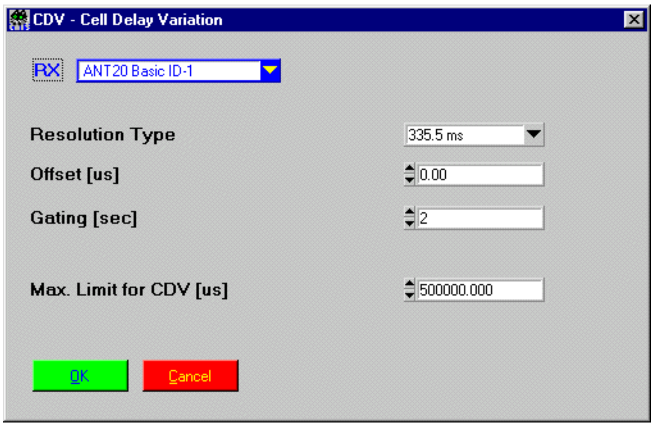
Notes:

9 Check Cell Delay Variation w/Auto Resolution

Function Name

```
_check_CDV
            .\atm\celdelay.obj
```

User Interface



Description

Purpose: Measures cell transfer delay and cell delay variation of ATM test cells. The measurement takes place with a fixed resolution and offset value. To achieve maximum accuracy it is advisable to use the “_check_auto_CDV” testcase before in order to check the expected result range.

Caution: The ATM payload must be set to 0.191 testcells.
See testcase “Set RX AAL & Scrambler” (_set_rx_atm_adaptation).

PASS / FAIL Conditions

PASS: Maximum cell delay is below threshold.

FAIL: Maximum cell delay above threshold, or cell delay value not valid (e.g. due to alarms).

Test Report

```
CDV Res 0.16 us            PASS
   Resolution Range:0.16 us
   Offset [us]:                0.000
   Gating [s]:                 1.000
   Max. Limit for CDV [us]:    2.000
   Cell Delay Variation   [us] =       1.60
   Mean Cell Delay        [us] =       3.98
   Min. Cell Delay        [us] =       3.36
   Max. Cell Delay        [us] =       4.96
   All Cells                   =       353207
   Errored Cells               =       0
   Alarmtime Errored Cells [s] =       0
```

Applications

See sequence “atmdelay.squ”.

Parameters

RX Id: Integer, 1 .. 10

Resolution: Enumeration
0.16 us = 0, 1.28 us = 1, 10.24 us = 2, 81.92 us = 3, 655.36 us = 4,
5.242 ms = 5, 41.940 ms = 6, 335.500 ms = 7.

Offset: Double, 0.00 .. 167769.60 [us]

Gating: Double, 1 .. 10 000 000 [sec]

CDV Limit: Double, 0.00 .. 10000000.00 [us]

Remote Control of Testcase

Input: %d#%d,%lf,%lf,%lf
RX Id,Resolution,Gating,Limit,Offset

Output: %lf,%lf,%lf,%lf,%lf,%lf,%lf,%lf
Max. Delay Limit,CDV [us],Mean Cell Delay [us],Min. Cell Delay [us],
Max. Cell Delay [us],All Cells,Errored Cells,Alarmtime Errored Cells [sec].

Notes:

WWG-CATS Testcase Library

Jitter Testcases

Jitter Testcases

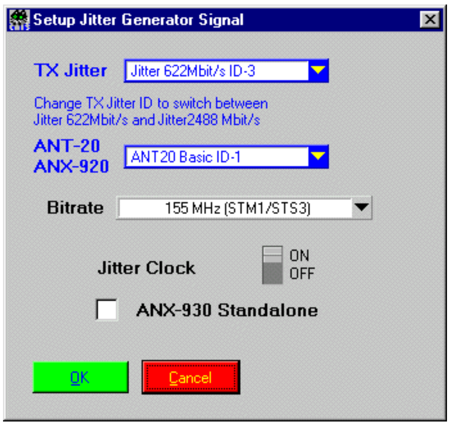
Notes:

1 Setup TX Jitter Module

Function Name

```
_setup_TX_jitter
               .\sdh_bas\jitter.obj
```

User Interface



Description

Purpose: Here you can select the bitrate for jitter measurements and jitter transfer measurements as well as the bitrate for jitter generation (FTJ, MTJ).

Caution: It is NOT sufficient to set the ANT-20 Basic RX and TX to the respective bitrates, for jitter TX and RX are handled as separate modules by CATS!

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup TX Jitter    CONTINUE
Bitrate = 155 MHz
```

Applications

–

Parameters

TX Jitter Id: Integer, 1 .. 10

BER Id: Integer, 1 .. 10

Bitrate: Enumeration
STM-1 / STS-3 = 10, STM-4 / STS-12 = 11, STS-1 =12, 2M / E1 = 30
8M / E2 = 31, 34M / E3 = 32, 140M / DS4NA = 33, 1.5M = 40, 44M = 42

Jitter Clock 1 = ON, 0 = OFF
Transmitter Clock is taken from the Jitter Module

ANX-930 only used if ANX-930 is used standalone

Remote Control of Testcase

Input: %d,%d#%d,%d,%d
TX Jitter Id,BER Id,Bitrate,Jitter Clock,ANX-930

Output: No Results

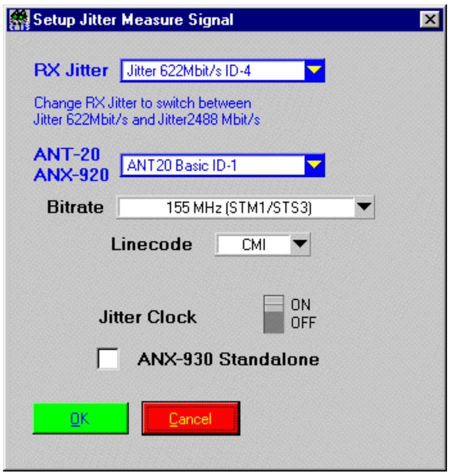
Notes:

2 Setup RX Jitter Module

Function Name

```
_setup_RX_jitter
              .\sdh_bas\jitter.obj
```

User Interface



Description

Purpose: Here you can select the bitrate for jitter measurements and jitter transfer measurements as well as the bitrate for jitter generation (FTJ, MTJ).

Caution: It is NOT sufficient to set the ANT-20 Basic RX and TX to the respective bitrates, for jitter TX and RX are handled as separate modules by CATS!

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup RX Jitter    CONTINUE
Bitrate = 155MHz,  linecode = CMI
```

Applications

–

Parameters

RX Jitter Id:	Integer, 1 .. 10
BER Id:	Integer, 1 .. 10
Bitrate:	Enumeration STM-1 / STS-3 = 10, STM-4 / STS-12 = 11, STS-1 =12, 2M / E1 = 30 8M / E2 = 31, 34M / E3 = 32, 140M / DS4NA = 33, 1.5M = 40, 44M = 42
Code:	Enumeration CMI_CCITT = 1, NRZ = 2, HDB3 = 3
Jitter Clock	ON = 1, OFF = 0
ANX-930	only used if ANX-930 is used standalone

Remote Control of Testcase

Input:	%d#%d,%d RX Jitter Id,Bitrate,Code
Output:	No Results

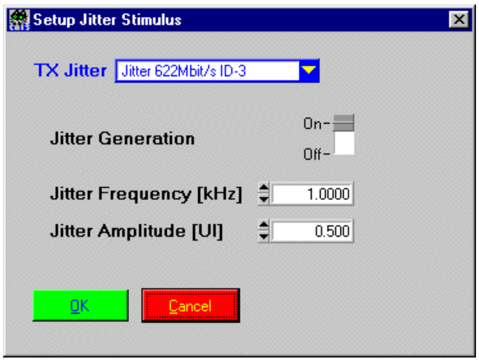
Notes:

3 Insert Jitter

Function Name

```
_setup_jitter_stimulus
               .\sdh_bas\jitter.obj
```

User Interface



Description

Purpose: Setup jitter stimulus.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup Jitter                  CONTINUE
    Frequency = 1.000 kHz,  amplitude = 0.500 UI
```

Applications

–

Parameters

TX Jitter Id:	Integer, 1 .. 10
TX Id:	Integer, 1 .. 10
Switch:	Enumeration Jitter generation ON = 1, Jitter generation OFF = 0
Frequency:	Double, 0.002 .. 5000.000 [Hz]
Amplitude:	Double, 0.002 .. 256.000

Remote Control of Testcase

Input:	%d#%d,%lf,%lf TX Jitter Id,TX Id,Switch,Frequency,Amplitude
Output:	No Results

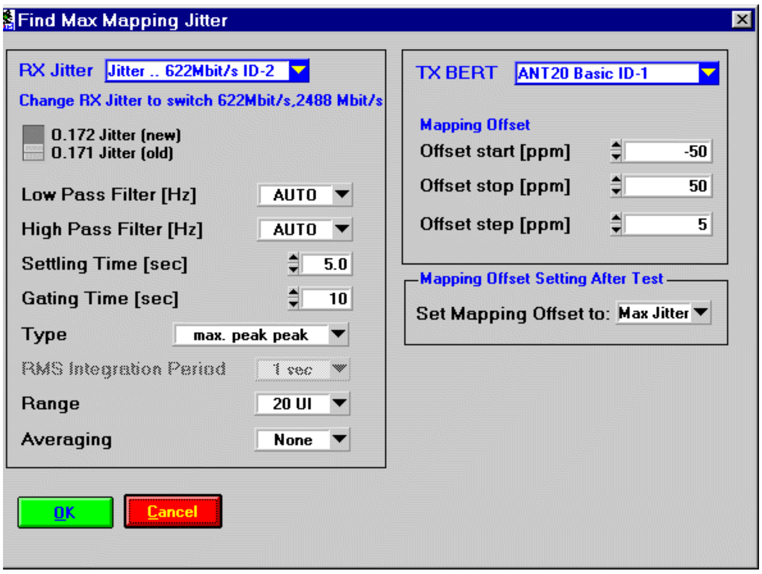
Notes:

4 Find Max. Mapping Jitter

Function Name

```
_find_max_mapping_jitter
              .\sdh_bas\mapp_jit.obj
```

User Interface



Description

Purpose Find critical mapping offset which leads to a jitter maximum.
It starts with the mapping offset start, measures jitter and step to the next mapping offset and measures jitter ... until the stop mapping offset is reached. At the end the result is the most critical offset. Mapping offset after test is selectable to set it to critical mapping offset or to set it to 0 ppm.

Caution: This testcase requires for each step settling time + gating time. If you have defined a big range it may take some minutes.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Low pass filter:   AUTO
High pass filter:  AUTO
Range:             1.6 UI
Averaging:         OFF
Gating:            10 s
Settling Time:     5.0 s
```

Applications

See the following test sequences in the Operating Manual:

- i_mj_e1.squ
- a_mj_ds1.squ
- i_cj_e1.squ
- a_cj_ds1.squ

Parameters

rx_Id	Integer 1..10
settlingTime	Double, 0 .. 99.9 [s]
tx_BERT_Id	Integer 1..10
lpFreq	Enumeration 40 KHz = 40000, 60 KHz = 60000, 100 KHz = 100000, 400 KHz = 400000, 800 KHz = 800000, 1300 KHz = 1300000, 3500 KHz = 3500000, 5000 KHz = 5000000
hpFreq	Enumeration 2 Hz = 2, 4 Hz = 4, 10 Hz =10, 20 Hz = 20, 40 Hz = 40, 100 Hz = 100, 200 Hz = 200, 400 Hz =400, 500 Hz = 500, 700 Hz = 700, 1 KHz = 1000, 3 KHz = 3000, 8 KHz = 8000, 10 KHz = 10000, 12 KHz = 12000, 18 KHz = 18000, 20 KHz = 20000, 30 KHz = 30000, 65 KHz = 65000, 80 KHz = 80000, 250 KHz = 250000
gating	Integer, 1 .. 10 000 [s]
typeOfJitter	Enumeration Max. Peak - Peak = 1, Peak Peak = 2, RMS = 3, Positive Peak = 4, Negative Peak = 5
range	Enumeration 1.6 UI = 1.6, 2 UI = 2, 6.4 UI = 6.4, 20 UI = 20, 80 UI = 80, 200 UI = 200, 800 UI = 800
averaging	Enumeration Off = 0, 1 s = 1, 2 s = 2, 3 s = 3, 4 s =4, 5 s = 5
rms_integ_per	Enumeration 1 = 1 s, 2 = 2 s, 5 = 5 s, 10 = 10 s, 20 = 20 s, 40 = 40 s, 80 = 80 s
start_offset	start ppm -100 .. 100 ppm
stop_offset	stop ppm -100 .. 100 ppm
step_offset	step ppm -100 .. 100 ppm
end_setting_type	0 = set to 0 ppm after test 1 = set to critical mapping offset which was found by this testcase

Remote Control of Testcase

Input:	"%d,%d#%ld,%ld,%lf,%d,%d,%d,%d,%lf,%d,%d,%d,%d,%d", rx_Id,tx_BERT_Id,lpFreq,hpFreq,settlingTime, gating,typeOfJitter,range,averaging,rms_integ_per,start_offset, stop_offset,step_offset,end_setting_type
Output:	"%d,%.3e,%d", status,max_jitter,max_mapping_offset

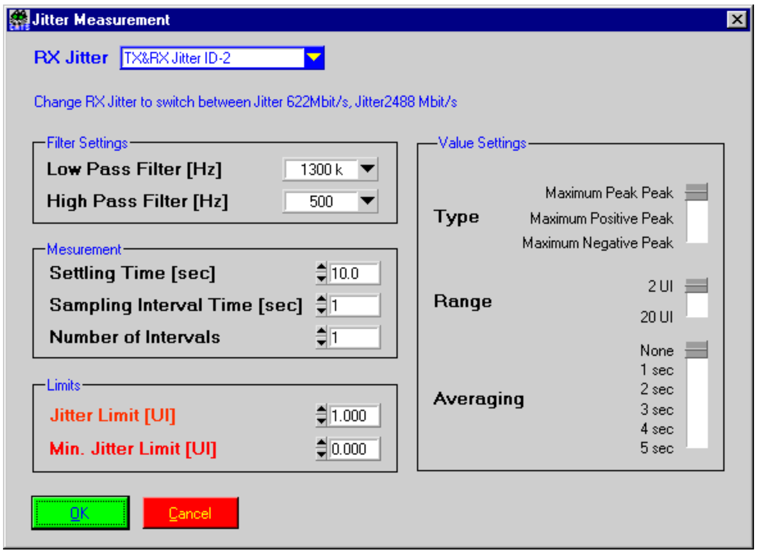
Notes:

5 Check RX Jitter

Function Name

```
_check_jitter
              .\sdh_bas\jitter.obj
```

User Interface



Description

Purpose: Jitter measurement, e.g. in the presence of TX stimuli like offset, jitter, pointer bursts that had been set by preceding testcases.
This testcase can be preceded by a testcase that generates one or more of the above mentioned stimuli.

Caution: “Settling Time” will elapse once before the first jitter measurement takes place. It is advisable to allow for a settling time of at least 12 s to get a fully valid first measurement.

PASS / FAIL Conditions

PASS: Testcase assumed PASS if max. jitter was not reached and no alarm found for any of the measured intervals.

FAIL: Measured jitter value beyond limit or alarm found during measurement.

Test Report

```
Jitter Measurement    PASS
    Max. Limit =      0.20 UI
    Min. Limit =      0.00 UI
    peak-peak =       0.002 UI:PASS
```

Applications

–

Parameters

RX Jitter Id:	Integer, 1 .. 10
Type of Jitter:	Enumeration Peak - Peak = 1, Positive Peak = 2, Negative Peak = 3
Range:	Enumeration 2 UI = 2, 20 UI = 20
Averaging:	Enumeration Off = 0, 1 s = 1, 2 s = 2, 3 s = 3, 4 s =4, 5 s = 5
HP Filter:	Enumeration 2 Hz = 2, 4 Hz = 4, 10 Hz =10, 20 Hz = 20, 40 Hz = 40, 100 Hz = 100, 200 Hz = 200, 400 Hz =400, 500 Hz = 500, 700 Hz = 700, 1 KHz = 1000, 3 KHz = 3000, 8 KHz = 8000, 10 KHz = 10000, 12 KHz = 12000, 18 KHz = 18000, 20 KHz = 20000, 30 KHz = 30000, 65 KHz = 65000, 80 KHz = 80000, 250 KHz = 250000
LP Filter:	Enumeration 40 KHz = 40000, 60 KHz = 60000, 100 KHz = 100000, 400 KHz = 400000, 800 KHz = 800000, 1300 KHz = 1300000, 3500 KHz = 3500000, 5000 KHz = 5000000
Intervals:	Integer, 1 .. 999
Settling Time:	Double, 0.0 .. 20.0 sec
Samp. Interval:	Integer, 1 .. 10 [s]
Min. Limit:	Double, 0.00 .. 99.00 [UI]
Max. Limit:	Double, 0.00 .. 99.00 [UI]

Remote Control of Testcase

Input:	%d#%ld,%ld,%lf,%d,%d,%d,%d,%d,%lf,%lf RX Jitter ID,LP Filter,HP Filter,Settling Time,Samp. Interval,Intervals, Type of Jitter,Range,Averaging,Max. Limit,Min. Limit
Output:	%d{,%lf,%d} Intervals,{Result,ResultStatus}

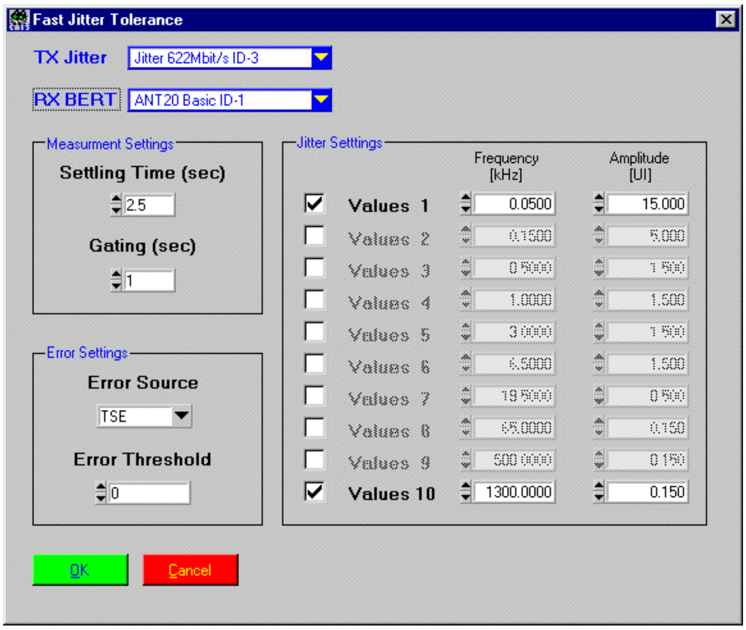
Notes:

6 Fast Jitter Tolerance Check

Function Name

```
_check_jitter_tolerance
              .\sdh_bas\jitter.obj
```

User Interface



Description

Purpose: Up to 20 user selectable values for jitter amplitude / jitter frequency can be set. A bit error check is performed at each of these points.

Caution: Make sure to use a suitable settling time between measurements (will elapse after jitter generator is switched to new value and prior to starting the measurement).

PASS / FAIL Conditions

PASS: Testcase is assumed PASS, if no bit error and no alarm occurred during the measurement period.

FAIL: Bit errors or alarms occurred during measurement.

Test Report

```
FTJ                          PASS
    Threshold of TSE =       0
    0.050 kHz, 15.00 UI      : PASS
    1300.000 kHz, 0.15 UI    : PASS
```

Applications

–

Parameters

TX Jitter Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
TX Id:	Integer, 1 .. 10
Settling Time:	Double, 1.0 .. 99.0 [s]
Gating:	Integer, 1 .. 999 [s]
Error Source:	Enumeration TSE = 0, HP-FEBE = 15, LP-FEBE = 17, E-Bit = 47
Err. Threshold:	Integer, 0 .. 999
Values (10 pairs)	
Flag:	Enumeration Off = 0, On = 1
Frequency:	Double, 0.002 .. 4500.000 [KHz]
Amplitude:	Double, 0.00 .. 20.00 [UI]

Remote Control of Testcase

Input:	%d,%d#%lf,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%ld,%ld,%ld,%ld, %ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%hd TX Jitter Id, RX Id,TX Id,Settling Time,Gating,Error Source,Flag 1, .., Flag 10, Frequency 1, ..., Frequency 10, Ampltude 1, ..., Amplitude 10, Err. Threshold
Output:	%d{,%ld,%d} Number of Points,{Frequency [Hz],Result Status}
Result Status:	Enumeration Valid = 1, Greater than = 2, Less than = 3, Fail valid = -1, Fail greater than = -2, Fail less than = -3, Overflow = -65, Underflow = -66, Unvalid = -67, Alarm or error = -68, Err. reference = -129

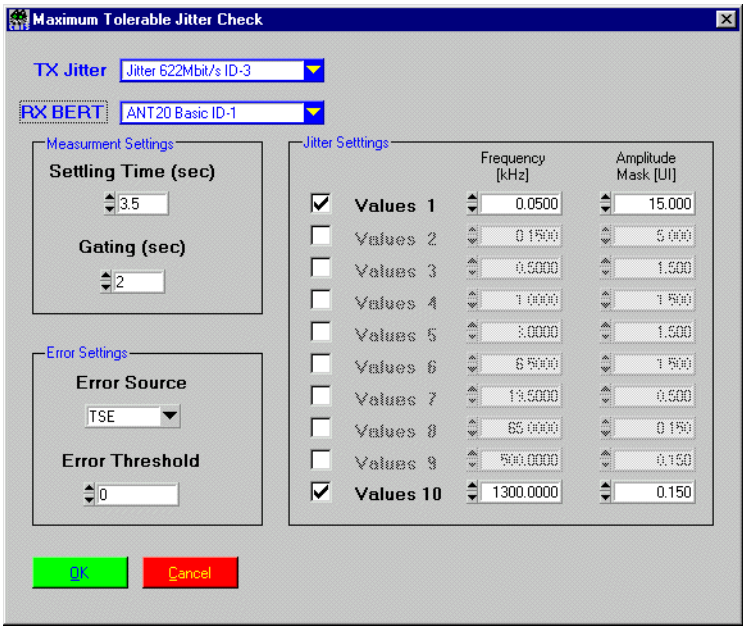
Notes:

7 Maximum Jitter Tolerance Measurement

Function Name

```
_check_mtj
              .\sdh_bas\jitter.obj
```

User Interface



Description

Purpose: Up to 10 user selectable values for jitter frequencies can be set. A bit error check is performed at each of these points for the given expected maximum jitter amplitude.
Measurement is started using the “Mask” value of jitter amplitude.
If no bit error and no alarm occurr during the measurement period (“mask” value), jitter is increased until an error or alarm is found. The jitter amplitude below the one that caused an error or alarm is considered the MTJ result.

If a bit error or alarm occurred during the first measurement period, jitter is decreased until no error or alarm is found. This value (frequency and amplitude) is considered the MTJ result.

Caution: Make sure to use a suitable settling time between measurements.

PASS / FAIL Conditions

PASS: No bit error and no alarm occurres during first measurement period (“mask” value).

FAIL: Bit error or alarm occurres during first measurement period (“mask” value).

Test Report

```
MTJ                           PASS
    Threshold of TSE          =0
    0.050 kHz, >64.00 UI     : PASS
    1300.000 kHz,   0.59 UI  : PASS
```

Applications

–

Parameters

TX Jitter Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
TX Id:	Integer, 1 .. 10
Settling Time:	Double, 0.0 .. 10.0 [s]
Gating:	Integer, 1 .. 10 [s]
Error Source:	Enumeration TSE = 0, HP-FEBE = 15, LP-FEBE = 17, E-Bit = 47
Err. Threshold:	Integer, 0 .. 999
Values (10 pairs)	
Flag:	Enumeration Off = 0, On = 1
Frequency:	Double, 0.002 .. 4500.000 [KHz]
Amplitude:	Double, 0.00 .. 64.00 [UI]

Remote Control of Testcase

Input:	%d,%d#%lf,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%ld,%ld,%ld,%ld, %ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%hd TX Jitter Id, RX Id,TX Id,Settling Time,Gating,Error Source,Flag 1, .., Flag 10, Frequency 1, ..., Frequency 10, Ampltude 1, ..., Amplitude 10,Err. Threshold
Output:	%d{,%ld,%lf,%d} Number of Points,{Frequency [Hz],Amplitude [UI],Result Status}
Result Status:	Enumeration Valid = 1, Greater than = 2, Less than = 3, Fail valid = -1, Fail greater than = -2, Fail less than = -3, Overflow = -65, Underflow = -66, Unvalid = -67, Alarm or error = -68, Err. reference = -129

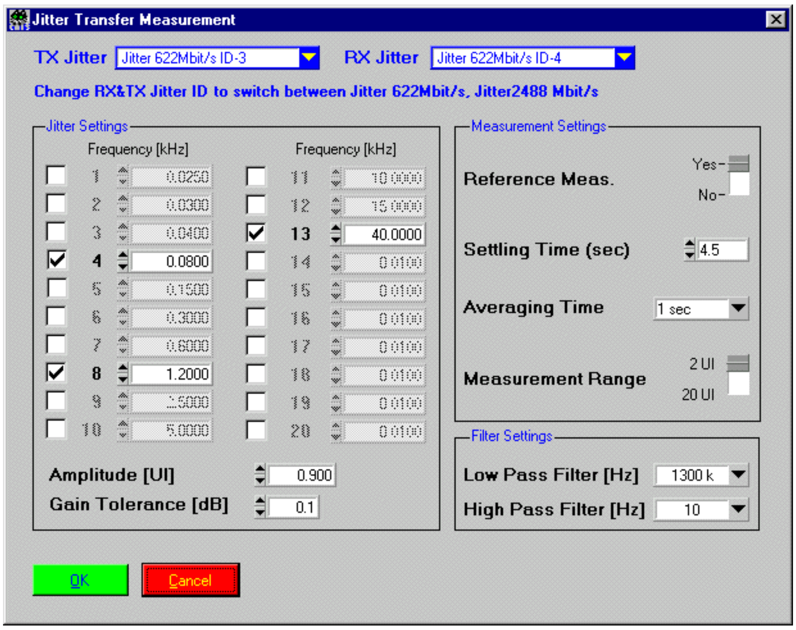
Notes:

8 Jitter Transfer Function

Function Name

```
_check_jitter_transfer
              .\sdh_bas\jitter.obj
```

User Interface



Description

Purpose: A fixed jitter amplitude is set at TX. Jitter frequency list is user selectable (20 values).
Jitter amplitude at RX is measured for each frequency point. Reference Measurement (Yes/No), used to eliminate any error caused by the measurement setup. Values will be stored for all following runs of this testcase. When a sequence is loaded, reference results are deleted.

Caution: No complex tolerance curve is given. To evaluate against such tolerances, use spreadsheet programs, e.g. Microsoft EXCEL. To carry out JTF measurement with changing amplitude values, call this testcase more than once with respective values.

PASS / FAIL Conditions

PASS: Testcase is assumed PASS if no alarm occurs during measurement of jitter amplitudes and if maximum gain tolerance is below the tolerance limit.

FAIL: Alarm occurs during jitter measurement or maximum gain tolerance is beyond tolerance limit.

Test Report

```
JTF                         PASS
    0.080 kHz,  -0.059 dB   : PASS
    1.200 kHz,  -0.020 dB   : PASS
    40.000 kHz, 0.020 dB    : PASS
```

Applications

–

Parameters

TX Jitter Id:	Integer, 1 .. 10
RX Jitter Id:	Integer, 1 .. 10
TX Id:	Integer, 1 .. 10
Settling Time:	Double, 0.0 .. 10.0 [s]
Type of Jitter:	Enumeration Peak - Peak = 1
Tolerance:	Double, -99.0 .. 9.9 [dB]
Averaging:	Enumeration Off = 0, 1 s = 1, 2 s = 2, 3 s = 3, 4 s = 4, 5 s = 5
Range:	Enumeration 2 UI = 2, 20 UI = 20
LP Filter:	Enumeration 40 KHz = 40000, 60 KHz = 60000, 100 KHz = 100000, 400 KHz = 400000, 800 KHz = 800000, 1300 KHz = 1300000, 3500 KHz = 3500000, 5000 KHz = 5000000
HP Filter:	Enumeration 2 Hz = 2, 4 Hz = 4, 10 Hz = 10, 20 Hz = 20, 40 Hz = 40, 100 Hz = 100, 200 Hz = 200, 400 Hz = 400, 500 Hz = 500, 700 Hz = 700, 1 KHz = 1000, 3 KHz = 3000, 8 KHz = 8000, 10 KHz = 10000, 12 KHz = 12000, 18 KHz = 18000, 20 KHz = 20000, 30 KHz = 30000, 65 KHz = 65000, 80 KHz = 80000, 250 KHz =250000
Amplitude:	Double, 0.1 .. 9.9 [UI]
Ref. Meas.:	Enumeration Take reference measurement = 1, No reference measurement = 0

Remote Control of Testcase

Input:	%d,%d#%ld,%ld,%lf,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld,%lf TX Jitter Id,RX Jitter Id,TX Id,LP Filter,HP Filter,Settling Time,Type of Jitter, Range,Averaging,Ref. Meas.,Flag 1, ...,Flag 20,Frequency 1, ...,Frequency 20, Amplitude,Tolerance
Output:	%d{,%ld,%lf,%d} Number of Points,{Frequency,Jitter Gain [dB],Result Status}
Result Status:	Enumeration Valid = 1, Greater than = 2, Less than = 3, Fail valid = -1, Fail greater than = -2, Fail less than = -3, Overflow = -65, Underflow = -66, Unvalid = -67, Alarm or error = -68, Err. reference = -129

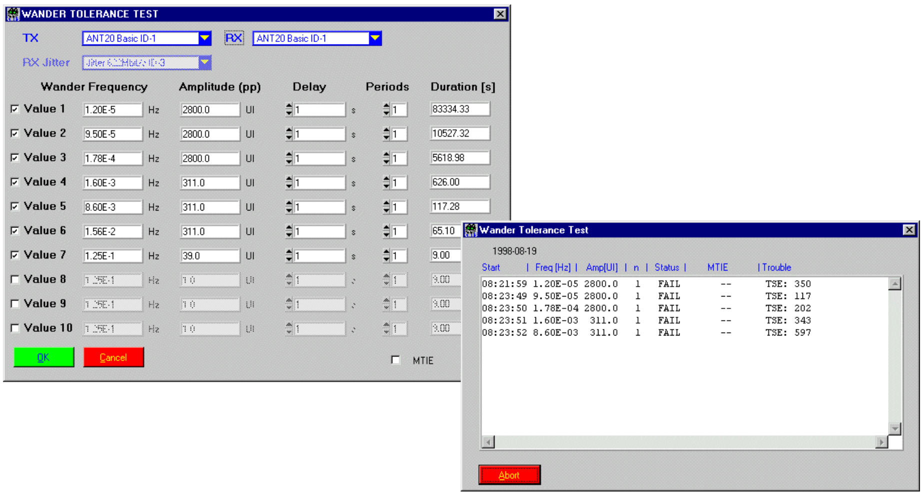
Notes:

9 Check Wander Tolerance

Function Name

```
_check_wander_tolerance
               .\sdh_bas\wander.obj
```

User Interface



Description

Purpose: Wander can be applied sequentially at ten different “frequency / UI” points. A bit error test (TSE errors) and alarm check are carried out for each individual testpoint.
Wander is applied in an approximated sinusoidal way, by slowly changing the ANT-20’s TX signal against an external reference clock signal. A table allows to enter the individual testpoints by giving the wander amplitude value in [Unit Intervals] peak-peak and the respective wander frequency in [Hz]. Due to the very low frequencies used, the number of wander periods for each given frequency is limited to five.
The duration field automatically calculates the estimated time for each single testpoint run. This estimate includes the count of periods as well as the preceding delay time. The delay time allows the UUT circuitry to get into a stable state before the next wander stimulus is applied.
The signal bitrate is read from the ANT-20 automatically, so that it is not necessary to set it with the wander tolerance testcase. The testcase “_set_signal_structure” should preceed testcase “ _check_wander_tolerance” to set the desired bitrate.

Caution: It is absolutely necessary to synchronize the ANT-20 to an external reference clock signal. You should use testcases “ _set_tx_adaptation2” and “_set_rx_adaptation2” for the proper setup of synchronization.

PASS / FAIL Conditions

PASS: No alarms and no TSE errors during the complete measurement.

FAIL: Any alarms or TSE errors.Bit errors or alarms occurred during measurement.

Test Report

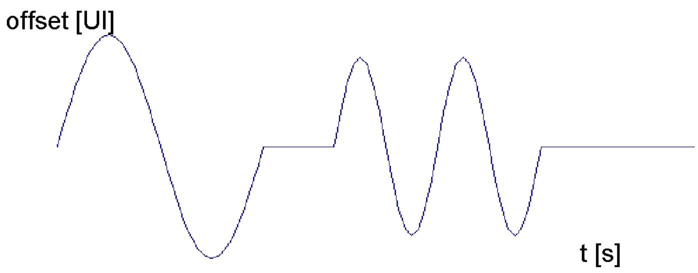
```
Wander Tolerance           FAIL
    Start      | Freq[Hz]|        | n   | Status   | Trouble
                          Amp[UI]
    ---------- +-------- +------- +---- +-------- +---------
    07:54:11   1.25E-01   10.0    1     FAIL       Alarm: AU-AIS 0.3 s
    07:54:21   6.00E-02  100.0    1     PASS       -
    07:54:41   1.00E-01  200.0    1     FAIL       TSE: 194
    07:54:52   1.25E-01   40.0    2     FAIL       Alarm: MS-AIS 3.5 s
    07:55:10   1.25E-01   50.0    1     FAIL       Alarm: AU-AIS 1.0 s
    07:55:23   1.25E-01   10.0    1     FAIL       Alarm: AU-LOP 0.5 s
```

Applications

This testcase may be used to test the wander tolerance of SDH, SONET or PDH network elements. Any ANT-20 can be used to carry out a wander tolerance test; the Jitter/Wander option is not required to run this testcase.

If an ANT-20 with Wander measurement option is available, an additional MTIE measurement function can be added to this testcase: if the MTIE field of the user interface is checked, the accumulated maximum time interval error for each wander frequency/UI pair will be measured and shown in the result display.

Typical TX wander output (f1 = 1 period, f2 = 2 periods):



Parameters

TX Id:	Integer, 1 .. 10
RX Id:	Integer, 1 .. 10
RX Jitter Id:	Integer, 1 .. 10
Values (10 records)	
Flag:	Enumeration Off = 0, On = 1
Frequency:	Double, 0.000012 .. 0.125 [Hz]
Amplitude:	Double, 0.0 .. 999999.0 [UI]
Delay:	Double, 1.0 .. 99999.0 [s]
Periods:	Double, 1 .. 5
MTIE Flag:	Enumeration Off = 0, On =1

Remote Control of Testcase

Input:	%d,%d,%d#{,%lf,%lf,%lf,%lf,%lf,}%d,%d TX Id,RX Id,RX Jitter Id{,Flag,Frequency,Amplitude,Delay,Periods,}MTIE Flag
Output:	{%d,%e,%e,%e,} {Status,Alarm Time [s],Bit error count,MTIE value}
Status:	Enumeration FAIL = 0, PASS =1, Not valid (not selected) = -1.

Notes:

WWG-CATS Testcase Library

Optics Testcases
(only available with BN 3045/93.41
or BN 3045/93.42)

Optics Testcases

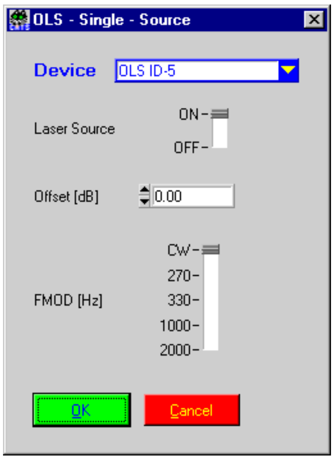
Notes:

1 Set Single Laser Source

Function Name

```
_setup_OLS
                .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Sets the parameter of the single laser source.

Caution: Only for single sources. For dual laser sources use _setup_OLS_WDM instead.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup OLS      PASS
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Flag:	Enumeration Laser Off = 0, Laser On = 1
Offset:	Double, -0.5 ... 0.5 [dB]
FMOD:	Integer, 0,270,330,1000,2000 [Hz]

Remote Control of Testcase

Input:	%d#%d,%lf,%d Id,Flag,Offset,FMOD
Output:	No Results

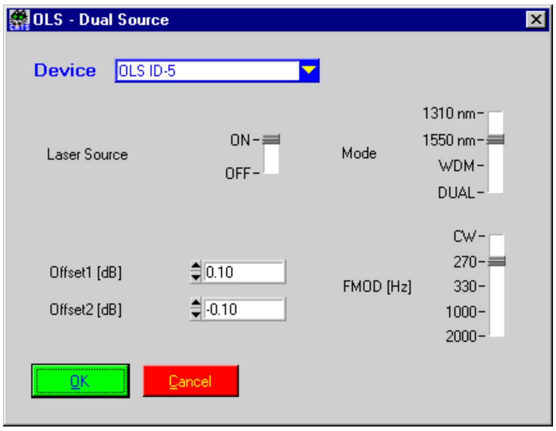
Notes:

2 Set Dual Laser Source

Function Name

```
_setup_OLS_WDM
                .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Sets the parameter of the dual laser source.

Caution: Only for dual sources. For single laser sources use _setup_OLS_WDM instead.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup OLS WDM     PASS
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Flag:	Enumeration Laser Off = 0, Laser On = 1
Mode:	Enumeration 1310 nm = 1, 1550 nm = 2, WDM = 3, Dual = 4
Offset1:	Double, -0.5 ... 0.5 [dB]
Offset2:	Double, -0.5 ... 0.5 [dB]
FMOD:	Integer, 0,270,330,1000,2000 [Hz]

Remote Control of Testcase

Input:	%d#%d,%d,%lf,%lf,%d Id,Flag,Mode,Offset1,Offset2,FMOD
Output:	No Results

Note: For the use of the WDM mode refer to the OLS user manual.

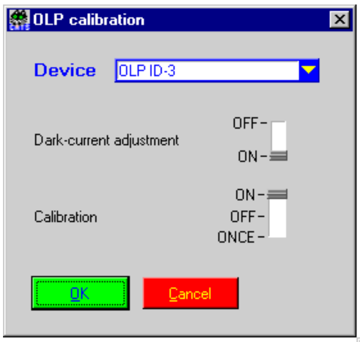
Notes:

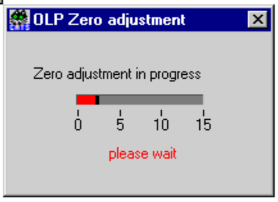
3 Initialize OLP Zeroing (not for OLP-90)

Function Name

```
_init_OLP
               .\optic\ovxoptic.obj
```

User Interface





Description

Purpose: Initializes the optical power meter calibration and starts zeroing.

Caution: Not for OLP-90.

PASS / FAIL Conditions

PASS: Not applicable

FAIL: Not applicable

Test Report

```
ini OLP                        PASS
    OLP Dark-current adjusted
    OLP automatic calibration ON
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Dark-curr. adj.:	Enumeration On = 1, Off = 0
Calibration:	Enumeration ON =1, OFF = 2, ONCE = 3

Remote Control of Testcase

Input:	%d#%d,%d Id,Dark-Current Adjustment,Calibration
Output:	No Results

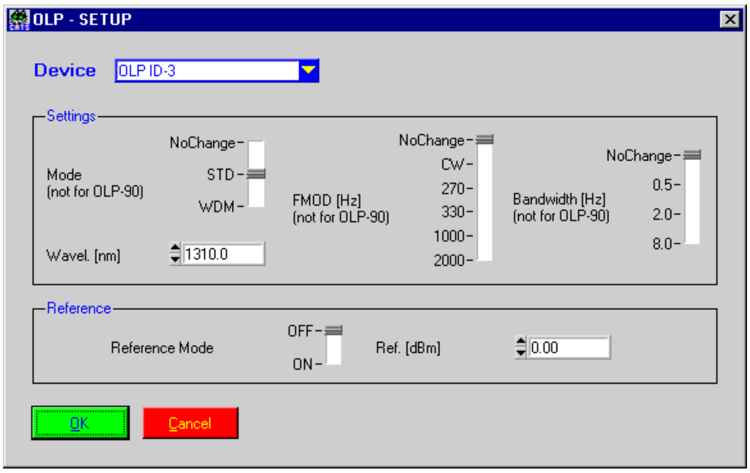
Notes:

4 Set Optical Power Meter

Function Name

```
_setup_OLP
                .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Sets the parameter of the optical power meter.

Caution: The parameter Mode, FMOD and Bandwidth could not be used with the OLP-90, therefore set the parameter to NoChange.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup OLP       PASS
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Mode:	Enumeration NoChange = 1, STD = 2, WDM = 3
Wavelength:	Double, 1260.0 .. 1600.0 [nm]
Ref. Mode:	Enumeration On = 1, Off = 0
Ref. Level:	Double, -120.0 .. 120.0 [dBm]
FMOD:	Integer, 0, 270, 330, 1000, 2000 [Hz]
Bandwidth:	Integer, 0, 5, 20, 80 [Hz/10]

Remote Control of Testcase

Input:	%d#%d,%lf,%d,%lf,%d,%d Id,Mode,Wavelength,Ref.Mode,Ref. Level,FMOD,Bandwidth
Output:	No Results

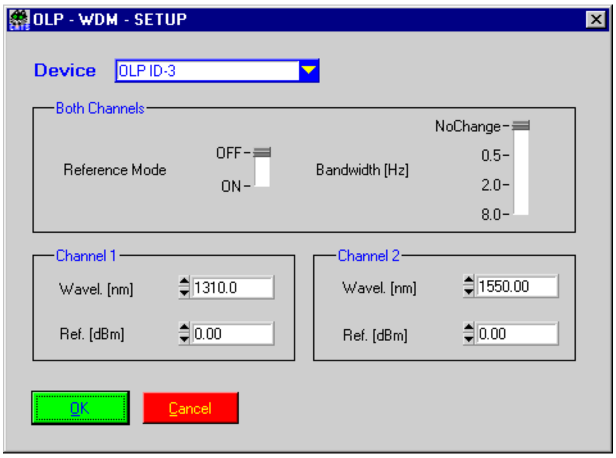
Notes:

5 Set Optical Power Meter in WDM Mode

Function Name

```
_setup_OLP_WDM
                .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Sets the parameter of the optical power meter in WDM mode.

Caution: Could not be used with the OLP-90.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup OLP     PASS
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Wavelength1:	Double, 1260.0 .. 1600.0 [nm]
Wavelength2:	Double, 1260.0 .. 1600.0 [nm]
Ref. Mode:	Enumeration On = 1, Off = 0
Ref. Level1:	Double, -120.0 .. 120.0 [dBm]
Ref. Level2:	Double, -120.0 .. 120.0 [dBm]
Bandwidth:	Integer, 0,5,20,80[Hz/10]

Remote Control of Testcase

Input:	%d#%lf,%lf,%d,%lf,%lf,%d Id,Wavelength1,Wavelength2,Ref.Mode,Ref. Level1,Ref. Level2,Bandwidth
Output:	No Results

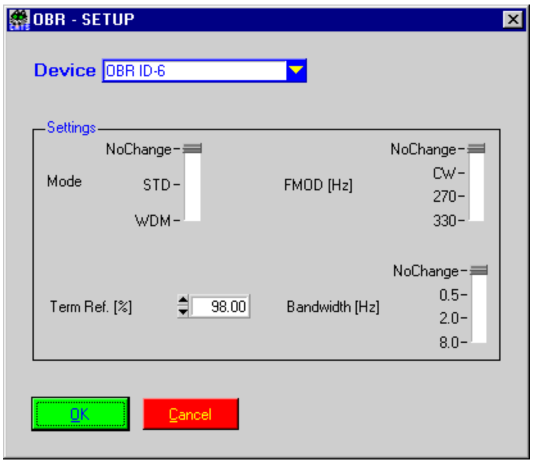
Notes:

6 Set Optical Back Reflection Meter

Function Name

```
_setup_OBR
            .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Sets the parameter of the optical back reflection meter.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup OBR        PASS
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Mode:	Enumeration NoChange = 1, STD = 2, WDM = 3
FMOD:	Integer, 9999 = NoChange; 0, 270, 330[Hz]
Term Ref:	Double, 0 ... 100[%]
Bandwidth:	Integer, 9999 = NoChange; 0,5,20,80[Hz/10]

Remote Control of Testcase

Input:	%d#%d,%d,%lf,%d Id,Mode,FMOD,Term Ref.,Bandwidth
Output:	No Results

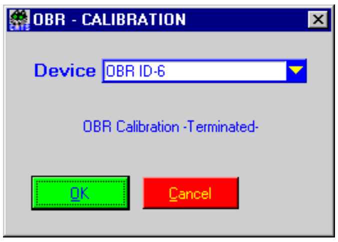
Notes:

7 Calibrate OBR -terminated-

Function Name

```
_cal_term_OBR
              .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Calibrates the optical back reflection meter.

Caution: Ending must be terminated.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Calibrate OBR      PASS
```

Applications

–

Parameters

Id:	Integer, 1 .. 10

Remote Control of Testcase

Input:	%d# Id
Output:	No Results

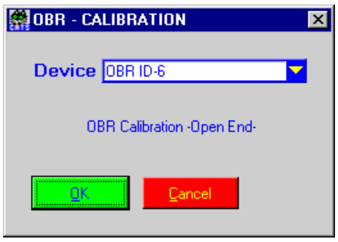
Notes:

8 Calibrate OBR -open port-

Function Name

```
_cal_term_OBR
               .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Calibrates the optical back reflection meter.

Caution: Ending must be open.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Calibrate OBR         PASS
```

Applications

–

Parameters

Id: Integer, 1 .. 10

Remote Control of Testcase

Input: %d#
Id

Output: No Results

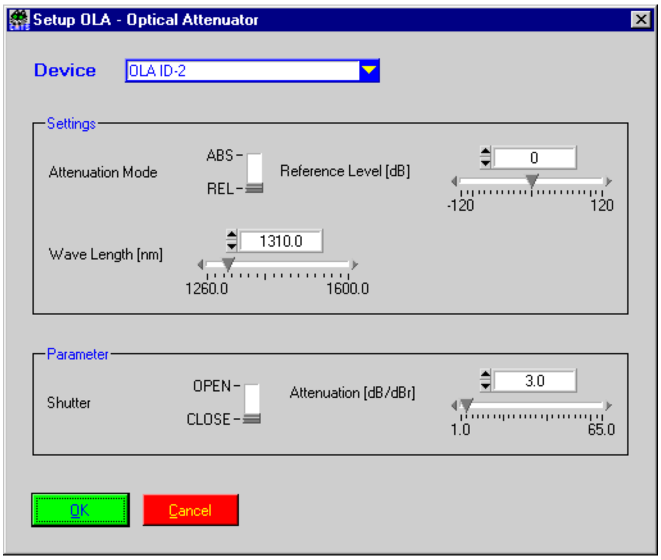
Notes:

9 Set Optical Attenuation

Function Name

```
_setup_OLA
               .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Sets the parameter of the optical attenuator.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup OLA                PASS
   Wave Length:          1310.0 nm
   Attenuation:          3.0 dB/W
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Shutter:	Enumeration ON =1, OFF = 0
Mode:	Enumeration Absolute = 1, Relative = 0
Ref. Level:	Double, -120.0 .. 120.0 [dB]
Wavelength:	Double, 1260.0 .. 1600.0 [nm]
Attenuation:	Double, 0.8 .. 65.0 [dB/dBr]

Remote Control of Testcase

Input:	%d#%d,%d,%lf,%lf,%lf Id,Shutter,Mode,Wavelength,Ref. Level,Attenuation
Output:	No Results

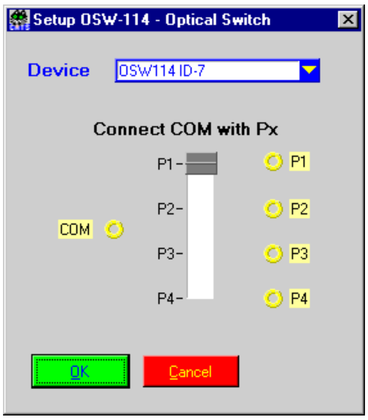
Notes:

10 Set Optical Switch (OSW114)

Function Name

```
_setup_OSW114
                .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Sets the optical switch OSW114. Selects one out of four.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup OSW114     PASS
   Pos P2
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Position:	Enumeration P1 = 1, P2 = 2, P3 = 3, P4 = 4

Remote Control of Testcase

Input:	%d#%d Id,Position
Output:	No Results

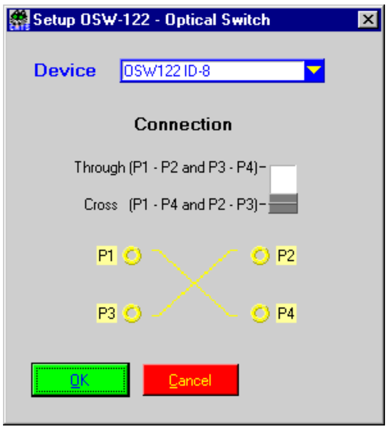
Notes:

11 Set Optical Switch (OSW122)

Function Name

```
_setup_OSW122
                .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Sets the optical cross switch OSW122.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup OSW122                    PASS
    Cross (P1 - P4 and P2 -
    P3)
```

Applications

–

Parameters

Id: Integer, 1 .. 10

Position: Enumeration
(P1 <-> P2 &P3 <-> P4) = 1, (P1 <-> P4 &P2 <-> P3) = 2

Remote Control of Testcase

Input: %d#%d
Id,Position

Output: No Results

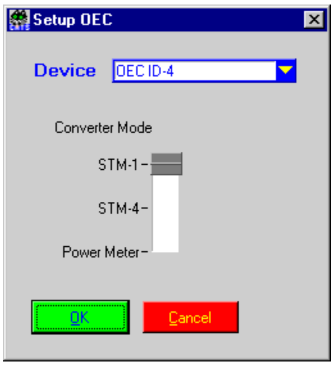
Notes:

12 Set Optical - Electrical Converter

Function Name

```
_setup_OEC
             .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Sets the parameter of the optical / electrical converter.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
setup_OEC                      PASS
     Convert to STM-1 Electric
```

Applications

–

Parameters

Id: Integer, 1 .. 10

Position: Enumeration
STM-1 = 1, STM-4 = 2, Power Meter = 3

Remote Control of Testcase

Input: %d#%d
Id,Position

Output: No Results

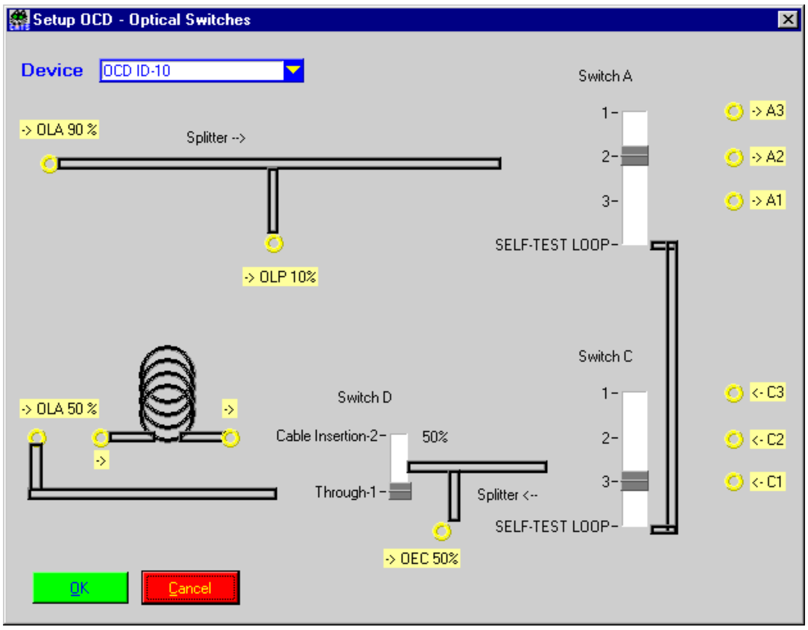
Notes:

13 Set Optical Customer Module

Function Name

```
_setup_OCD
                .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Sets the optical switches of the OCD.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Setup  OCD   A1C1       CONTINUE
A      C     D
1      1     1
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Switch D:	Enumeration Through = 1, Cable Insertion = 2
Switch A:	Enumeration ->A3 = 1, ->A2 = 2, ->A1 = 3, Self-Test Loop = 4
Switch C:	Enumeration <-C3 = 1, <-C2 = 2, <-C1 = 3, Self-Test Loop = 4

Remote Control of Testcase

Input:	%d#%d,%d,%d Id,Switch D,Switch A,Switch C
Output:	No Results

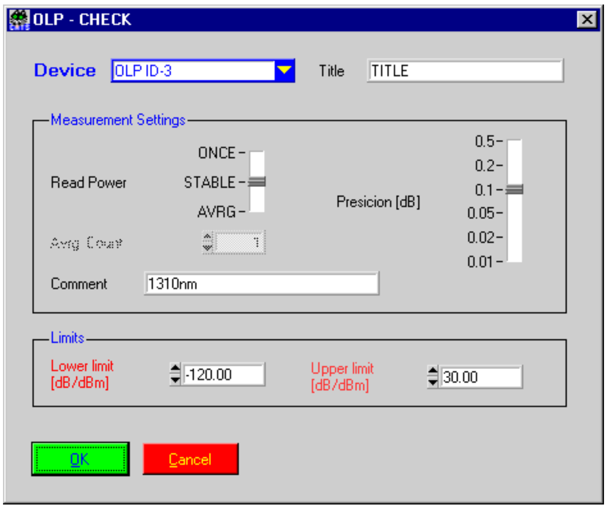
Notes:

14 Check Optical Power

Function Name

```
_check_OLP
            .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Measures the optical power and checks the result against the given limits.

Caution: –

PASS / FAIL Conditions

PASS: Optical power value is within the given limits.

FAIL: Power value beyond limits or value not stable in the mode ‘Stable’.

Test Report

```
Check OLP                    PASS
     Title
     comment (1310 nm) level: 0.10 dBm
```

Applications

The testcase supports three measure modes:

Once: One single measurement is performed.

Stable: Several measurements are performed until the result value is in the range defined with the parameter *Precision*. Only the last stable value is evaluated. If the result value is not stable after 25 measurements the testcase will stop with a FAIL condition. *Precision* defines the upper and lower deviation.

Average: Several measurement up to the number defined in the parameter *Average Count* are taken. The average value is calculated and evaluated.

Parameters

Id:	Integer, 1 .. 10
Mode:	Enumeration Once =1, Stable = 2, Average = 3
Precision:	Enumeration 0.5 dB = 0.500, 0.2 dB = 0.200, 0.1 dB = 0.100, 0.05 dB = 0.050, 0.02 dB = 0.020
Average Cnt.:	Integer, 1 ... 30
Upper Limit:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Lower Limit:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Title:	String, Title for measurement
Comment:	String, Comment

Remote Control of Testcase

Input:	%d#%d,%lf,%d,%lf,%lf,#%s#,#%s# Id,Mode,Precision,Average Count,Upper Limit,Lower Limit,Title,Comment
Output:	%d,%lf Valid?,Power Valid?: Valid = 1, Not valid = 0

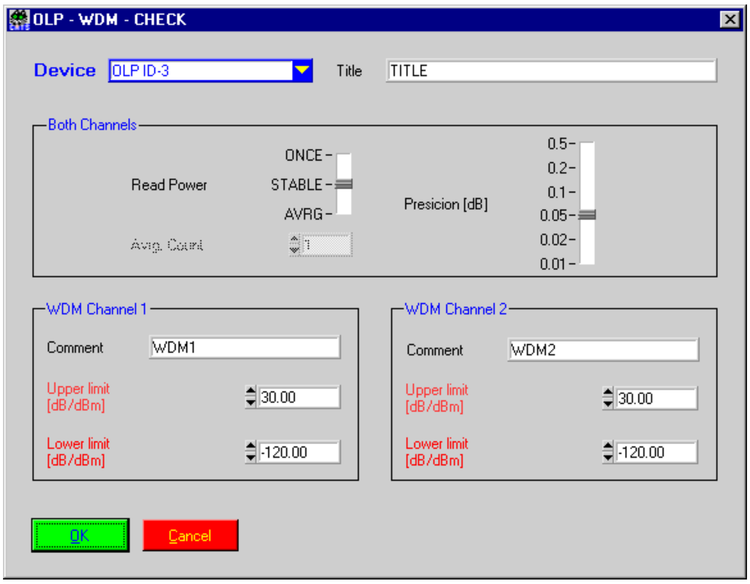
Notes:

15 Check Optical Power in WDM Mode

Function Name

```
_check_OLP_WDM
              .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Measures the optical power in WDM mode and checks the result against the given limits.

Caution: –

PASS / FAIL Conditions

PASS: Optical power value is within the given limits.

FAIL: Power value beyond limits or value not stable in the mode ‘Stable’.

Test Report

```
Check OLP                    PASS
    Title
    WDM1 level: 0.10 dBm
    WDM2 level: -0.10 dBm
```

Applications

The testcase supports three measure modes:

Once:	One single measurement is performed.
Stable:	Several measurements are performed until the result value is in the range defined with the parameter *Precision*. Only the last stable value is evaluated. If the result value is not stable after 25 measurements the testcase will stop with a FAIL condition. *Precision* defines the upper and lower deviation.
Average:	Several measurement up to the number defined in the parameter *Average Count* are taken. The average value is calculated and evaluated.

Parameters

Id:	Integer, 1 .. 10
Mode:	Enumeration Once =1, Stable = 2, Average = 3
Precision:	Enumeration 0.5 dB = 0.500, 0.2 dB = 0.200, 0.1 dB = 0.100, 0.05 dB = 0.050, 0.02 dB = 0.020
Average Cnt.:	Integer, 1 ... 30
Title:	String, Title for measurement
Upper Limit1:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Lower Limit1:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Comment1:	String, Comment
Upper Limit2:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Lower Limit2:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Comment2:	String, Comment

Remote Control of Testcase

Input: %d#%d,%lf,%d,%lf,%lf,#%s#,#%s#
Id,Mode,Precision,Average Count,Title,Upper Limit1,Lower Limit1,Comment1, Upper Limit2,Lower Limit2,Comment2

Output: %d,%lf,%d,%lf
Valid1?,Power1,Valid2?,Power2
Validx?: Valid = 1, Not valid = 0

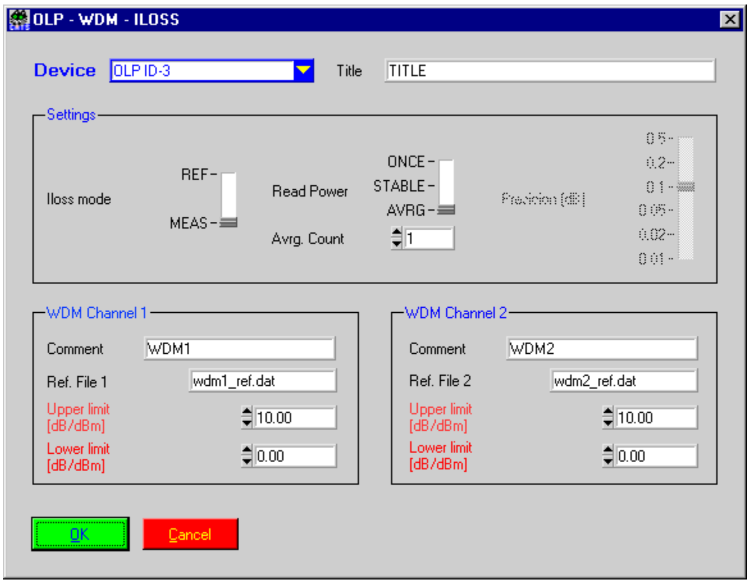
Notes:

16 Check Insertion Loss

Function Name

```
_iloss_OLP
            .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Measures the insertion loss of a UUT and checks the result against the given limits.

Caution: –

PASS / FAIL Conditions

PASS: Insertion loss value is within given limits.

FAIL: Insertion loss beyond limits or value not stable in the mode ‘Stable’.

Test Report

```
iloss OLP                    PASS
Title
    comment(1310 nm) PASS level: -0.1 dBm
```

Applications

For the insertion loss measurement the testcase must be performed two times. First a reference measurement without the UUT is made. Then the UUT has to be connected and the insertion loss is measured.

The testcase supports three measure modes:

Once:	One single measurement is performed.
Stable:	Several measurements are performed until the result value is in the range defined with the parameter *Precision*. Only the last stable value is evaluated. If the result value is not stable after 25 measurements the testcase will stop with a FAIL condition. *Precision* defines the upper and lower deviation.
Average:	Several measurement up to the number defined in the parameter *Average Count* are taken. The average value is calculated and evaluated.

Parameters

Id:	Integer, 1 .. 10
Mode:	Enumeration Once =1, Stable = 2, Average = 3
Precision:	Enumeration 0.5 dB = 0.500, 0.2 dB = 0.200, 0.1 dB = 0.100, 0.05 dB = 0.050, 0.02 dB = 0.020
Average Cnt.:	Integer, 1 ... 30
Title:	String, Title for measurement
Upper Limit:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Lower Limit:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Comment:	String, Comment
Ref. Mode:	Enumeration Reference Mode: ON =1, OFF = 0
Ref. File:	String, Reference File

Remote Control of Testcase

Input:	%d#%d,%lf,%d,#%s#,%lf,%lf,#%s#%d,#%s# Id,Mode,Precision,Average Count,Upper Limit,Lower Limit,Title,Comment, Ref. Mode,Ref. File
Output:	%d,%lf Valid?,Power Valid?: Valid = 1, Not valid = 0

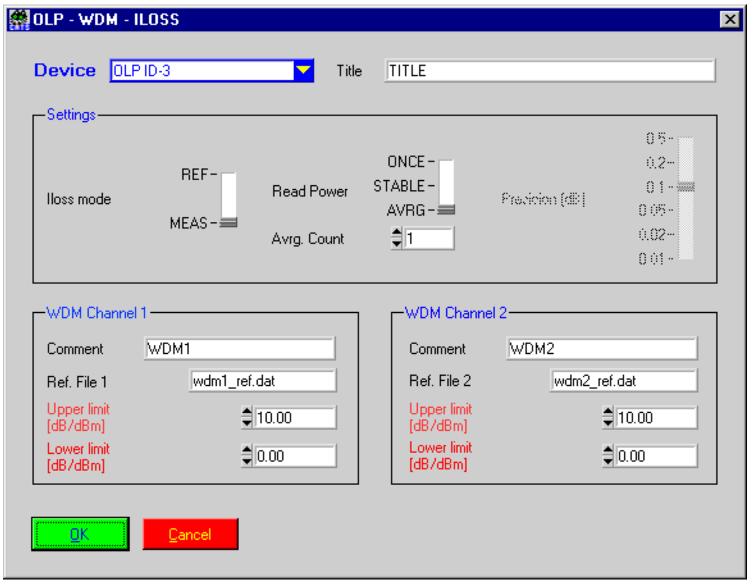
Notes:

17 Check Insertion Loss in WDM Mode

Function Name

```
_iloss_OLP_WDM
              .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Measures the insertion loss of a UUT in WDM mode and checks the result against the given limits.

Caution: –

PASS / FAIL Conditions

PASS: Insertion loss value is within the given limits.

FAIL: Insertion loss beyond limits or value not stable in the mode ‘Stable’.

Test Report

```
iloss OLP WDM PASS
Title
    WDM  PASS level: -0.1 dBm
    WDM  PASS level: -0.8 dBm
```

Applications

For the insertion loss measurement the testcase must be performed two times. First a reference measurement without the UUT is made. Then the UUT is connected and the insertion loss is measured.

The testcase supports three measure modes:

Once: One single measurement is performed.

Stable: Several measurements are performed until the result value is in the range defined with the parameter *Precision*. Only the last stable value is evaluated. If the result value is not stable after 25 measurements the testcase will stop with a FAIL condition. *Precision* defines the upper and lower deviation.

Average: Several measurement up to the number defined in the parameter *Average Count* are taken. The average value is calculated and evaluated.

Parameters

Id:	Integer, 1 .. 10
Mode:	Enumeration Once =1, Stable = 2, Average = 3
Precision:	Enumeration 0.5 dB = 0.500, 0.2 dB = 0.200, 0.1 dB = 0.100, 0.05 dB = 0.050, 0.02 dB = 0.020
Average Cnt.:	Integer, 1 ... 30
Title:	String, Title for measurement
Upper Limit1:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Lower Limit1:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Comment1:	String, Comment
Upper Limit2:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Lower Limit2:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Comment2:	String, Comment
Ref. Mode:	Enumeration Reference Mode: ON =1, OFF = 0
Ref. File1:	String, Reference File1
Ref. File2:	String, Reference File2

Remote Control of Testcase

Input:	%d#%d,%lf,%d,#%s#,%lf,%lf,#%s#,%lf,%lf,#%s#,%d,#%s#,#%s# Id,Mode,Precision,Average Count,Title,Upper Limit1,Lower Limit1,Comment1, Upper Limit2,Lower Limit2,Comment2,Ref. Mode,Ref. File1,Ref. File2
Output:	%d,%lf,%d,%lf Valid1?,Power1,Valid2?,Power2 Validx?: Valid = 1, Not valid = 0

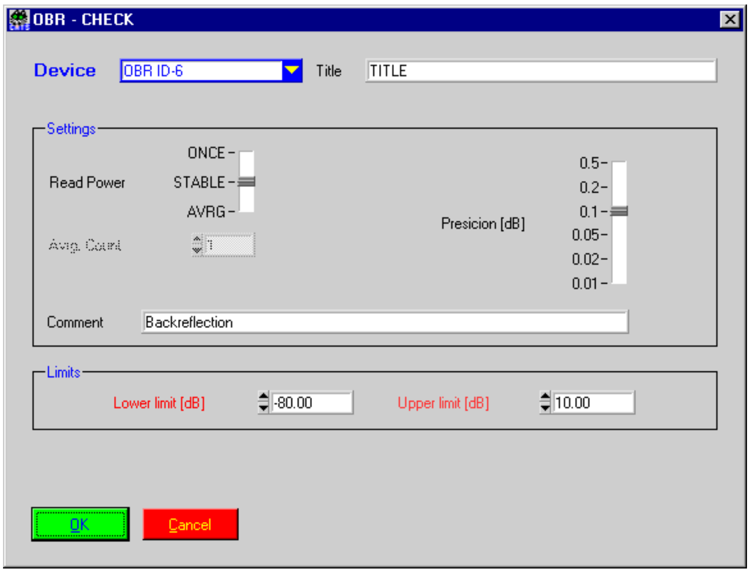
Notes:

18 Check Optical Back Reflection

Function Name

```
_check_OBR
              .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Measures the optical back reflection and checks the result against the given limits.

Caution: –

PASS / FAIL Conditions

PASS: Optical power value is within the given limits.

FAIL: Power value beyond limits or value not stable in the mode ‘Stable’.

Test Report

```
Check OBR                    PASS
     Title
     comment (1310 nm) level: 0.10 dBm
```

Applications

The testcase supports three measure modes:

Once:	One single measurement is performed.
Stable:	Several measurements are performed until the result value is in the range defined with the parameter *Precision*. Only the last stable value is evaluated. If the result value is not stable after 25 measurements the testcase will stop with a FAIL condition. *Precision* defines the upper and lower deviation.
Average:	Several measurement up to the number defined in the parameter *Average Count* are taken. The average value is calculated and evaluated.

Parameters

Id:	Integer, 1 .. 10
Mode:	Enumeration Once =1, Stable = 2, Average = 3
Precision:	Enumeration 0.5 dB = 0.500, 0.2 dB = 0.200, 0.1 dB = 0.100, 0.05 dB = 0.050, 0.02 dB = 0.020
Average Cnt.:	Integer, 1 ... 30
Upper Limit:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Lower Limit:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Title:	String, Title for measurement
Comment:	String, Comment

Remote Control of Testcase

Input: %d#%d,%lf,%d,%lf,%lf,#%s#,#%s#
Id,Mode,Precision,Average Count,Upper Limit,Lower Limit,Title,Comment

Output: %d,%lf
Valid?,Power
Valid?: Valid = 1, Not valid = 0

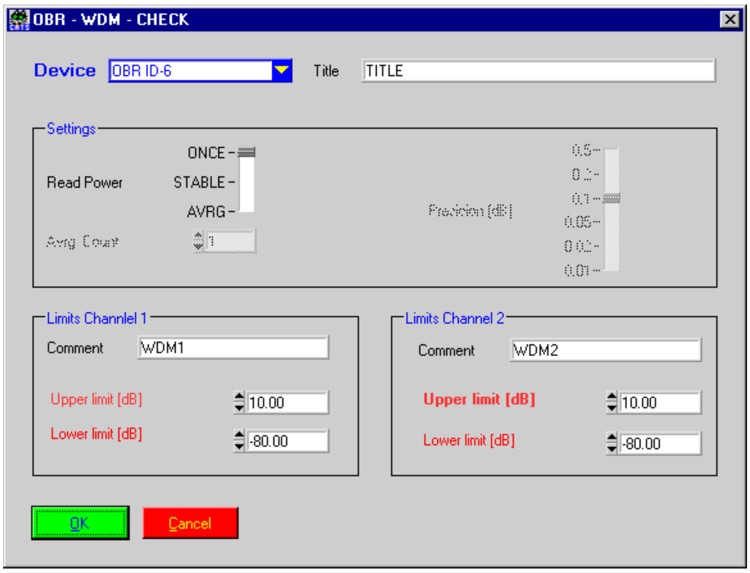
Notes:

19 Check Optical Back Reflection in WDM Mode

Function Name

```
_check_OBR_WDM
                .\optic\ovxoptic.obj
```

User Interface



Description

Purpose: Measures the optical back reflection in WDM mode and checks the result against the given limits.

Caution: –

PASS / FAIL Conditions

PASS: Optical power value is within the given limits.

FAIL: Power value beyond limits or value not stable in the mode ‘Stable’.

Test Report

```
Check OBR                    PASS
    Title
    WDM1 level: 0.10 dBm
    WDM2 level: -0.10 dBm
```

Applications

The testcase supports three measure modes:

Once: One single measurement is performed.

Stable: Several measurements are performed until the result value is in the range defined with the parameter *Precision*. Only the last stable value is evaluated. If the result value is not stable after 25 measurements the testcase will stop with a FAIL condition. *Precision* defines the upper and lower deviation.

Average: Several measurement up to the number defined in the parameter *Average Count* are taken. The average value is calculated and evaluated.

Parameters

Id:	Integer, 1 .. 10
Mode:	Enumeration Once =1, Stable = 2, Average = 3
Precision:	Enumeration 0.5 dB = 0.500, 0.2 dB = 0.200, 0.1 dB = 0.100, 0.05 dB = 0.050, 0.02 dB = 0.020
Average Cnt.:	Integer, 1 ... 30
Title:	String, Title for measurement
Upper Limit1:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Lower Limit1:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Comment1:	String, Comment
Upper Limit2:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Lower Limit2:	Double, -120.0 0.8 .. 30.0 [dB/dBr]
Comment2:	String, Comment

Remote Control of Testcase

Input:	%d#%d,%lf,%d,%lf,%lf,#%s#,#%s# Id,Mode,Precision,Average Count,Title,Upper Limit1,Lower Limit1,Comment1, Upper Limit2,Lower Limit2,Comment2
Output:	%d,%lf,%d,%lf Valid1?,Power1,Valid2?,Power2 Validx?: Valid = 1, Not valid = 0

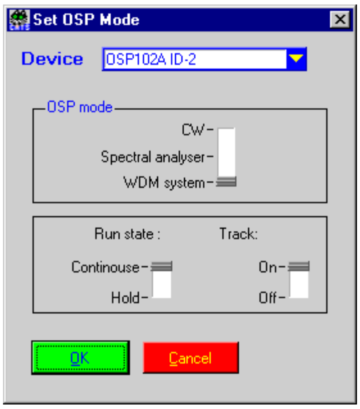
Notes:

20 Set OSP Mode

Function Name

```
_OSP_set_mode
                .\optic\ospoptic.obj
```

User Interface



Description

Purpose: Sets operation mode of the OSP-102A. Available modes are spectral analyser, D-WDM tester and power meter.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
set mode OSP                 PASS
    OSP mode = Spectral
    analysis
    Run state = Continuous
    Sweep = Run
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Mode:	Enumeration CW = 1, Spectral Analyzer = 2,WDM-System = 3
Run State:	Enumeration Continuous = 1,Hold/Single = 0
Track:	Enumeration On = 1,Off = 0

or (in Spectral Analyser Mode)

Sweep State:	Enumeration Run = 1, Stop = 0

Remote Control of Testcase

Input:	%d#%d,%d,%d Id,Mode,Run State,Track/Sweep State
Output:	No Results

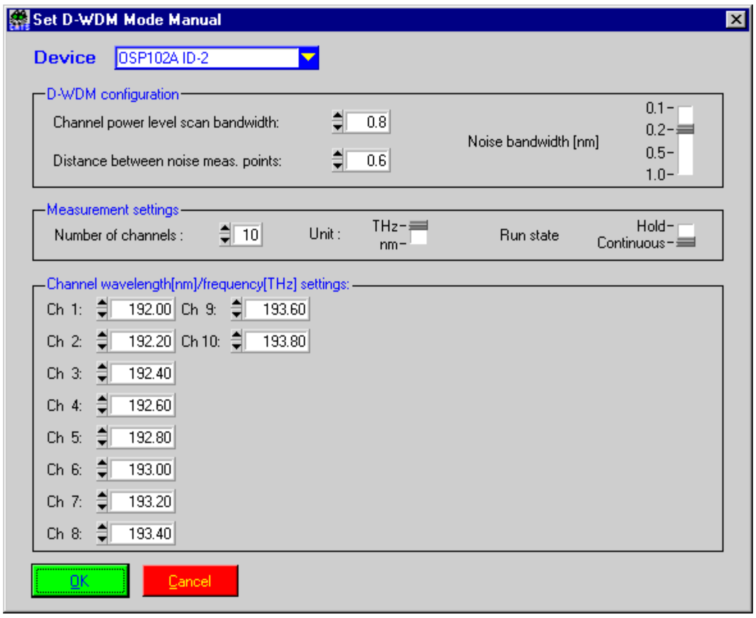
Notes:

21 Set OSP Channels Manually in D-WDM Mode

Function Name

```
_WDM_set_man
              .\optic\ospoptic.obj
```

User Interface



Description

Purpose: Sets parameter of the OSP-102A in D-WDM mode. Channels are selected manually.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable

FAIL: Not applicable

Test Report

```
set  WDM                         PASS

    OSP mode = WDM system

    Measurement is made continuously

    OSP WDM system settings:

     Number of channels: 10

     Channel power level scan bandwidth: 0.8 nm

     Distance between noise meas. points: 0.6 nm

     Noise bandwidth for S/N meas.: 0.2 nm

    OSP WDM channel settings:

    Channel  1 frequency: 192.00 THz
    Channel  2 frequency: 192.20 THz
    Channel  3 frequency: 192.40 THz
    Channel  4 frequency: 192.60 THz
    Channel  5 frequency: 192.80 THz
    Channel  6 frequency: 193.00 THz
    Channel  7 frequency: 193.20 THz
    Channel  8 frequency: 193.40 THz
    Channel  9 frequency: 193.60 THz
    Channel 10 frequency: 193.80 THz
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Unit:	Enumeration THz=1, nm=0
# of Chan.:	Integer, 1 .. 40
dλ-Scan:	Double, 0.1 .. 60.0[nm]
dλ-Noise:	Double, 0.1 .. 30.0[nm]
BW Noise:	Double, 0.1,0.2,0.5,1[nm]
Mode:	Enumeration Hold = 0,Continuous = 1
Wavelength1:	Double, 1530.0 .. 1570.0 [nm] Double, 190.95 .. 195.94 [THz]
Wavelength2:	Double, 1530.0 .. 1570.0 [nm] or Double, 190.95 .. 195.94 [THz] *
. . .	
Wavelengthx:	Double, 1530.0 .. 1570.0 [nm] or Double, 190.95 .. 195.94 [THz] *

Remote Control of Testcase

Input:	%d#%d,%d,%lf,%lf,%lf,%d,%lf,%lf,..%lf Id,Unit,Number of Channels,*dλ*-Scan,*dλ*-Noise,BW-Noise,Mode,Wavelength 1 / Frequency 1, .. Wavelength x / Frequency x
Output:	No Results

x = number of channels

* depending on unit setting

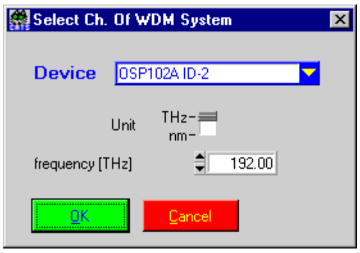
Notes:

22 Select OSP Channel in Power Meter Mode

Function Name

```
_CW_select_channel
               .\optic\ospoptic.obj
```

User Interface



Description

Purpose: Sets the power meter mode of the OSP-102A at a specified frequency

Caution: -

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
sel ch                    PASS
    Selected wavelength: 1561.40 nm
```

Applications

In this mode the input signal is put to the monitor output of the OSP-102A. This could be used to filer out a channel of a D-WDM system.

Parameters

Id:	Integer, 1 .. 10
Unit:	Enumeration THz = 1, nm = 0
Wavelength:	Double, 1530.0 .. 1570.0 [nm] Double, 190.95 .. 195.94 [THz]

Remote Control of Testcase

Input:	%d#%d,%lf Id,Mode,Wavelength / Frequency
Output:	No Results

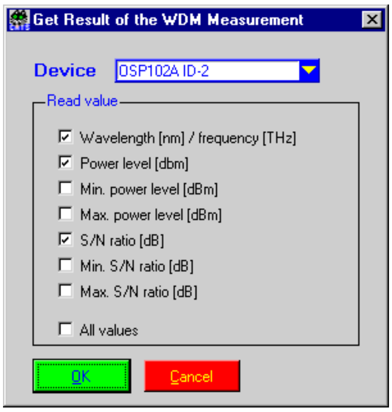
Notes:

23 Read Results in D-DWDM Mode

Function Name

```
_WDM_get_result
             .\optic\ospoptic.obj
```

User Interface



Description

Purpose: Reads the optical parameter of the D-WDM channels. The channels are defined in the OSP-102A with _WDM_set_man testcase.

Caution: —

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
get value WDM                PASS
    OSP mode = WDM system
    OSP WDM system settings:
    Number of channels: 8
    Channel power level scan bandwidth:  0.8 nm
    Distance between noise meas. points: 0.6 nm
    Noise bandwidth for S/N meas.:       0.5 nm
     Measurement is running -> get actual values
    Channel number:  1
     wavelength: 1531.40 nm
     power level: -1.0 dBm
     S/N ratio: 20.0 dB
    Channel number:  2
     wavelength: 1532.20 nm
     power level: -1.3 dBm
     S/N ratio: 19.3 dB
    Channel number: 3
     wavelength: 1533.00 nm
     power level: -1.6 dBm
     S/N ratio: 18.6 dB
    Channel number:  4
     wavelength: 1533.80 nm
     power level: -1.9 dBm
     S/N ratio: 17.9 dB
    Channel number:  5
     wavelength: 1534.60 nm
     power level: -2.2 dBm
     S/N ratio: 17.2 dB
    Channel number:  6
     wavelength: 1535.40 nm
     power level: -2.5 dBm
     S/N ratio: 16.5 dB
    Channel number: 7
     wavelength: 1536.20 nm
     power level: -2.8 dBm
     S/N ratio: 15.8 dB
    Channel number: 8
     wavelength: 1537.00 nm
     power level: -3.1 dBm
     S/N ratio: 15.1 dB
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Wavelength.:	Enumeration Check = 1,Ignore = 0
Power:	Enumeration Check = 1,Ignore = 0
Min. Power:	Enumeration Check = 1,Ignore = 0
Max. Power:	Enumeration Check = 1,Ignore = 0
S/N:	Enumeration Check = 1,Ignore = 0
Min. S/N:	Enumeration Check = 1,Ignore = 0
Max. S/N:	Enumeration Check = 1,Ignore = 0
All values:	Enumeration Check = 1,Ignore = 0

Remote Control of Testcase

Input:	%d#%d,%d,%d,%d,%d,%d,%d Id,All Values,Wavelength,Power,Min. Power,Max. Power,S/N,Min. S/N,Max. S/N
Output:	No Results

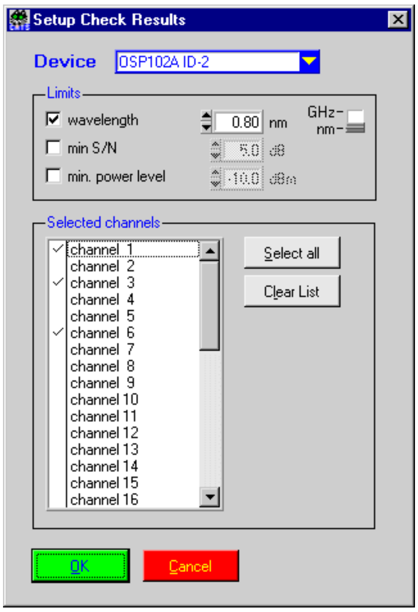
Notes:

24 Check Results in D-WDM Mode

Function Name

```
_WDM_check_result
               .\optic\ospoptic.obj
```

User Interface



Description

Purpose: Read result of the selected D-WDM channels and checks them against the given limits.

Caution: —

PASS / FAIL Conditions

PASS: If the results are in the limits.

FAIL: If the results are out of the limits or if there is no signal at a selected channel.

Test Report

```
check result                 FAIL
    OSP mode = WDM system
    OSP WDM system settings:
     Number of channels: 8
     Channel power level scan bandwidth: 0.8 nm
     Distance between noise meas. points: 0.6 nm
     Noise bandwidth for S/N meas.: 0.5 nm
     Measurement is running -> get actual values
    Channel number: 1
    mid. wavelength 1531.00 nm
     wavelength: 1531.40 nm
     S/N ratio: 20.0 dB
    Channel number: 3
    mid. wavelength: 1532.60 nm
     wavelength: 1533.00 nm
     S/N ratio: 18.6 dB
    FAIL S/N below limit
    Channel number: 6
    mid. wavelength: 1535.00 nm
     wavelength: 1535.40 nm
     S/N ratio: 16.5 dB
    FAIL S/N below limit
```

Applications

–

Parameters

Id:	Integer, 1 .. 10
Check wave.:	Enumeration Check = 1,Ignore = 0
Wavelength:	Double, 0.8 .. 9.95 [nm] Double, 100 .. 600 [GHz]
Check S/N:	Enumeration Check = 1,Ignore = 0
Min. S/N:	Double, 0.5 .. 50[dB]
Check Pow:	Enumeration Check = 1,Ignore = 0
Min. Power:	Double, -60 .. 20[dBm]
Unit:	Enumeration THz = 1, nm = 0
Check Ch1:	Enumeration Check = 1,Ignore = 0
Check Ch2:	Enumeration Check = 1,Ignore = 0
. . .	
Check Ch40:	Enumeration Check = 1,Ignore = 0

Remote Control of Testcase

Input: %d#%d,%lf,%d,%lf,%d,%lf,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d,%d
Id,Ckeck Wav.,Wavelength,Check S/N,Min. S/N,Check Pow.,Min. Power,Unit,Check Ch1,Check Ch2,Check Ch3,Check Ch4, Check Ch5, Check Ch6, Check Ch7,Check Ch8, Check Ch9, Check Ch10, Check Ch11, Check Ch12, Check Ch.13, Check Ch14, Check Ch15, Check Ch16, Check Ch17, Check Ch18, Check Ch19, Check Ch20, Check Ch21, Check Ch22, Check Ch 23, Check Ch24, Check Ch 25, Check Ch26, Check Ch27, Check Ch28, Check Ch29, Check Ch30, Check Ch 31, Check Ch32, Check Ch33, Check Ch34, Check Ch35, Check Ch36, Check Ch37, Check Ch38, Check Ch39, Check Ch40

Output: No Results

Notes:

WWG-CATS Testcase Library

Auxiliary Testcases (Toolkit)

Auxiliary Testcases (Toolkit)

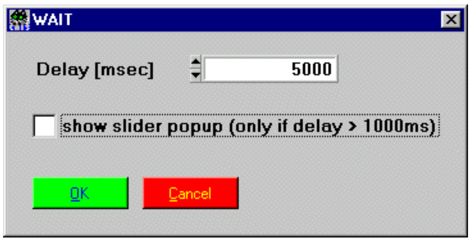
Notes:

1 Waiting Time (Delay in [ms])

Function Name

```
_wait
        .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: Wait for a specified period. This testcase is mainly used to produce error/alarm/offset scenarios, e.g. Wander generation and G.826 event generation.

Caution: When writing scenarios using this testcase please be aware that the overall resolution of a time delay can't be better than around 0.1 sec. If better time resolution is required, internal functions of the testset have to be used.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Delay 2 s                        CONTINUE
     Wait 2000 [msec]
```

Applications

Typical applications are G.821/G.826 stimulus scenarios or wander scenarios (used in conjunction with “_set_alarm” or “_set_stimulus”). See sequences “wander.squ” and “g826.squ”.

Parameters

Delay: Integer, 100 .. 10000000 [ms]

Remote Control of Testcase

Input: %d
Delay

Output: No Results

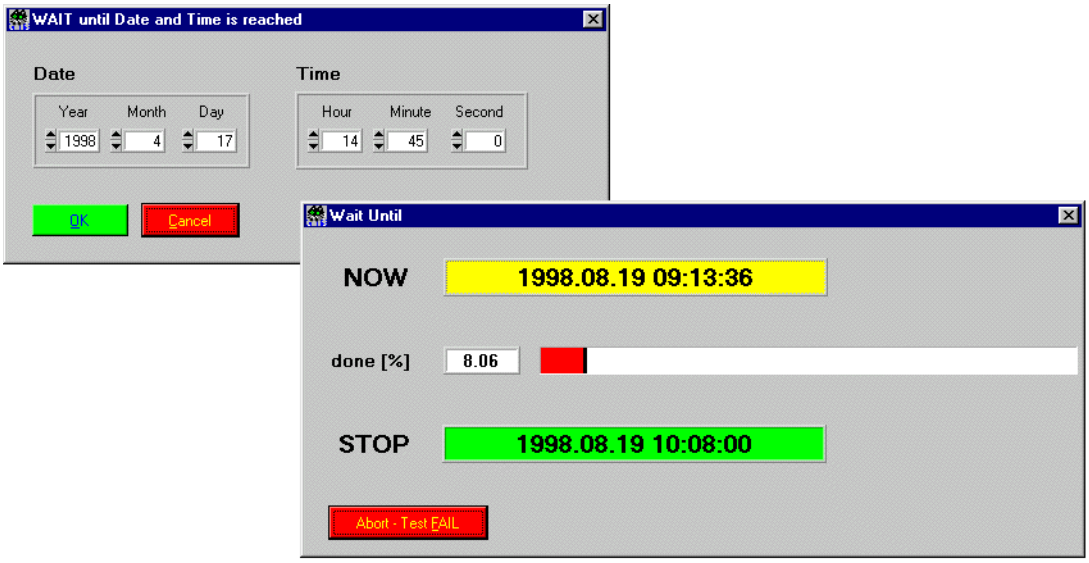
Notes:

2 Wait Until a Predefined Date/Time is Reached

Function Name

```
_wait_until_time
              .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: Waits until a specific point in time.

Caution: —

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
wait until        CONTINUE
```

Applications

To start a specific test or test sequence at a specific date and time. This may be useful when testing longterm performance monitoring sensors (e.g. to G.826) of network elements.

Parameters

Year:	Integer, 1997 .. 2020
Month:	Integer, 1.. 12
Day:	Integer, 1.. 31
Hour:	Integer, 0 .. 23
Minute:	Integer, 0 .. 59
Second:	Integer, 0 .. 59

Remote Control of Testcase

Input:	%d,%d,%d,%d,%d,%d Year,Month,Day,Hour,Minute,Second
Output:	No Results

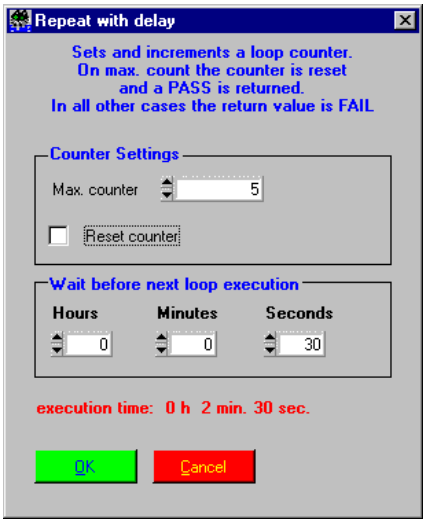
Notes:

3 Wait a defined time and increments an loop counter

Function Name

```
_repeat_wait
              .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: Determines how many times a loop is to be executed consecutively and defines the delay between each execution of the loop.

Definition of a loop: see Applications

Caution: -

PASS / FAIL Conditions

PASS: Maximum counter value reached or reset triggered.

FAIL: Maximum counter value not reached.

Test Report

```
repeat_wait                PASS
```

Applications

The testcase is used in conjunction with a “GOTO” command to produce loops within sequences.

The “GOTO” command is placed after the testcase in the sequence and points to the sequence. A precondition is also defined for the “GOTO” command which causes the “GOTO” command to be executed only if the testcase generates “FAIL” status. As soon as the required number passes through the loop has been completed, the testcase generates “PASS” status and the “GOTO” command is no longer executed.

Example:

```
_repeat_wait
>Insert the testcases
>that are to be within
>the loop

Goto _repeat_wait
>Testcases outside
>the loop
```

A precondition with content “FAIL _repeat_until” is defined for “Goto _repeat_until”.

Parameters

Max. counter:	Integer 1..100000
Wait time [sec.]:	Double 30..651599
Reset:	Enumeration ON = 1, OFF = 0

Remote Control of Testcase

Input Buffer: 1#,%d,%lf,%d

Output Buffer: No result

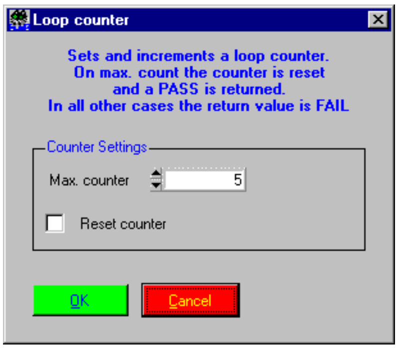
Notes:

4 Increment an loop counter

Function Name

```
_loop_counter
             .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: Defines the number of times a loop is executed.

Definition of a loop: see Applications

Caution: -

PASS / FAIL Conditions

PASS: Maximum counter value reached or reset triggered.

FAIL: Maximum counter value not reached.

Test Report

```
loop_counter                PASS
```

Applications

The testcase is used in conjunction with a “GOTO” command to produce loops within sequences.

The “GOTO” command is placed after the testcase in the sequence and points to the sequence. A precondition is also defined for the “GOTO” command which causes the “GOTO” command to be executed only if the testcase generates “FAIL” status. As soon as the required number passes through the loop has been completed, the testcase generates “PASS” status and the “GOTO” command is no longer executed.

Example:

```
_loop_counter
>Insert the testcases
>that are to be within
>the loop

Goto _loop_counter
>testcases outside
>the loop
```

A precondition with content “FAIL _loop_counter” is defined for “Goto _loop_counter”.

Parameters

Max. Counter:	Integer 1..100000
Reset:	Enumeration: ON = 1, OFF = 0

Remote Control of Testcase

Input Buffer:	1#,%d,%d 1# Max. Counter, Reset
Output Buffer:	No result

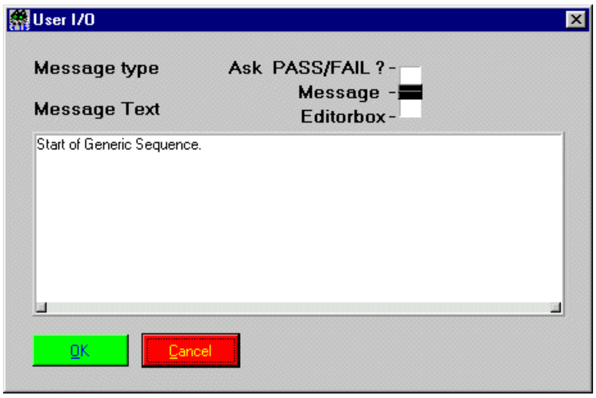
Notes:

5 Pop-Up Message or Question towards User

Function Name

```
_user_io
          .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: This testcase allows an interaction with the user. A user-defined message-text is displayed when running this test, next testcase starts after “OK” is pressed. In a second mode (“Ask PASS/FAIL”) the user is asked a question that can be answered “Yes” or “No”.

Caution: Do not use the “#” character.
Do not use empty lines between lines containing text.

PASS / FAIL Conditions

PASS: If the user answers “Yes”, this testcase is considered PASS.

FAIL: Testcase is considered FAIL, if the answer is “No”.

Test Report

```
Message                         CONTINUE
     Start of Generic Sequence.
```

Applications

This feature may be used to ask the user for a specific action (like: Please connect ....) or give information about a following testcase (like: Please make sure ... before continuing). Testcase relevance in that case is normally “Don’t Care”.

This testcase may also be used to ask the user for test-specific details that cannot be automated but that do influence the outcome of a test (like “Alarm LED on?”).

Testcase relevance has to be “Relevant”.

Parameters

Op. Mode:	Enumeration Ask PASS / FAIL = 1, Message = 2
Text:	String, up to ten lines

Remote Control of Testcase

Input:	%d,%d,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s Op. Mode,Number of Lines,Line 1, ..., Line 10
Output:	No Results

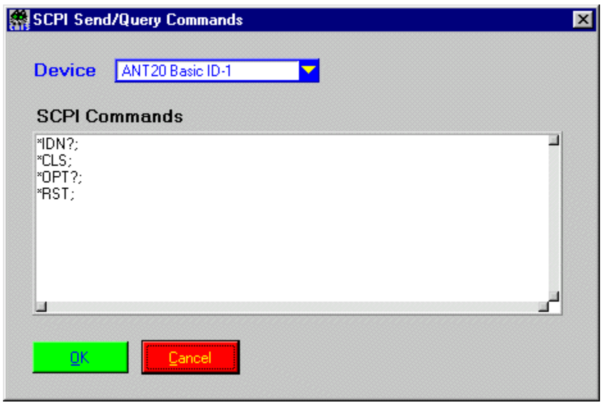
Notes:

6 Send SCPI Command to Instrument and Read Result

Function Name

```
_scpi_io
            .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: Sends user-defined SCPI commands to the specified instrument via the opened interface. If a query is sent, the response is displayed.

No PASS/FAIL is derived from the contents of the received result string.

Caution: Please keep in mind that commands sent to an instrument in this way may conflict with settings that are done using preceding or subsequent testcases.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
SCPI Commands                    CONTINUE
      *IDN?;
     WANDEL&GOLTERMANN,ANT-20/01A7BA850000,FM-203,5.00/3035/01/0C01/
     0252
      *CLS;
      *OPT?;
      3035/90.01,3035/90.02,3035/90.03,3035/90.04,3035/90.05,3035/
     90.10
```

Applications

This testcase may be used for setups or to read out and display results that might not be covered by any Wavetek Wandel Goltermann testcase.

Example for SCPI string:

“:INP:TEL:LINE:EQU ON; :INP:TEL:LINE:EREF -22”
Sets the equalizer reference level to - 22 dB.

Users of the full version of CATS may also use this testcase to control 3rd party instruments via IEEE 488.2 or V.24 interface.

Parameters

Device Id: Integer, 1 .. 10

Commands: String(s), Up to 10 SCPI command lines.

Remote Control of Testcase

Input: %d#%d,#%s#,#%s#,,#%s#,#%s#,#%s#,#%s#,#%s#,#%s#,#%s#,#%s#
Device Id,Number of commands,command 1, ..., command 10

Output: %s
Response

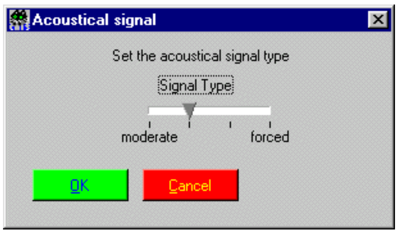
Notes:

7 Beep

Function Name

```
_acoustical_signal
               .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: Generates an acoustical signal (beep).

Caution: —

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Beep            CONTINUE
```

Applications

May be used to alert the user when a specific test case is reached, or when a test sequence is about to end. This testcase may also be used to alert the user when a specific preceding test failed (see test preconditions).

Parameters

Signal Type: Integer, 0 .. 3
0 = moderate, 3 = forced

Remote Control of Testcase

Input: %d
Signal Type

Output: No Results

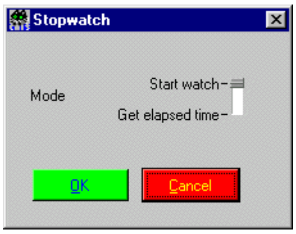
Notes:

8 Measure Elapsed Time between 2 Calls of this Test

Function Name

```
_stopwatch
           .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: Measures elapsed time between first call of this testcase (start of stop watch) and subsequent calls of the same testcase.

Caution: —

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
stop watch start                CONTINUE
     start stopwatch
-------------------
time passed                     CONTINUE
     stop stopwatch, time passed: 2149 [msec]
```

Applications

Can be used to measure the exact time used by a specific test sequence.

Parameters

Mode:	Enumeration (Re-)Start stop watch = 1, Get elapsed time = 2

Remote Control of Testcase

Input:	%d Mode
Output:	%i Elapsed time (=0 in case of “Start stop watch”) [ms]

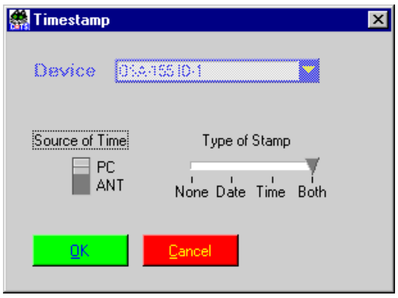
Notes:

9 Print Time Stamp to Report

Function Name

```
_time_stamp
               .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: Inserts a timestamp in the report file.

Caution: -

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Time_stamp                        PASS
```

Applications

-

Parameters

Device ID:	Integer 1..10
Source:	Enumeration PC = 0, ANT-20 = 1
Type:	Enumeration None = 0, Date = 1, Time = 2, Both = 3

Remote Control of Testcase

Input Buffer:	%d#,%d,%d Device ID#,Source,Type
Output Buffer:	No result

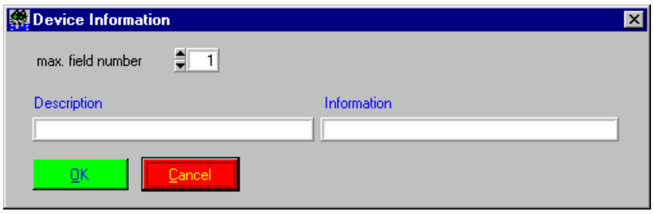
Notes:

10 Entry Fields for UUT / User Info

Function Name

```
_show_info_field
               .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: Inserts UUT information to the report file.

Caution: –

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applcable.

Test Report

```
show_info_field          OK
UUT Type: XYZ
Serial Number: 123
Technican: John Smith
```

Applications

–

Parameters

max. field number:	Integer 1..15
Description 1..15:	string
Information 1..15:	string

Remote Control of Testcase

Input Buffer: %#,%d,#%s#,#%s#,...#%s#

%# max field number,#Description1#,#Information1# ...
#Description15#,#Information15#

Output Buffer: No result

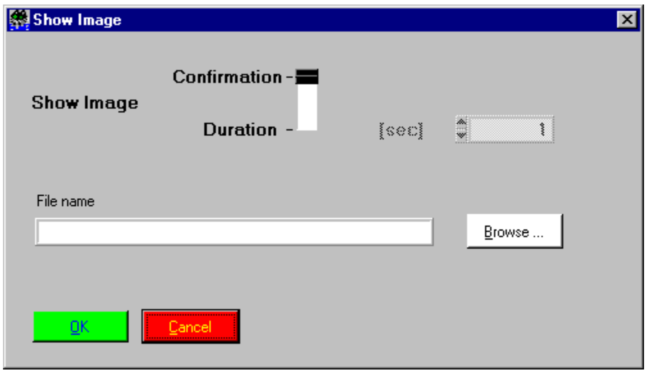
Notes:

11 Show Bitmap (e.g. Picture of Cabling)

Function Name

```
_show_image
              .\toolkit\toolkit.obj
```

User Interface



Description

Purpose: Displays a picture for some information.

Caution: -

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
show_image:          OK
```

Applications

–

Parameters

Type	Enumeration: Confirmation = 1, Duration = 2
Wait time:	Double 1..1000
File name:	string

Remote Control of Testcase

Input Buffer:	1#%d,%lf,#%s# 1#Type,Wait Time,#File Name
Output Buffer:	No results.

Notes:

WWG-CATS Testcase Library

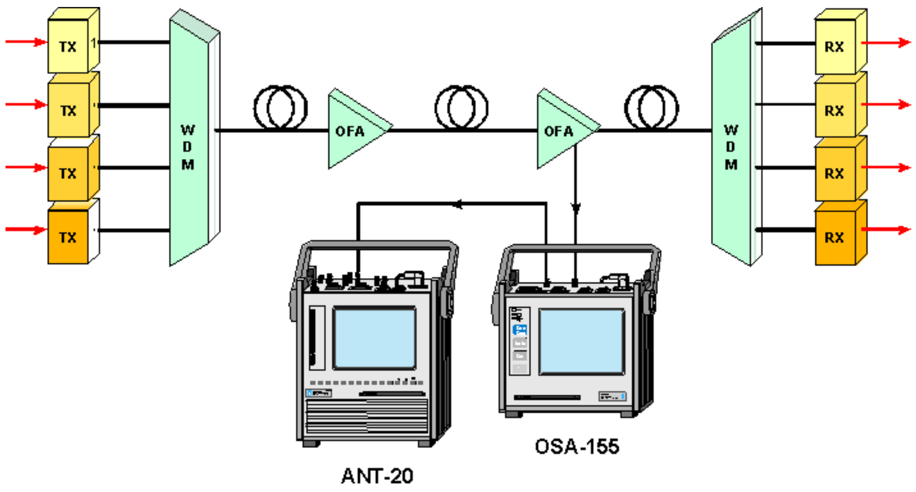
D-WDM Testcases

(Requires BN 3045/93.43)

D-WDM Testcases

Notes:

1 Applications



CATS DWDM includes a set of testcases and CATS sequences that combine the power of the ANT-20 Advanced Network Tester and the OSA-155 DWDM System Analyzer to perform fully automatic quality analysis of DWDM systems. The OSA-155 is connected to the ANT-20 via RS-232 cable or the GPIB-Bus. The CATS DWDM software runs directly on the ANT-20's internal PC and controls the measurement functions of both instruments.

OSA-155: Measurements of wavelength, power, and optical SNR of the different carriers of multi-wavelength signals in the range 1450 to 1650 nm with 50 GHz/0.4 nm or higher spacing.

ANT-20/E: Measurements of parity errors, SDH/SONET/PDH alarms, jitter etc. of the different embedded SDH/SONET channels. It is even possible to perform a complete analysis of all hierarchy levels of a particular wavelength signal and show higher order and lower order path alarms, error rates, path trace and signal label information as well as PDH alarms and FAS error rate.

A typical test sequence would first of all verify the levels, wavelengths and S/N of all optical carriers, then set the OSA-155 filter to feed all wavelengths, one after the other, to the ANT-20 optical RX, where a quick check for alarms and errors is performed. For in depth analysis, longterm measurements or scans of the complete structure of particular SDH/SONET signals can be performed. At the end of a sequence run a complete ASCII test protocol is generated that shows all relevant results plus a clear PASS/FAIL statement.

2 Sequences

i_dwdm.squ

This sequence only works together with the OSA-155 Optical Spectrum Analyser, which is controlled from the ANT-20 via an RS.232 or GPIB interface. The test starts with the setup of DWDM parameters (number of channels, spacing, threshold), then a measurement of DWDM parameters (level, optical signal/noise ratio, wavelength) is carried out for each ‘color’ and evaluated against user-defined thresholds. The next test sets the OSA-155 into ‘filter mode’ which means that one specific wavelength is filtered out and is fed into the ANT-20 optical RX. Now any ANT-20 specific test can be carried out for the filtered wavelength, in this example just a B3-Error Rate check for all STM-1 tributaries of an STM-16 signal. This sequence uses the CATS loop feature, an outer loop to switch between 8 colcors of a DWDM signal, and an inner loop to switch between the 16 STM-1 channels of the STM-16 signal.

a_dwdm.squ

This sequence only works together with the OSA-155 Optical Spectrum Analyser, which is controlled from the ANT-20 via an RS.232 or GPIB interface. The test starts with the setup of DWDM parameters (number of channels, spacing, threshold), then a measurement of DWDM parameters (level, optical signal/noise ratio, wavelength) is carried out for each ‘color’ and evaluated against user-defined thresholds. The next test sets the OSA-155 into ‘filter mode’ which means that one specific wavelength is filtered out and is fed into the ANT-20 optical RX. Now any ANT-20 specific test can be carried out for the filtered wavelength, in this example just a B3-Error Rate check for all STS-1 tributaries of an OC-48 signal. This sequence uses the CATS loop feature, an outer loop to switch between 8 colors of a DWDM signal, and an inner loop to switch between the 48 STS-1 channels of the OC-48 signal.

3 Testcases

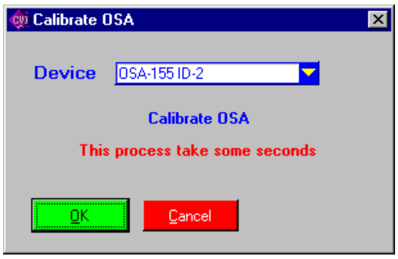
The D-WDM test cases are described in this section.

3.1 Calibrate OSA

Function Name

```
_calibrate_OSA
               .\optic\osa155.obj
```

User Interface



Description

Purpose: Calibrates the OSA-155.

Caution: The autocalibration of the OSA-155 is switched off. Use this testcase to calibrate the OSA in long term applications.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Calibrate OSA DC
```

Applications

–

Parameters

Device-Id:	Integer, 1 .. 10

Remote Control of Testcase

Input:	%d# Device-Id
Output:	No Results

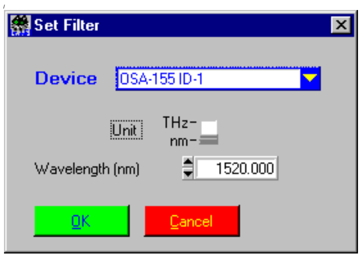
Notes:

3.2 Set filter to specified wavelength/frequency

Function name

```
_set_filter_man
          .\optic\osa155.obj
```

User interface



Description

Purpose: Sets the optical filter of the OSA to a defined wavelength/frequency.

Caution: -

PASS / FAIL conditions

PASS: Not applicable.

FAIL: Not applicable.

Test report

```
Set OSA FilterDC    DC
      Filter set to 1535.000 nm
```

Applications

With this setting, the OSA-155 can be used as a tunable optical filter. The internal optical filter is set to the specified wavelength/frequency. The signal output from the filter is sent to the monitor output of the OSA-155.

Parameter

Device-Id: Select device ID address

Integer, 1 ... 10

Unit: Select display in units of wavelength or frequency

Explanation
wavelength in nanometer [nm] = 0, frequency in tera hertz [THz] = 1

Wavelength / Frequency: Select wavelength or frequency range

Double, 1500.00 ... 1620.00 [nm] / 185.057 ... 199.862 [THz], depending on Unit

Testcase remote control

Input: %d#%d,%lf
Device-Id, Unit, Filter Wavelength / Frequency

Output: No Results

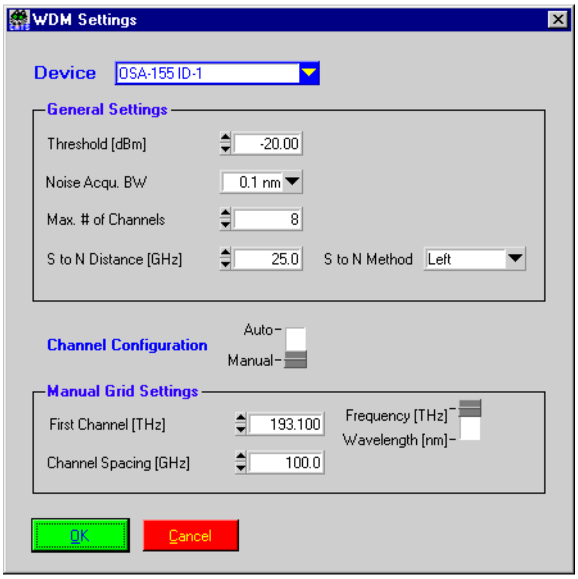
Notes:

3.3 Set parameter of DWDM mode

Function Name

```
_WDM_set_parameter
               .\optic\osa155.obj
```

User Interface



Description

Purpose: Sets the general parameters for D-WDM mode.

Caution: The meaning of the maximum number of channels differs in the AUTO and MANUAL mode.
In AUTO mode the maximum number of channels represents the maximum number of channels the OSA-155 will search for.
In Manual mode the parameter is the maximum number of channels for the channel grid.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
OSA  WDM Parameters  DC
     ---Set OSA-155 to WDM mode---
     Threshold                   :   -20.00 dBm
     S to N Distance             :    25.00 GHz
     S to N Method               :     Left
     Noise Aqu. BW               :     0.10 nm
     Max. # of Channels          :     8
     Channel Configuration       :   MANUAL
     ---Manual Channel Settings---
     First Channel               :  193.100 THz
     Channel Spacing             :  100.000 GHz
```

Applications

In the typical setup the AUTO mode is used. The AUTO mode enables the automatic search for the D-WDM channels. A channel is detected according to the parameters level threshold and S to N distance. For a detailed description refer to the OSA-155 manual.

Parameters

Device-Id:	Integer, 1 .. 10
Mode:	Enumeration MANUAL mode = 0, AUTO mode = 1
Unit:	Enumeration wavelength in nanometer [nm] = 0, frequency in tera hertz [THz] = 1
Threshold:	Double, -85 .. 20 [dBm]
SN Distance:	Double, 25 .. 1000 [GHZ]
SN Method:	Enumeration Left = 0, Right = 1, Both = 2
Noise BW	Double, 0.01 .. 10 [nm]
First Chnl.:	Double, 1500 .. 1620 [nm]/185.057 .. 199.862 [THz], depending on Unit
Channel Spacing:	Double, 50 .. 1000 [GHz]
Max. # of channels:	Integer, 1 .. 255

Remote Control of Testcase

Input:	%d#%d,%d,%lf,%lf,%d,%lf,%lf,%lf,%d Device-Id,Mode,Unit,Threshold,SN Distance,SN Method,Noise BW,First Channel,Channel Spacing,Max. Number of Channels
Output:	No Results

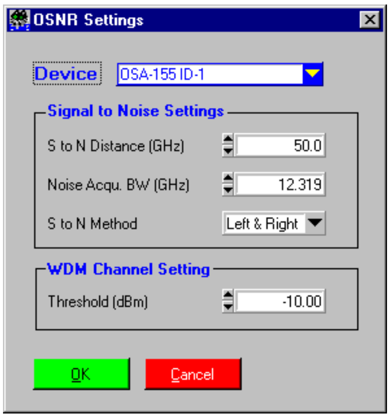
Notes:

3.4 Set S to N parameter of WDM Mode

Function name

```
_WDM_set_OSNR_parameter
               .\optic\osa155.obj
```

User interface



Description

Purpose: Sets the S to N parameters for D-WDM mode.

Caution: -

PASS / FAIL conditions

PASS: Not applicable.

FAIL: Not applicable.

Test report

```
OSNR Parameter
    ---Set OSA-155 to WDM mode---
    Threshold             :    -10.00 dBm
    S to N Distance       :     50.00 GHz
    S to N Method         :    Left & Right
    Noise Aqu. BW         :     12.319 nm
```

Applications

AUTO mode is used for the typical setup. AUTO mode enables automatic search for D-WDM channels. Channel detection is determined by the level threshold and S to N distance parameters. Refer to the OSA-155 manual for a detailed description.

Parameters

Device-Id:	Select device ID address. *Integer, 1 ... 10*
S to N Distance:	Specify frequency spacing between measurement points for signal level and measurement points for noise level. *Double, 25 ... 8000 [GHZ]*
Noise Acqu. BW:	Specify wavelength or frequency bandwidth within which noise is detected and averaged. *Double, 0.01 ... 10 [nm]*
S to N Method:	Select location for measuring noise level for calculating OSNR. Explanation *Left = 0, Right = 1, Both = 2*
Threshold:	Channel display threshold value *Double, -85 ... 20 [dBm]*

Testcase remote control

Input: %d#,%lf,%lf,%d,%lf
Device-Id, SN Distance, Noise Acqu. BW, SN Method, Threshold

Output: No results

Notes:

3.5 Read all results of found channels

Function Name

```
_WDM_read_all_channels
               .\optic\osa155.obj
```

User Interface

Description

Purpose: Reads selected measurement results for all D-WDM channels set in MANUAL mode or found in AUTO mode and writes them to the report.

Caution: —

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Read all channels        PASS
     OSA is in auto mode
      16 channels found.
        Ch.# Waveln[nm]  Pwr[dBm]    SN[dB]
           1 1577.025    -1.40       20.40
           2 1576.196    -1.30       21.30
           3 1575.368    -1.20       21.20
           4 1574.540    -5.60       15.60
           5 1573.714    -0.80       20.80
           6 1572.888    -1.40       21.40
           7 1572.063    -0.40       20.40
           8 1571.239    -0.80       20.80
           9 1570.416    -5.10       15.10
          10 1569.594    -1.30       21.30
          11 1568.772    -0.80       20.80
          12 1567.952    -1.10       21.10
          13 1567.132    -0.40       20.40
          14 1566.313    -0.60       20.60
          15 1565.495    -1.00       21.00
          16 1564.678    -1.10       21.10
```

Applications

Gives an overview without evaluating for PASS and FAIL.

Parameters

Device-Id:	Integer, 1.. 10
Unit:	Enumeration wavelength in nanometer [nm] = 0, frequency in tera hertz [THz] = 1
Frequency/ Wavelen. Flag:	Enumeration read and display frequency / wavelength: ON = 1, OFF = 0
Power Flag:	Enumeration read and display power: ON = 1, OFF = 0
S to N Flag:	Enumeration read and display S to N: ON = 1, OFF = 0

Remote Control of Testcase

Input:	%d#%d,%d,%d,%d Device-Id,Unit,Frequency / Wavelength Flag,Power Flag,S to N Flag
Output:	%d{,%d[,%lf,%lf,%lf,]}{,%d[,%lf,%lf,%lf,]}, ... {,%d[,%lf,%lf,%lf,]} Max. Number of Channels{,Channel Number[,Frequency / Wavelength, Power,S to N]} {} up to maximum number of channels [] depending on Frequency / Wavelength Flag,Power Flag and S to N Flag

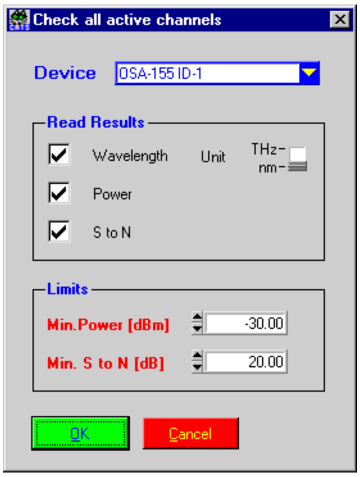
Notes:

3.6 Check all results of found channels

Function Name

```
_WDM_check_all_channels
               .\optic\osa155.obj
```

User Interface



Description

Purpose: Reads selected measurement results of all D-WDM channels set in MANUAL mode or found in AUTO mode and checks them against the given limits.

Caution: —

PASS / FAIL Conditions

PASS: Measured results are above the given limits.

FAIL: Measured results are equal or below the given limits.

Test Report

```
Check all channels      PASS
     OSA is in auto mode.
      16 channels found.
     Min. Power Limit:   -6.00 dBm
     Min. S to N Limit:  15.00 dB
     -------------------------------------
     Results:
        Ch.# Freq.[THz]  Pwr[dBm]    SN[dB]      State
           1 190.125     -1.40       20.40     | PASS
           2 190.195     -1.30       21.30     | PASS
           3 190.303     -1.20       21.20     | PASS
           4 190.370     -5.60       15.60     | PASS
           5 190.500     -0.80       20.80     | PASS
           6 190.612     -1.40       21.40     | PASS
           7 190.698     -0.40       20.40     | PASS
           8 190.835     -0.80       20.80     | PASS
           9 190.900     -5.10       15.10     | PASS
          10 191.002     -1.30       21.30     | PASS
          11 191.096     -0.80       20.80     | PASS
          12 191.203     -1.10       21.10     | PASS
          13 191.326     -0.40       20.40     | PASS
          14 191.405     -0.60       20.60     | PASS
          15 191.495     -1.00       21.00     | PASS
          16 191.608     -1.10       21.10     | PASS
```

Applications

Check the overall optical parameter of a D-WDM system.

Parameters

Device-Id:	Integer, 1.. 10
Unit:	Enumeration wavelength in nanometer [nm] = 0, frequency in tera hertz [THz] = 1
Frequency/ Wavelen. Flag:	Enumeration read and display frequency / wavelength: ON = 1, OFF = 0
Power Flag:	Enumeration read and check power: ON = 1, OFF = 0
S to N Flag:	Enumeration read and check S to N: ON = 1, OFF = 0
Min. Power:	Double, -60 .. +15 [dBm]
Min. S to N:	Double, 0 .. 100 [dB]

Remote Control of Testcase

Input:	%d#%d,%d,%d,%d,%lf,%lf, Device-Id,Unit,Frequency / Wavelength Flag,Power Flag,S to N Flag, Minimum Power,Minimum S to N
Output:	%d{,%d[,%lf,%d,%lf,%d,%lf,]}{,%d[,%lf,%d,%lf,%d,%lf,]}, ... {,%d[,%lf,%d,%lf,%d,%lf,]} Max. Number of Channels{,Channel Number[,Frequency / Wavelength, Status,Power,Status, S to N]} {} up to maximum number of channels [] depending on Frequency / Wavelength Flag,Power Flag and S to N Flag

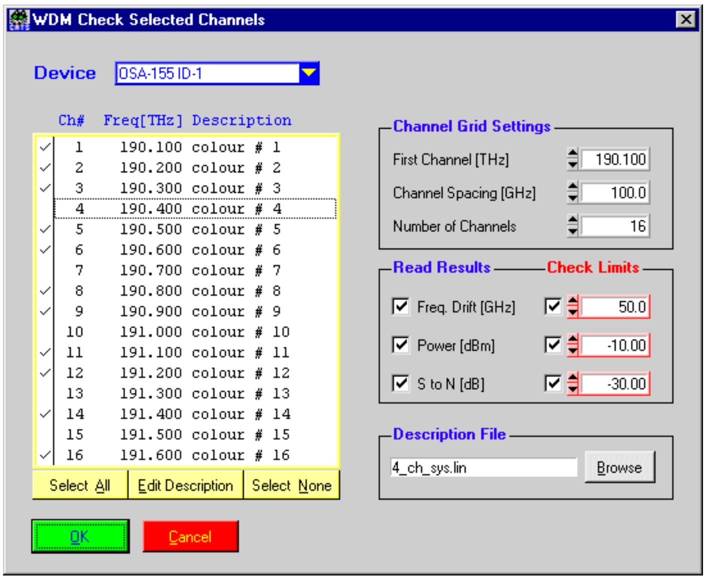
Notes:

3.7 Check results of selected channels

Function Name

```
_WDM_check_sel_chnls
              .\optic\osa155.obj
```

User Interface



Description

Purpose: Searches for the selected channels of a grid and evaluates the measured results against the defined limits.

Caution: The WDM measurement mode of the OSA-155 must be set to “AUTO”. Refer to the “WDM Set Parameter” testcase.

PASS / FAIL Conditions

PASS: Measured results are within the given limits and limit check is enabled.

FAIL: Measured results beyond limits and limit check is enabled.

Test Report

```
Check Sel. DWDM Ch.      PASS
   SIMULATION MODE
   OSA is in auto mode
    16 channels found.
    Ch. #     Freq[THz] | State     | Description
         2       190.200 | PASS      | colour # 2
         3       190.400 | PASS      | colour # 4
```

Applications

The testcases searches for the selected channels of a frequency grid.

The grid can be defined with the parameter “First Channel”, “Channel Spacing” and “Number of Channels”. For example the setting “First Channel = 193.100 THz” , “Channel Spacing = 100 GHz” and “Number of Channels = 16” will define a grid conform to ITU-T Rec. G.692.

In a first step the testcase reads the results of all channels found with the OSA-155.
To enable this measurement the OSA-155 is set to AUTO mode. In this mode the OSA searches for active D-WDM channels. (Refer to the OSA-155 manual for more information about the AUTO mode.)
In the second step the testcase tries to map the found channels to the related channel defined with the grid settings. If for example a channel at a frequency of 193.225 is found it will map it to the second channel (f = 193.2 THz) defined in the example grid above. the testcase will now evaluate the Power and SN to the given limits and calculate the frequency drift of this related channel.

Parameters

Device-Id:	Integer, 1 .. 10
Unit:	Enumeration wavelength in nanometer [nm] = 0, frequency in tera hertz [THz] = 1
First Channel:	Double, 185.057 .. 199.862 [THz]
Chnl. Spacing:	Integer, 25 .. 1000 [GHz]
# of Channels:	Integer, 1 .. 255
Freq Drift Flag:	Enumeration no Measurement and no limit check = 0, measurement without limit check = 1, measurement and limit check enabled = 3
F. Drift Limit:	Double, 0.0 .. 1000.0
Power Flag:	Enumeration no Measurement and no limit check = 0, measurement without limit check = 1, measurement and limit check enabled = 3
Power Limit:	Double, -60 .. 20 [dBm]
S to N Flag:	Enumeration no Measurement and no limit check = 0, measurement without limit check = 1, measurement and limit check enabled = 3
S to N Limit:	Double, 0 .. 100 [dB]
Description	
File Name:	String, Name of the file including the channel descriptions.
Chnl Flag x:*	Enumeration evaluate channel = 1, ignore channel = 0

* x is the number of the channel [x = 1 .. 255]

Remote Control of Testcase

Input:

%d#%d,%lf,%lf,%d,%d,%lf,%d,%lf,%d,%lf,#%s#,%d,%d, .. %d
Device-Id,Unit,First Channel,Chnl. Spacing,# of Channels,Freq. Drift Flag,Freq. Dift Limit,Power Flag,Power Limit,S to N Flag,S to N Limit, Decription File Name,Chnl Flag1,Chnl Flag2, ... Chnl Flag255

Output:

%d,%d,%lf[,%d,%lf][,%d,%lf][,%d,%lf],%d,%lf[,%d,%lf][,%d,%lf][,%d,%lf] ..
# of Channels,Channel Number,Channel Mid Frequency[,Status Frequency Drift,Frequency Drift][,Status Power, Power][,Status S to N,S to N],Channel Number,Channel Mid Frequency[,Status Frequency Drift,Frequency Drift][,Status Power, Power,][,Status S to N,S to N,] ..

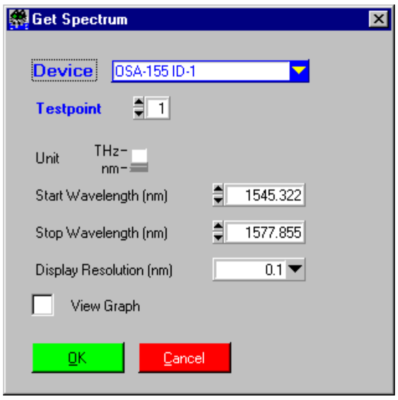
Values in [] brackets are only returned when the related flag is set in the input buffer.
Only the results of the selected channels are returned.
“Channel Number” is set to -1 if a second channels is found in the range defined with channel spacing, or if no signal is found for the selected channel. In the case of no signal the “Channel Mid Frequency” is additionally set to 0.0.

3.8 Get the actual spectrum

Function name

```
_get_spectrum
               .\optic\osa155.obj
```

User interface



Description

Purpose: Defines the spectrum display.

Caution: If the [View Graph] switch is selected, the “View Spectrum” window opens each time an error is detected and the current spectrum is displayed. The measurement will not resume until the “Spectrum window” is closed by clicking the [OK] button.

PASS / FAIL conditions

PASS: Not applicable.

FAIL: Not applicable.

Test report

```
Get_spectrum                  PASS
Spectrum data written to file:
.\osa_data\TP1_10_55-23_02_2000\SPEC11_49-
23_02_2000.csv
```

Applications

-

Parameters

Device:	Select device.
Testpoint:	Test point selector physical connection *Integer, 1 ... 10* *Default: 1*
Unit:	Select display in units of wavelength or frequency
Start Wavelength:	Set limit for spectrum display; enter smallest wavelength or highest frequency *Integer: 185057 ... 199862 THz or 1500.000 ... 1620.000*
Stop Wavelength:	Set limit for spectrum display; enter largest wavelength or lowest frequency *Integer: 185057 ... 199862 THz or 1500.000 ... 1620.000*
Display Resolution:	Set display resolution. *THz: 1, 10, 100* *nm: 1, 0.1, 0.01*

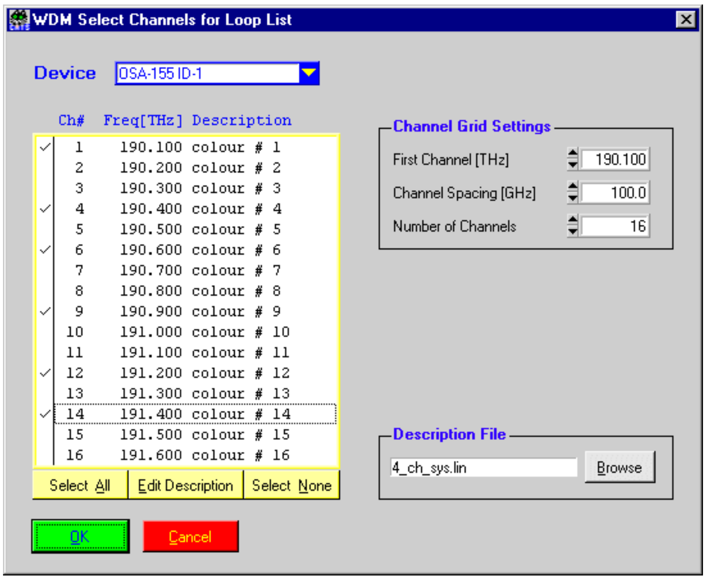
Notes:

3.9 Define channel list for loop

Function Name

```
_WDM_loop_select_list
              .\optic\osa155.obj
```

User Interface



Description

Purpose: Defines a list of D-WDM channels which will be set with the _WDM_loop_set_chnl testcase. In combination with a GOTO this list can be used to scan several channels and carry out any number of additional testcases for each channel form the defined list. (See application part for further information, or see example sequences _demdwdm.squ)

Caution: The WDM measurement mode of the OSA-155 must have been set to “AUTO”. Refer to the “Set WDM Parameter” testcase.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
DWDM Channel List        PASS
     SIMULATION MODE
     OSA is in auto mode.
      16 channels found.
        Ch.# Freq.[THz]  | Description
           1 190.100        190.125     | colour # 1
           2 190.200        190.195     | colour # 2
           3 190.300        190.303     | colour # 3
           4 190.400        190.370     | colour # 4
```

Applications

The testcases searches for the selected channels of a frequency grid. The grid is defined with the parameter “First Channel” , “Channel Spacing” and “Number of Channels”. For example the setting First Channel = 193.100 THz , Channel Spacing = 100 GHz and Number of Channels = 16 will define a grid conforming to ITU-T Rec. G.692.

The combination of the testcases _WDM_loop_select_list, _WDM_loop_set_chnl and a GOTO allows a loop of measuring testcases over several D-WDM channels e.g. with the ANT-20 when connected to the OSA-155 monitoring output.

Loop control is done in the following way:

_WDM_loop_select_list	definition of list
_WDM_loop_set_chnl	Set channel and increment entry of defined list
_test1	Testcase (loop body)
_test2	
_test_n	
GOTO **_WDM_loop_set_chnl**	GOTO is performed until the end of the list is reached

_further testcases, which are not inside the loop.

The GOTO shall only be performed if the testcase _WDM_loop_set_chnl is FAIL. The FAIL state indicates that there is still an entry in the list. To enable this mechanism a precondition for the GOTO must be defined. With this precondition the GOTO is only performed if the testcase _WDM_loop_set_chnl is FAIL. For more information about preconditions and how to set them refer to the chapter 1.4.3.4.

When the end of the list is reached ‘_WDM_loop_set_chnl’ testcase is set to the PASS state and therefore the GOTO will be skipped.

Please have a look at the example sequence ‘_demdwdm.squ’.

Parameters

Device-Id:	Integer, 1 .. 10
Unit:	Enumeration wavelength in nanometer [nm] = 0, frequency in tera hertz [THz] = 1
First Channel:	Double, 185.057 .. 199.862 [THz]
Chnl. Spacing:	Integer, 25 .. 1000 [GHz]
# of Channels:	Integer, 1 .. 255
Description	
File Name:	String, Name of the file including the channel descriptions.
Chnl Flag x:*	Enumeration evaluate channel = 1, ignore channel = 0

* x is the number of the channel [x = 1 .. 255]

Remote Control of Testcase

Input:	%d#%d,%lf,%d,%d,%d,%lf,%d,%lf,#%s#,%d,%d, .. %d Device-Id,Unit, First Channel,Chnl. Spacing, # of Channels,Decription File Name,Chnl Flag1,Chnl Flag2, ... Chnl Flag255
Output:	No Results

Notes:

3.10 Set next channel in loop

Function Name

```
_WDM_loop_set_chnl
               .\optic\osa155.obj
```

User Interface

Description

Purpose: Sets next channel from the list that was defined with testcase _WDM_loop_select_list.

Caution: The testcase _WDM_loop_select_list must be executed in front of this testcase.

PASS / FAIL Conditions

PASS: Not applicable.

FAIL: Not applicable.

Test Report

```
Incr. DWDM Ch.                    FAIL
     Filter set to  190.125 THz
     CH#  1, colour # 1
```

Applications

Refer to _WDN_loop_select_list.

Parameters

Device-Id:	Integer, 1 .. 10

Remote Control of Testcase

Input:	%d# Device-Id
Output:	No Results

Notes:

WWG-CATS Testcase Library

List of WWG-CATS Testcases

List of WWG-CATS Testcases

Notes:

1 SDH Version

This section contains the test cases that are included in the SDH version of the Test Sequencer.

1.1 Basic Testcases

See section 2

(included in BN 3035/95.90, 3045/93.12 and 3045/93.13)

Set ITU-T Signal Structure RX/TX	_set_signal_structure	.\sdh_bas\sig_stru.obj
Set ITU-T Channel Number RX/TX	_set_tributaries	.\sdh_bas\sdh_bas.obj
Set TX Code, PRBS, Optics, Clock, Trigger, Level	_set_tx_adaptation2	.\sdh_bas\sdh_bas.obj
Set RX Code, PRBS, Optics, Sensitivity, Equalizer	_set_rx_adaptation2	.\sdh_bas\sdh_bas.obj
Set SDH Overhead Byte	_set_SOH_POH_byte	.\sdh_bas\sdh_bas.obj
Set Parameters for G826 and Starts G826	_set_and_start_g826	.\sdh_bas\g826.obj
Set ITU-T Parameters for APS and Starts APS	_set_and_start_APS	.\sdh_bas\aps.obj
Set 2 MBit/s CRC SA Bit Sequences	_set_sa_bit_sequence	.\sdh_bas\sa_bits.obj
Setup ANX950 Module (VXI only)	_control_NRZ	.\sdh_bas\nrz.obj
Set Mapping Offset	_set_mapp_offset	.\sdh_bas\mapp_jit.obj
Define Trib.list for Scanning of Multiple Tribs.	_setup_mscan_trib	.\sdh_bas\mscantri.obj
Increment Trib-Number of defined Trib. List	_mscan_set_inc_trib	.\sdh_bas\mscantri.obj
Add / Remove Software Options	_set_options	.\sdh_bas\options.obj
Check Transparency of Selected Channel	_check_transparency	.\sdh_bas\sdh_bas.obj
Check Transparency of SDH DCCs, E1, F1, K1/K2	_check_byte_group_transparency	.\sdh_bas\sdh_bas.obj
Check SDH Overhead Byte Against Expected Value	_check_SOH_POH_byte	.\sdh_bas\sdh_bas.obj
Check 2 Mbit/s CRC Sa-Bit Sequence	_check_sa_bit_sequence	.\sdh_bas\sa_bits.obj
Check Clock Pulling Range (BER vs. Offset)	_check_clock_pulling_range	.\sdh_bas\sdh_bas.obj
Get Current G.826 Results (after G.826 Start)	_monitor_g826	.\sdh_bas\g826.obj
Get Final G.826 Results (after G.826 Start)	_check_g826	.\sdh_bas\g826.obj
Check APS Results	_check_APS	.\sdh_bas\aps.obj
Check Presence of a Group of Options (n of n)	_check_options	.\sdh_bas\options.obj
Check Presence of at Least One Option (1 of n)	_check_options_one_of	.\sdh_bas\options.obj
Check Round-Trip Pattern Delay	_check_pattern_delay	.\sdh_bas\delay.obj

1.2 Monitoring Testcases

See section 3

(included in BN 3035/95.90, 3045/93.12 and 3045/93.13)

Set SDH Pointer Actions and Sequences	_set_ptr_action	.\sdh_bas\ptr_move.obj
View SDH Pointer Movements Over Time	_ptr_move	.\sdh_bas\ptr_move.obj
Check Received SDH Ptr. Against Expected Value	_check_ptr_value	.\sdh_bas\ptr_move.obj
Show Current Alarm Situation	_read_actual_alarms	.\sdh_bas\actalarm.obj
Show Current SDH Overhead	_read_actual_SOH_POH	.\sdh_bas\soh_poh.obj
Scan PDH/SDH Hierarchies (TSE BIP/FAS and Alarms)	_scan_monitor	.\sdh_bas\scan_mon.obj

Check All Events (TSE, BIP, FAS, Alarms, ...)	_check_trouble	.\sdh_bas\trouble.obj
Read Frequency Offset	_read_rx_offset	.\sdh_bas\monitor.obj
Read Optical Level	_read_rx_optical_level	.\sdh_bas\monitor.obj

1.3 Alarm Sensor Testcases

See section 4

(included in BN 3035/95.90, 3045/93.12 and 3045/93.13)

Insert ITU-T Alarm	_set_alarm	.\al_sens\al_sens.obj
Insert Dynamic ITU-T Alarm	_set_alarm_N_in_M	.\al_sens\al_dyna.obj
Insert SDH Path Trace	_set_path_trace	.\al_sens\al_sens.obj
Insert ITU-T Errors or Offset Stimulus	_set_stimulus	.\al_sens\al_sens.obj
Insert ITU-T Alarm or Errors and Check Alarms	_check_alarm	.\al_sens\al_sens.obj
Insert ITU-T Errors or Offsets and Check Response	_check_stimulus_response	.\al_sens\al_sens.obj
Check SDH Path Trace Against Expected Value	_check_path_trace	.\al_sens\al_sens.obj
Read SDH Path Trace	_read_rx_path_trace	.\al_sens\al_sens.obj

2 SONET Version

This section contains the test cases that are included in the SONET version of the Test Sequencer.

2.1 Basic Testcases

See section 2

(included in BN 3035/95.90, 3045/93.12 and 3045/93.13)

Set SONET Signal Structure RX/TX	_set_sonet_structure	.\sdh_bas\sig_stru.obj
Set SONET Channel Number RX/TX	_set_sonet_tributaries	.\sdh_bas\sonetbas.obj
Set TX Code, PRBS, Optics, Clock, Trigger, Level	_set_tx_adaptation2	.\sdh_bas\sdh_bas.obj
Set RX Code, PRBS, Optics, Sensitivity, Equalizer	_set_rx_adaptation2	.\sdh_bas\sdh_bas.obj
Set SONET Overhead Byte	_set_sonet_TOH_POH_byte	.\sdh_bas\sonetbas.obj
Set SONET Parameters for APS and Start APS	_set_and_start_sonet_APS	.\sdh_bas\aps.obj
Setup ANX950 Module (VXI only)	_control_NRZ	.\sdh_bas\nrz.obj
Set Mapping Offset	_set_mapp_offset	.\sdh_bas\mapp_jit.obj
Define Trib.List for Scanning of Multiple-Tribs	_setup_mscan_trib	.\sdh_bas\mscantri.obj
Increment Trib-Number of Defined Trib. List	_mscan_set_inc_trib	.\sdh_bas\mscantri.obj
Add / Remove Software Options	_set_options	.\sdh_bas\options.obj
Check Transparency of Selected SONET Tributary	_check_sonet_transparency	.\sdh_bas\sdh_bas.obj
Check SONET Overhead Byte Against Expected Val.	_check_sonet_TOH_POH_byte	.\sdh_bas\sonetbas.obj
Check Clock Pulling Range (BER vs. Offset)	_check_clock_pulling_range	.\sdh_bas\sdh_bas.obj
Check APS Results	_check_APS	.\sdh_bas\aps.obj
Check Presence of a Group of Options (n of n)	_check_options	.\sdh_bas\options.obj
Check Presence of at Least One Option (1 of n)	_check_options_one_of	.\sdh_bas\options.obj
Check Round-Trip Pattern Delay	_check_pattern_delay	.\sdh_bas\delay.obj

2.2 Monitoring Testcases

See section 3

(included in BN 3035/95.90, 3045/93.12 and 3045/93.13)

Set SONET Pointer Actions and Sequences	_set_sonet_ptr_action	.\sdh_bas\ptr_move.obj
View SONET Pointer Movements vs. Time	_ptr_sonet_move	.\sdh_bas\ptr_move.obj
Check Received SONET Ptr. Against Expected Value	_check_sonet_ptr_value	.\sdh_bas\ptr_move.obj
Show Current SONET Alarm Situation	_read_sonet_actual_alarms	.\sdh_bas\actalarm.obj
Show Current SONET Overhead	_read_actual_TOH_POH	.\sdh_bas\tohpohso.obj
Scan PDH/SONET Hierachies (BIP/FAS and Alarms)	_scan_monitor	.\sdh_bas\scan_mon.obj
Check All Events (TSE,BIP,FAS,Alarms, ...)	_check_trouble	.\sdh_bas\trouble.obj
Read Frequency Offset	_read_rx_offset	.\sdh_bas\monitor.obj
Read Optical Level	_read_rx_optical_level	.\sdh_bas\monitor.obj

2.3 Alarm Sensor Testcases

See section 4

(included in BN 3035/95.90, 3045/93.12 and 3045/93.13)

Insert SONET Alarm	_set_sonet_alarm	.\al_sens\al_sens.obj
Insert Dynamic SONET Alarm	_set_sonet_alarm_N_in_M	.\al_sens\al_dyna.obj
Insert SONET Path Trace	_set_sonet_path_trace	.\al_sens\al_sens.obj
Insert SONET Errors or Offset Stimulus	_set_sonet_stimulus	.\al_sens\al_sens.obj
Insert SONET Alarm or Errors and Check Alarms	_check_sonet_alarm	.\al_sens\al_sens.obj
Insert SONET Errors or Offsets and Check Response	_check_sonet_stimulus_response	.\al_sens\al_sens.obj
Check SONET Path Trace Against Expected Value	_check_sonet_path_trace	.\al_sens\al_sens.obj
Read SONET Path Trace	_read_rx_sonet_path_trace	.\al_sens\al_sens.ob

3 SDH and SONET Version

This section contains the test cases that are included in the SDH version as well as the SONET version of the Test Sequencer.

3.1 Jitter Testcases

See section 6

(included in BN 3035/95.90, 3045/93.12 and 3045/93.13)

Setup TX Jitter Module	_setup_TX_jitter	.\sdh_bas\jitter.obj
Setup RX Jitter Module	_setup_RX_jitter	.\sdh_bas\jitter.obj
Insert Jitter	_setup_jitter_stimulus	.\sdh_bas\jitter.obj
Find Max. Mapping Jitter	_find_max_mapping_jitter	.\sdh_bas\map_jit.obj
Check RX Jitter	_check_jitter	.\sdh_bas\jitter.obj
Fast Jitter Tolerance Check	_check_jitter_tolerance	.\sdh_bas\jitter.obj
Maximum Jitter Tolerance Measurement	_check_mtj	.\sdh_bas\jitter.obj
Jitter Transfer Function	_check_jitter_transfer	.\sdh_bas\jitter.obj
Check Wander Tolerance	_check_wander_tolerance	.\sdh_bas\wander.obj

3.2 ATM Testcases

See section 5

(included in BN 3035/95.90, 3045/93.12 and 3045/93.13)

Set TX VPI/VCI, UNI/NNI and Load	_set_tx_vpi_vci_load	.\atm\atm_bas.obj
Set RX VPI/VCI & UNI/NNI	_set_rx_vpi_vci	.\atm\atm_bas.obj
Set TX AAL & Scrambler	_set_tx_atm_adaptation	.\atm\atm_bas.obj
Set RX AAL & Scrambler	_set_rx_atm_adaptation	.\atm\atm_bas.obj
Scan VCI and Show Load	_scan_vci_for_load	.\atm\atm_bas.obj
ATM QoS Check (SONET or SDH or PDH)	_check_atm	.\atm\atm.obj
Check Transparency of a List of VCI/VPI	_check_vci_vpi_transparency	.\atm\atmtrans.obj
Check Cell Delay Variation w/Fixed Resolution	_check_CDV	.\atm\celdelay.obj
Check Cell Delay Variation w/Auto Resolution	_check_auto_CDV	.\atm\celdelay.obj

3.3 Auxiliary Testcases (Toolkit)

See section 8

(included in all packages)

Waiting Time (Delay in [ms])	_wait	.\toolkit\toolkit.obj
Wait Until a Predefined Date/Time is Reached	_wait_until_time	.\toolkit\toolkit.obj
Wait a defined time and increments an loop counter	_repeat_wait	.\toolkit\toolkit.obj
Increments an loop counter	_loop_counter	.\toolkit\toolkit.obj
Pop-Up Message or Question towards User	_user_io	.\toolkit\toolkit.obj
Send SCPI Command to Instrument and Read Result	_scpi_io	.\toolkit\toolkit.obj
Beep	_acoustical_signal	.\toolkit\toolkit.obj
Measure Elapsed Time between 2 Calls of this Test	_stopwatch	.\toolkit\toolkit.obj
Print Time Stamp to Report	_time_stamp	.\toolkit\toolkit.obj
Entry Fields for UUT / User Info	_show_info_field	.\toolkit\toolkit.obj
Show Bitmap (e.g. Picture of Cabling)	_show_image	.\toolkit\toolkit.obj

3.4 Testpoint Selector Control

See section 3

(included in BN 3045/93.20)

Set Switch Position of Testpoint Selector	_scan_select_channel	.\scanner\scan.obj
Setup Channel List for Testpoint Selector	_setup_mscan_ch	.\scanner\mscan.obj
Set & Increment Number of Channel List	_mscan_set_inc_ch	.\scanner\mscan.obj
Reset Channel List or Counter	_mscan_reset_list	.\scanner\mscan.obj

3.5 Optical Testcases

See section 7

(included in BN 3045/93.41 and 3045/93.42)

Set Single Laser Source	_setup_OLS	.\optic\ovxoptic.obj
Set Dual Laser Source	_setup_OLS_WDM	.\optic\ovxoptic.obj
Initialize OLP Zeroing (not for OLP-90)	_init_OLP	.\optic\ovxoptic.obj
Set Optical Power Meter	_setup_OLP	.\optic\ovxoptic.obj
Set Optical Power Meter in WDM Mode	_setup_OLP_WDM	.\optic\ovxoptic.obj
Set Optical Attenuation	_setup_OLA	.\optic\ovxoptic.obj
Set Optical Switch (OSW-114)	_setup_OSW114	.\optic\ovxoptic.obj
Set Optical Switch (OSW-122)	_setup_OSW122	.\optic\ovxoptic.obj
Set Optical-Electrical Converter	_setup_OEC	.\optic\ovxoptic.obj
Set Optical Customer Module	_setup_OCD	.\optic\ovxoptic.obj
Read Optical Power	_check_OLP	.\optic\ovxoptic.obj
Read Optical Power in WDM Mode	_check_OLP_WDM	.\optic\ovxoptic.obj
Check Insertion Loss	_iloss_OLP	.\optic\ovxoptic.obj
Check Insertion Loss in WDM Mode	_iloss_OLP_WDM	.\optic\ovxoptic.obj

Customized testcases combining ANT-20 with OMS-150/200 are available on request. These include optical sensitivity, overload, BER vs attenuation and similar testcases that can be specially adapted to customer requirements.

3.6 D-WDM Testcases

See section 9

(included in BN 3045/93.43)

Calibrate OSA	_calibrate_OSA	.\optic\osa155.obj
Set filter to specified wavelength/frequency	_set_filter_man	.\optic\osa155.obj
Set parameter of DWDM mode	_WDM_set_parameter	.\optic\osa155.obj
Set S to N parameter of WDM Mode	_WDM_set_OSNR_parameter	.\optic\osa155.obj
Read all results of found channels	_WDM_read_all_channels	.\optic\osa155.obj
Check all results of found channels	_WDM_check_all_channels	.\optic\osa155.obj
Check results of selected channels	_WDM_check_sel_chnls	.\optic\osa155.obj
Get the actual spectrum	_get_spectrum	.\optic\osa155.obj
Define channel list for loop	_WDM_loop_select_list	.\optic\osa155.obj
Set next channel in loop	_WDM_loop_set_chnl	.\optic\osa155.obj

Notes:

Developers Guide

Developers Guide

1 Introduction

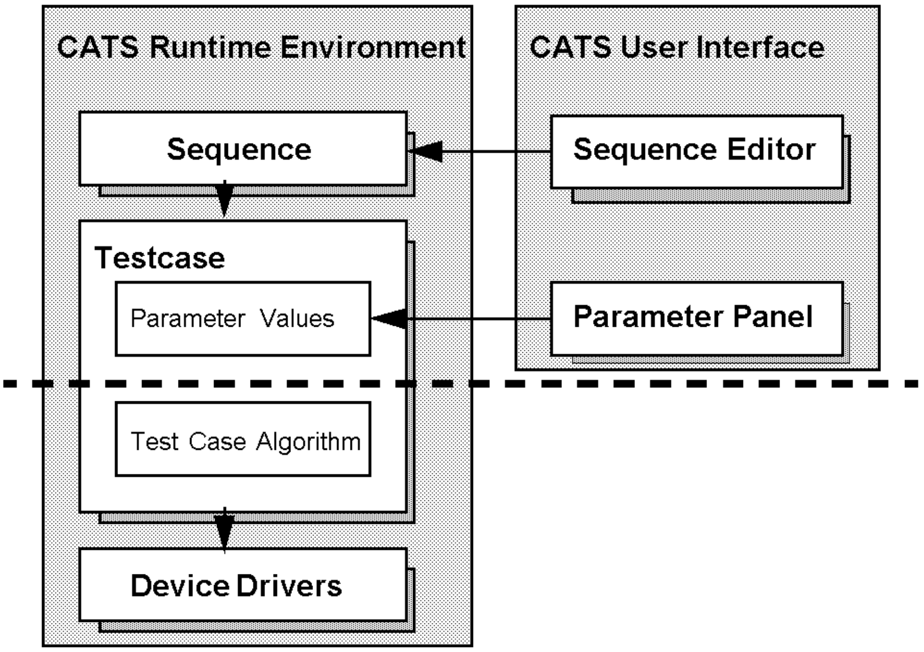
CATS and the ANT-20 CATS Test Sequencer derived from it are LabWindows CVI applications based on National Instruments “Test Executive”. The software packages form a platform for producing test sequences for automating tests that can be employed during development, production and network operation.

Preassembled Test Case Templates are used to produce the sequences. Concrete testcases are derived from the templates by setting specific parameters. Specially tailored user interface panels provide support for setting the test parameters. Since each testcase generally results in a PASS/FAIL statement, permitted limits and expected results can also be set as parameters. Each template is implemented using the C programming language and specifies the test algorithm.

The testcases (or the underlying templates) perform their functions through the instrument drivers, which handle communications to the measuring equipment.

CATS itself, the Test Case Templates and the instrument drivers are developed using LabWindows CVI. A complete LabWindows CVI development environment is therefore required to perform specific extensions.

The following diagram shows the general structure in simplified form.



The dotted line in the above diagram indicates the interface between a pure CATS application (above the line) and working with the complete development environment. The production of user-defined sequences and the providing of parameter sets for the Test Case Templates is possible without the CVI development environment and is described in the “CATS Operator’s Manual”.

This manual is concerned with all aspects of working with CATS that require the use ofLabWindows CVI. The description assumes prior experience in application development under CVI, i.e. it is not a substitute for the corresponding documentation provided by National Instruments (NI). All information directly concerned with handling and operating CVI should be taken from the appropriate manuals from NI.

The following software packages are required:

- LabWindows CVI
- CATS Base Package (BN 3045/01)
- Test Case Package (BN 3045/93.12 or BN 3045/93.13 including the source code if existing testcases are to be modified)
- Optional further application packages.

The following topics will be covered:

- Directory structure
  The files associated with CATS are contained in various directories. This structure cannot be simply altered, as it is reflected in parts of the software. Knowledge of this structure is therefore of particular interest when you wish to create your own files.
- Integration of additional application packages
  The standard CATS package includes an executable file containing the drivers necessary for controlling the ANT-20 modules. With this configuration, you can execute all applications that are possible using one (or more) ANT-20 (or ANT-20E, Domino, ANX-9xx). If you want to integrate further application packages defined by Wavetek Wandel Goltermann (generally consisting of device drivers, sequences and Test Case Templates), then this section will be of interest to you.
- Integration of additional instruments
  This section covers the inclusion of instruments that have not been integrated into CATS by Wavetek Wandel Goltermann. By building on LabWindows CVI drivers, CATS provides the opportunity of including existing equipment into the new CATS environment.
- Creating / modifying Test Case Templates
  This section explains the structure of the Test Case Templates implemented in C. This information is of interest if you want to alter the algorithm of an existing test template (e.g. to introduce additional parameters, add wait times, change the way results are displayed, etc.), or to develop your own templates based on the built-in device drivers. The way that the user interface panel is linked to the templates is also described in this section.
- Creating test sequencer applications
  The ANT-20 CATS Test Sequencer is a product derived directly from CATS. It is therefore logical that users of the CATS development environment should have the opportunity to create applications that can be directly executed on the ANT-20 (e.g. for controlling the “internal” ANT-20 and an “external” OMS-200). For this reason, purchase of the CATS Base Package (BN3045/01) also includes the right to run the executable files that you generate on a specific ANT-20. This section describes how this is accomplished.
- Integrating CATS into existing applications
  A special feature of CATS applications is the ease of integration into existing environments (Visual Basic, Microsoft Excel, HP-VEE, etc.). This section explains the concept of “Test Case Servers” which allows simple integration of CATS using standard Windows/UNIX mechanisms.

2 Directory structure

The following section describes the directory structure of CATS. To avoid problems when changing to later versions of CATS, it makes sense is to retain this structure when you are developing your own applications.

Note: CATS should only be installed in the suggested directory, otherwise run time problems will result (when loading the sequences).

The following directories are created during installation:

- config
  The “config” directory contains all the files required for operation of the Configuration Window. You should not make any changes to this directory.
- include
  This directory contains all the header files (*.h) required for translating the Test Case Templates . The headers belonging to the platform and the include files for the drivers used are located here. The headers located in other directories are only used locally.
- drivers
  The actual LabWindows CVI drivers are located in this directory. The associated Function Panels (*.fp files) are also stored here although they do not have any meaning within the CATS environment.
- platform
  All the files which permit integration of the drivers into the CATS environment are located here. Auxiliary and service functions are also found in this directory.
- report
  The report files created during the execution of a test are stored here.
- sequence
  The sequences (*.squ) are saved in the “sequence” directory. These include the examples created by Wavetek Wandel Goltermann and the user-defined sequences.
- txengine
  This directory contains the files belonging to the actual platform. These are basically files that were originally developed by National Instruments. You should not make any changes or additions to this directory.

In addition to these platform directories, other directories also exist for the individual applications (“sdh_bas” for SDH/SONET/PDH, “atm” for ATM, etc.). It is recommended to create a separate directory for each application to include the associated Test Case Templates (object and UIR files).

3 Integration of application packages

The architecture of CATS makes it possible to expand existing applications step-by-step by the addition of further application packages (consisting of drivers, test cases and sequences). For this reason, further application packages (e.g. Optics, BN3045/93.41 or BN3045/93.42) are available as options to the CATS Base Package (BN3045/01).

Subsequent integration of these packages requires the creation of a new executable file, which requires the use of the LabWindows CVI development environment.This section describes how this is done.

Briefly, the following steps are required:

- Installation of the application package
- Creation/adaptation of the project file
- Adaptation of the configuration
- Generation of the new executable file

3.1 Installation of the application package

Application packages are installed using the installation disk(s) by executing the program "a:\setup.exe".

Note: The suggested installation path should not be altered. If it is altered, the sequences included with the application will not run and will need to be edited.

3.2 Creation/adaptation of a project file

LabWindows CVI uses a project file (extension ".prj") to handle all the files belonging to a particular project or executable file. A project file is always required if you wish to make use of the CVI development environment (e.g. to translate source files or to create a new executable file).

To create/adapt a CATS project file, please proceed as follows:

1. Start LabWindows CVI.
2. Load the appropriate project file (if it has already been generated) or create a new project.
3. Add the following files to the project:
   - all object files contained in ".\txengine"
   - all object files contained in ".\platform"
   - all object files contained in ".\drivers"
   - "conf.c" from ".\config"
   - "wgwsdrvw.dll"
4. Set an include path to ".\include".
5. Save the project file under the name of your choice.

Note: The method described results in an increase in the size of the executable file, since all the available device drivers will be linked. It is therefore recommended to remove devices that are no longer required. The files to be removed can be found in the listing contained in Sec. 9, Page 1-24 (Annex: "List of device-specific files") or the "read_me.txt" file included on the corresponding installation disk.

3.2.1 Adaptation of the configuration

The file ".\config\conf.h" is used to define the actual system configuration. The content of this file determines which devices will or will not be visible in the Configuration Window.

The following information must be included for each device:

- Unique identification number of the device class.
  The ID is set using "#define"; the define statement always begins with "VI_". When the program is run, this ID is used to select the appropriate adaptation functions (XX_open_device, XX_close_device, etc.). To prevent possible multiple assignment of IDs, you should use IDs > 500 for devices that you integrate.
- Prototype adaptation functions
- Table of adaptation functions ("device_tab")
  A name, the unique ID, a mask value and the names of the device-specific adaptation functions are entered in these tables of the "DEVICE_CLASS" type (see ".\include\devices.h") for each device integrated into CATS.

Notice: The value of the "VImask" component must on no account be altered. If it is altered, pointer errors will cause the program to crash. If you integrate your own devices, this value should be set to "0".

To simplify handling, additional define statements should be included to control the configuration. These define statements begin with "CFG_" and are set via the "Options", "Compiler Defines" command.

Devices integrated by Wavetek Wandel Goltermann are normally already included in "conf.h", so that it is only necessary to activate the corresponding compiler define statement. If the device is not included, you will find information about the device ID and the names of the adaptation functions in the "read_me.txt" file on the installation disk.

3.2.2 Generating a new executable file

It is advisable to generate an executable program for test purposes within the LabWindows CVI development environment. You can generate such a test version using the "Build Project" function. If errors occur during linking, the files contained in the project file do not match the desired configuration.

The last step is to generate an executable program that will run outside the LabWindows CVI environment using the "Build Stand-Alone Executable" function.

4 Integration of additional instruments

As an advanced user, you will need to integrate further devices into your CATS environment that are not included in the standard scope of delivery from Wavetek Wandel Goltermann.

This section describes the basic ways in which this aim can be achieved.

Briefly, they are:

- Controlling the device using the “Generic Driver”
- Integration of an additional device driver

4.1 Integration using the Generic Driver

The “Generic Driver” is included as standard with CATS and allows communication with any device which is IEEE 488.2-compatible. Communications can be via any interface (v.24, GPIB, GPIB-VXI, etc.).

A Test Case Template for exchanging SCPI commands (“SCPI-IO Test Case”) based on this driver has been developed. More information about this test case is found in the “CATS Operator’s Manual”.

This mechanism is sufficient to integrate devices that only require setting (e.g. test point scanners, voltage sources, attenuators, synthesizers, etc.).
This method reaches its limits, however, as soon as results are to be read out, since the results read out can only be checked “manually” rather than automatically by the test case.

4.2 Integration of an additional device driver

If integration using the “Generic Driver” as described above does not meet the requirements, a device-specific LabWindows CVI driver must be integrated. If no CVI driver is available or if the driver functions are insufficient, the first step is to (further) develop the driver. The functions to be implemented are governed by the application that is to be realized, i.e. the desired test cases. Development of the driver is not handled here (see documentation provided by NI).

When integrating a driver into CATS the following should be done:

- Linking the driver-specific “open” function to loading of a sequence (Configuration Window).
- Linking the driver-specific “close” function to exiting from a sequence.
- Optional linking of the “send-command” and “query” functions to the SCPI-IO test case.

The integration of Configuration Window, test case and driver is accomplished using five standardized adaptation functions and a data structure that takes care of the exchange of information between these software modules. The adaptation functions are saved in a file (file name beginning “ini_”) in the “.\platform” directory.

To simplify orientation, the function names should begin with the device name (represented by “XXX” below). The file “ini_bsp.c” contains example implementations.

The sections below explain the following aspects:

- DEVICE structure
- The “XXX_open_device” function
- The “XXX_close_device” function
- The “XXX_send_command” function
- The “XXX_query” function
- The “XXX_init” function

4.2.1 DEVICE structure

This data structure is used to exchange information between the Configuration Window, test case and the functions used for integration of the driver. The “DEVICE” structure is defined as follows in the “devices.h” header in the “.\include” directory:

```
typedef struct
{
int aux[3];                /* Auxiliary variables                    */
char name[20];             /* Description of device purpose (TX, ...)*/
int monitor;               /* Monitoring ON / OFF                    */
int dev_type;              /* Device class id, VI_SF60, ...          */
int interface;             /* Interface (GPIB, GPIB-VXI, ...)        */
int addr;                  /* Primary address                        */
int sec_addr;              /* Secondary address                      */
unsigned long handle;      /* Device handle returned by “OpenDev”    */
unsigned long aux_handle;  /* Additional handle allowing handling of*/
                           /* device combinations (ANX-920 + NRZ)    */
int simulation;            /* Flag indicating simulation mode        */
} DEVICE;
```

The first seven components in the structure are filled with the contents of the Configuration Window. The information “Aux. Select” for addressing the OMS/OVX modules is saved in “aux[0]”.

4.2.2 The XXX_open_device function

This function has the task of opening the device (over the interface selected by the user) with the aid of the function(s) provided by the driver for this purpose. For subsequent addressing of the device from the application, the handle supplied is saved in the corresponding structure components. A pointer value pointing at the device structure is transferred to the function as a parameter. “Error check” should always be “ON”. You should activate or deactivate the monitoring (if supported by the driver) depending on the content of the “monitor” structure component.

Many drivers support a simulation mode which allows applications to be run without the device being connected. Simulation is automatically selected whenever opening the device fails.

Notice: The user must be informed that the device could not be opened. Simulation must not be activated without any comment, as the user may not then be able to distinguish between real and simulated results under certain circumstances. The “simulation” flag must be set to the value “1”.

4.2.3 The XXX_close_device function

The function closes the device with the aid of the corresponding driver function. A pointer value pointing at the device structure is transferred to the function as a parameter.

4.2.4 The XXX_send_command and XXX_query commands

The CATS platform realizes the link to the “SCPI-IO Test Case” with the aid of these adaptation functions. Commands are sent to the device and results queried by calling up the corresponding driver functions. In each case, “dp->handle” is used for addressing.

4.2.5 The XXX_init function

This function is used internally by Wavetek Wandel Goltermann and does not therefore require filling out. “return 0” is sufficient for the function body.

5 Creating/modifying test cases

The following section describes the implementation of customized Test Case Templates and their integration into CATS. This information is also useful where existing test cases developed by Wavetek Wandel Goltermann are to be modified.

The first section contains a general explanation of the structure. The second section describes the code template (file ".\codefram.c") that is included as standard with the CATS package.

The following aspects are covered:

- General structure
- Code template

5.1 General structure

The CATS concept distinguishes between the creation and the execution of an application. In the first step, the parameters are set for each test case ("Double Click Left") required to perform the overall task . No parameters are set during actual execution of the test ("Test UUT", "Single Pass" or "Run Test"); the application makes use of the pre-defined values.

This mechanism is reflected in the structure of the test cases. Each Test Case Template consists of the function that actually realizes the test (this function is basically independent from LabWindows CVI and CATS) and a function which integrates the test function into the CATS environment. The (optional) user interface panel is also linked in with the aid of this function.

The sections below describe the following parts:

- User interface for entering the parameters
- Test function for performing the actual test
- Adapter function for adapting the test function to the CATS-specific calling conventions.

5.1.1 User interface panel

A special user interface panel is developed for each test case. This interface appears after "Double Click Left" on the name of the corresponding test case.The layout and content of the interface depend on the aim and purpose of the test. All relevant parameters should be included. In rare cases, development of an interface can be omitted (whenever no special parameter setting is required). It is, however, recommended to always provide a means for entering the logical device IDs.

To ensure correct linking to the code template described below, the "OK" (Constant Name "OK") and "Cancel" (Constant Name "CANCEL") buttons must exist.

Each user interface panel is saved in its own file (file name extension ".uir").

Note: Make sure that the UIR file and the object file containing the test case are saved in the same directory, otherwise it will not be possible to load the interface when the program is run.

5.1.2 Test function

This function has the task of performing the actual test. The test algorithm is defined in this function. Programming is done on the basis of the devices or their drivers that are required to perform the test. The function parameters can be freely defined. The return values are always of the “Status” type, i.e. “PASS”, “FAIL” or “ABORT”.

The two following sections look at two special features of the CATS architecture necessary for the correct function of the test cases.

Use of the service functions

Use of the service functions (which can be recognized by function names starting with “tc_”) is a major requirement for the correct functioning of test cases in all CATS operating modes. Linking of the test cases to the CATS platform is performed by calling up service functions. The developer does not need to be concerned with these routine tasks.

The task and usage of each service function are described in detail in Sec. 8, Page 1-20. In addition, the code template supplied (see Sec. 5.2, Page 1-12) gives information about the basic structure. At this point, only those points which must be observed without fail will be handled briefly:

- The first command in a test function must be to call “tc_start”. If this command is missing, the program will crash due to non-initialized pointers.
- A test function must not be exited without calling “tc_return”, otherwise errors in dynamic memory management will result.
- Information which is to be included in the report must be written with the aid of “tc_print”.
- Results which are to be delivered to the controlling platform (see Sec. 7, Page 1-19) via DDE or
  TCP/IP are generated using “tc_results”. The string transferred as a parameter contains the results in CSV format (“comma separated values”).

In general, the use of LabWindows-specific functions should be avoided if a corresponding service function is available. For example, always use “tc_operator_prompt” instead of “MessagePopup”.

Addressing the devices

The platform takes over the opening and closing of the devices integrated into CATS. After exiting from the Configuration Window all selected devices are opened. They are all closed again when the sequence is exited.

Note: The correct function of these mechanisms requires that the corresponding drivers have been integrated as described in Sec. 4.2, Page 1-6.

On leaving the Configuration Window, the platform assigns “logical device IDs” to all devices that are to be opened. The ID is already shown in the Configuration Window at the beginning of every line, corresponding to the line number. Within the test function, the instruments are addressed via a device table and these logical IDs. Thus, for example, the handle for the instrument with the logical ID “1” is in “device[1].handle”. This allows all components in the device structure to be addressed.

To keep the test function as flexible as possible, it is advisable to transfer the logical IDs of the devices involved in the test as parameters and to allow them to be set using the user interface panel. This will allow a test case to be used with different device configurations.

5.1.3 Adapter function

The adapter function serves to integrate the test function into the CATS environment. The function name is derived from the test function name by preceding this with an underscore (_).

The adapter function fulfills the following tasks which are described below:

- Adaptation of the test function to the call up conventions of CATS/Test Executive
- Linking of the user interface panel to allow input of the parameters

Adaptation of the test function to the call up conventions of CATS/Test Executive

To allow any test case to be integrated into CATS, all tests must present the platform with the same interface. This interface is fixed within CATS/Test Executive and consists of the two structures described below:

- “data” of the type “tTestData”
  The exchange of information between the test case and the platform is accomplished with the aid of this structure. The following structural components are of importance:
  - inBuffer
    CATS/Test Executive handles the parameter values for every single test case in the sequence currently loaded. These values are loaded and saved together with the sequence. These parameter values are transferred from the platform to the test case using the “inBuffer” component (also called the “Input Buffer”) of the type “char *”. The values are read out of this string into the variables that represent the parameters using the “sscanf” function.
  - hook
    The adapter function is called up for the actual performance of the test function **and** for entering the parameters. The contents of the “hook” component is used to distinguish between these two cases when the test case is called. On exiting from the test, the test results required for remote controlled operation are returned using this pointer.
  - result
    “result” returns the status of the test function to the platform. The possible values (defined using the “#define” function) are “PASS”, “FAIL” and “ABORT” (indicating a run-time error).
  - outBuffer
    The “outBuffer” character buffer contains the test results as a string. The buffer is shown in the display by the platform software and is saved in the report file.
- “error” of type “tTestError”
  This second structure is used to handle errors. The following components are employed:
  - errorFlag
    “errorFlag” is set to “1” if a run-time error occurs in the test function. Otherwise, “0” is returned.
  - errorCode
    “errorCode” returns the error code to the platform.
  - errorMessage
    “errorMessage” gives plain text information on the cause of the error.

The adapter functions do not differ much between (simple) test cases. In particular, the use of the above-described structures can be largely standardized. The file “codefram.c” described in Sec. 5.2, Page 1-12 contains a code template which can be easily adapted.

Linking in the user interface panel

The user interface panel (saved in a UIR file) is treated using standard LabWindows CVI functions. There are no restrictions or special conditions due to their use in the CATS environment.

The procedure can be generally divided up as follows:

- Loading the UIR file and starting the user interface panel
  After loading the UIR file (“LoadPanel”), the user interface panel is started. This is done using the “InstallPopup” function to ensure that the panel is exited before the sequence is started.
- Setting the parameters
  The panel controls are initialized using the parameter values read out of the input buffer (“SetCtrlVal”). If the input buffer is empty (i.e. the first time the panel is called) or the values cannot be read correctly, the default values will be set.
- Using the panel
  In the simplest case, the system waits in a loop (“GetUserEvent”) until the user clicks on “OK” (accept new values) or “CANCEL” (retain old values). If the parameters are interdependent, this will also be handled within the loop or by means of separate call backs. The panel is removed in conclusion (“RemovePopup”).
- Reading the parameters
  If the parameter values are to be set (“OK” button) they are read into the corresponding parameters (“GetCtrlVal”). The values are written to a string buffer using “sprintf” and then transferred to the platform by calling up the “TX_SetParameters” function.

5.2 Code template

The file “codefram.c

```
/*------------------------------------------------------------------\
File:     ** 1 **.c
Version:  1.00
Copyright Wandel & Goltermann 1996
--------------------------------------------------------------------

Description:
This file contains test cases to ** 2 **.

Each test consists of two functions. One builds the test itself, the
other adapts the test to Wavetek Wandel Goltermann CVI test executive.

The name of all CVI specific test cases start with the prefix "_".

The test case function which executes the test returns "Status".

No platform specific code should be included in the test case itself
to ensure portability to other test platforms.

To build platform specific objects use the following compiler defines:
CVI:    NI Test Executive

--------------------------------------------------------------------
History of changes:
** 3 **  [M00]  ** 4 **   - Original Release
** 3 **  [M01]  ** 4 **   - ** 5 **
------------------------------------------------------------------*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
#ifdef CVI
#include <txengine.h>
#include <userint.h>
#include <utility.h>

/* Include Files generated from the UIR-Files */
#include "** 6 **.h"
#endif

#include "tc.h"
#include "util.h"
#include "devices.h"

/* Driver Include Files */
#include "** 7 **.h"

/*----------- Copyright 1996 Wandel & Goltermann  -------------------
Description:  ** 8 **.
Return Value: Test status (PASS, FAIL, ABORT)
-------------------------------------------------------------------*/

Status ** 9 ** (** 10 **)

  {
  tc_start ("** 11 **");
  ** 12 **
  return (tc_return (PASS));
  }


/*----------- Copyright 1996 Wandel & Goltermann  --------------------
Description:  ** 8 ** (GUI - TEST EXECUTIVE adaption).
Return Value: None
-------------------------------------------------------------------*/


 void _** 9 ** (tTestData *data, tTestError *error)
  {
  int pnl_handle;                   /* CVI panel handle             */
  int panel;                       /* Id of panel creating the event */
  int control;                     /* Id of control creat. the event */

  char path[50];                    /* Path name of UIR file        */
  char *buffer;                    /* Pointer to temporal buffer    */

  int tx_id;                        /* Device id                    */
  int rx_id;                        /* Device id                    */
  ** 13 **

  if (sscanf (data->inBuffer, "** 14 **", ** 15 **) != ** 16 **)
   {
   /* Set test parameters to default values */

   tc_warning ("Parameter set to default values");
```

```
    tx_id = 1;
    rx_id = 2;
    ** 17 **
    }

  if (data->hook == (void *)-1)
    {

#ifdef CVI

    sprintf (path, "%s%s", data->modPath, "** 18 **.uir");
    pnl_handle = LoadPanel (0, path, ** 19 **);

    /* Pre-set Controls of user interface panel */
    SetCtrlVal (pnl_handle, ** 20 **, ** 21 **);

    InstallPopup (pnl_handle);

    /* Handle user interface panel */
    do
     {
     GetUserEvent (1, &panel, &control);
     }

    while (control != ** 19 **_OK && control != ** 19 **_CANCEL);

    /* Get actual values from user interface panel */
    GetCtrlVal (pnl_handle, ** 20 **, &** 21 **);

    RemovePopup (0);
    DiscardPanel (pnl_handle);

    if (control == ** 19 **_OK)
     {
     buffer = (char *)malloc (** 22 **);
     sprintf (buffer, "** 23 **", ** 24 **);
     TX_SetParameters (buffer);
     }

#endif

    }
  else
    {
    /* Store parameters (necessary for remote operation) */
    tc_param (data->inBuffer);

    /* Call function performing the test case */
    data->result = ** 9 ** (** 25 **);
    if ((error->errorCode = tc_get_err_msg (&(error->errorMessage))) !=
    0)
```

```
    {
    error->errorFlag = 1;
    }
   else
    {
    /* Return test report */
    data->outBuffer = tc_get_report ();

    /* Return test results (necessary for remote operation) */
    data->hook = tc_get_results ();
    }
  }
}
```

The file “codefram.c” contains the framework of a C source file for creating Test Case Templates. This makes it possible to integrate your own test cases into CATS with the minimum of effort.

This section explains the content and structure of the code template. Changes to the code template should only be made with extreme caution and if you have precise knowledge of the structures.

The sections marked with ** x ** indicate the corresponding locations within the file “codefram.c”.

- ** 1 **
  Name of the C file. The name should not exceed 8 characters in length (even if run under Win95/NT or UNIX) to allow simple transfer to Windows 3.11. Customized test cases should normally be saved in separate source files to avoid problems when updating. If test cases developed by Wavetek Wandel Goltermann are modified, these should also be copied to separate source files.
- ** 2 **
  General description of the test cases saved in the source file and their application. The individual test cases are described in the header of the corresponding function.
- ** 3 **
  Modification date. Every modification after a given date (at the latest when the file is supplied to a third party) should be recorded in the file header. Each modification is indicated by a consecutive number (format [Mxx]). This number is inserted as a comment at the modified point in the code to allow these modifications to be located easily. the modification date is entered in the format “dd.mm.yy”.
- ** 4 **
  Name of the person making the modification.
- ** 5 **
  Brief description of the modification.
- ** 6 **
  Names of the header files belonging to the user interface panels. As a rule, each test case has one user interface panel (UIR file) for entering the parameters. For each UIR file, CVI creates a header file with the same name (“jitter.uir” ==> “jitter.h”). The header files for all test cases stored in the file must be included at this point.

Notice: The contents of this automatically generated header must not be altered.

- ** 7 **
  Names of the header files belonging to the drivers used. There is a header file for each CVI device driver. The header files for all device drivers addressed in the file must be included at this point.

- ** 8 **
  Short description of the test case.
- ** 9 **
  Name of the C function. The name should describe the task performed as closely as possible.
- ** 10 **
  Parameters of the C function. There are no restrictions on the possible data types at this point. All the possibilities available in “C” can be used. In general, each parameter is linked to an input facility (Control) within the user interface panel. Constants which cannot be altered via the panel may also be used.
- ** 11 **
  Test name. The string entered here has no meaning in the CATS environment. Despite this, the call up of “tc_start” at this point is absolutely necessary. The command must not be removed, because important pointers that are necessary for correct function are initialized at this point.
- ** 12 **
  Test case algorithm. The algorithm of the test case is programmed here.
- ** 13 **
  Parameter definitions.
- ** 14 **
  The format command describes the input buffer format, i.e. the way in which the individual parameters are saved. The precise format depends on the format used for writing the input buffer (see ** 23 **).
- ** 15 **
  List of parameters to be read.
- ** 16 **
  Number of parameters to be read. “sscanf” returns the number of parameters read correctly from the buffer. If this number does not correspond to the number of parameters actually stored, all parameters will be set to their default values. This may occur for two reasons:
  - The input buffer is empty.
    This is always the case when the test is called for the first time. In this case, setting the parameters to their default values is both intended and useful. The warning can therefore be ignored.
  - The input buffer was written using the wrong format command (see ** 23**).
    In this case, the user interface panel is ineffective, the test can only be performed with the default values. The built-in warning draws attention to this error situation.
- ** 17 **
  A default value should be specified here for each parameter. If certain parameters are dependent on other parameters, attention must be paid to setting defaults that are compatible with each other, otherwise this may lead to run-time errors.
- ** 18 **
  Name of the UIR file. The name should not exceed 8 characters in length (even if run under Win95/NT or UNIX) to allow simple transfer to Windows 3.11. Customized user interface panels should generally be stored in separate files to avoid problems when updating. If the user interface panels developed by Wavetek Wandel Goltermann are altered, these should also be copied into separate files.
- ** 19 **
  Name of the panel. The name is entered as the “Constant Name” of the panel when creating the user interface panel. The definition is included in the header file generated by CVI.
- ** 20 **
  Name of the control. Every control is assigned a name when the panel is created. The control is addressed in the C code using this name, in that it must always be preceded by the panel name (see ** 19**) and an underscore (_).

- ** 21 **
  Name of the variable into which the value is to be read.
- ** 22 **
  Length of the buffer required for storing the parameter. The length of the string must not exceed 1000 characters.
- ** 23 **
  This format command describes the format of the input buffer, i.e. the way in which the individual parameters are stored. Special attention is required if general texts are to be saved. These texts must not contain “Newline \n” as this may lead to problems in editing using the sequence editor. Texts of more than one line must be saved line-by-line, (i.e. as separate parameters). If the text can contain “blanks”, arrangements should be made to allow the parameters to be read correctly using “sscanf” (see ** 14 **). In such cases, it is advisable to enclose the text in special characters (e.g. “#”) that may not be included in the actual text string.
- ** 24 **
  List of variables that represent the parameters.
- ** 25 **
  List of current parameters. The parameters were either read from the input buffer or have been “hard coded” at this point. The latter is always the case if not all of the parameters required for calling the function are to be set or can be set via the panel.

6 Creating Test Sequencer applications

The preceding sections have shown how you can create your own applications on the basis of the CATS platform and the existing Test Case Templates by integrating additional devices and developing further test cases. The flexibility of this arrangement means that it is now possible to run these new applications directly on the ANT-20/ANT-20E or DominoCom (since these devices are complete PCs). To do this, the following steps are required:

- Standard installation
  The first step prepares the ANT-20 for operating CATS. To do this, please follow the steps described in the Operating Manual under “WG-CATS (BN 3045/01)”. Use the same floppy disks that you used for the original installation on your development PC.

Notice: The CATS platform (BN 3045/01) and Test Case Packages (BN 3045/93.xx) must not be installed.

- Creating installation disks
  The next step is to generate a so-called “distribution kit” on your development PC. This consists of installation disks that contain all the files belonging to your application in compressed form. Normally, no source files are installed on the target machine. The executables, objects and UIR files are sufficient.

Notice: The DLL “wgwsdrvw.dll” present on your development PC must on no account be installed on the ANT-20. This is a version-dependent DLL that is already present on the ANT-20.

Notice: Make sure that the directory structure of the target machine (of the ANT-20) corresponds to the conventions described in Sec. 2, Page 1-3 of this manual. If this is not done, severe functional errors may result.

- Installing the application
  Installation on the target machine is accomplished by calling “A:\setup.exe”.

7 Integrating CATS into other applications

The integration of applications created under CATS into existing environments (e.g. Visual Basic, Excel, HP VEE, etc.) is accomplished most easily using the CATS remote control interface. This interface is available for the Test Sequencer for TCP/IP (BN 3035/95.91) and DDE (BN 3035/95.92) as a standard option.
Both interfaces are also available for WG-CATS (BN 3045/01) (based on a customized solution). Please contact your local Wavetek Wandel Goltermann Sales Organization in this regard.
Since the two solutions use the same concept, they are handled together here.

Although the details of integration differ depending on the application to be used as the controlling platform, the general procedure can be described. After starting the remote control interface, CATS sets up a server (TCP/IP or DDE) which waits for connection requests from clients. If a connection is established, information can be exchanged between the client and the server. In real terms, the client issues commands (see Operating Manual) that the server responds to with result strings. The string contains the results in CSV format.

The job of the integrator is therefore to implement a corresponding client on the side of the application to be controlled. Such servers are included with many Windows (Visual Basic, Excel, etc.) or UNIX (HP VEE, etc.) software tools, so that integration is possible with little effort.

The operation of the test case server is not described here. The information is found in the Operating Manual.

8 Annex: Service functions

The main objective in developing this library was the encapsulation of platform-specific conditions. This ensures hat test cases developed under CATS can be used in various operating modes (interactively via the user interface panel or by remote control via LAN or DDE).

Note: Although at first glance it may seem that alternatives for the following functions are available from the stock of “normal” LabWindows CVI functions, correct functioning of the test cases is only guaranteed if the functions listed here are used. For example, directly invoking the CVI “operator_message” function leads to a non-resolvable hang-up in remote controlled operation.

The use of the functions governing interaction between the test case and the platform is also seen from the code template.

8.1 tc_exec_error

“tc_exec_error” is invoked when a run-time error occurs during execution of the test. The description of the driver function detecting the error or the description of the device is transferred via the “comment” parameter.

The range from 5000 to 6000 is reserved for “user defined error codes”.

Note: When a run-time error occurs, the test function should be exited with the return value “ABORT”. Renewed invoking of “tc_exec_error” with new parameters leads to the old information being overwritten.

Declaration

```
int tc_exec_error (char *comment, char *err_msg, int err_code)
```

8.1.1 tc_get_err_msg

“tc_get_err_msg” returns information to the CATS platform regarding a possible run-time error indicated by “tc_exec_error” during execution of the test. The return value returns the error code. The “global_error_msg” parameter contains the error message.

Note: The function may only be invoked within the adapter function. The function always supplies information about the last error indicated by “tc_exec_error” (see “tc_exec_error”).

Declaration

```
int tc_get_err_msg (char **global_error_msg)
```

8.1.2 tc_get_report

“tc_get_report” returns the test report that is built up line by line during execution of the test to the CATS platform.

Note: The function may only be invoked within the adapter function.

Declaration

```
char *tc_get_report (void)
```

8.1.3 tc_get_results

“tc_get_results” returns the results saved in CSV format by “tc_results” during execution of the test to the CATS platform.

Note: The function may only be invoked within the adapter function.

Declaration

```
void *tc_get_results (void)
```

8.1.4 tc_operator_message

“tc_operator_message” interrupts the test sequence and outputs the message that was transferred as a parameter. The test resumes when the user has acknowledged the message by clicking “OK”.

Declaration

```
int tc_operator_message (char *message)
```

8.1.5 tc_operator_prompt

“tc_operator_prompt” is similar to “tc_operator_message”, the difference being that “tc_operator_prompt” poses a question that must be answered by the user with “yes” or “no”. The response to the question affects the test status (“yes” ==> “PASS”, “no” ==> “FAIL”).

Declaration

```
int tc_operator_prompt (char *query, int *response)
```

8.1.6 tc_param

“tc_param” is used to incorporate the input buffer into the result string that can be called up with “tc_get_results”.

Note: The function may only be invoked at a specific point within the adapter function (see “codefram.c”).

Declaration

```
void tc_param (char *pointer)
```

8.1.7 tc_print

The test report is built up line by line with the ais of “tc_print”. Each time it is invoked, one line of the report is generated.

Note: “New-Line \n” is automatically added to the end of a line by the platform. The text should only contain another new line command if a blank line is required.

Declaration

```
int tc_print (char *text)
```

8.1.8 tc_results

The test results saved in CSV format are incorporated into the result string with the aid of “tc_results”.

Declaration

```
int tc_results (char *results)
```

8.1.9 tc_return

“tc_return” is invoked using the test status (“PASS”, “FAIL” or “ABORT”) before exiting from the test function.

Note: “tc_return” is essential if the test case is to function correctly within CATS. Exiting from the test function without invoking this function leads (at least in the case of remote control of CATS via LAN or DDE) to unpredictable results. In the best case scenario, only the test report will be generated incorrectly.

Declaration

```
Status tc_return(Status test_status)
```

8.1.10 tc_show_message

“tc_show_message” displays a message for as long as the platform deems necessary. In contrast to “tc_operator_message”, the user is not required to acknowledge “tc_show_message”.

Declaration

```
int tc_show_message (char *message)
```

8.1.11 tc_show_message_some_sec

"tc_show_message_some_sec" displays a message for a specific length of time. In contrast to "tc_operator_message", the user is not required to acknowledge "tc_show_message_some_sec".

Declaration

```
void tc_show_message_some_sec (char *message, double sec)
```

8.1.12 tc_sleep

Invoking "tc_sleep" allows interruption of the test sequence for n milliseconds.

Note: The function is realized using a "busy wait", i.e. no task change occurs and the event queue is not processed and it is not possible to abort the test during the wait period.

Declaration

```
int tc_sleep (int delay_msec)
```

8.1.13 tc_start

Invoking "tc_start" indicates the start of a new test case to the platform. Although the name transferred in the CATS environment as a parameter is not used further in the current version, it is advisable to use a descriptive name (e.g. you could use the name of the C function).

Note: As "tc_start" initializes internal variables essential for the subsequent test algotirhm, it must always be the first action invoked in a test function.

Declaration

```
int tc_start (char *test_name)
```

8.1.14 tc_warning

"tc_warning" is used to handle situations requiring a warning to be given to the operator.

Note: The menu item "Warning Level" is used to determine whether invoking "tc_warning" leads to interruption of the test by an operator message, or to a note in the test report, or to no reaction at all.

Declaration

```
int tc_warning (char *warning)
```

9 Annex: List of device-specific files

The following lists indicate which files belong specifically to certain devices or drivers. This information is useful if you do not require certain devices for a given application and you do not want to clutter up the executable file with unnecessary files. Please note that the “conf.h” file must be adapted to suit (by commenting out the define statements for devices that are not needed). If this is not done, the linker will issue error messages.

- Generic Driver
  - .\drivers\wgvxipnp.obj
  - .\platform\ini_gene.obj

- ANT-20 Basic / ANX-920
  - .\drivers\wgantx20.obj
    .\drivers\wgant_2.obj
    .\platform\ini_antb.obj
    .\platform\sdh_ant.obj
    .\platform\sdh_antf.obj
    .\platform\set_ant.obj

- ANT-20 Jitter / ANX-930
  - .\drivers\wgantx30.obj
    .\platform\ini_antj.obj

- ANT-20 STM-16 Jitter / ANX-960
  - .\drivers\wgantx60.obj
    .\platform\ini_a960.obj

- ANX-950 / NRZ
  - .\drivers\wganx950.obj
  - .\platform\ini_nrz.obj

- SF-60
  - .\drivers\wgsf60.obj
    .\drivers\wgsf60_2.obj
    .\platform\ini_sf60.obj
    .\platform\sdh_sf60.obj
    .\platform\sdh_60_f.obj
    .\platform\set_sf60.obj

- OMS / OVX
  - .\drivers\wgomscvi.obj
  - .\platform\ini_oms.obj
  - .\platform\omslayer.obj

- Test Point Selector - Pickering Scanner
  - .\drivers\pil20.obj
  - .\platform\ini_pil.obj
  - .\platform\scanlay.obj

10 What are the components of the CATS project?

CATS with c-sourcefiles has the BN 3045_01. Components of this packet are: - CATS Test Executive Runtime Package - Testcase packages like transmission, optics, ... for ANT20, OMS, Testpoint Selector (Pickering Scanner) ... To work with these parts, and to create new testcases, it is necessary to have the Lab Windows CVI Developers Environment, Version 4.01 or higher. Additional to this, depending on what you want to do, you should have: - Card & Socket software
if PCMCIA cards are used - GPIB for Windows card software if a GPIB card is used - VXI Plug&Play software with the VISA DLL

Explanation:

CATS CVI Application for Test Sequencing

Testcases:

A testcase is a normal c-function. It consists of the test algorithm and the graphical user interface to set the parameters. It uses the standarsized calling interface of the NI Test Executive Engine. The Prototype is: The data and error structures will be described later.

Abstractionlayer:

The purpose of using the abstractionlayer was to ensure that testcases developed under CATS can be used without changes for different instruments with similar functionality.

Example:

setup 2 MBit TX Signal Structure (is device independent)

For further information see Chapter 8:

Auxillary testcases Toolkit testcases (are device independent)

- _wait waits a selected time
- _user_io the operator is asked to make a YES/NO decision or whether he just has to confirm
- _scpi_io sends SCPI commands to an instrument and receives the response
- _stopwatch prints the time since program started
- _acoustical_signal makes an acoustical signal on a PC - Loudspeaker
- _wait_until_time wait until time is reached
- _show_info_field shows info fields
- _show_image shows an image for a selected duration
- _time_stamp prints a time/date - stamp in tc_print function

11 Directory Structure

main folder: \CATSxx\
(xx = version number)

***.exe CATS Executable:**

- codefram.c Example template for a testcase
- readme.txt Describes new features in CATS 3.5
- testlist.txt List of available testcases, this file is displayed if you press the Select TC ... button in the Sequence Editor
- ini_bsp.c Example of an ini-File to integrate new devices

\CATSxx\al_sens\

Alarmsensor testcases:

- al_sens.obj Object file for Alarm Sensor testcases
- al_dyna.obj Object file for Generate Alarm Stimulus testcases
- *.uir various graphical user interfaces to set parameters

\CATSxx\atm\

ATM testcases:

- atm.obj testcases for ATM (Telia)
- atm_bas.obj Basic testcases for ATM
- atmtrans.obj ATM Transparency Check
- celdelay.obj ATM Cell Delay Varition testcase
- *.uir Various Graphical User Intefaces to set parameters

\CATSxx\bitmaps\

Bitmaps:

- ant_loop.bmp Bitmap shows the ANT20 in loop measuring
- ant_uut-bmp Bitmap shows the cabling of the measuring configuration:

\CATSxx\config\

Device Configuration:

- conf.c Contains a function which returns a pointer to the DEVICE_CLASS table, see devices.h
- config.c Does the configuration (open and initializes all selected devices which are used for a testcase)

\CATSxx\drivers\

Instrument drivers:

- pil20_wg.c Driver for the Pickering Scanner
- wgant_2.c Higher functions, which uses the basic driver functions of the ANT20 wgantx20.c Driver for the ANT20, ANX920, ANX940 BasicModule
- wgantx30.c Driver for ANT20/ANX930 Jitter Generator/Analyzer Module wgantx60.c Driver for ANT20/ANX960 Jitter Generator/Analyzer STM16 Module
- wganx950.c Driver for the ANX950 NRZ Module
- wgomscvi.c Driver for OMS
- wgsf60.c Driver for the SF60
- wgsf60_2.c functions, which uses the basic driver functions of the SF60
- wgvxipnp.c Generic driver, basics for developing of new drivers
- *.fp Function panels for the device drivers
- *.obj Object files of the drivers

\CATSxx\include\

Include files

devices.h Definition of the structures DEVICE, DEVICE_CLASS and MAKROs

vi_dev.h Definition of the instrument IDs

conf.h Declaration of instrument specific driver functions

*.h other header files of drivers and needed components

\CATSxx\platform\

Platform:

- tc.c Utility functions to make testcases platform independent
- ini*.c Files with functions to integrate drivers into the CATS project
- *.obj Object files of these c- files

\CATSxx\report\

Report

- *.rpt Contains test reports of the executed test sequence
- *.csv File with comma separated values (CSV)

\CATSxx\sdh_bas\

SDH Basic testcases

- actalarm.obj Testcase, which reads actual alarms
- aps.obj Testcase APS delay.obj Makes the DELAY_TEST of the ANT20
- g826.obj Testcase for G826
- jitter.obj Testcases for jitter applications
- monitor.obj Monitoring testcase
- mscantri.obj Scan testcase für selected tributaries
- nrz.obj Testcases to control the NRZ module ANX-950
- options.obj Testcase, which checks the available options
- pf140.obj PDH alarm monitoring testcase
- ptr_mov.obj Testcase for pointer (generating/monitoring) actions
- sa_bits.obj Set SA4-SA8 bit and check one testcases
- scan_mon.obj PDH/SDH FAS error monitoring testcases
- sdh_bas.obj Basic SDH testcases
- sig_stru.obj Testcase, to set signalstructure for TX and RX
- soh_poh.obj Testcase reads the actual SOH POH arrays
- sonetbas.obj Basic SONET testcases
- tohpohso.obj Testcase reads the actual TOH POH arrays
- trouble.obj PDH/SDH trouble monitoring testcase
- wander.obj Testcase for _check_wander_tolerance
- *.uir SDH basic testcase Graphical User Intefaces

\CATSxx\sequence\

CATS Sequences

*.squ Sequences arranged by WWG

\CATSxx\toolkit\

Toolkit:

- toolkit.c Device independent Testcases, like I/O, Wait, Message, ...
- *.uir Graphical User Interfaces for the toolkit testcases
- *.h Header files

\CATSxx\txengine\

CATS Test Executive Engine:

- cvitx.obj Test Executive Main module
- txedpc.obj Contains functions to edit preconditions
- txedseq.obj Contains functions to edit sequences
- txengine.obj Main Code for the Test Executive Engine
- txlogin.obj Contains functions dealing with logging into the Test Executive
- txprecnd.obj Contains functions dealing with the preconditions
- txreport.obj Contains the functions concerned with writing test reports
- txsavres.obj Contains the functions concerned with saving test results
- cvitxuir.uir Contains the Graphical User Interfaces of CATS

12 CATS Global Header Files, Structures, Tables and Makros

Header Files:

vi_dev.h: contains Id defines for the integrated instruments like:

- #define VI_ANT20 1
- #define VI_ANX930 2
- #define VI_ANX960 3
- #define VI_GENERIC 20
- #define VI_OMS 40 ...

This Ids are used in conf.h, in the DEVICE_CLASS structure at parameter position 2, where the variable class_id is filled with these (i.e. VI_ANT20) values.

devices.h: This header file contains the definition of the DEVICE and the DEVICE_CLASS structures, describing the devices and their configuration. It also contains the makros for SEND_COMMAND, QUERY and DEVICE_SIMULATION. And there are the definitions for some functions like device_open, device_close, device_query ... , which are used in devices.c. conf.h This header file contains the information required to run the user interface “config.uir”, and to open the devices. There are definitions for the instrument specific open, close, init ... functions.

Structures

DEVICE_CLASS: defined in devices.h

```
typedef struct
                    {
                    char class_name[20];
                    Class Name filled in conf.h, i.e. "Generic", or
                    "ANT20 Basic" ...

                    int class_id;
                    Class Id filled in conf.h, i.e. VI_GENERIC,
                    VI_ANT20, ...

                    int VImask;
                    VImask filled in conf.h, i.e. 0, 1, 3, ...Only
                    useful for ANT (VImask=3), otherwise Vimask=0

                    int (*open) (DEVICE *);
                    Function pointer to open function

                    int (*send_command) (DEVICE *, char *);
                    to send command

                    int (*query) (DEVICE *, char *, char *);
                    FP to query function

                    int (*init) (DEVICE *, int, int);
                    FP to init function

                    } DEVICE_CLASS;
```

DEVICE:defined in devices.h

```
typedef struct
                    {
                    int aux[3]; //Auxiliary variables

                    char name[40]; //Description of device purpose
                    (TX,..)

                    int monitor; //Monitoring ON / OFF

                    int dev_type; //Device class id, VI_SF60, ...

                    int interface; //Interface (GPIB, GPIB-VXI, ...)

                    int addr; //GPIB address int sec_addr; Secondary
                    address

                    unsigned long handle; //Device handle returned by
                    "OpenDev"

                    unsigned long aux_handle; //Additional handle
                    allowing to deal with device combinations (ANX-
                    920 NRZ)

                    int simulation; //Flag indicating simulation mode

                    } DEVICE
```

Tables

static DEVICE_CLASS device_tab[ ];

This device_tab table is filled in conf.h with entrys for different devices. Depending on which defines are activated, additional to the standard devices (Generic, ANT20 Basic, Jitter .. 622Mbit/s and Jitter 2488Mbit/s) others (SF-60, OLA, PIL-20 ...) were added. The defines were set in the project window in Options>>

Compiler Defines .

extern DEVICE device[ ]; This device table is declared in devices.h.

In the function init_devices (in config.c) it is initialized for 21 entrys. After that the values (for interface, device, address ...) which are set in the config.uir window will be written into the table.

Makros

#define SEND_COMMAND(i,A) device_send_command(&(device[i]),A)

When SEND_COMMAND( ) is called (i.e. in toolkit.c, testcase scpi_io), the macro pictures the function to the device_send_command( ) function (in devices.c), which finds out the corresponding function for the specified instrument, and then calls this function.

Example: toolkit.c testcase SCPI_IO the following function call exists:

SEND_COMMAND (id, line_cmd[i]); i.e. SEND_COMMAND (2, “*RST;”]); where 2 is the device id, selected in the config.uir window, and “*RST;” is the command to send. Now the function device_send_command is called the following way: device_send_command (DEVICE *dp, char *command); i.e. device_send_command (DEVICE *dp, “*RST;”);

Because of the definition of the macro, ...&(device[i]),... the entrys for device number i=2 (i.e. ANT20) are searched in the DEVICE device[ ] table, and are returned to the device_send_command( ) function. Now the function gets the device_tab[ ] table, and the class_id of the device_tab[ ] table is compared with the dev_type of the returned device[ ] list entrys until they are equal. Then the corresponding send function is called (i.e. WGANT_send_command)

#define QUERY(i,A,B) device_query(&(device[i]),A,B)

With the macro QUERY it is the same, exept that it is pictured to the function device_query( )

#define DEVICE_SIMULATION(i) device[i].simulation

This macro handles only the simulation variable. When DEVICE_SIMULATION ( ) is called, the value of the device[i].simulation variable is returned (i.e. in jitlayer.c) These macros are part of the abstractionlayer

13 Test Executive Engine Structures (from National Instruments)

tTestData: defined in txengine.h

```
typedef
struct ClassData_Rec
                      {
                      Status result;
                      Wether the test is passed or not double
                      measurement; Measurement taken by test function

                      char *inBuffer;
                      For passing parameters into test

                      char *outBuffer;
                      For output messages from the test

                      char *modPath;
                      Path of module containing test

                      char *modFile;
                      Base file of module containing test

                      void *hook;
                      User defined expansion hook

                      }tTestData;
```

tTestError: defined in txengine.h

```
typedef
struct ClassError_Rec
                      {
                      Boolean errorFlag;
                      Wether error occurred in test function or not

                      tErrLoc errorLocation;
                      Where error occurred (pretest, test ...)

                      int errorCode;
                      User defined error code

                      char *errorMessage;
                      User defined error message

                      }tTestError;
```

14 Device initializing

In CATS Sequences the “Setup Sequence Function” is set to “CATS Configure”, and the “Cleanup Sequence Function” is set to “CATS Close”. See the sequence attributes in the sequence editor.

Description of what happens during the loading of a sequence:

The function CATS_configure (in config.c) gets the global device_tab[ ] list (yet filled in conf.h) via the function get_device_tab (in conf.c). In this device_tab[ ] the instruments listed in the conf.h file are yet registered. After this the global table device[ ] (defined in devices.c as DEVICE device[21]) is initialized. Then the parameter string is scanned and the device[ ] table is filled with this values (maybe default values). Then the Device Configuration Window is opened. Now the settings for the devices, addresses, interface types, monitor mode ... can be made. Those values are scanned, and in the function store_configuration these values are written into the device[ ] list.

Description of what happens during open and initialize of a device:

Directly after the device[ ] list was filled, the function open_and_initialize (in config.c) is called. Here the class_id of the device_tab[ ] list is compared (in a while ... loop) with the dev_type of the device[ ] list. This is done until the device_tab.class_id is equal to device.dev_type. Now the function device_open (in devices.c) is called and a pointer to the device[ ] list is returned. Now again an entry belonging to the device (in the device list) is searched in the device_tab[ ] list. Via the found entry the abstract open function is called (i.e. if device=ANT20 à WGANT_open_device). This function can be found in the inixxxxx.c file belonging to the device (i.e. ini_antb.c). And this function calls the driver function wgxxxxx_init (i.e. wgant20_init).

Description of what happens when other functions like query were called:

If a testcase communicates with a device and i.e. has to make a query, so always a function of the abstraction layer (i.e. QUERY) is called. The driver functions (i.e. wgantx20_get_response_string) were never called directly. As there exists the device_open (in devices.c) function, there are some more abstract functions. These are the device_query, device_init, device_close and the device_send_command functions. Like described above at the device_open function, these functions also search the corresponding entry (in the device_tab[ ] list) and then call the abstract init, close, query, send_command function (i.e. WGANT_query). And from those abstract functions the driver function is called.

15 Frequently asked questions

The DWDM test cases are described in this section.

15.1 What is the core driver, construction, usefulness, purpose?

The core driver, or generic driver is the frame of a device driver. It can be used as basic entry point for developing new device drivers. The basic functions for initializing and for communication with devices are implemented. The driver even contains the controlling of various interfaces and a simulation mode. Additionally an error query function, monitoring of the device communication and output of messages into files are implemented.

To control a new device only few modifications are necessary. This means a reduce of work for later implementations of new devices. How this works is described later. Description of the driver:

The generic driver is written to access the following interface types:

- GPIB
- GPIB-VXI (i.e. VXI rack controlled via GPIB)
- VXI/MXI
- RS-232

It allows multiple instances of the same instrument on the same interface type (e.g. GPIB) or on several interface types (e.g. GPIB, RS-232). To be in accordance with the VXIplug&play specifications, only the NI VISA Library is allowed. For interface types which are not supported by VISA, the specific interface libraries must be used. (Not in accordance with the VXIplug&play specifications!). Required interface types with accompanying libraries can be selected by definitions for conditional compilation in the include file wgvxipnp.h. For more instructions on how to use this template instrument driver file, see the LabWindows/CVI Instrument Library Developer’s Guide. To alter this program for use with a specific instrument, perform the following steps:

1. Perform a global change to change all occurrences of the word “wgvxipnp” to the prefix “wgxyz123”, where xyz123 is the name of your instrument. Save the file as “wgxyz123.c.
2. Perform a global change to change all occurrences of the word “wgvxipnp” to the prefix “wgxyz123” in the include file “wgvxipnp.h”. Save the new include file as “wgxyz123.h”.
3. Instrument-dependent commands are marked with the comment CHANGE. Search for occurrences of this comment and make appropriate changes (in both files wgxyz123.h and wgxyz123.c).
4. Save the file wgvxipnp.fp as “wgxyz123.fp”, where xyz123 is the name of your instrument. Edit the Instrument Node: Change the Name to be the name of your instrument and the Prefix to “wgxyz123”.
5. Insert your instrument-dependent functions. Be sure to declare all of the user callable functions in the include file “wgxyz123.h”
6. Search all occurrences of text marked with comment brackets in the file wgvxipnp.doc and replace it in the help information with instrument-dependent text.

If you are interested to control your device via only one specific interface (i.e. RS232), delete all entrys which are needed for other interfaces (i.e. GPIB, GPIB-VXI, VXI/MXI).

The basic functions are:

- find_instruments searches with the Visa function viFindRsrc all available instruments and writes them into a list
- init initializes the specified device, using the viOpen function
- configure_rs232_port configures the RS232 port to the specified values. Must be called before initializing the serial device
- reset resets the device to default values
- close closes the instrument, and resets some variables - set_timeout this function sets the timeout value for I/O accesses to the instrument
- set_test_mode this function sets the test parameters for simulation, error check, monitor mode ...
- get_test_mode this function gets the test parameters
- q_idn this function queries the instrument by the command *IDN? (Identification Query)
- q_opt this function queries the instrument by the command *OPT? (Option Query)
- self_test this function queries the instrument by the command *TST? (Selftest Query)
- send_command_string this function sends the command string to the instrument.
- get_response_string this function sends the query message to the instrument and receives the response message
- receive_string This function receives a message and other, not important functions

15.2 What is the fastest way to create a new testcase?

even if there is no CVI driver for this device

If you want to write a new testcase for a new device, and you dont have a driver for this device do the following things:

First you have to write the testcase functions, for example setting the instrument, make a measurement and then reading out the results. See the template file codefram.c.

Use the functions implemented in the generic driver, because the generic driver can send messages, receive messages to almost all devices which understand SCPI commands, and use the GPIB, VXI or RS232 interface.

This solution works proper, if you use basic generic functions. For extended testcases with a lot of read message/write message function calls with internal dependencies you have to write an own driver and integrate it in the project.

15.3 What has to be done to write a new driver for my device?

First you have to make sure that it is necessary to write a new driver. If there are only few tests to do, or if only the communication with the device has to be tested it is not necessary to develope a new instrument driver. For such smaller tasks use the generic driver. With this driver you can open the device (if it is controlled via GPIB, VXI or RS 232 interface), send SCPI commands, receive results and make error queries. This should be enough for basic tests.

If this is not enough, i.e. if device specific functions are needed, you have to write a new driver, especially for this device.

The developing of this driver can base on the frame of the generic driver.

To describe which files of CATS belong to a device, as an example the ANT20 Basic Module is selected.

File	Description
wgantx20.c	Driver source file
wgantx20.fp	Function panel for the driver
wgantx20.h	Header file for the driver
ini_antb.c	Contains functions to integrate the ANT20 Basic Module into the CATS environment, i.e. WGANT_open_device, WGANT_query ...
set_def.h	Contains abstract define values for signal, mapping, ...
sdh.h	Contains the definition of the SDH/PDH structure set.h Contains the definition of the SET structure, which contains functionpointers to set the signal structure
sdh_ant.c	Contains measuring functions, like setting the pattern, the measuring time, start and stop measuring, insert errors ...
set_ant.c	Contains functions to set the TX/RX signal structure, the interface (koax, optics, symm ...), the code (HDB3, AMI, NRZ ...) ...
sdh_antf.c	Contains functions which calls the wgantx20_fetch_... functions to get SDH/PDH measuring results
sdh.c	Contains abstract RESULT functions
wgantss.c	For mapping the abstract definitions on the ANT20 defines Additional to these files some of the testcases in the folders \atm\ , \al_sens\ and \sdh_bas\ belong to the device.

15.4 What is the abstractionlayer (construction, usefulness, purpose)?

In general CATS has two abstractionlayers. These are on the one hand the tc-functions, the layer between the Test Executive Engine and the testcases, and on the other hand the abstract functioncalls in the file devices.c

Instead of calling LabWindows specific functions, the tc- functions are called. These tc-functions behave like the LabWindows functions, but they make the Testcases platform independent. So whenever you want to popup a message, wait some time, ... you should use the tc-functions. For example, instead of Delay(sec) use the tc_sleep(msec) function.

Instead of directly calling any driver functions you should use the abstract functions defined in devices.c (i.e. device_open, device_query, ...).

Example: You can run a SDH Check Transparency Test with an ANT20, SF60, PRA, ... how works the it?

There is the testcase check_transparency (from sdh_bas.c).

You can run this testcase either on the ANT20, or on a SF60. You have to select the device in the config.uir window. Then you have to set the parameters of the test. if device = ANT20 common if device = SF60 check_transparency RESULTS (...) SDH_WGANT20_RESULTS (...) SDH_WGSF60_RESULTS (...) wgantx20_fetch_all_configure_delete wgsf60_conf_counters wgantx20_fetch:all_configure RX_GATING_TIME (...) WGANT_RX_GATING_TIME (...) WGSF60_RX_GATING_TIME (...) wgantx20_set_measure_time_interm wgsf60_conf_gating_time wgantx20_set_measure_time RX_START (...) This call of the corresponding function is based on the entrys in the device table and the device selected in the parameters of the testcase.

15.5 How integrate a self written testcase into the project?

To create your own testcase use the template file codefram.c. To make sure that your self written testcase works proper.

pay attention to the following things:

- if an abstract layer exists use only abstract driver functioncalls and no direct calls of driver functions
- use the tc-functions and not Lab Windows functioncalls - make the prototype of your testcase function like void _testcase_name (tTestData *data, tTestError *error)
- create a Graphical User Interface to set the parameters of your testcase After you have created your testcase you should save it in your own folder .\my_testcase_folder\ as my_testcase.c.
- create an object file from your c-source file which has the name my_testcase.obj
- edit the file testlist.txt the following way: -Insert a separation line, after the last entry in the file, like:

#---

Add the testcase name, function name and the path to your testcase like

This is my testcase ;_my_testcase_function ;0;. \my_testcase_folder\my_testcase.obj

The number before the pathname indicates whether the status of the test is “relevant”(1) or if it is “dont care”(0). If the number is 1 the testcase will be shown in red colour, otherwise in blue colour.

Now run CATS, choose Sequence, Edit Sequence and then press the Select TC... button. Now your self created testcase will appear at the bottom of the Testcase List window. There you can select it by double click, and insert it into an existing or a new sequence.

15.6 How integrate a new device into the project?

The instrument in this example is an ABC123

First you have to write an instrument driver with header file for your device. The functions should have the prefix abc123 . Save the source file in the folder \drivers\, and the header file in \include\ . If no abstract layer exists call driver functions, if no driver exists call generic driver functions. You can use the generic driver wgvxipnp.c as a frame to create a driver. Make the changes like described in the chapter core driver.

Then write your testcases, and if an abstract layer exists call abstract high level functions i.e. ABC123_SET_MEASURING_TIME (for setting, measuring ...). You can use the template file codfram.c in for creating your testcases. Save your testcases in an extra folder. See the toolkit testcases as examples.

Make a define entry in the file vi_dev.h

the following way: #define VI_ABC123 520 where the number 520 (or another number higher than 500) is your instrument ID. Use this ID in all other functions created for this device!

Make some entrys in the conf.h header file the following way:

#ifdef CFG_ABC123
extern int ABC123_open_device (DEVICE *) ... Do this only for the open, close, send, query and init functions ... extern int ABC123_init (DEVICE *, int, int)
#endif

Then make additional entrys at the end of this header file, where the DEVICE_CLASS table is filled:

#ifdef CFG_ABC123 “ABC 123”, VI_ABC123, 0,
ABC123_open_device, ...
... ABC123_init, #endif
See the description of the DEVICE_CLASS structure.
Create an ini_ABC123.c file which contains high level basic functions, this means the functions defined in conf.h. These functions are i.e. ABC123_open_device, ..., ABC123_init. These functions call the driver functions abc123_init, ...

As an example see the files ini_antb.c or ini_gene.c

For measuring, setting ... functions create another file with the name meas_ABC123.c, where meas stands for the kind of measuring you want to do (i.e. sdh_ant.c makes SDH measurings with the ANT20). This file should contain high level measuring, setting, controlling ... functions which call low level driver functions like send_command, set_measure_time ... These high level functions should have the type i.e. ABC123_SET_MEASURING_TIME Create a header file of the functions you implemented.

Before you compile your modified project you have to set a compiler define in Options>>Compiler Defines . Add the entry \DCFG_ABC123 .

Notice: Follow exactly what is described here. Dont modify or delete any other files or entries! This could affect the proper work of CATS.

15.7 How cut an integrated device out of the CATS project?

The instrument in this example is the ANT20 Basic module Make modifications to the following files:

vi_def.h delete the define for the ANT20

conf.h delete all the function definitions of the ANT20, i.e. extern int WGANT_open_device (DEVICE *); delete the entry in the DEVICE_CLASS device_tab[ ], i.e. “ANT20 Basic”, VI_ANT20, 3, ...

config.c delete the wgantx20_... (...) function calls in the function special_open_ANT_V24 (...)

Remove the following files from the project: (extensions depend on which files you have added to the project, either the *.c files or the *.obj files)

wgantx20.c or wgantx20.obj wgantx_2.c or wgantx_2.obj ini_antb.c or ini_antb.obj ini_cnt.c or ini_cnt.obj sdh_bas.c or sdh_bas.obj sdh_ant.c or sdh_ant.obj sdh_antf.c or sdh_antf.obj sdh.c or sdh.obj set_ant.c or set_ant.obj set.c or set.obj sig_stru.c or sig_stru.obj atm.c or atm.obj delay.c or delay.obj options.c or options.obj

In general you have to remove the files which use functions belonging to the ANT20 Basic module. Do exactly what is described here. Dont modify or delete any other files! This could affect the proper work of CATS.

Notes: