


Agilent 81250 Parallel Bit Error Ratio Tester

## System Setup Examples



**Agilent Technologies**



## Important Notice

This document contains propriety information that is protected by copyright. All rights are reserved. Neither the documentation nor software may be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of Agilent Technologies.

© Copyright 2002 by:  
Agilent Technologies  
Herrenberger Straße 130  
D-71034 Boblingen  
Germany

The information in this manual is subject to change without notice. Agilent Technologies makes no warranty of any kind with regard to this manual, including but not limited to the implied warranties of merchantability and fitness for a particular purpose.

Agilent Technologies shall not be liable for errors contained herein or direct, indirect, special, incidental, or consequential damages in connection with the furnishing, performance, or use of this manual.

Brand or product names are trademarks or registered trademarks of their respective companies or organizations.

Authors: 13 medien GmbH

# Contents

About this Manual	7
<hr/>	
BER Test on a Single System Using PRBS Data	9
<hr/>	
Focus of this Example	10
Hardware Setup	11
Test Setup	12
Specifying the Connections	13
Setting the Clock Module Characteristics	14
Setting Voltage Levels and Termination	17
Checking the Measurement Mode	20
Specifying the Test Sequence	20
Test Execution Using Auto Delay Alignment	22
Specifying Auto Delay Alignment	23
Running the Test	24
Test Execution Using Auto Bit Synchronization	26
Specifying Auto Bit Synchronization	27
Running the Test	29
<hr/>	
BER Test on a Single System Using Memory Data	31
<hr/>	
Focus of this Example	32
Hardware Setup	33
Test Setup	34
Specifying the Connections	35
Setting the System Clock Frequency	36
Setting Voltage Levels and Termination	38
Checking the Measurement Mode	40
Creating the Data Sequence and Segment	41
Editing a Memory Data Segment	43
Test Execution With Manual Analyzer Delay Adjustment	45
Specifying a Common Analyzer Start Delay	46
Running the Test	47

Test Execution With Automatic Analyzer	
Delay Adjustment	50
Specifying Auto Delay Alignment	50
Running the Test	52
<b>Capturing and Analyzing Data on a Single System</b>	<b>55</b>
<hr/>	
Focus of this Example	56
Hardware Setup	57
Test Setup	58
Specifying the Connections	58
Checking the System Clock Frequency	60
Setting Timing, Voltage Levels and Termination	62
Comparing and Acquiring Data Around Error	65
Setting the Measurement Mode	65
Checking the Test Sequence	66
Running the Test	68
Inspecting the Results	69
Comparing and Capturing Incoming Data	71
Setting the Measurement Mode	71
Changing the Test Sequence	72
Running the Test	74
Inspecting the Results	74
Capturing Incoming Data	81
Setting the Measurement Mode	81
Changing the Test Sequence	82
Running the Capture Operation	83
Inspecting and Saving the Captured Data	84

Using Events on a Single ParBERT System	87
Focus of this Example	88
Hardware Setup	89
Test Setup	90
Creating the Test Sequence	92
Defining Events	96
Specifying Reactions Upon Events	98
Executing the Test	100
Return to Standard Mode Sequence Editor?	102





# About this Manual

This is a collection of examples on how to use the Agilent 81250 Parallel Bit Error Ratio Tester for various applications.

## Intended Audience

Newcomers to the ParBERT are encouraged to execute these examples on their own systems. This is probably the fastest way to learn how to operate the system.

Experienced users of the ParBERT may find one or more examples that assist them in setting up a particular test or solving a current problem.

## About the Examples

The examples are sorted according to their complexity. The first examples are fairly simple. However, they provide the basic knowledge and experience which is presumed in the later examples.

Every example starts with a short description. So, you can decide in forehand whether you feel the example important or wish to skip it.

Every example contains not only instructions and screenshots of the results, but also explanations of requirements, parameters, and limitations. While proceeding from one example to the next, you will not only learn how to use the system's features, but also get an understanding of the underlying principles.

The explanations, however, are not going into all the details. The source for detailed information is the *Agilent 81250 ParBERT System User Guide*.

## Prerequisites

Many of these examples can be reproduced without a real DUT (device under test). A few shielded cables with SMA connectors (such as Agilent 15443A) will suffice for connecting generators with analyzers.

**Cable and propagation delay compensation** If a real DUT is used, at least a *cable delay compensation* must be performed. Recommended is the *cable and propagation delay compensation* procedure that compensates for both signal delays in the cables and propagation delays on the DUT board.

**Zero adjust** If the system configuration has been changed by replacing or adding modules or frontends, the *zero adjust* procedure has to be performed to synchronize the new generators or analyzers with the ones already installed.

Delay compensation is done with the Deskew Editor.

**Reader assumptions** Readers should have a basic understanding of the system's purpose and components. Such information is given in the chapter "*Introduction to the System*" of the *Agilent 81250 ParBERT System User Guide*.

**Multi-Media Guided Tour, Tutorial and Getting Started** As an additional source of information, the Multi-Media Guided Tour, Tutorial and Getting Started provide a comprehensive overview of the Agilent 81250 Parallel Bit Error Ratio Tester.

If it has been installed on your system, you will find it in the Windows start menu under *Programs – Agilent 81250 Tutorial*.

If not, you can download it from the web through

- <http://www.agilent.com/find/81250demo>





# BER Test on a Single System Using PRBS Data

This example demonstrates how to set up a bit error rate (BER) test on a single system using pseudo random bit stream (PRBS) data.

See:

- *“Focus of this Example” on page 10*
- *“Hardware Setup” on page 11*
- *“Test Setup” on page 12*
- *“Test Execution Using Auto Delay Alignment” on page 22*
- *“Test Execution Using Auto Bit Synchronization” on page 26*

# Focus of this Example

This example concentrates on the minimum setup requirements:

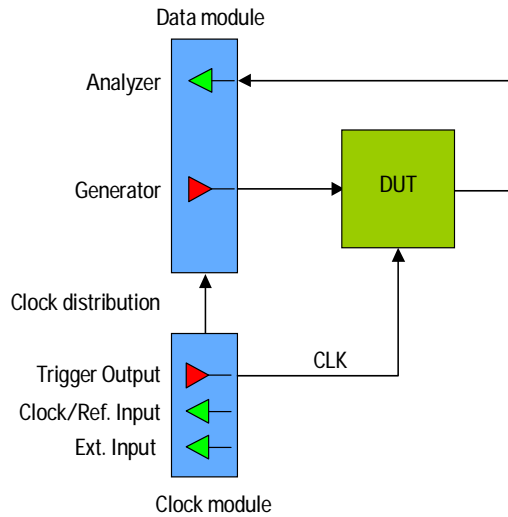
- We use one ParBERT system that generates stimulating data and analyzes the response of the device under test (DUT).
- The DUT has one input and one output terminal.
- Pure, undistorted PRBS data is generated and expected.
- The sampling delay of the analyzer is automatically set.
- The bit error rate (BER) is measured.

**What you will learn** You will learn:

- How to connect the DUT with the frontends
- How to set the system frequency
- How to set the signal levels
- How to create the sequence of generated and expected data
- How to distinguish between Automatic Delay Alignment and Automatic Bit Synchronization
- How to run the test

# Hardware Setup

The hardware setup is illustrated in the figure below:



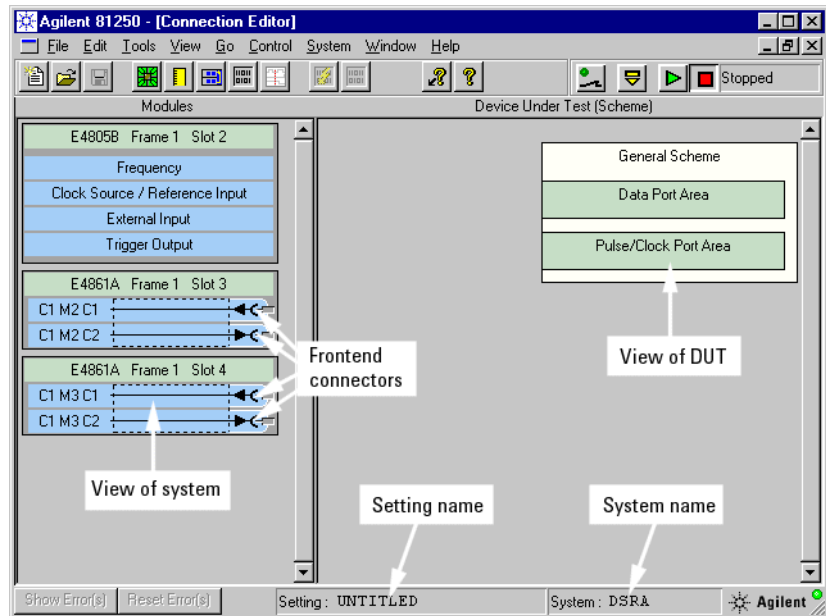
The TRIGGER OUTPUT of the clock module is used to provide the system clock to the DUT. A generator could be used as well.

To reproduce this example without a DUT, you can use a shielded SMA cable and connect the analyzer with the generator.

Any ParBERT system that has a generator frontend and an analyzer frontend can be used.

# Test Setup

The first window that appears automatically after starting the Agilent 81250 User Software is the Connection Editor.



**View of system** The left-hand side identifies the modules plugged into the VXI frame. The module on the top is the master clock module. The view of system identifies also the connectors of the modules. The connectors are provided by data generator or analyzer frontends plugged into the modules.

**View of DUT** The view of DUT shows two types of ports:  
Data ports are used for sourcing test data to and capturing data from the DUT. They are divided into DUT input ports and DUT output ports.  
Pulse/Clock ports are used for sourcing clock pulses to the DUT, if such pulses are generated by generator frontends.

**System name** The system name becomes important if the tester comprises more than one system. It informs you about the system that is currently operated.

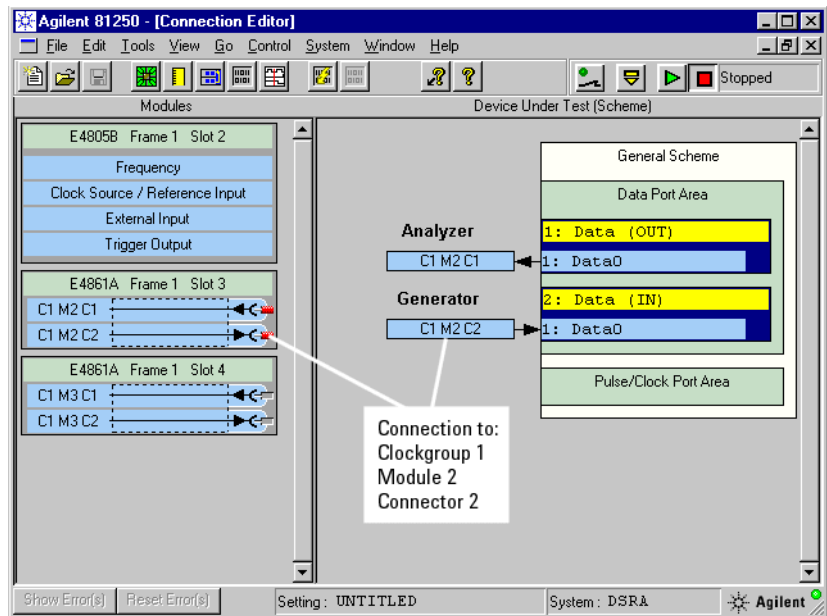
**Setting name** A setting comprises all the setup parameters used for testing the device. It can be stored and reloaded at any time.

## Specifying the Connections

Proceed as follows:

- 1 Click the *Data Port Area* with the right mouse button.
- 2 Insert an output port and an input port with one terminal each.
- 3 Click the terminals with the left mouse button and drag them to the appropriate frontend connectors.

This establishes the connections.



It is possible to assign individual names to the ports. Terminals are automatically named after the port but can also be renamed.

As we will use the TRIGGER OUTPUT of the clock module to provide the system clock to the DUT, no pulse port is needed.

**TIP** You can also create a data port with terminals and connect the terminals in one go. This is done by clicking the connector and dragging it over the *Data Port Area*.

The same way, a terminal can be added to a port and connected in one go. This is done by clicking the connector to be used and dragging it over the data port to the desired position.

The method will be demonstrated in a later example.

## Setting the Clock Module Characteristics

We will use a system frequency of 500 MHz in this example. You will get similar results for any system frequency between 334 and 675 MHz.

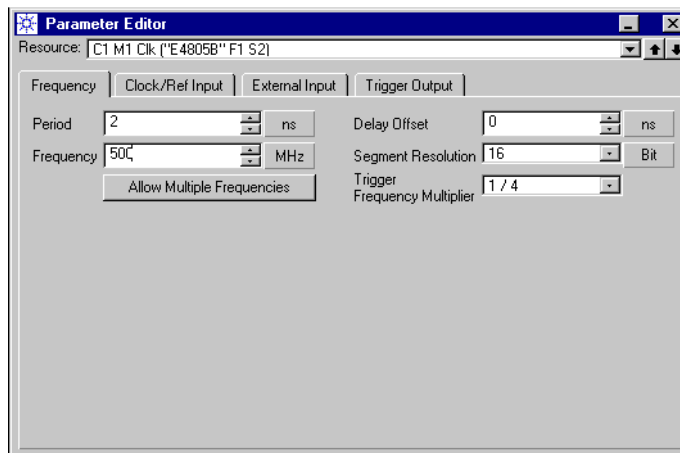
- 1 Double-click the *Frequency* box of the clock module.

This opens the Frequency page of the Parameter Editor for the clock module.

The Parameter Editor has special windows for setting up the clock module, data ports, and single frontend connectors. Each window has several pages.

- 2 Set the system *Frequency* to 500 MHz and press Enter.

Pressing Enter causes the Parameter Editor to check the input and to accept or reject it.



The system *Period* is always the reciprocal of the *Frequency*. In this example, it is 2 ns.

### Delay Offset

*Delay Offset* can be used for specifying an initial delay. Then, individual ports or channels can start earlier than the rest.

### Segment Resolution

Note that the *Segment Resolution* is set to 16. The Segment Resolution has to be set according to the chosen frequency and the data generator/analyzer module. It defines the word length used for

addressing the data memory built into the modules. The data memory is used to store the patterns of generated or expected data. For E4832A modules, 16 is the maximum. For E4861A modules, this is the minimum. The greatest flexibility is provided by E4861B modules.

The capabilities are shown in the tables below:

**Table 1 Clock Rates, Segment Resolution, and Memory Depth for E4832A Modules**

System Clock Frequency Mbit/s	Segment Resolution bits	Memory Depth bits	Frequency Multiplier Range
≤ 42.1875	1	131,008	1, 2, 4, 8, 16
≤ 84.375	2	262,016	1/2, 1, 2, 4, 8
≤ 168.750	4	524,032	1/4, 1/2, 1, 2, 4
≤ 337.500	8	1,048,064	1/8, 1/4, 1/2, 1, 2
≤ 675.000	16	2,097,152	1/16, 1/8, 1/4, 1/2, 1

**Table 2 Clock Rates, Segment Resolution, and Memory Depth for E4861A Modules**

System Clock Frequency Mbit/s	Segment Resolution bits	Memory Depth bits	Frequency Multiplier Range
333.334 to 675.000	16	2,097,152	1, 2, 4
≤ 1,350.000	32	4,194,304	1/2, 1, 2
≤ 2,700.000	64	8,388,608	1/4, 1/2, 1

**Table 3 Clock Rates, Segment Resolution, and Memory Depth for E4861B Modules**

System Clock Frequency Mbit/s	Segment Resolution bits	Memory Depth bits	Frequency Multiplier Range
20.834 to 41.666	1	131,072	1, 2, 4, 8, 16, 32, 64, 128
≤ 82.333	2	262,144	1/2, 1, 2, 4, 8, 16, 32, 64
≤ 166.666	4	524,288	1/4, 1/2, 1, 2, 4, 8, 16, 32
≤ 333.333	8	1,048,576	1/8, 1/4, 1/2, 1, 2, 4, 8, 16
≤ 666.666	16	2,097,152	1/16, 1/8, 1/4, 1/2, 1, 2, 4, 8
≤ 1,333.333	32	4,194,304	1/32, 1/16, 1/8, 1/4, 1/2, 1, 2, 4

Table 3 Clock Rates, Segment Resolution, and Memory Depth for E4861B Modules

System Clock Frequency Mbit/s	Segment Resolution bits	Memory Depth bits	Frequency Multiplier Range
≤ 2,700,000	64	8,388,608	1/64, 1/32, 1/16, 1/8, 1/4, 1/2, 1, 2
≤ 3,350,000	128	16,777,216	1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2, 1

The modules have a memory capacity of 128 K words. The Segment Resolution defines the word length and hence the available storage capacity.

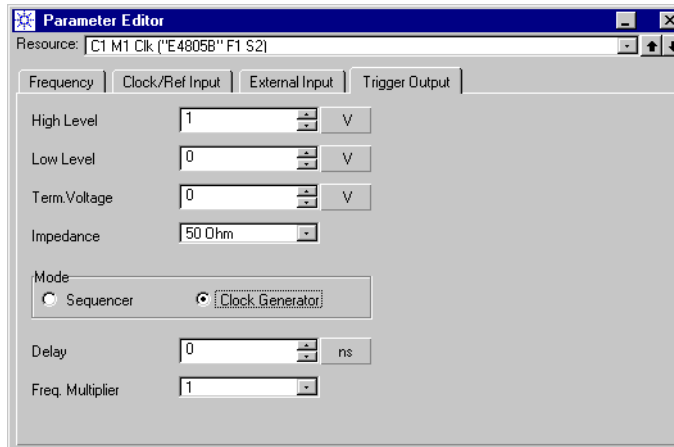
As we are using E4861A modules and 500 MHz in this example, we could set the Segment Resolution to 32 or 64 to make better use of the hardware memory resources.

For details please refer to the *Agilent 81250 ParBERT System User Guide*.

#### Trigger Frequency Multiplier

Note also that the *Trigger Frequency Multiplier* is set to 1/4. This means, the pulse provided by the TRIGGER OUTPUT of the clock module will have a frequency of 125 MHz.

- 3 Open the Trigger Output page.
- 4 Ensure that the voltage levels and the termination conform to the requirements of the DUT.
- 5 Set the operating *Mode* of the TRIGGER OUTPUT to *Clock Generator* and set the *Trigger Frequency Multiplier* to 1.



Once a test is running, the TRIGGER OUTPUT will now produce the 500 MHz system clock.



**NOTE** The TRIGGER OUTPUT is limited to frequencies below 675 MHz. Higher clock frequencies can be generated by the generator frontends.

In *Sequencer* mode, the TRIGGER OUTPUT can be used to issue a single pulse at the beginning of a block of the overall data sequence. It can also be used to issue a single pulse indicating that a certain event has occurred. For details please refer to the *Agilent 81250 ParBERT System User Guide*.

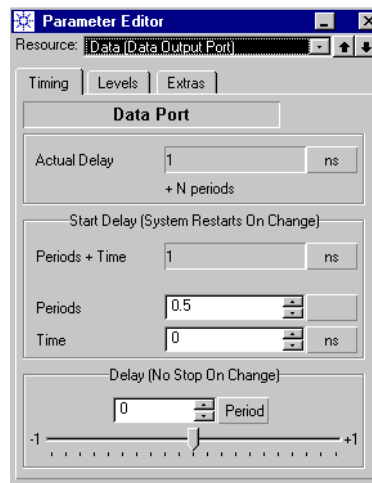
6 Close the Parameter Editor.

## Setting Voltage Levels and Termination

The properties of signal generators and analyzers have to be set up according to the requirements of the DUT. To keep it simple, you can set up all the channels connected to a port in one go.

Output port parameters 1 Double-click the DUT output port.

This opens the Parameter Editor for that port. It comes up with the Timing page.



Note that the *Start Delay* is set to 0.5 periods by default. In this example, this is equivalent to 1 ns.

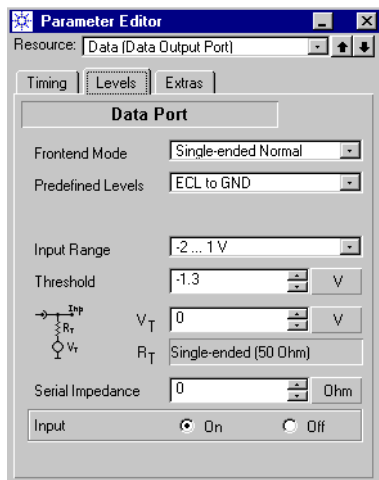
2 Open the Levels page and choose the *Frontend Mode Single-ended Normal*.

As we are using one simple cable for substituting the DUT, this is the right choice.

- 3 Choose a suitable level range and place the threshold into the middle.

We use **ECL to GND** in this example. For predefined levels, the threshold is automatically set.

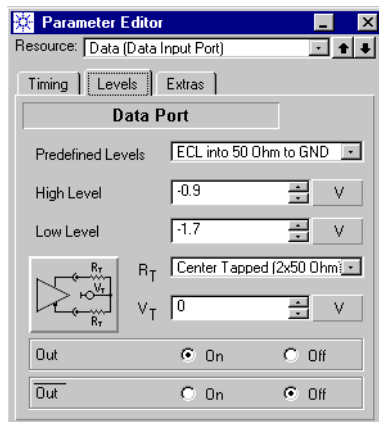
- 4 Click the On button of the *Input* field, as shown in the figure below. This switches the analyzer on and is physically indicated by the green LED above the connector.



- Input port parameters
- 1 Double-click the DUT input port.

This opens the Parameter Editor's Timing page for that port.

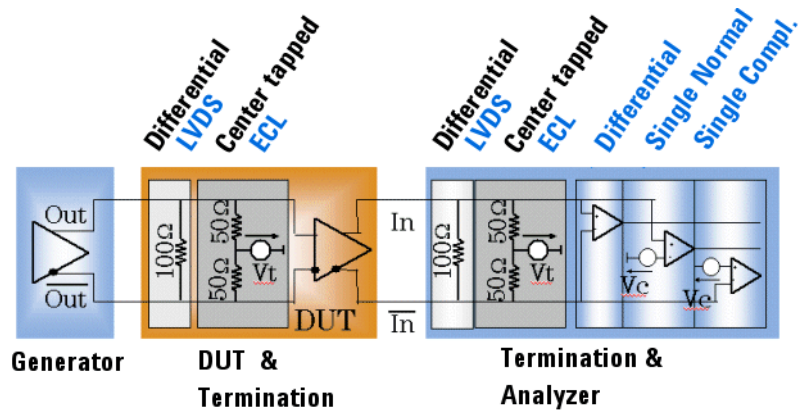
- 2 Open the Levels page and choose a level range that fits to the analyzer setting.
- 3 Click the On button of the *Out* connector.



- 4 Make sure that the green LED is on.
- 5 Close the Parameter Editor.

- 6 Use an SMA cable and connect the generator and analyzer physically.

The generators and analyzers support many predefined levels and signal terminations, as shown in the figure below:



Generators can generate signals for differential or center-tapped termination. The termination setting of a generator has an impact on the generated signal levels. The specified signal levels are met when the generator is correctly terminated.

The input termination and operation of analyzers can be adapted to the output characteristics of the DUT. Differential as well as center-tapped termination is available. The incoming signal can be analyzed in differential or single-ended mode (the latter either normal or inverted). For details please refer to the *Agilent 81250 ParBERT System User Guide*.

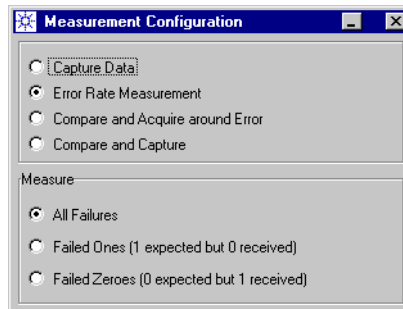
As we have connected the generator directly with the analyzer, the generator output termination is provided by the analyzer input circuitry.

## Checking the Measurement Mode

The measurement mode has an impact on the segments that can be used in the data sequence. For example, you cannot acquire data if a BER test is specified.



- 1 Click the Measurement Configuration button.
- 2 Ensure that *Error Rate Measurement* is enabled. This is the default.



- 3 Close the Measurement Configuration window.

## Specifying the Test Sequence

The sequence of generated and expected data can be specified with one of three available Sequence Editors. We will use the Standard Mode Sequence Editor in this example.



- 1 Click the Sequence Editor button.

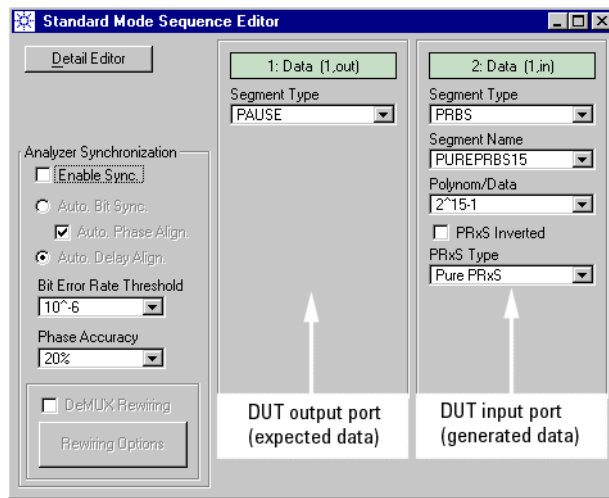
The Sequence Editor shows the two ports. The default segments are PAUSE0 for the input port and PAUSE for the output port. That means for the associated generator “keep zero voltage” and for the analyzer “ignore”.

Input port (generated data)

Set up the DUT input port first, because this defines the data the generator will send:

- 2 Set the *Segment Type* to **PRBS**.

3 Type the *New Segment Name*: **PUREPRBS15**, and click *Create*.



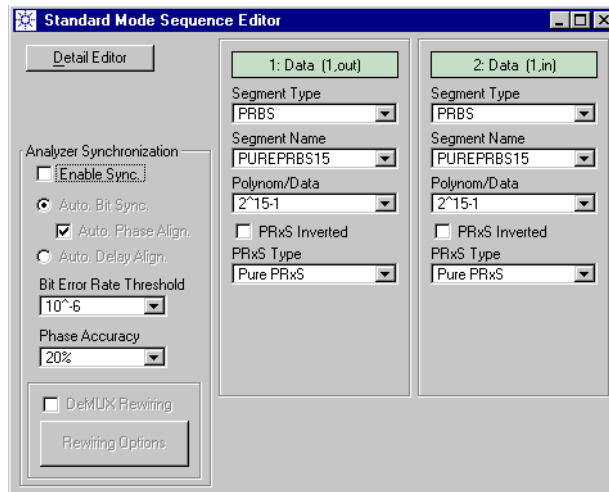
A pure, undistorted PRBS of polynomial  $2^{15}-1$  is the default. You could also choose from other polynomials, but a pure PRBS of polynomial  $2^{15}-1$  is what we are going to generate.

Pure PRBS do not consume hardware memory, neither on the generating nor on the analyzing side. They are generated by special feedback shift registers built into the data modules.

Output port (expected data)

4 Set the *Segment Type* of the DUT output port to **PRBS**.

5 From the *Segment Name* list, choose the segment **PUREPRBS15**. This defines the expected data.



**Save the setting** Now that the test setup is complete, it is time to save the setting:

- 1 Open the File menu and choose *Save Setting As ...*
- 2 Save the setting under the name **PRBS\_1A**.

Once the setting has been saved, you can always return to the present status.



After you have saved the setting for the first time, you can update it and save any changes by clicking the Save Setting button.

## Test Execution Using Auto Delay Alignment

Both methods for automatic analyzer sampling delay adjustment can be specified directly in the Standard Mode Sequence Editor.

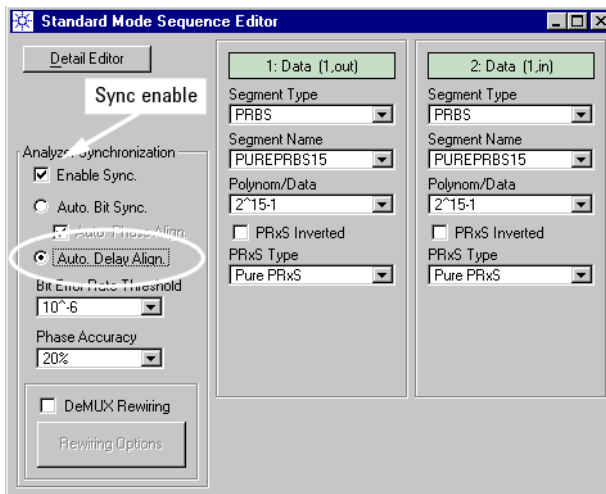
**TIP** If you are using the Detail Mode Sequence Editor or the Data/Sequence Editor, open the Edit menu and choose *Sync*.

## Specifying Auto Delay Alignment

Do the following:

- 1 Enable *Analyzer Synchronization*.
- 2 Enable *Auto Delay Alignment*.

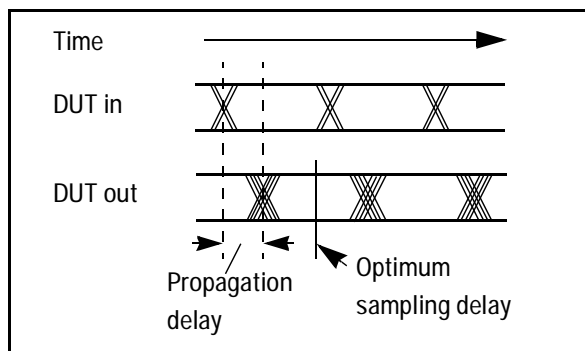
In the Standard Mode Sequence Editor, this looks as follows:



Auto Delay Alignment will fully automatically place the sampling delay of the analyzer into optimum position, which means in the middle of the eye diagram of the received signal.

There is only one restriction: Auto Delay Alignment searches around the analyzer start delay for a certain time span. This time span is  $\pm 50$  ns for E4832A modules and  $\pm 10$  ns for E4861A modules.

So, the analyzer start delay must be set to a value that allows the synchronization process to capture the incoming signal within that time span with an adequate precision. This precision is defined by the *Bit Error Rate Threshold*.



Because our DUT consists of a cable of roughly 60 cm length, we do not expect a signal delay between the generator output and the

analyzer input connectors of more than a few nanoseconds. Therefore we did not care much about the start delay.

The *Phase Accuracy* refers to the phase optimization where the analyzer measures the width of the signal's eye diagram. It can be set to values between 1 % and 20 %. It defines whether the analyzer performs up to 100 BER measurements or just five. In this example, we stay with five, which means 20 %.

- 3 Close the Sequence Editor window.

## Running the Test

We will now run the test:



- 1 Open the Bit Error Rate Display.

Opening this display before starting the test has the advantage that you can view the test results from the beginning.



- 2 Click the Run button.

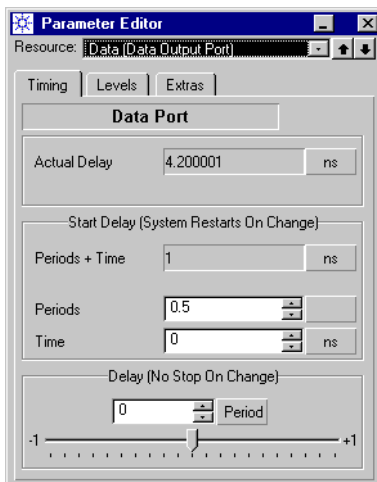
The Bit Error Rate Display is updated approximately every second. *Actual* values report the results of the last measurement interval—the time since the last update.

Port 1: Data								
Term	Rst	S	Actual Number of Bits	Actual Number of Errors	Actual Bit Error Rate	Accum. Number of Bits	Accum. Number of Errors	Accum. Bit Error Rate
1: Data0	R	<input checked="" type="checkbox"/>	3.155248e+007	0.000000e+000	0.000000e+000	1.351155e+010	0.000000e+000	0.000000e+000
Summary			3.155248e+007	0.000000e+000	0.000000e+000	1.351155e+010	0.000000e+000	0.000000e+000

If the system could not synchronize, then the bit error rate counters would be disabled and display just empty fields.



- 3 In the Connection Editor, double-click the DUT output port. This opens the Timing page of the port. Here you can inspect the result of the synchronization.



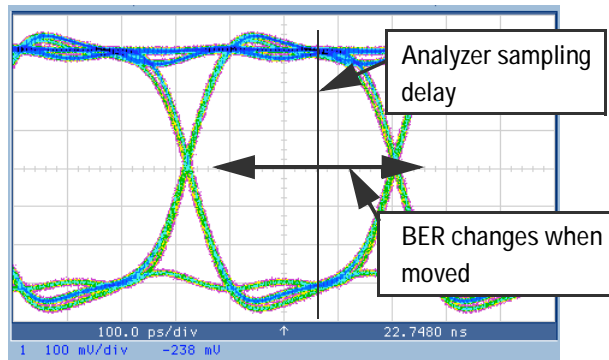
**Actual delay** After Auto Delay Alignment, the *Actual Delay* shows the full analyzer delay since starting the clock. In this example, it is 4.2 ns. Auto Delay Alignment has increased the default start delay of one half period by 3.2 ns. This is the traveling time of the signal. The *Actual Delay* of 4.2 ns represents the optimum moment for sampling the incoming signal.

**NOTE** Auto Delay Alignment is the method of choice if you have a rough idea of the signal delay between generator and analyzer and set this as the *Start Delay*. From this point, the expected delay must be within  $\pm 50$  ns for E4832A modules and  $\pm 10$  ns for E4861A modules. Auto Delay Alignment works with all kinds of data and informs you about the total delay.

**Delay vernier** You may move the delay vernier while the test is running. This changes the actual delay of the analyzer sampling point immediately. When you move the delay vernier, observe the *Actual Bit Error Rate* in the Bit Error Rate Display. On both sides of zero, you will find a point where the actual bit error rate increases rapidly. As zero represents the optimum, these points are generally close to

$\pm 0.5$  periods. The area between these points indicates the width of the signal's eye opening.

This is illustrated in the figure below.



#### Start delay

If you change one of the *Start Delay* settings while a test is running, then the test is aborted and automatically restarted, beginning with the synchronization. This is the message of *System Restarts on Change*.



- 4 Click the Stop button to terminate the test.

## Test Execution Using Auto Bit Synchronization

Automatic Bit Synchronization is another method for adjusting the analyzer sampling delay to the incoming data.

Automatic Bit Synchronization does not consider the start time of the system clock. Instead, it shifts the phase of the sampling edge relative to the clock.

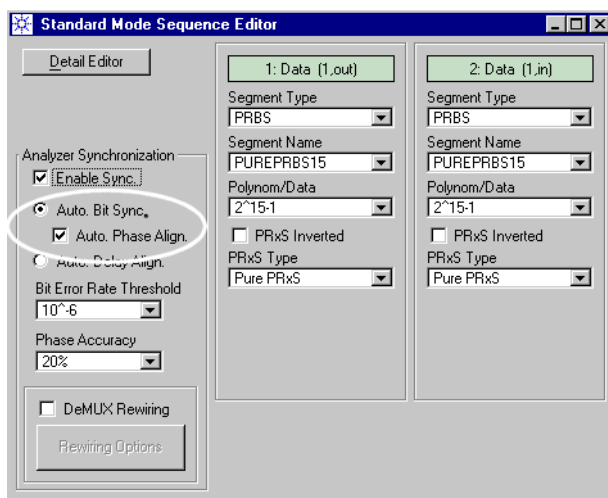
You may specify an analyzer start delay. If this exceeds one system period, all full periods are ignored.

## Specifying Auto Bit Synchronization

We use the Standard Mode Sequence Editor once more.



- 1 Click the Sequence Editor button.
- 2 Enable *Auto Bit Synchronization*.



Auto Bit Synchronization works fine with pure PRBS data.

Distorted PRBS or memory-type data cannot be used on a single system that generates *and* analyzes data.

If distorted PRBS or memory-based data is used, Auto Bit Synchronization can be used on a separate analyzing system.

Auto Bit Synchronization shifts the phase of the sampling edge until the incoming signal is recognized with an adequate precision. This precision is defined by the *Bit Error Rate Threshold*.

A reasonable analyzer start delay (such as 0.5 periods) may accelerate the process. If the specified analyzer start delay exceeds one period, all full periods are discarded.

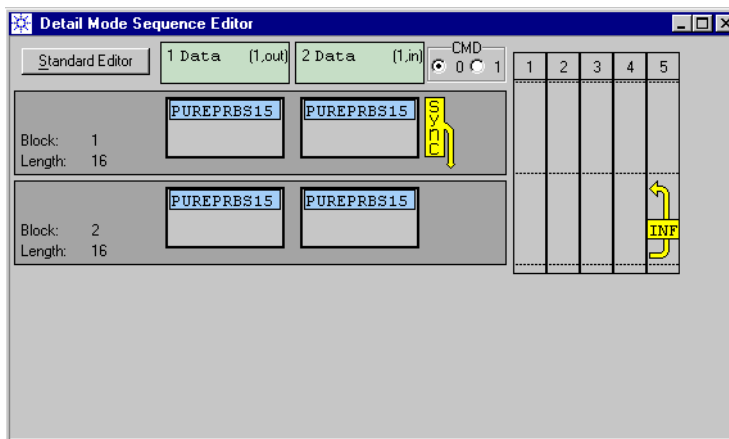
If *Auto Phase Alignment* is enabled—which is the default—, then the analyzer measures the width of the signal's eye diagram. It fully automatically places the sampling delay of the analyzer to optimum position, which means in the middle of the eye diagram of the received signal. This takes slightly more time, of course.

The *Phase Accuracy* refers to the *Auto Phase Alignment* and has the same meaning as for Auto Delay Alignment.

### 3 Click *Detail Editor*.

This opens the Detail Mode Sequence Editor that shows the data sequence which is going to be used for the test.

This is just for information—everything comes from the Standard Mode Sequence Editor.



The sequence consists of two blocks. Both reference for both ports the same data segment: PUREPRBS15.

The first block is the synchronization block. It is automatically repeated until the synchronization criteria are met.

The second block is used for the BER test. It is repeated infinitely. That means, you have to stop the test by clicking the Stop button.

#### NOTE

As soon as one of the automatic analyzer synchronization methods is enabled, the test sequence needs a synchronization block. The synchronization block has to be the first block to be executed. The Standard Mode Sequence Editor creates this block automatically by duplicating the specified block.

The Standard Mode Sequence Editor is meant for setting up BER tests quickly and efficiently. Therefore, it adds an infinite loop to the measurement block.

All this can be changed with the Detail Mode Sequence Editor.

Changes, however, may have the effect that you cannot return to the Standard Mode Sequence Editor.

### 4 Close the Sequence Editor window.



4.25 ns. But if you execute only Auto Bit Sync, the total delay remains unknown.

You may wonder about the difference between 4.20 ns and 4.25 ns. This is due to the chosen *Phase Accuracy* of just 20%.

If *Auto Phase Alignment* has been enabled, then the resulting phase delay is the optimum moment for sampling the incoming signal. If not, the resulting phase delay is just a suitable sampling point.

#### NOTE

Auto Bit Synchronization is used if the signal propagation delay is unknown. It reports only the analyzer phase delay relative to the clock.

On a single system that generates and analyzes data, only pure PRBS data can be used.

Auto Bit Synchronization with Auto Phase Alignment is the default. It optimizes the sampling point.

Auto Bit Synchronization without Auto Phase Alignment can be used to speed up the synchronization, especially if a certain phase delay is expected and set in the Parameter Editor's Timing page.

Delay vernier, start delay

The other functions of the window are the same as for Auto Delay Alignment. You may move the delay vernier while the test is running. You may also change the *Start Delay* settings.



- 4 Finally click the Stop button to terminate the test.



# BER Test on a Single System Using Memory Data

This example demonstrates how to set up a bit error rate (BER) test on a single system using memory data. Memory data is a pattern stored in a file.

See:

- *“Focus of this Example” on page 32*
- *“Hardware Setup” on page 33*
- *“Test Setup” on page 34*
- *“Test Execution With Manual Analyzer Delay Adjustment” on page 45*
- *“Test Execution With Automatic Analyzer Delay Adjustment” on page 50*

# Focus of this Example

Highlights of this example:

- We use one ParBERT system that generates stimulating data and analyzes the response of the device under test (DUT).
- A DUT with two input and two output terminals is tested.
- A certain data pattern is generated and expected.
- The sampling delay of the analyzers is set manually and automatically.
- The bit error rate (BER) is measured.

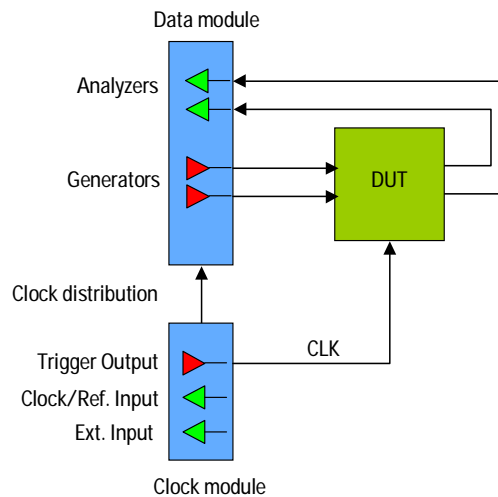
What you will learn You will learn:

- How to use the Connection Editor quickly and efficiently
- How to create the test sequence
- How to create data segments that define generated and expected data
- How to set the analyzer delay to the optimum sampling point
- How to run the test



# Hardware Setup

The hardware setup is illustrated in the figure below:

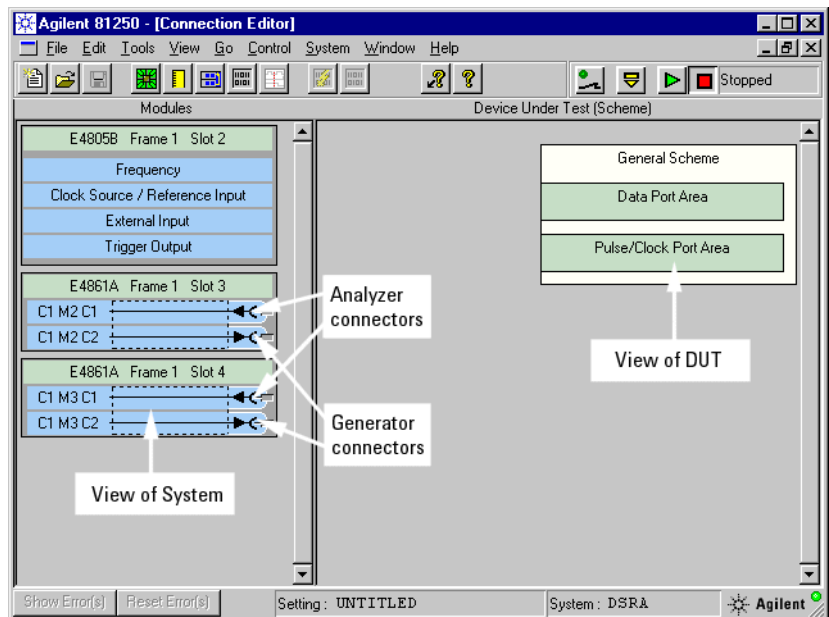


The TRIGGER OUTPUT of the clock module is used to provide the clock pulse to the DUT. A generator could be used as well.

To reproduce this example without a DUT, you can use two shielded SMA cables and connect the analyzers with the generators. Any ParBERT system that has two generators and two analyzers can be used.

# Test Setup

As we have not loaded any setting when starting the Agilent 81250 User Software (which is also possible), the Connection Editor shows just the system configuration and an empty “General Scheme” of the DUT.



**View of system** The left-hand side identifies the modules plugged into the VXI frame. The module on the top is the master clock module.

Note that the arrows associated with the connectors of data modules indicate the signal direction and hence the connector type—generator or analyzer.

**View of DUT** Data ports are used for sourcing test data to and capturing data from the DUT. They are divided into DUT input ports and DUT output ports.

Pulse/Clock ports are used for sourcing clock pulses to the DUT, if such pulses are generated by generator frontends. We will not use a pulse/clock port in this example.

## Specifying the Connections

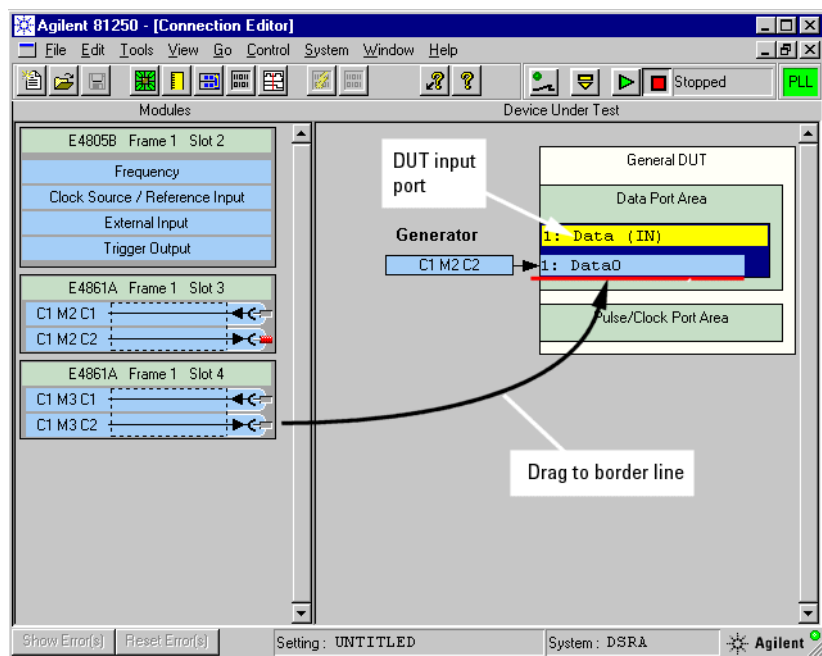
In this example, we will not use the menus of the Connection Editor. We will create data ports with terminals and connect them in one go using the drag and drop feature. Proceed as follows:

- 1 Click the upmost generator connector. Keep the mouse button depressed and drag the connector over the *Data Port Area*.

This creates a DUT input port with one terminal and establishes the connection.

**NOTE** If the chosen module had additional generator connectors in a row, the port would contain several terminals, all readily connected.

- 2 Click the second generator connector and drag it over the lower border line of the existing terminal. When the border turns red, release the mouse button.



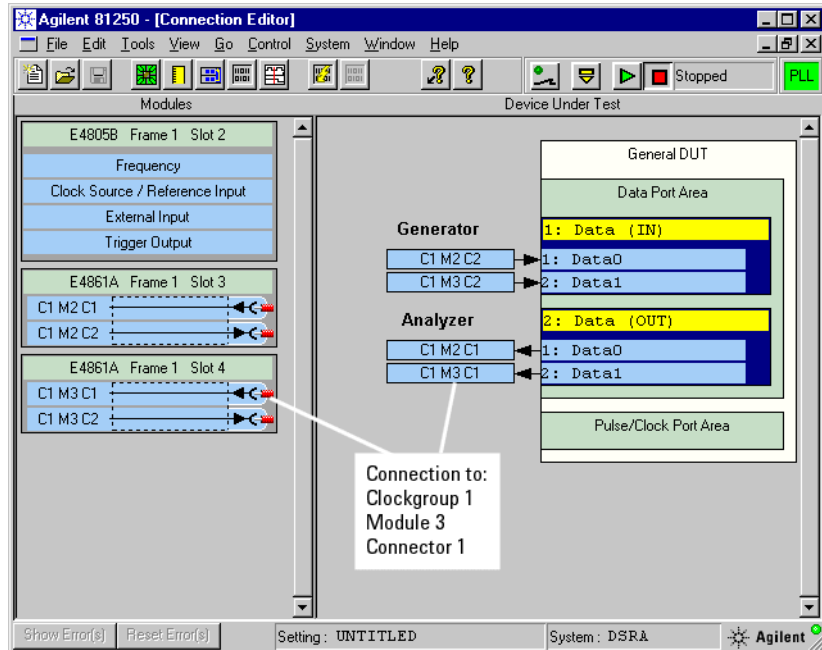
This adds a second terminal to the port and connects it.

Note that ports and terminals are automatically named and numbered, but this can be changed at any time.

Note also that you cannot drag an analyzer connector to a DUT input port, and vice versa.

### 3 Repeat the steps 1 and 2 for the analyzer connectors.

This creates a DUT output port and establishes the connections.



Because we will be using the TRIGGER OUTPUT of the clock module to provide a fraction of the system clock to the DUT, no pulse port is needed. Our DUT has a built-in frequency multiplier.

We would have to set up a pulse port, if we were using an extra generator to supply the clock to the DUT. An extra generator can supply clock frequencies above 675 MHz—the clock module’s TRIGGER OUTPUT cannot.

**NOTE** A generator connected to a pulse port terminal is automatically put into clock mode. This means that it generates a rectangular pulse with 50 % duty cycle and the frequency set with the Parameter Editor.

## Setting the System Clock Frequency

We will use a system clock frequency of 1250 MHz in this example.

1 In the Connection Editor, double-click the *Frequency* box of the clock module.

This opens the Frequency page of the Parameter Editor for the clock module.

2 Set the *Segment Resolution* to 32 and press Enter.

**NOTE** If you wish to change the system clock frequency, always start with checking and eventually correcting the Segment Resolution. This helps to avoid error messages.

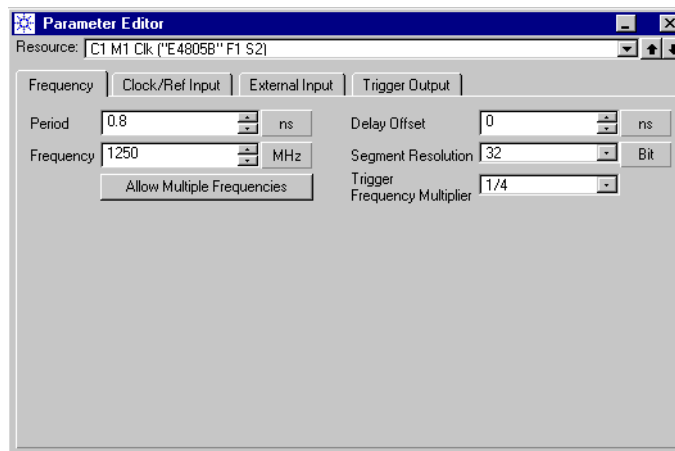
Refer to the tables given in “*Clock Rates, Segment Resolution, and Memory Depth for E4832A Modules*” on page 15 and following.

For E4861A modules, the minimum Segment Resolution for frequencies between 675 MHz and 1.35 GHz is 32.

We could also set the Segment Resolution to 64 to make better use of the hardware memory resources.

**3** Set the system frequency to 1250 MHz and press Enter.

When you press Enter, the Parameter Editor checks the value and accepts or rejects it.



#### Period/Frequency

The system *Period* is always the reciprocal of the *Frequency*, and vice versa. If you change the *Period*, the *Frequency* will be updated.

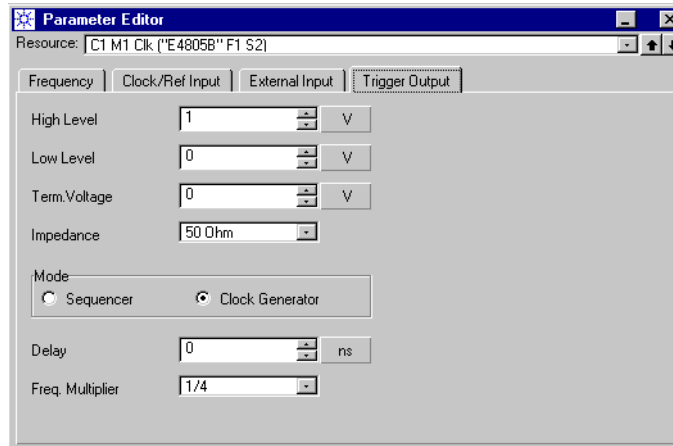
#### Trigger Frequency Multiplier

Note also that the *Trigger Frequency Multiplier* is set to 1/4. This means that the pulse provided by the TRIGGER OUTPUT of the clock module will have a frequency of 312.5 MHz.

You could also set the *Trigger Frequency Multiplier* to 1/2. The TRIGGER OUTPUT can supply a clock frequency of 625 MHz. But if the DUT would need the system clock frequency of 1.25 GHz, then you would have to install an additional generator.

**4** Click the *Trigger Output* tab.

- 5 Ensure that the voltage levels and the termination conform to the requirements of the DUT and that the operating *Mode* of the TRIGGER OUTPUT is set *Clock Generator*.



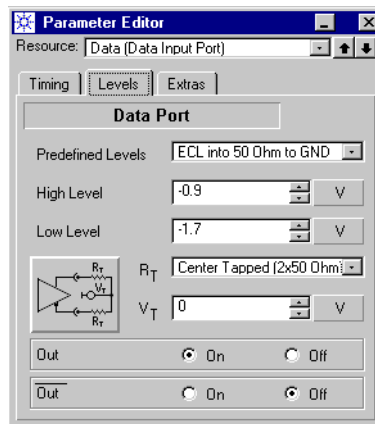
- 6 Close the Parameter Editor.

## Setting Voltage Levels and Termination

The properties of signal generators and analyzers have to be set up according to the requirements of the DUT.

- Input port parameters
- 1 In the Connection Editor, double-click the DUT input port.  
This opens the Parameter Editor's Timing page for that port. We are not going to specify any delays for the generators.
  - 2 Open the Levels page and choose a suitable signal level.  
The predefined levels are easy to use and consider many devices. You can also specify custom signal levels.

### 3 Enable the *Out* connector.



Note that the generator expects a load of 50  $\Omega$  to ground. This is illustrated on the graphical button.

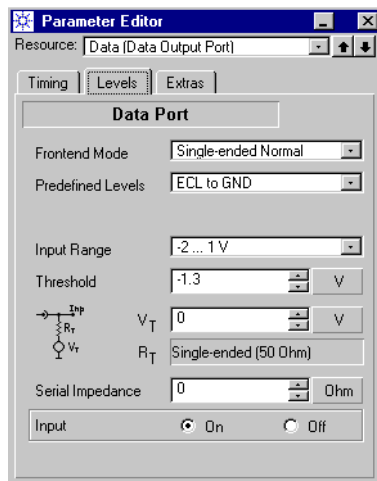
**NOTE** The Parameter Editor has a third tab named Extras. On this page, you can specify how the disconnection is made if the Connectors Off/On button is actuated.



For details see the *Agilent 81250 ParBERT System User Guide*.

- Output port parameters**
- 1 In the Connection Editor, double-click the DUT output port. This opens the Parameter Editor for that port. It comes up with the Timing page. We will adjust the analyzer timing later in this example.
  - 2 Open the Levels page and choose the *Frontend Mode Single-ended Normal*. This fits to the generators which expect a 50  $\Omega$  termination.
  - 3 Choose a suitable signal level. In this example, we use the same as for the generators: **ECL to GND**.  
For the predefined levels, *Input Range* and *Threshold* are automatically adapted. If you had generated custom levels, you would now choose a suitable *Input Range* and place the *Threshold* into the middle of the expected signal high and low levels.

4 Enable the *Input*, as shown in the figure below.



5 Make sure that the green LEDs of the connectors are lit.

6 Close the Parameter Editor.

7 Use two SMA cables and connect the generators and analyzers physically.

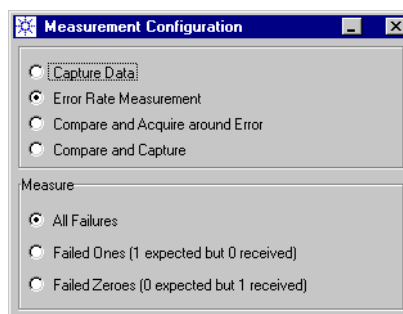
## Checking the Measurement Mode

The measurement mode has an impact on the segments that can be used in the data sequence. For example, you cannot acquire data if a BER test is specified.



1 Click the Measurement Configuration button.

2 Ensure that *Error Rate Measurement* is enabled. This is the default.



3 Close the Measurement Configuration window.



## Creating the Data Sequence and Segment

We will use the Standard Mode Sequence Editor in this example.



- 1 Click the Sequence Editor button.

The Sequence Editor shows the two ports. The default segments are PAUSE0 for the input port and PAUSE for the output port. That means for the associated generators “keep zero voltage” and for the analyzers “ignore”.

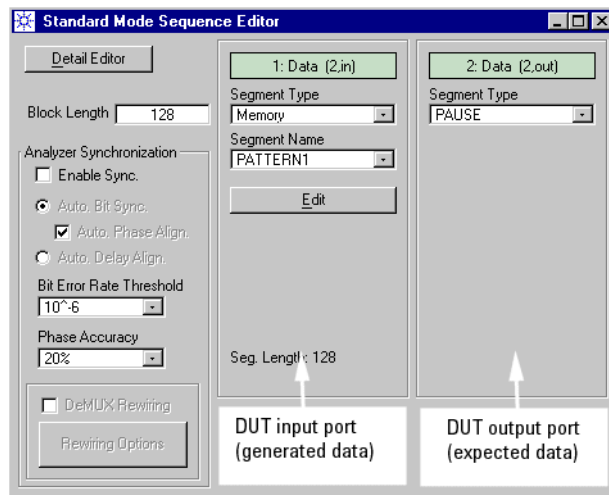
Input port (generated data)

Set up the DUT input port first, because this defines the data the generator will send:

- 2 Set the *Segment Type* to **Memory**.
- 3 Type the *New Segment Name*: **PATTERN1**.
- 4 Enter the *Segment Length* and click *Create*.

### NOTE

The segment length has to be an integer multiple of the Segment Resolution, which in this example is 32. We use a segment length of 128 vectors.



The *Block Length* shown on the window’s left-hand panel is automatically adjusted to the segment length.

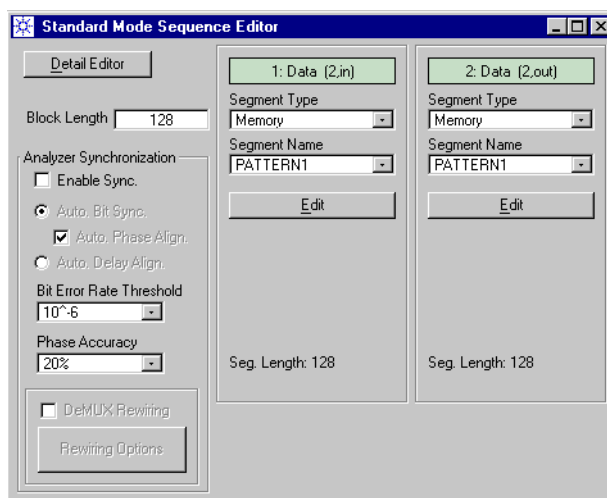
Every sequence consists of blocks. A block spans over all data ports of the DUT. Blocks contain references to the data segments that will be used for the test. Different data ports can reference different segments. Blocks can be executed once or repeatedly.

**NOTE** The Standard Mode Sequence Editor accepts only one block. This block is endlessly looped.

If automatic analyzer synchronization is enabled, the Standard Mode Sequence Editor duplicates this block to create a sync block. The sync block is repeated until all analyzers have finished the synchronization process. If automatic analyzer synchronization is disabled, the Standard Mode Sequence Editor removes the sync block.

Additional blocks referencing different data segments can be created and maintained with the Detail Mode Sequence Editor or the Data/Sequence Editor.

- Output port (expected data)
- 5 Set the *Segment Type* of the DUT output port to **Memory**.
  - 6 From the *Segment Name* list, choose the segment **PATTERN1**. This defines the expected data.



**Save the setting** Now that the test setup is almost complete, it is time to save the setting:

- 1 Open the File menu and choose *Save Setting As ...*
- 2 Save the setting under the name **MEMO\_1A**.

Once the setting has been saved, you can always return to the present status.



After you have saved the setting for the first time, you can save it occasionally by clicking the Save Setting button. This updates the saved setting.

## Editing a Memory Data Segment

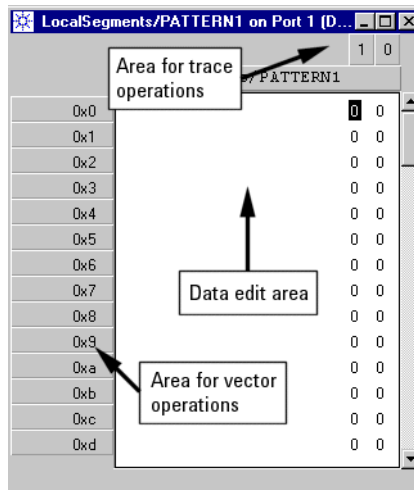
Now that we have created a memory segment, we have to fill it with data. We will do this manually in this example, in order to demonstrate some features of the Segment Editor.

Experienced users will rarely proceed this way. They will convert simulation data into segments for the generators. They will capture the output of a golden device and use this as a template for the analyzers.

However, because the Segment Editor allows you to examine and modify all memory segments, no matter how they were created, you should be familiar with its appearance and capabilities.

- 1 In the Standard Mode Sequence Editor window, click the *Edit* button of the DUT input port.

This opens the Segment Editor for the chosen segment.



The segment has two traces (it was created for a two-terminal port) and 128 vectors. Trace and vector numbers start from zero.

The window has three “active” areas where you can open individual context menus (with the right mouse button) providing appropriate functions.

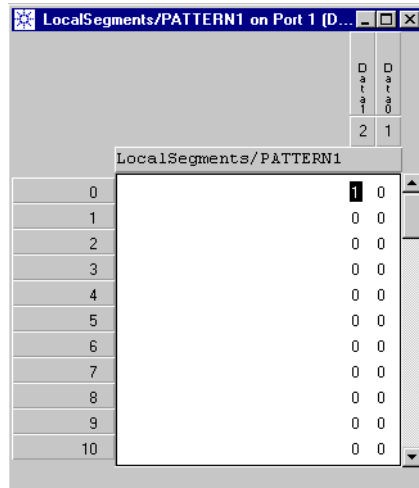
- 2 In the area for vector operations, open the context menu and choose the *Decimal* address format.

This displays the vector numbers in integer format.

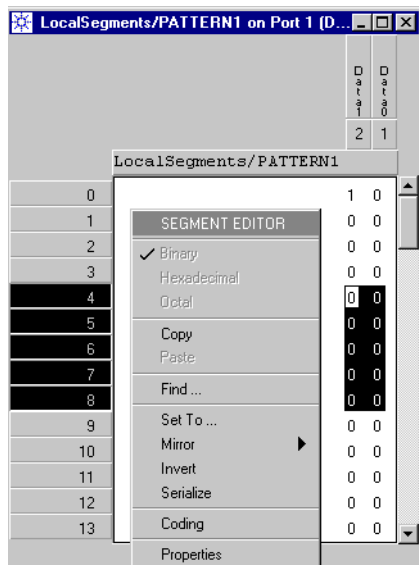
- 3 In the area for trace operations, open the context menu and choose the *Terminal View*.

This displays the terminal names.

- 4 Click any bit and type “1” or “0” to change its contents.  
The window now looks as shown below.



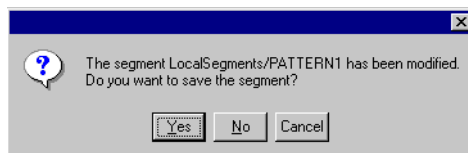
- 5 Using the mouse, highlight some vectors or traces and then open the context menu of the data edit area.



Because we have chosen the *Terminal View*, only binary vector display is available. Otherwise you could combine traces to show octal or hex numbers.

- 6 Click *Invert* to change all highlighted zeros to ones and vice versa.  
Refer to the online Help or the *Agilent 81250 ParBERT System User Guide* for an explanation of all the additional options.

- 7 When you are done, close the Segment Editor. You will be asked whether you wish to save your changes.



- 8 Click *Yes* to save the updated segment.

## Test Execution With Manual Analyzer Delay Adjustment

An analyzer start delay that is entered in the Parameter Editor takes effect as soon as a test is started. This is the fastest way to execute device tests.

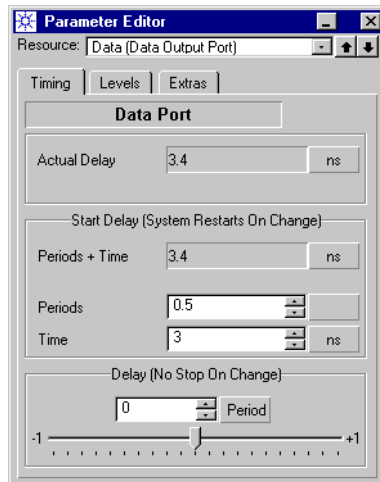
The methods for automatic analyzer delay adjustment require some more execution time, because the actual measurement does not start until all analyzers have finished the synchronization process—either successfully or with error.

To get an impression of what means “some more time”, we will use both methods and start with manual analyzer delay adjustment.

## Specifying a Common Analyzer Start Delay

In the previous example, we have found the signal traveling time for a simple cable connection to be approximately 3.2 ns.

- 1 Open the Parameter Editor for the DUT output port.
- 2 Set the *Start Delay* to 3.4 ns. This can be done by adding 3 ns to the default delay of 0.5 periods.



Note that the delay vernier at the bottom of the window offers a sweep range of  $\pm 1$  periods. Because we have set the system clock period to 0.8 ns (1250 MHz), the vernier covers a total delay range from 2.6 ns to 4.2 ns.

## Running the Test

To run the test:



- 1 Open the Bit Error Rate Display.

The Bit Error Rate Display shows all the terminals to be analyzed.

Opening this display before starting the test has the advantage that you can view the test results from the beginning.



- 2 Click the Run button.

The Bit Error Rate Display reports no errors.

The screenshot shows the Agilent 81250 software interface. The main window is titled "Bit Error Rate - Port 2: Data" and displays a table of test results. The "Time Since Start" is 00:00:47. The table shows data for two terminals, Data0 and Data1, with columns for Actual Number of Bits, Actual Number of Errors, Actual Bit Error Rate, Accum. Number of Bits, Accum. Number of Errors, and Accum. Bit Error Rate. A summary row is also present. Below the table is a "Parameter Editor" window for the "Data (Data Output Port)" resource, showing settings for "Data Port" such as Actual Delay (3.4 ns), Start Delay (System Restarts On Change) with Periods + Time (3.4 ns), Periods (0.5), and Time (3 ns), and a "Delay (No Stop On Change)" slider set to 0.

Port 2: Data			Actual Number of Bits	Actual Number of Errors	Actual Bit Error Rate	Accum. Number of Bits	Accum. Number of Errors	Accum. Bit Error Rate
Term	Rst	S						
1: Data0	R	<input checked="" type="checkbox"/>	1.250000e+009	0.000000e+000	0.000000e+000	5.897500e+010	0.000000e+000	0.000000e+000
2: Data1	R	<input checked="" type="checkbox"/>	1.250000e+009	0.000000e+000	0.000000e+000	5.900000e+010	0.000000e+000	0.000000e+000
Summary			2.500000e+009	0.000000e+000	0.000000e+000	1.179750e+011	0.000000e+000	0.000000e+000

- 3 While the test is running, click the down-arrow of the delay vernier to move the slider to the left. After each click, observe the *Actual Bit Error Rate*. Continue until the *Actual Bit Error Rate* increases. This happens at  $-0.3$  periods. The *Actual Delay* is displayed as 3.16 ns, as shown in the figure below.

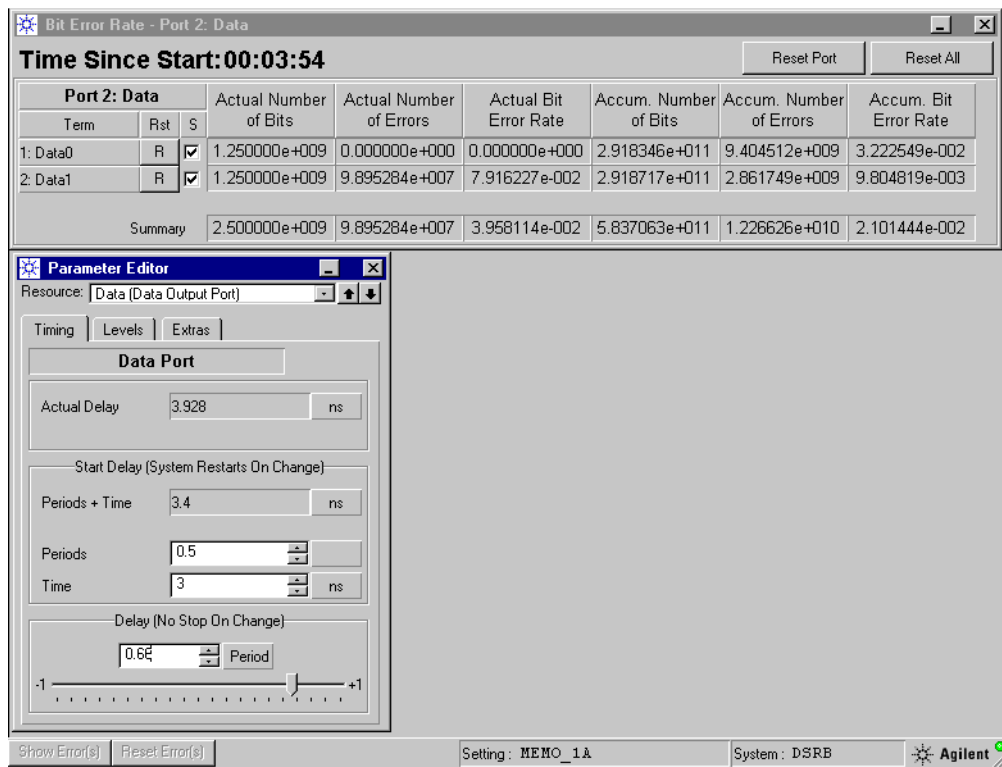
The screenshot displays the Agilent 81250 software interface. The main window, titled "Bit Error Rate - Port 2: Data", shows test results for two channels: Data0 and Data1. The "Time Since Start" is 00:02:23. The "Actual Delay" is set to 3.16 ns. The "Parameter Editor" window is open, showing the "Data Port" settings. The "Actual Delay" is 3.16 ns, "Start Delay (System Restarts On Change)" is 3.4 ns, "Periods" is 0.5, and "Time" is 3 ns. The "Delay (No Stop On Change)" is set to -0.3 Period. The status bar shows "Setting: MEMO\_1A" and "System: DSRB".

Port 2: Data			Actual Number of Bits	Actual Number of Errors	Actual Bit Error Rate	Accum. Number of Bits	Accum. Number of Errors	Accum. Bit Error Rate
1: Data0	R	<input checked="" type="checkbox"/>	1.250000e+009	1.668027e+008	1.334421e-001	1.779312e+011	2.967986e+009	1.668053e-002
2: Data1	R	<input checked="" type="checkbox"/>	1.225000e+009	2.116300e+004	1.727592e-005	1.779328e+011	3.883670e+005	2.182661e-006
Summary			2.475000e+009	1.668238e+008	6.740358e-002	3.558640e+011	2.968374e+009	8.341317e-003

Note that the two channels show different bit error rates. This is due to the fact that we have set a common analyzer delay for the whole port. Individual channels, however, have usually slightly different optimum sampling points. The deviations become visible, when you approach the borders of the signal's eye opening. It is therefore possible to assign an individual delay to each analyzer.



- 4 Click the up-arrow of the delay vernier to move it to the right. Again, watch the *Actual Bit Error Rate* after each step. Continue until the *Actual Bit Error Rate* increases again.



Bit Error Rate - Port 2: Data

Time Since Start: 00:03:54

Port 2: Data			Actual Number of Bits	Actual Number of Errors	Actual Bit Error Rate	Accum. Number of Bits	Accum. Number of Errors	Accum. Bit Error Rate
Term	Rst	S						
1: Data0	R	<input checked="" type="checkbox"/>	1.250000e+009	0.000000e+000	0.000000e+000	2.918346e+011	9.404512e+009	3.222549e-002
2: Data1	R	<input checked="" type="checkbox"/>	1.250000e+009	9.895284e+007	7.916227e-002	2.918717e+011	2.861749e+009	9.804819e-003
Summary			2.500000e+009	9.895284e+007	3.958114e-002	5.837063e+011	1.226626e+010	2.101444e-002

Parameter Editor

Resource: Data [Data Output Port]

Timing Levels Extras

Data Port

Actual Delay: 3.928 ns

Start Delay (System Restarts On Change)

Periods + Time: 3.4 ns

Periods: 0.5

Time: 3 ns

Delay (No Stop On Change)

0.6 ns

Setting: MEMO\_1A System: DSRB Agilent

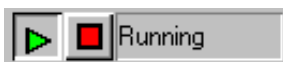
At this point, the *Actual Delay* is 3.928 ns.

- 5 Calculate the optimum sampling delay.

The optimum sampling delay for the two analyzers (and hence the port) must be in the middle:  $(3.16 + 3.928) / 2 = 3.54$  ns.

This is the common analyzer start delay that should be set for further tests to ensure optimum performance.

**NOTE** We just actuated the up/down arrows of the delay vernier. More precise measurements of the eye opening can be made by dragging the slider with the mouse.



- 6 Click the Stop button to terminate the test.



- 7 Click the Save Setting button to save the actual status of the setting MEMO\_1A for re-use in a later example.

# Test Execution With Automatic Analyzer Delay Adjustment

Since we are using memory-based data on one single ParBERT system, automatic analyzer delay adjustment is restricted to Auto Delay Alignment. This refers also to distorted PRBS/PRWS data, because distorted PRxS is generated on the workstation and downloaded into the data memories of the modules.

**NOTE** The second method of automatic analyzer delay adjustment, Auto Bit Sync, cannot be used in combination with memory-based data on a single ParBERT system that generates *and* analyzes data.

However, Auto Bit Sync with memory-based data can be used on a separate system that uses only analyzers. This will be demonstrated in a later example.

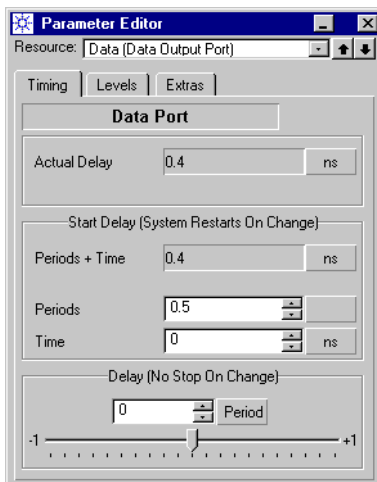
## Specifying Auto Delay Alignment

Auto Delay Alignment works with all kinds of data. It has only one drawback: The capturing range is limited to  $\pm 50$  ns for E4832A modules and  $\pm 10$  ns for E4861A modules around the analyzer start delay.

The analyzer start delay must therefore be set to a value that allows the synchronization process to capture the incoming signal within that time span with an adequate precision.

For our example with minimum delays, this is no problem. We can stay with the default analyzer start delay which is 0.5 periods.

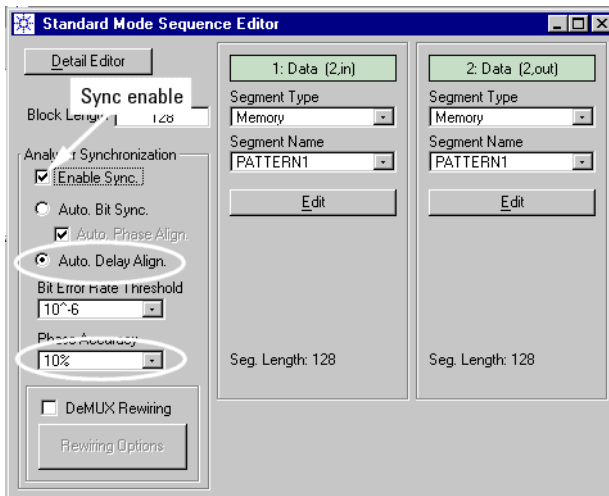
- 1 In the Connection Editor, double-click the DUT output port.  
This opens the timing page of the port. Re-establish default conditions, as shown below.



Note that the delay vernier has to be in zero position.



- 2 Click the Sequence Editor button.  
This opens the Standard Mode Sequence Editor.
- 3 Enable *Synchronization* and *Auto Delay Alignment*.
- 4 Set the *Phase Accuracy* to 10 %.  
This increases the synchronization time slightly, but yields better optimization of the sampling moment.



- 5 Close the Sequence Editor.

## Running the Test

To run the test:



- 1 Open the Bit Error Rate Display.

Now you can observe the test results right from the beginning.



- 2 Click the Run button.

It may take a second until the Bit Error Rate Display gets updated. This is the time used for the synchronization. After that, the BER test is running.

### TIP

The various phases of a test—synchronizing, running, finished, or stopped—are indicated by the side of the Run/Stop buttons.

- 3 While the test is running, open the Parameter Editor for one of the DUT terminals (the simplest method is: Double-click the terminal in the Connection Editor).

The screenshot shows the Agilent 81250 software interface. The main window is titled "Bit Error Rate - Port 2: Data" and displays a table of test results. The "Time Since Start" is 00:00:22. The table shows data for two terminals, Data0 and Data1, with columns for Actual Number of Bits, Actual Number of Errors, Actual Bit Error Rate, Accum. Number of Bits, Accum. Number of Errors, and Accum. Bit Error Rate. A summary row is also present.

Port 2: Data		Actual Number of Bits	Actual Number of Errors	Actual Bit Error Rate	Accum. Number of Bits	Accum. Number of Errors	Accum. Bit Error Rate
1: Data0	R	1.225000e+009	0.000000e+000	0.000000e+000	2.485000e+010	0.000000e+000	0.000000e+000
2: Data1	R	1.225000e+009	0.000000e+000	0.000000e+000	2.485000e+010	0.000000e+000	0.000000e+000
Summary		2.450000e+009	0.000000e+000	0.000000e+000	4.970000e+010	0.000000e+000	0.000000e+000

The Parameter Editor window is open, showing the "Data Terminal E4863A" configuration. The "Actual Delay" is set to 3.600001 ns. The "Start Delay (System Restarts On Change)" section includes "Periods + Time" (0.4 ns), "Periods" (0.5), and "Time" (0 ns). The "Delay (No Stop On Change)" section includes a slider set to 0 Period.

The Parameter Editor shows the *Actual Delay* which has been automatically determined for this analyzer. It is 3.6 ns. For this

analyzer, this is the optimum moment for sampling the incoming signal.

**NOTE** Every delay value that has been automatically determined refers to a particular analyzer. Both methods for automatic analyzer delay adjustment assign individual delays to every analyzer.

The actual delay for the whole port is neither calculated nor displayed. But if you study the individual analyzer delays, you should be able to determine an appropriate common start delay for the port.

**TIP** By clicking the down/up arrows in the upper right-hand corner of the Parameter Editor window, you can switch directly from one channel to the next and inspect the delays of the additional analyzers.



4 Click the red Stop button to terminate the test.

**NOTE** You may recall the results of the very first example “BER Test on a Single System Using PRBS Data”. We measured the signal traveling time from generator to analyzer as 3.2 ns. By definition, the optimum sampling point for an analyzer is one half period later. In this example, this has led to an actual delay of 3.6 ns.





# Capturing and Analyzing Data on a Single System

This example demonstrates how to capture and analyze received data on a single ParBERT system. Memory data is generated. The data that is actually received is compared with expected data.

See:

- *“Focus of this Example” on page 56*
- *“Hardware Setup” on page 57*
- *“Test Setup” on page 58*
- *“Comparing and Acquiring Data Around Error” on page 65*
- *“Comparing and Capturing Incoming Data” on page 71*
- *“Capturing Incoming Data” on page 81*

# Focus of this Example

The highlights of this example are:

- We use one ParBERT system that generates stimulus data and analyzes the response of the device under test (DUT).
- A DUT with one input and two output terminals is tested.
- A certain data pattern is generated and expected.
- The sampling delay of the analyzers is manually set.
- The analyzers compare incoming data with expected data.
- The results are evaluated.

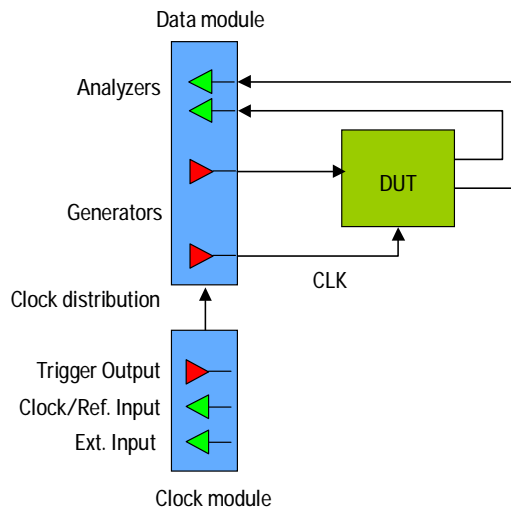
What you will learn You will learn:

- How to compare and capture data from a device
- How to check the results with the Error State Display
- How to use the Waveform Viewer
- How to save captured data for re-use



# Hardware Setup

The hardware setup is illustrated in the figure below:



The DUT has one input and two output terminals. A separate generator is used to supply the system clock to the DUT.

A separate generator channel is required if the DUT needs a clock frequency above 675 MHz or if the TRIGGER OUTPUT of the clock module is used for triggering an external instrument. It is also required if the DUT needs a precision clock signal, because a generator channel produces less jitter than the TRIGGER OUTPUT.

You can reproduce this example without a DUT. We will connect one generator to two analyzers and explain the setup.

Any ParBERT system that has two generators and two analyzers can be used.

# Test Setup

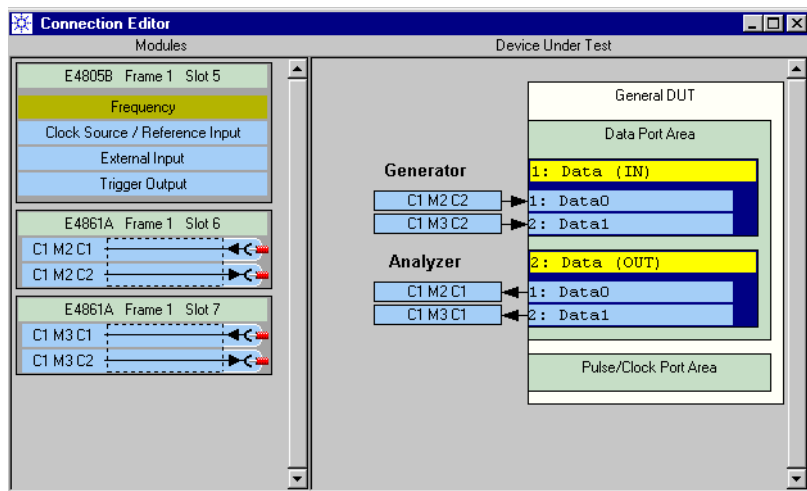
If you are testing a DUT similar to one you have already tested, and if the test conditions are the same, it saves time if you begin the setup with a stored setting.

In this example, we will begin with the setting we have used and saved in the example “*BER Test on a Single System Using Memory Data*” on page 31.



- 1 Click the Open Setting button and load the setting “MEMO\_1A”.

The Connection Editor shows on the left-hand side a ParBERT system with two data modules, and on the right a DUT with one input and one output port.



## Specifying the Connections

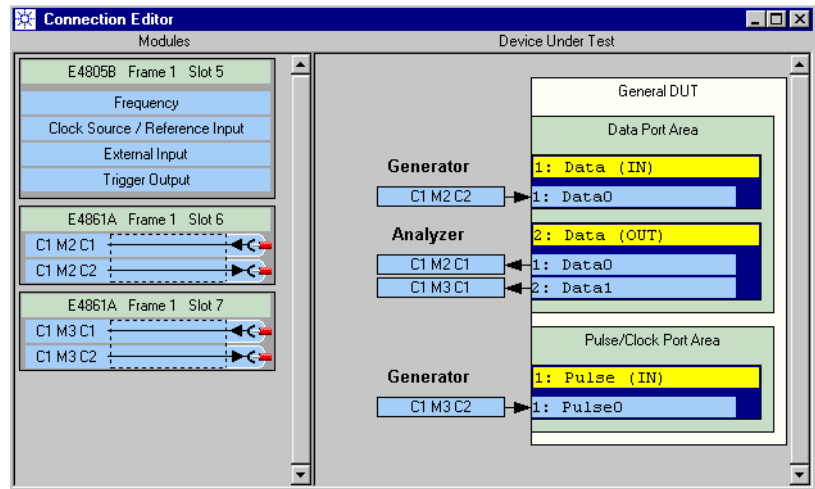
Because the DUT we are testing now differs from the DUT stored in the setting, we have to change two connections:

- 1 Click the terminal “Data1” of the DUT input port with the right mouse button and delete the terminal.

The DUT of the present example has only one input terminal.

- 2 Drag the connector of the second generator over the *Pulse/Clock Port Area*.

This creates a pulse port with one terminal. The result is illustrated in the figure below.



The Connection Editor now corresponds to the configuration described in “*Hardware Setup*” on page 57.

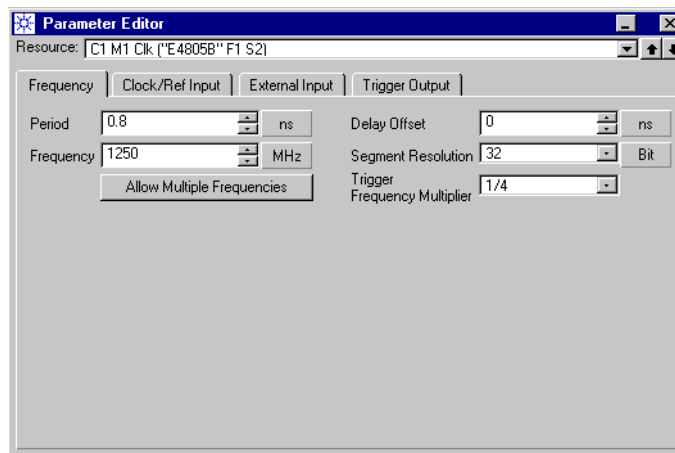
Note that only generator connectors can be connected to pulse port terminals.

## Checking the System Clock Frequency

We will not change the system clock frequency in this example. But you may want to know how the terminals of a pulse port are handled.

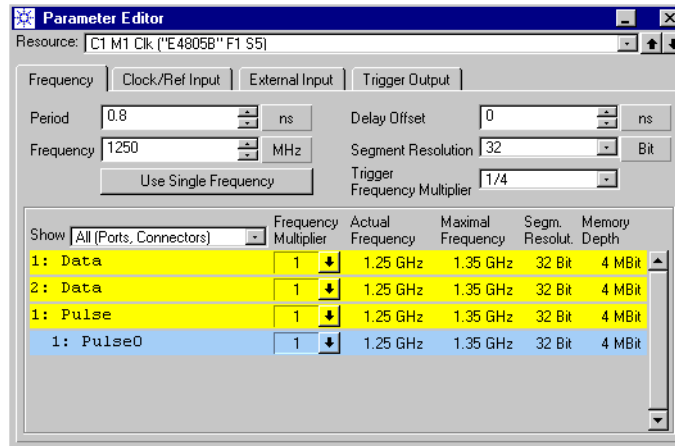
- 1 In the Connection Editor, double-click the *Frequency* field of the master clock module.

This opens the Parameter Editor's Frequency page for the clock module.



2 To see the details, click *Allow Multiple Frequencies*.

This view shows all the ports:



Note that the ports have not only names but also numbers. The numbers make it easy to differentiate between ports with identical names.

The terminals of data ports are not shown. All terminals of a data port have to have the same frequency and Segment Resolution. For pulse ports, this is different. If a pulse port contains more than one terminal, you can assign different pulse frequencies to its terminals.

**NOTE** A generator connected to a pulse port terminal is automatically put into clock mode. This means that it generates a rectangular pulse with 50 % duty cycle and the frequency set with the Parameter Editor.

We will not change the frequency of the pulse generator in this example. It shall provide the system clock to the DUT.

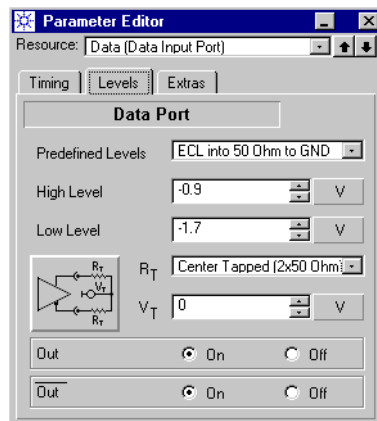
3 Close the Parameter Editor.

## Setting Timing, Voltage Levels and Termination

We are not going to change the parameters of the DUT input port. We keep these parameters as defined in the setting.

**Input port parameters** Just to recall the levels and termination of the generator:

- 1 Open the Levels page of the DUT input port.



The termination is illustrated on the graphic button. The generator expects for each output a load of 50  $\Omega$  to ground and a termination voltage  $V_T$  of zero.

- 2 Ensure that both outputs—normal and inverted—are enabled.

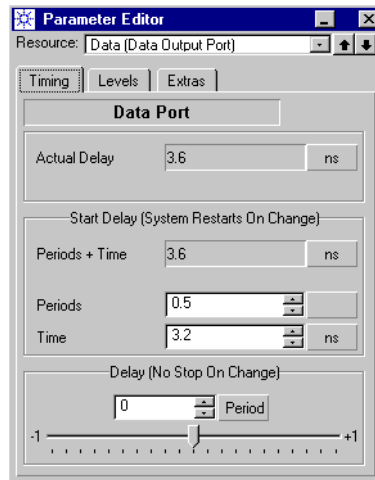
**Output port parameters** We will specify a common start delay for the analyzers and enable their differential sampling mode.

- 1 In the Connection Editor, double-click the DUT output port.  
This opens the Parameter Editor's Timing page for that port.

- 2 Set the *Start Delay* to **3.6 ns**.

You know from the previous example “*BER Test on a Single System Using Memory Data*” on page 31 that 3.6 ns is the optimum

analyzer sampling delay for a setup where the DUT is represented by 60 cm SMA cables and a clock frequency of 1.25 GHz is used.



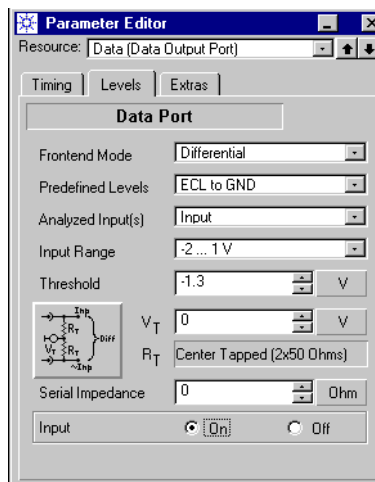
**3** Open the Levels page and choose the *Frontend Mode Differential*.

The *Frontend Mode* defines the way the incoming signal is sampled. It does not define the signal termination.

**4** Keep the signal level as **ECL to GND**.. This is the level specified for the generator.

**5** Set the *Analyzed Input(s)* to **Input**.

This alternative is only available in differential sampling mode.



Note that center-tapped termination—as shown on the graphic button—fits well to the generator which expects a 50  $\Omega$  termination to ground.

**6** Make sure that the analyzer inputs are switched on.

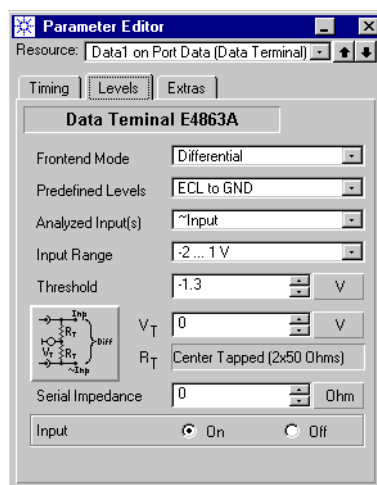
**Output terminal parameters** Now that we have set the common parameters for the DUT output port (they refer to both analyzers), we will change the input of the second analyzer.

- 1 In the Connection Editor, double-click the second terminal of the DUT output port. By default, the terminal is named “Data1”.

This opens the Parameter Editor’s Timing page for that terminal.

**TIP** You can also double-click the associated analyzer channel C1 M3 C1 on the left-hand side of the Connection Editor.

- 2 Open the Levels page and set the *Analyzed Input(s)* to **~Input**.



**NOTE** Individual terminal parameters override port parameters. If you would now return to the Levels page of the port, the Parameter Editor would display <no value> in the *Analyzed Input(s)* field.

- 3 Close the Parameter Editor.

Now you can make the physical connections:

Use two SMA cables. Connect the normal output OUT of the generator to the normal input IN of the first analyzer. Connect the inverted output \OUT\ of the generator to the inverted input \IN\ of the second analyzer.

**NOTE** With this setup, the first analyzer will receive the normal signal at its normal input. The second analyzer will receive the inverted signal at its complementary input. The result is that both analyzers will receive the same data provided by the generator.



**Save the setting** It is good practice to save the setting as soon as major changes have been made. We will keep the setting “MEMO\_1A” and create a new setting for this example:

- 1 Open the File menu and choose *Save Setting As ...*
- 2 Save the setting under the new name **MEMO\_1B**.

By loading this setting, you can always return to the present status.

## Comparing and Acquiring Data Around Error

This test allows you to compare the incoming data in real-time with expected data. The received data is stored from the beginning. If the analyzer memory is full, the oldest data is overwritten.

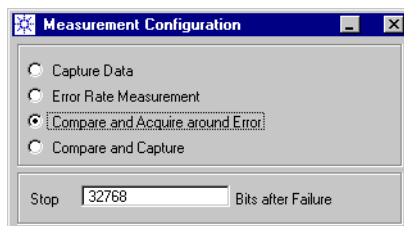
If an error occurs, the test continues for a specified number of vectors and then finishes. All errors that occur between the first error and the end of the test are also stored.

### Setting the Measurement Mode

The measurement mode has an impact on the segments that can be used in the test sequence. For example, you need to specify expected data before running this test, but you cannot specify expected data if you have enabled *Capture Data*.



- 1 Click the Measurement Configuration button.
- 2 Enable *Compare and Acquire around Error*.



By default, the number of *Bits after Failure* is 32768.

- 3 Close the Measurement Configuration window.

**NOTE** You generally use *Compare and Acquire around Error* if the length of the sequence exceeds the memory capacity of the analyzers. This is always the case if a block with data is infinitely looped.

Compare and Acquire around Error allows you to catch sporadic errors and to investigate their history and consequences.

## Checking the Test Sequence

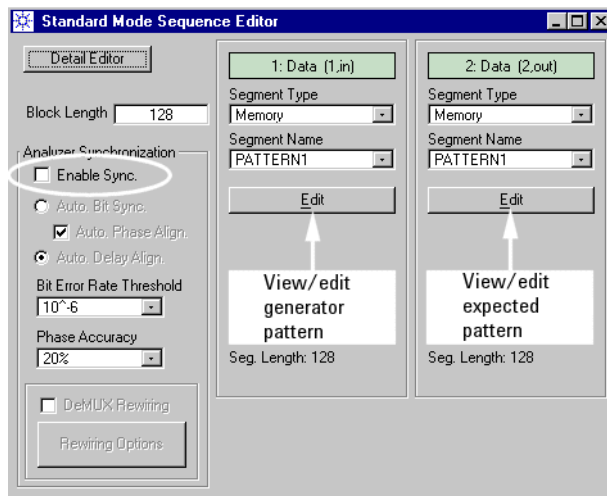
To check the test sequence:



1 Click the Sequence Editor button.

This opens the Standard Mode Sequence Editor.

2 Ensure that automatic *Analyzer Synchronization* is **disabled**.



The port identifiers tell you that port 1—the DUT input port—has one terminal whereas port 2—the output port—has two. However, the same segment is assigned to both ports.

- 3 Click the *Edit* button of the DUT input port.  
This shows you the pattern to be generated.

	Data 0	Data 1
0	1	0
1	0	0
2	0	0
3	0	0
4	1	1
5	0	1
6	1	1
7	1	0
8	1	1
9	0	0
10	0	0

The pattern has two traces. Because the port holds only one terminal, only the rightmost trace will be generated.

**NOTE** A segment may have more traces than fit to the port or more vectors than fit to the length of the block. Data that does not fit into the port or the block is ignored.

However, a segment must be at least as long as the block into which it is to be inserted, and at least as wide as the port.

- 4 Click the *Edit* button of the DUT input port.  
This shows you which trace is expected by which analyzer channel.

	Data 0	Data 1
0	1	0
1	0	0
2	0	0
3	0	0
4	1	1
5	0	1
6	1	1
7	1	0
8	1	1
9	0	0
10	0	0

Note that the data expected by the two analyzers is not identical. Errors will occur on the channel “Data1”, because only the data expected by the channel “Data0” will be generated.

**NOTE** This is the reason why automatic analyzer delay adjustment will not work properly with a setup like the one used in this example.

Our sequence consists of one block with 128 vectors that is infinitely looped. Only one single mismatch between received and expected data would yield a bit error rate of 0.78 % (1 divided by 128).

Both methods for automatic analyzer synchronization, however, require bit error rates below  $10^{-4}$ .

## Running the Test

To execute the test:



- 1 Click the Run button.

This downloads the sequence and starts the generator and analyzers.



- 2 Wait until the test has finished.
- 3 Click the Stop button.

## Inspecting the Results

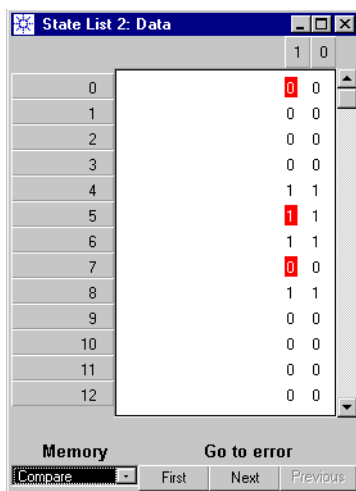
**Error State Display** Captured data and errors can be inspected with the Error State Display:



1 Click the Error State Display button.

This button is only available after the test has been **stopped**.

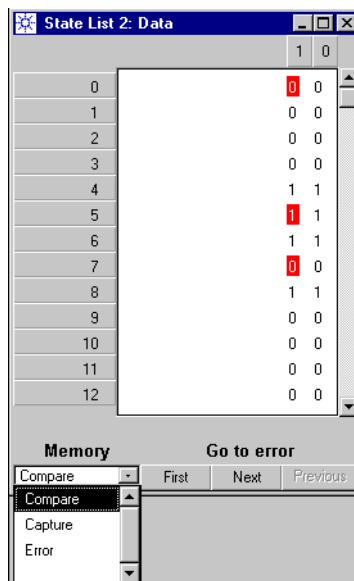
The Error State Display shows the vectors that have been captured.



Both traces are identical, as expected. The second (left-hand) analyzer has found errors from the beginning. They are highlighted. The Error State Display is very similar to the Segment Editor. Like the Segment Editor, it has three context menus offering different options. For example, you can change the address display format, the data format, or display terminal names instead of trace numbers.

Three buttons at the bottom allow you to move from one error to the next.

## 2 Open the *Memory* menu.



Using this menu, you can display

- the results of the compare and capture operation (as shown in the figure above)
- captured data only
- errors only

For details please refer to the online Help or the *Agilent 81250 ParBERT System User Guide*.

## 3 Close the Error State Display.

# Comparing and Capturing Incoming Data

This is another test that compares the incoming data in real-time with expected data. The received data is stored, as well as any errors that have occurred.

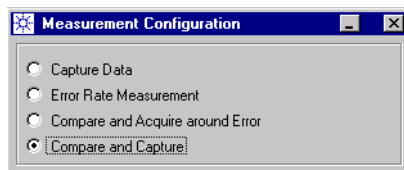
If the analyzer memory is full, the oldest data is overwritten. The test continues until the sequence expires.

## Setting the Measurement Mode

The measurement mode should always be set before opening the Sequence Editor.



- 1 Click the Measurement Configuration button.
- 2 Enable *Compare and Capture*.



- 3 Close the Measurement Configuration window.

**NOTE** You generally use *Compare and Capture* if you expect many errors or if the length of the sequence remains under the memory capacity of the analyzers.

If the incoming data exceeds the analyzer memory, the oldest data is overwritten. This may have the effect that sporadic errors cannot be identified.

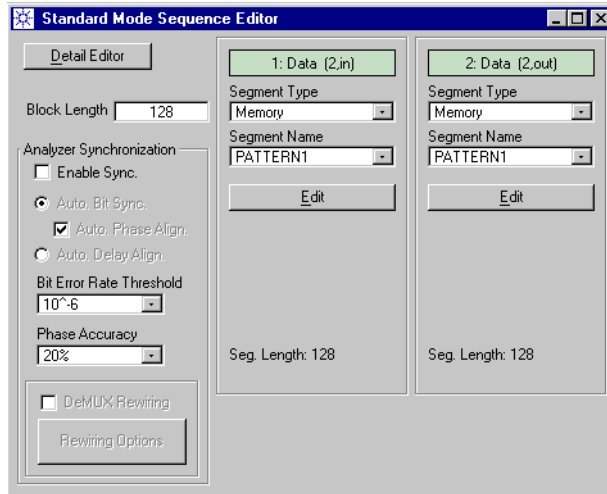
If the sequence contains an infinite loop, the test has to be stopped manually.

## Changing the Test Sequence

We will replace the infinite loop by a counted loop:

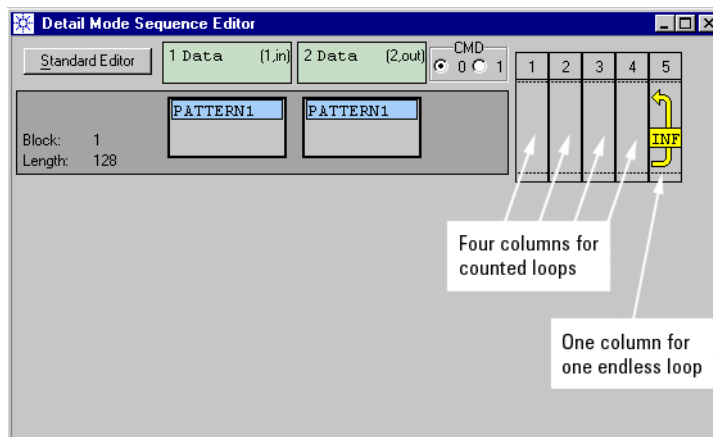


- 1 Click the Sequence Editor button.  
This opens the Standard Mode Sequence Editor.
- 2 Ensure that automatic *Analyzer Synchronization* is **disabled**.



- 3 Click the *Detail Editor* button.

This opens the Detail Mode Sequence Editor. It shows one block that is infinitely repeated.

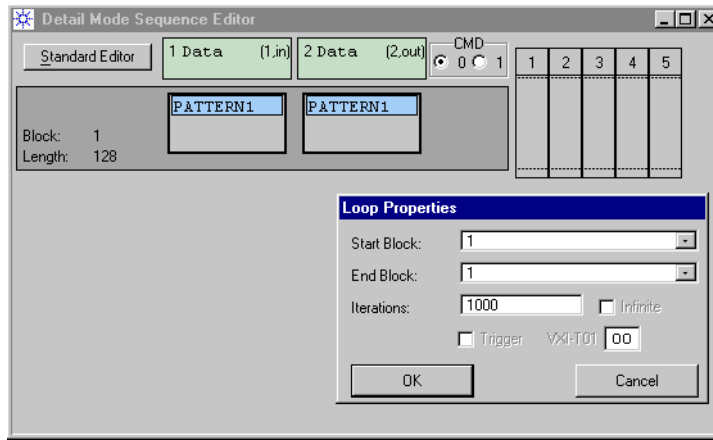


The editor provides five loop levels. The rightmost level is reserved for endless loops. The others can be used for repeating single blocks or groups of blocks a number of times.

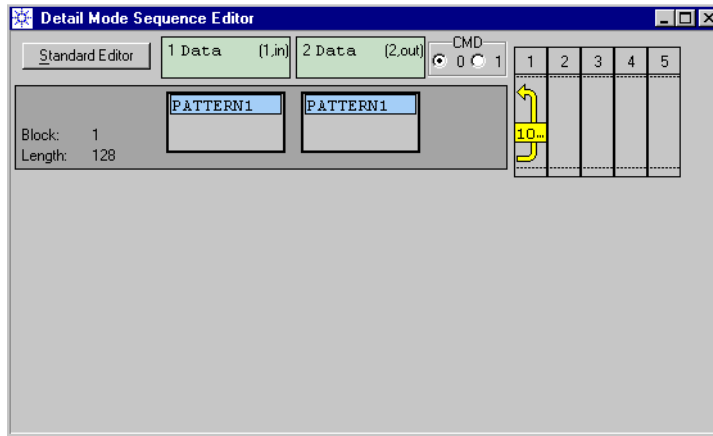
- 4 Open the context menu of the infinite loop (right-click the column) and delete the loop.



- 5 Open the context menu of loop level one (right-click the column) and choose *New Loop*. Set the number of *Iterations* to 1000.



- 6 Confirm. The system will now generate and capture 128,000 vectors.



**TIP** You can also click an empty column with the left mouse button. This inserts a loop with two repetitions. By double-clicking the loop symbol, you can then change its properties.

- 7 Close the Sequence Editor.

## Running the Test

To execute the test:



- 1 Click the Run button.

This downloads the sequence and starts the generator and analyzers.

TIP

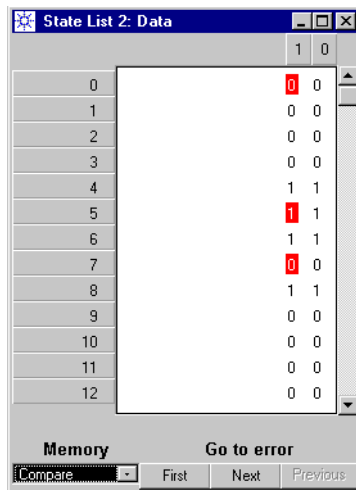
You can also click the Prepare button to download the sequence, and then actuate the Run button. If the same test sequence is executed repeatedly, this reduces the execution time of the following tests.



- 2 Wait until the test has finished.
- 3 Click the Stop button.

## Inspecting the Results

It is good practice to inspect captured data first with the Error State Display. If no errors were found, the *Go to Error* buttons would be disabled.



The result here is the same as for the previous test. This is no surprise, because already the first vector causes an error. The same error pattern repeats after every 128 vectors.

If we had kept the endless loop, the memory address of the first error would not be predictable.

TIP

If you are running a test repeatedly, you do not have to close the Error State Display. It will be updated as soon as the test has been stopped.

In this section, we will present the second tool – the Waveform Viewer. The Waveform Viewer allows you to display the results as a graphic.

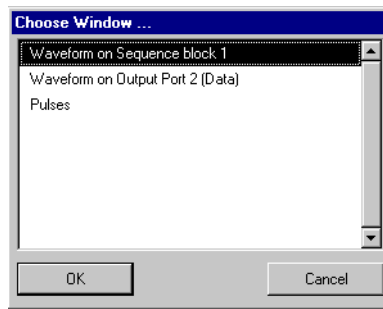
**NOTE** As its name says, the Waveform Viewer provides just another view. You can choose between **block view** (in time mode) and **output port view** (in sample mode).

Block view shows you generated and expected data. Output port view shows you captured data and test results.



1 Click the Waveform Viewer button.

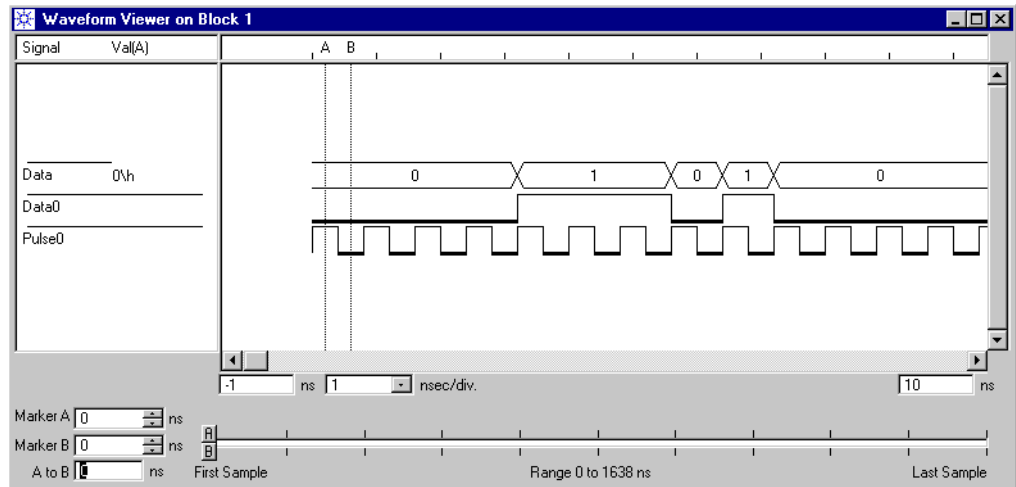
The Waveform Viewer offers information on the sequence block(s), on the DUT output port(s), and on the pulse port.



In this example, the sequence contains only one block and one output port.

Waveform Viewer in time mode 2 Choose *Waveform on Sequence block 1*.

This view shows you one block of the test sequence. Each block specifies for each data port the signals to be generated and expected.



Block information is displayed in **time mode**. This means, the horizontal scale is nanoseconds. In time mode, you can check the timing between the generators and analyzers.

The Waveform Viewer always starts with its last display configuration.

In the figure above, the DUT input port, the associated terminal, and the signal provided by the pulse generator are displayed. This is a synopsis of the generated data.

**NOTE** Port data is always shown in hex notation.

Because our DUT input port has only one terminal, the range of possible values is restricted to zero and one.

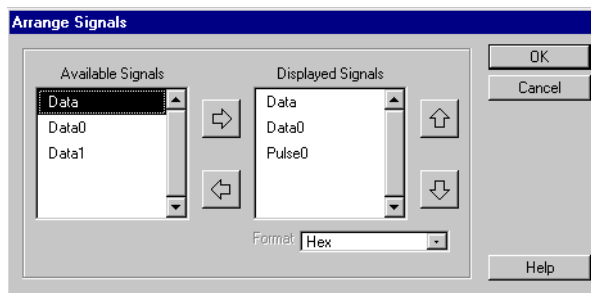
The configuration of the display can be changed at any time.

### 3 Open the context menu of the Waveform Viewer.



For assistance, please refer to the online Help and the *Agilent 81250 ParBERT System User Guide*.

### 4 Choose *Arrange Signals ...* to view additional or different data.

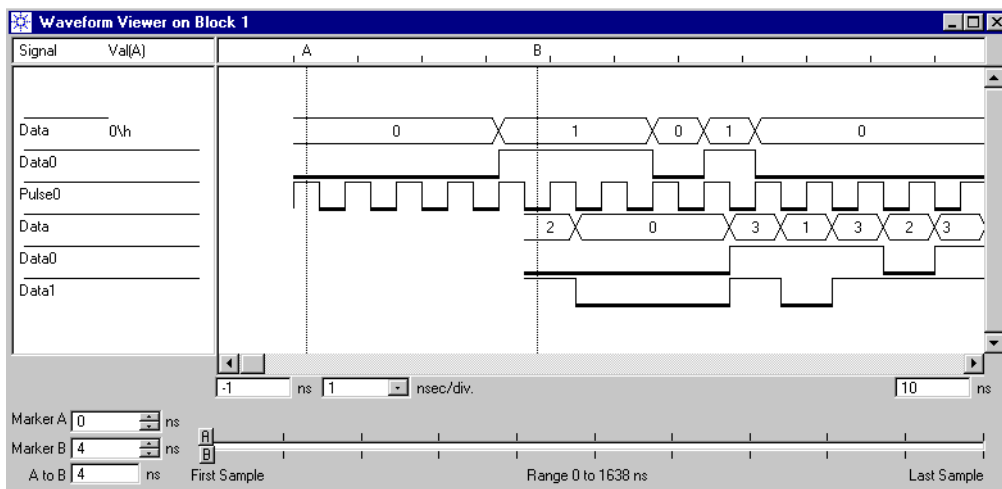


The left-hand box shows the DUT output port and its terminals.

Don't get confused by the identical port and terminal names. You can assign individual names to the ports and terminals with the Connection Editor.

More important is that these names should be as short as possible. The Waveform Viewer displays only the first 10 characters.

- 5 Enable the display of the DUT output port and its terminals (move the items from the left to the right-hand box and confirm).



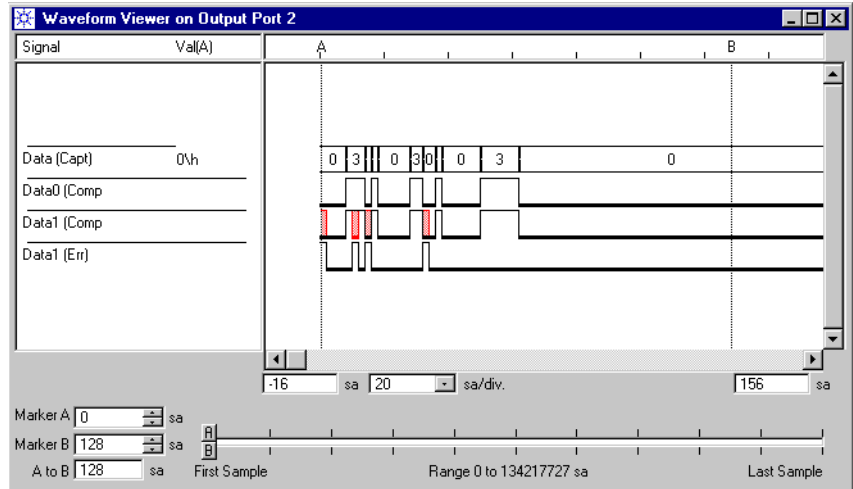
Now you can see the generated and—delayed—the expected data. The ports show the data (the vectors) in hexadecimal format. The terminals show the waveforms. You can move markers and measure delays.

**NOTE** The waveform display of sequence blocks shows the **specified** data. The information is extracted from the segments and from the timing parameters. Pure, undistorted PRBS is not displayed, because this is not stored in the memory but generated at runtime.

To view the **captured** data and **test results**, you have to use the output port view.

- Waveform Viewer in sample mode 6 Close the Waveform Viewer and re-open it to show the *Waveform on Output Port 2 (Data)*.

This display allows you to examine the errors and their environment. It provides a synopsis of the captured and errored data.

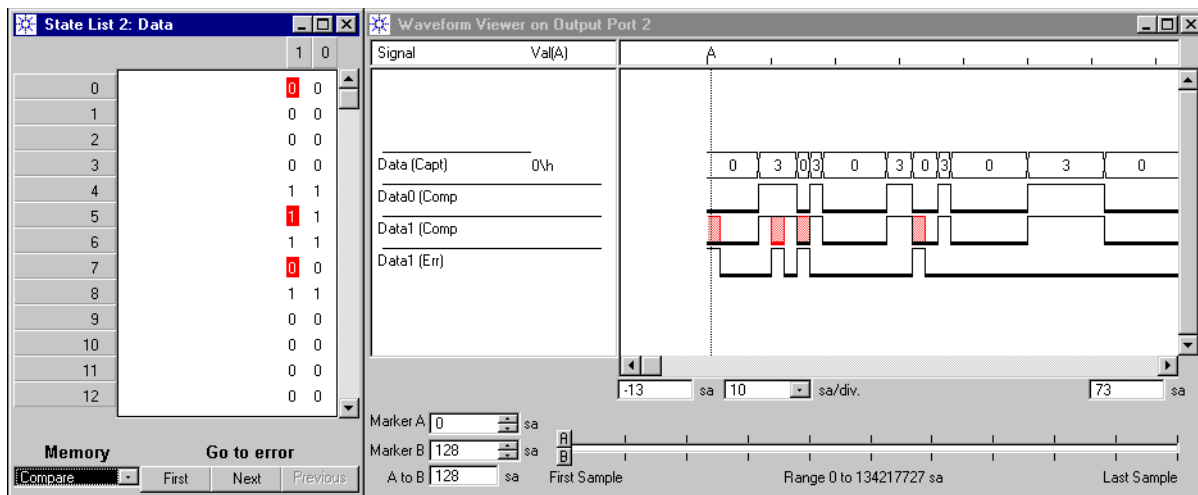


Port information is displayed in **sample mode**. This means, the horizontal scale is samples. In sample mode, you can view the data as it has been captured.

You will have to adjust this view to your needs. The context menu provides appropriate functions. If the display of comparison results is enabled, the errors are highlighted. You can also add a trace showing just the errors. This has been done in the figure above. You can also see why the port and terminal names should be as short as possible. The Waveform Viewer adds identifiers like

Capt(ured), Comp(ared), Err(ored) to the names but displays only the first 10 characters.

The informational contents is exactly the same as can be found in the Error State Display. This is illustrated in the figure below.



The captured vectors in hex notation are 4 times “0”, 3 times “3”, “0”, “3”, and so on. There are no errors for terminal “Data0”. Terminal “Data1” has had errors from the beginning. As the first bit, it has captured a low level signal while a high level was expected.

**NOTE** Samples are not the same as vectors. The Error State Display shows vectors. If you compare the Waveform Viewer with the Error State Display, you have to multiply the vector number with the port width to locate the corresponding samples in the Waveform Viewer. In the figure above, vector 5 is represented by the samples 10 and 11.

7 Close the Waveform Viewer.

Save the setting This is a good moment for updating the saved setting MEMO\_1B:



8 Click the Save Setting button.



# Capturing Incoming Data

Capturing the output data of a “golden device” is a common way to obtain the pattern that is expected from all other devices as soon as a certain input pattern is applied.

In capture mode, the analyzers capture data until the test sequence expires or their memory is full.

Captured data can be used to create a new segment. It can also be merged into an existing segment.

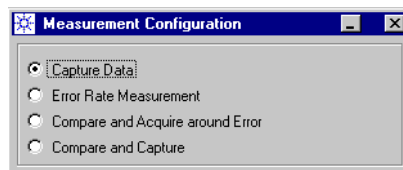
## Setting the Measurement Mode

It has already been mentioned that the measurement mode should be set before opening the Sequence Editor. This helps to avoid error messages at runtime.

For example, you cannot simply capture incoming data if you have specified expected data. If expected data is specified, the analyzers expect a compare operation.



- 1 Click the Measurement Configuration button.
- 2 Enable *Capture Data*.



- 3 Close the Measurement Configuration dialog.

**NOTE** Depending on the modules used for capturing and on the chosen Segment Resolution, you can acquire bulk data.

For example, if you were using four E4861A modules equipped with altogether eight analyzer frontends (for capturing data from an 8-bit bus) together with the highest Segment Resolution (which for these modules is 64 bits), you could capture 8 MB data.

Capturing that much data in one go is seldom useful, in particular if it is the response to a much smaller stimulating pattern that is repeated.

You should therefore know ahead of time, what you wish to achieve and how much data you need.

## Changing the Test Sequence

In order to acquire data, we have to change the test sequence:

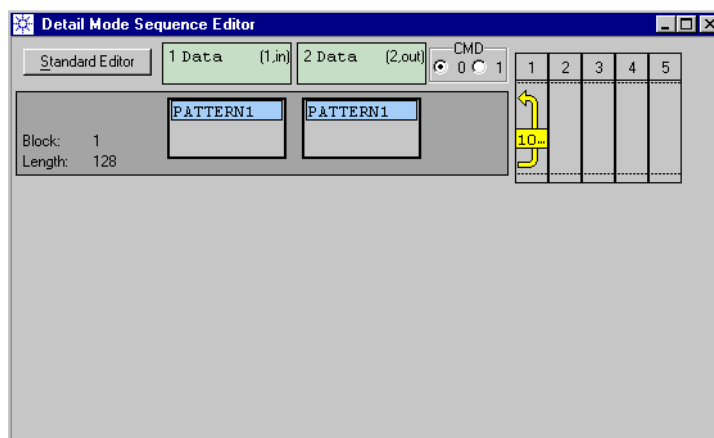


- 1 Click the Sequence Editor button.

This opens now the Detail Mode Sequence Editor.

Because we have replaced the infinite loop in the last test, the sequence can no longer be handled by the Standard Mode Sequence Editor. Therefore, the Detail Mode Sequence Editor is automatically started.

The sequence consists of one block that is looped 1,000 times.



With this setup, 128,000 vectors will be generated and captured.

If you had kept the infinite loop, the analyzers would store the incoming data until their memory is full. In our example—using E4861A modules and 32-bit Segment Resolution—4,194,304 vectors would be captured (see also the tables given in “*Clock Rates, Segment Resolution, and Memory Depth for E4832A Modules*” on page 15 and following).

In our example, considering a block of just 128 vectors that is looped, neither setup makes much sense. We need only the response to one block.

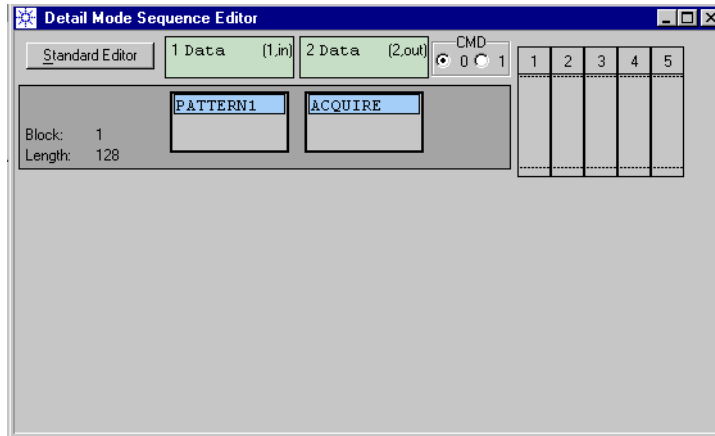
- 2 Delete the loop.

Captured data is meant for re-use. If it is the proven and correct answer to the stimulation with PATTERN1, any number of repetitions can be specified in the future test sequence.

- 3 Open the context menu of the DUT output port (right-click the segment area) and choose *Acquire*.

Due to the chosen measurement mode *Capture Data*, only two segments—*Acquire* or *Pause*—are available. These are pseudo segments that specify the operational mode of the analyzers. In

*Pause* mode, the analyzers would ignore any incoming data and do nothing.



**NOTE** If *Capture Data* is enabled, no expected data can be specified. This makes it impossible to use the automatic analyzer delay adjustment. The analyzer delays have to be set manually with the Parameter Editor.

## Running the Capture Operation

To capture the incoming data:



1 Click the Run button.



2 Wait until the operation has finished.

3 Click the Stop button.

## Inspecting and Saving the Captured Data

**Error State Display** To inspect the result, we use the Error State Display once more:



1 Click the Error State Display button.

The display shows the captured data.

	1	0
0x0	0	0
0x1	0	0
0x2	0	0
0x3	0	0
0x4	1	1
0x5	1	1
0x6	1	1
0x7	0	0
0x8	1	1
0x9	0	0
0xa	0	0
0xb	0	0
0xc	0	0

Memory      Go to error

Capture    First    Next    Previous

Both traces are identical, as expected.

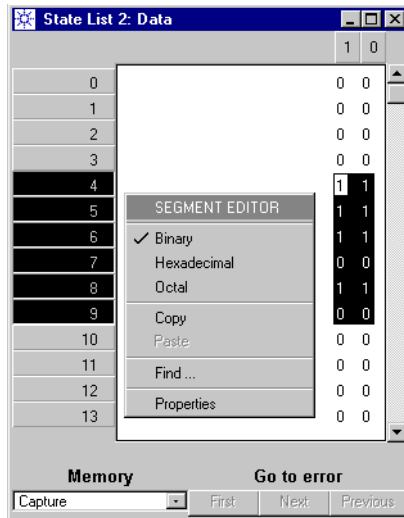
Like the Segment Editor, the Error State Display has three context menus offering different options.

2 In the area for vector operations, open the context menu and choose the *Decimal* address format.

This displays the vector numbers in integer format.

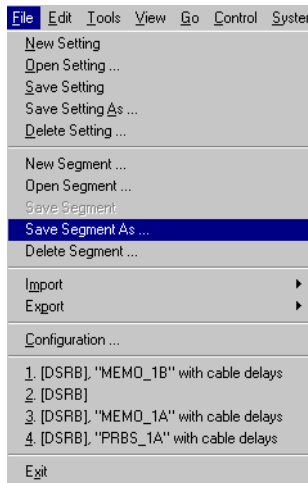
Dragging the cursor over traces or vectors highlights selected data.

- 3 In the data display area, open the context menu to see what you can do here.



For example, you can copy selected data to the clipboard and then paste it into any existing segment. For details please refer to the online Help or the *Agilent 81250 ParBERT System User Guide*.

- 4 To save the captured data as a new segment for future tests, open the File menu and choose *Save Segment as ...*.



- 5 Save the segment under the name **PATTERN2**.

In this example, PATTERN2 is the correct response to the generated data.

In further tests with the same setup, you would specify PATTERN1 for the DUT input port and PATTERN2 for the DUT output port.

If these tests should produce any errors, this can only be due to a poor cable contact.



# Using Events on a Single ParBERT System

This example demonstrates how events can be used to change the test sequence.

See:

- *“Focus of this Example” on page 88*
- *“Hardware Setup” on page 89*
- *“Test Setup” on page 90*
- *“Executing the Test” on page 100*
- *“Return to Standard Mode Sequence Editor?” on page 102*

# Focus of this Example

This example provides an introduction to the principles of event handling:

- We use one ParBERT system that generates stimulating data and analyzes the response of the device under test.
- A test sequence consisting of several blocks will be set up.
- Two events are defined.
- Reactions upon these events are assigned to the sequence blocks.
- The test is executed and the results are examined.

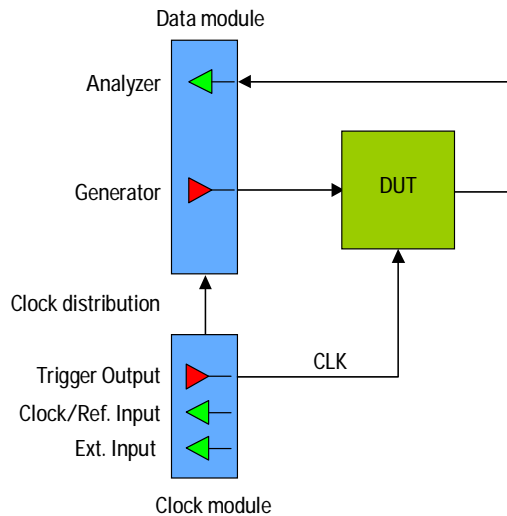
What you will learn You will learn:

- How to create a sequence with multiple blocks
- How to use distorted PRBS
- How to define events
- How to specify reactions on events
- How to force events manually
- How to verify the correct execution



# Hardware Setup

The hardware setup is the same as in the example “*BER Test on a Single System Using PRBS Data*” on page 9”:



The TRIGGER OUTPUT of the clock module is used to provide the system clock to the DUT. Therefore, we do not have to create a pulse port.

To reproduce this example without a DUT, you can use a shielded SMA cable and connect the analyzer with the generator.

Any ParBERT system that has a generator frontend and an analyzer frontend can be used.

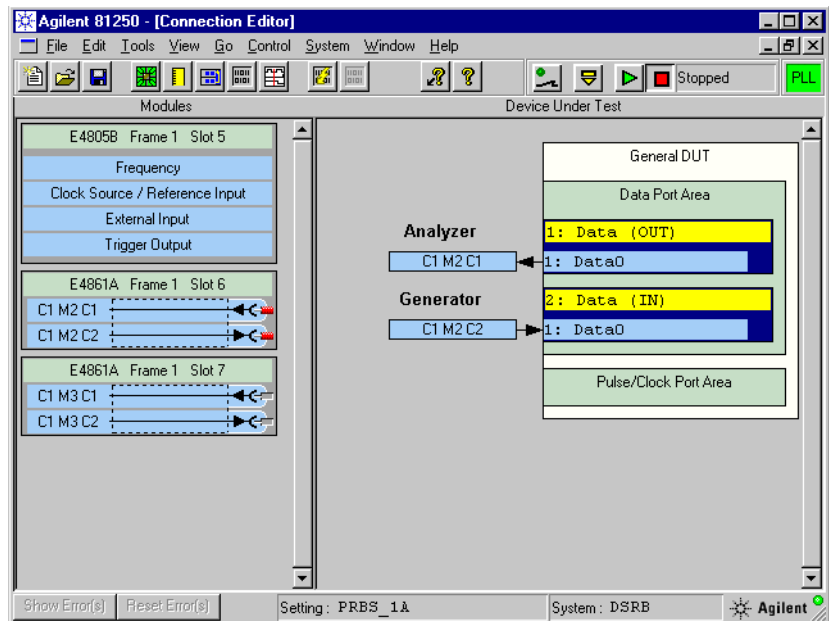
# Test Setup

Re-using a saved setting can reduce the effort for setting up a test considerably. In this example, we will re-use the setting of the previous example “*BER Test on a Single System Using PRBS Data*” on page 9.

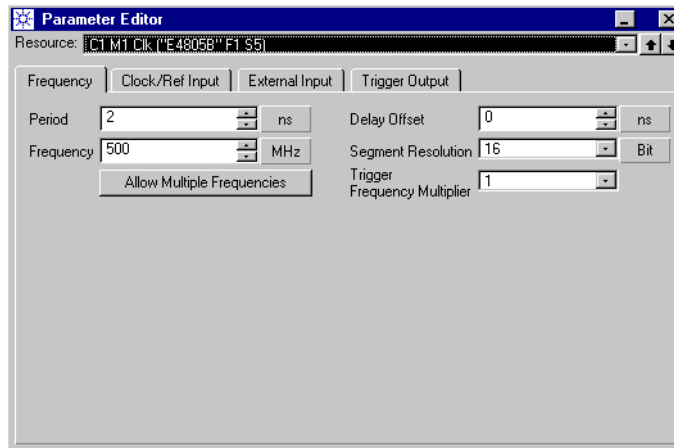


- 1 Click the Open Setting button and load the setting “PRBS\_1A”.

The Connection Editor shows the DUT with one input and one output port. Each port has one terminal.



- 2 Make sure that the system clock *Frequency* is set to 500 MHz and the *Trigger Frequency Multiplier* to 1.



Note that the *Segment Resolution* is 16. This is the minimum Segment Resolution for 500 MHz.

Because we have loaded a proven setting, there is no need to bother about signal levels, signal termination, or the measurement mode. All this has been set up and saved in the example “*BER Test on a Single System Using PRBS Data*” on page 9.

- 3 Use an SMA cable and connect the generator and analyzer physically.
- 4 Make sure that the green LEDs of the generator and analyzer are lit.

**NOTE** The frontends have built-in protection circuits that automatically disconnect a frontend if an attempt is made to operate the frontend under intolerable conditions.

If this happens, the user interface is neither informed nor updated.

You should therefore always inspect the green LEDs before running a test. They clearly indicate the physical connection status.

Once the termination conditions have been corrected, the Connectors Off/On button of the toolbar can be used to re-establish the connection.



## Creating the Test Sequence

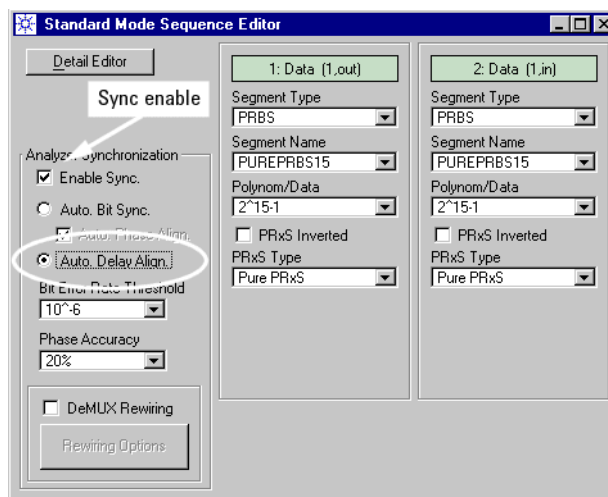
The sequence of generated and expected data can be specified with one of three available sequence editors. We will start with the Standard Mode Sequence Editor and then proceed to the Detail Mode Sequence Editor.



- 1 Click the Sequence Editor button.

The Standard Mode Sequence Editor shows one block. Pure PRBS data of polynomial  $2^{15}-1$  is generated and expected.

- 2 Enable *Analyzer Synchronization* and *Auto Delay Alignment*.

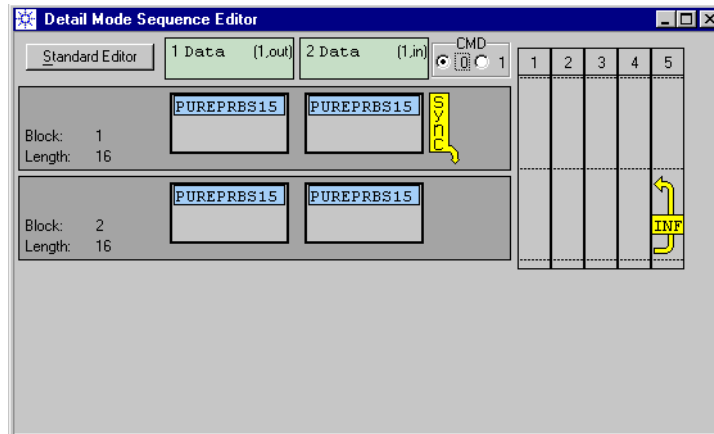


**NOTE** *Auto Bit Sync* cannot be used in this example, because we are going to add a block with distorted PRBS to the sequence. Distorted PRBS is memory data.

If a sequence contains a block with memory data, Automatic Bit Synchronization requires that the synchronization block holds also memory data. This, in turn, is only supported on a separate, analyzer-only system.

### 3 Click *Detail Editor*.

This opens the Detail Mode Sequence Editor which shows the test sequence. Up to now everything comes from the Standard Mode Sequence Editor.



After the synchronization has finished, block #2 will be executed and infinitely repeated.

You can see that the block length (which has been automatically set) corresponds to the Segment Resolution.

This is alright for PRBS data, pure as well as distorted. PRBS data continues with the next repetition of the block until the polynomial expires, and then restarts. A  $2^{15}-1$  PRBS, for example, restarts after 32767 bits.

If the blocks would reference memory-based data segments, only the first 16 vectors of the segments would be generated. The generation of memory data stored in segments restarts with every repetition.

**NOTE** Although non-pure PRBS data is memory data, it is handled differently when the block is looped.

**Assign a block label** 4 Double-click block #2 and name it **GOOD**.

Double-clicking opens the *Properties* dialog of the block. We call this block “GOOD”, because we do not expect any errors from this block—identical bit streams will be generated and expected.

**Add a block to the sequence** 5 Create a 3rd block by copying block #2 (open the context menu of block #2 and choose *Copy Block*; open the context menu once more and choose *Paste Block After*).

We will now replace the segments referenced by this block.

- Use errored PRBS **6** Double-click the generator segment of block #3 and create a new PRBS segment. Choose the *PRxS Type* “Errored PRxS”.

We have named the segment ERRPRBS15 and specified two errors. Two errors in a stream of 32767 bits ( $2^{15}-1$ ) correspond to a bit error rate of  $6.1037 \times 10^{-5}$ .

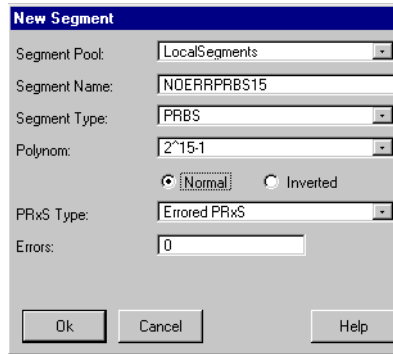
You could also choose from other distorted PRxS types. For details, please refer to the online Help and the *Agilent 81250 ParBERT System User Guide*.

**NOTE** Pure PRBS data is generated at runtime by hardware shift registers built into the data modules. Any non-pure PRBS is generated by software and downloaded to the memory of the generators or analyzers, respectively.

Due to the different signal paths with different internal delays, you cannot compare pure PRBS with distorted PRBS.

Therefore, you have to specify a distorted PRBS also for the analyzing side.

- 7 Double-click the analyzer segment of block #3 and create another new PRBS segment. Choose the *PRxS Type* “Errored PRxS” once more.

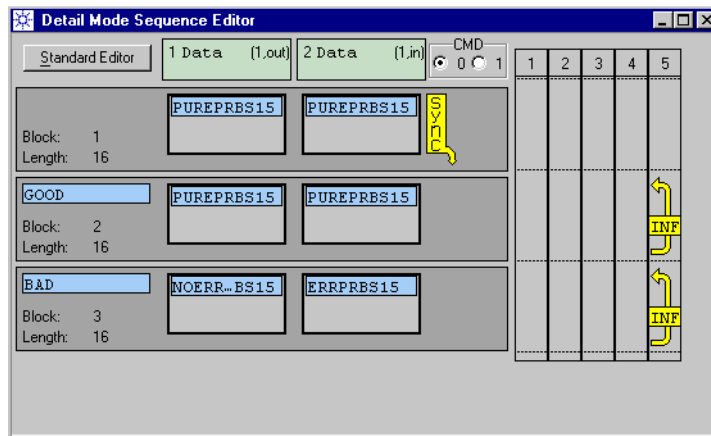


We have named the segment NOERRPRBS15 and specified zero errors.

- 8 Double-click block #3 and name it **BAD**.

We expect a bit error rate of  $6.1037 \times 10^{-5}$  when this block is executed and looped.

- Add a loop 9 Add an infinite loop to block #3 (click the rightmost loop column).



In this sequence, block #3 will never be executed, because block #2 has an infinite loop. Specifying and using events, however, makes it possible to leave a loop and to continue the test with a different block of the sequence.

- Save the setting We have made many changes. It is therefore time to save the setting. To keep the original setting “PRBS\_1A” unchanged:

- 1 Open the File menu and choose *Save Setting As ...*
- 2 Save the setting under the name **PRBS\_1B**.

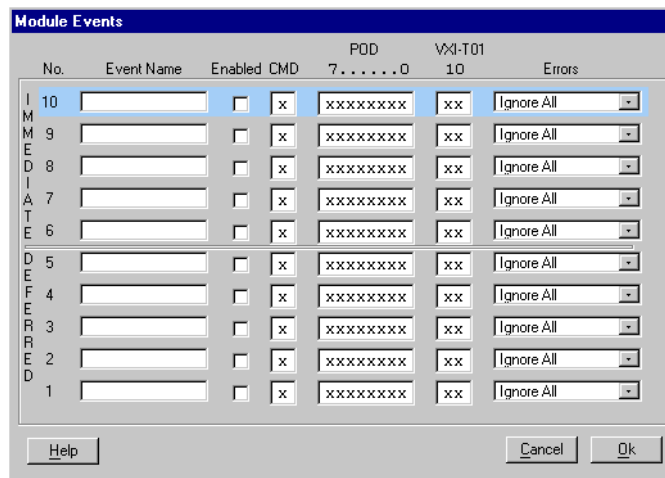
## Defining Events

To define events:

- 1 Choose *Events* from the Edit menu and select *Edit*.

**TIP** In the Detail Mode Sequence Editor or the Data/Sequence Editor, you can also open the context menu of an arbitrary block and choose *Edit Events*. Event definitions are independent of the blocks.

The Module Events window appears. This window deserves some explanations.



Every event has to have a *Name*. It becomes only effective if it is *Enabled*.

### Immediate vs. deferred

You can define up to ten events, five for immediate and five for deferred reaction.

Immediate events are immediately recognized. The reactions on deferred events occur at the end of a block.

The total response time after event recognition is predictable and must be taken into account, especially if immediate events are used. For details please refer to the *Agilent 81250 ParBERT System User Guide*.

### Event sources

Four columns allow you to define events. The items of these columns are logically ANDed. That means, the combination of whatever is activated and detected will cause an action.



**CMD** The *CMD* column refers to a manual or programmed command that causes an interrupt.

Manual interrupts can be generated from the Detail Mode Sequence Editor or the Data/Sequence Editor windows by clicking buttons. These are the events we will use in this example.

Acceptable input values in this column are x (don't care), 0, or 1.

**POD** The *POD* column refers to the trigger pod. The trigger pod—a hardware option of the master clock module—has eight sense lines that can be connected to external equipment.

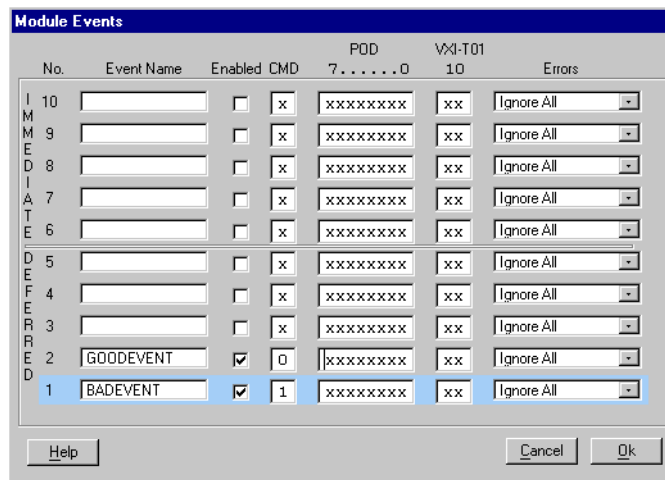
You can set the expected line status to x (don't care), 0, or 1.

**VXI-T01** The *VXI* column refers to the VXI trigger lines T0/T1. Their status may be changed by other VXI modules built into the mainframe. The default setting is 00. Acceptable inputs are x (don't care), 01, 11, 10.

Note: If you don't wish to react on their status, ensure they are set to xx. You can then set the VXI trigger lines as an answer to an event.

**Errors** The *Errors* column refers to the analyzer channels. You can open the pulldown menu and choose from the list.

2 Define the following events:



We have defined the events GOODEVENT and BADEVENT. GOODEVENT is associated with CMD0. BADEVENT occurs as soon as CMD1 is detected.

Remember that events have to have names and must be enabled to become effective.

We have defined deferred events, because we plan to change the sequence flow. The reaction on a deferred event occurs at the end of

the block. This ensures that the analyzers stay synchronized with the generators.

In addition, we have chosen the lowest priorities. That means, any additional event would override these two events.

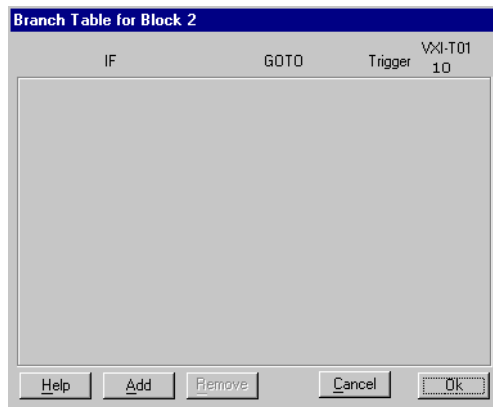
- 3 Close the Module Events window.

## Specifying Reactions Upon Events

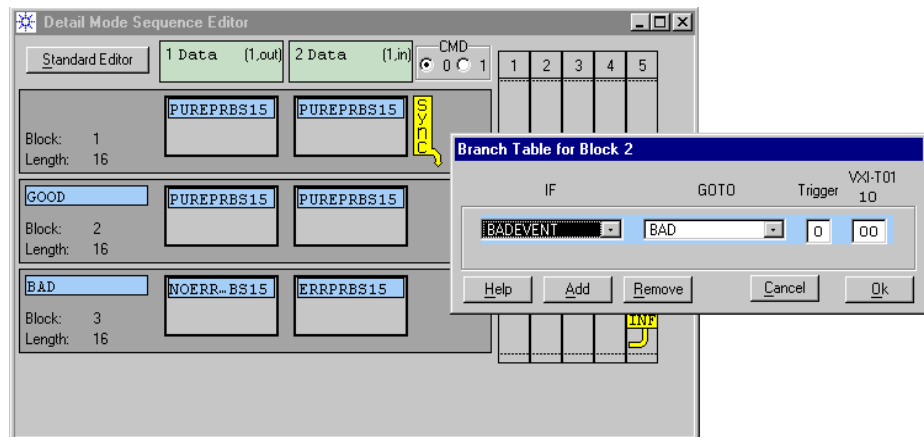
Now that we have defined events, we have to assign them to sequence blocks and to specify the desired reaction:

- 1 In the Sequence Editor, open the context menu of block #2 and choose *View/Edit Branches*.

This opens the Branch Table which starts with an empty window.



- 2 Click *Add*. Choose from the available *IF* conditions (events) and *GOTO* block labels. In this example, we wish to go to the BAD block if the event BADEVENT occurs.

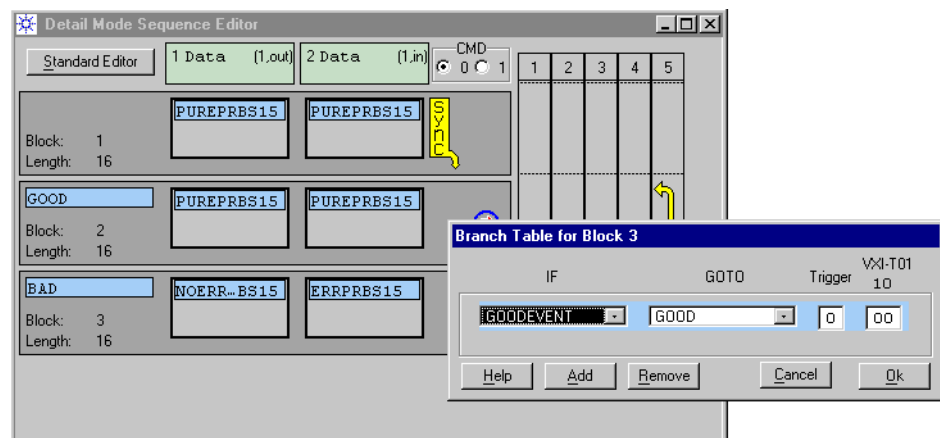


**NOTE** The reaction upon an event is specified in three columns:

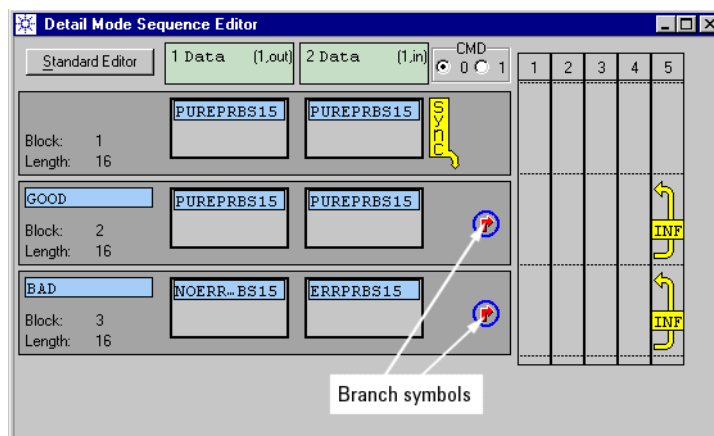
- *GOTO*: Go to a certain block of the sequence. This block must have a name (label).
- *Trigger*: Issue a trigger pulse at the TRIGGER OUTPUT of the master clock module. The TRIGGER OUTPUT must be in *Sequencer* mode.
- *VXI-T01*: Set the VXI trigger lines of the mainframe to a certain status. If you set the trigger lines upon an event, you must not use them for event recognition.

You can combine these reactions for one event. You can also specify the same or different reactions for additional events.

- 3 Repeat the steps 1 and 2 for block #3. Here, we wish to go to the GOOD block if the event GOODEVENT occurs.



The Sequence Editor shows that branches have been inserted:



**Do not** close the Sequence Editor. It has the buttons to force events manually.

# Executing the Test

To run the test:



- 1 Open the Bit Error Rate Display.

This display is only available, if the measurement mode is set to Error Rate Measurement. It is recommended to open it before starting the test.



- 2 Click the Run button.

After the synchronization process has finished, sequence block #2 is executed. The Bit Error Rate Display is updated approximately every second. Observe the *Actual Bit Error Rate*. It is zero.

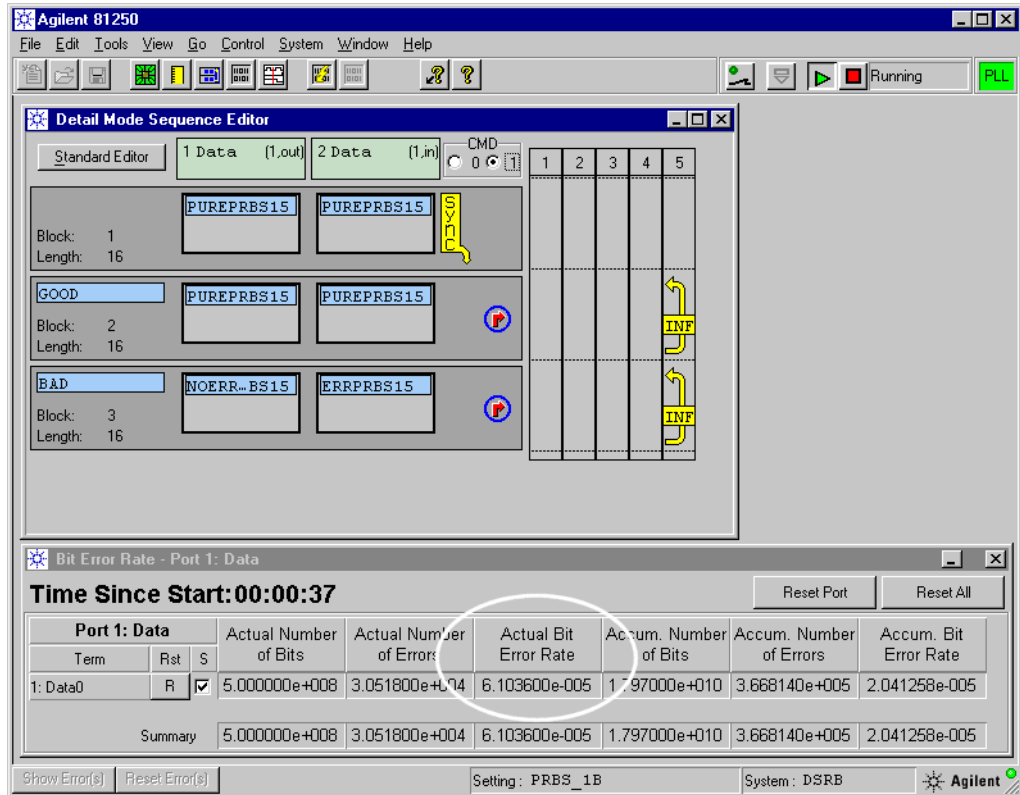
The screenshot shows the Agilent 81250 software interface. The top window is the 'Detail Mode Sequence Editor' with three blocks: 'Block 1' (GOOD), 'Block 2' (GOOD), and 'Block 3' (BAD). Each block contains two 'PUREPRBS15' or 'NOERR...BS15' and 'ERRRPRBS15' components. A 'CMD' column is visible with '0' and '1' values. A callout box points to the 'CMD0/1' buttons in the sequence editor, labeled 'Event command buttons CMD0/1'. The bottom window is the 'Bit Error Rate - Port 1: Data' display, showing 'Time Since Start: 00:00:09' and a table of error statistics.

Port 1: Data			Actual Number of Bits	Actual Number of Errors	Actual Bit Error Rate	Accum. Number of Bits	Accum. Number of Errors	Accum. Bit Error Rate
Term	Rst	S	5.000000e+008	0.000000e+000	0.000000e+000	3.980000e+009	0.000000e+000	0.000000e+000
1: Data0	R	<input checked="" type="checkbox"/>						
Summary			5.000000e+008	0.000000e+000	0.000000e+000	3.980000e+009	0.000000e+000	0.000000e+000

If the system could not synchronize, the bit error rate counters would be disabled and display just empty fields.

In the Detail Mode Sequence Editor you can see that the event command button *CMD0* is enabled by default.

- 3 In the Sequence Editor and while the test is running, click *CMD1*. This generates the event *BADEVENT*. Observe the *Actual Bit Error Rate*.



This event has changed the sequence flow. The repetition of block #2 has been aborted and block #3 is now executed. The *Actual Bit Error Rate* is as expected, toggling etween  $6.1036 \times 10^{-5}$  and  $6.1038 \times 10^{-5}$ .

- 4 You can now click *CMD0*. This generates the event *GOODEVENT* and returns to block #2. The *Actual Bit Error Rate* will return to zero.



- 5 Finally click the Stop button to terminate the test.

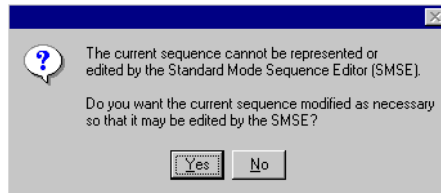
**NOTE** We have used the command events *CMD0/1* in this example, because they can be generated from the user interface or a test program—without any external equipment.

You can also react if one of the analyzers has detected an error, or if an external instrument has applied a certain pattern to the trigger pod or changed the status of the VXI trigger lines.

## Return to Standard Mode Sequence Editor?

We have created a sequence which the Standard Mode Sequence Editor cannot handle. The Standard Mode Sequence Editor is meant for setting up simple BER tests as quickly and efficiently as possible. It considers only one block and no events.

If you now in the Detail Mode Sequence Editor click the *Standard Editor* button, the following question appears:



If you have created a sophisticated sequence like the one in this example:

- 1 Click *No*.

The Standard Mode Sequence Editor would keep only one block to be used for the test and duplicated for the automatic analyzer synchronization. All additional blocks and branch tables would be lost.



- 2 Click the Save Setting button.

Now you can do whatever you want. The updated setting “PRBS\_1B” is saved. It can always be recalled and used again.



Copyright Agilent Technologies 2002  
Printed in Germany April 2002



5988-6093EN

S A