Bus Analyzer
User's Manual

adaptec

HAND DRIVE = 40

SDS2-3 VERS

TOP BD IS SE

BOTT IS DIFF ← SW TOWARDS CONN
IS DIFF
AWAY IS SE

SDS-310F
SCSI Bus Analyzer
User's Manual

PN: 481560-00
Rev: B

adaptec
Development Systems Operation
691 s. milpitas boulevard • milpitas, california 95035
(408) 945-8600

Copyright

Changes

The material in this manual is for information only and is subject to change without notice. Adaptec reserves the right to make changes in the product design without reservation and without notification to its users.

FCC Regulations

This equipment is intended for industrial/commercial application. It is, therefore, excluded from FCC Class B Part 15 regulations.

Technical Assistance

Technical assistance for this product is available by calling our Development Systems Operation (DSO) hot line (Monday through Friday between 8:30 AM and 5:30 PM Pacific Coast Time):

DSO Hot Line: (408) 945-2527

Equipment Returns

All equipment returns must be accompanied by a Return Material Authorization (RMA) number. To determine if a return is necessary and to obtain an RMA number, call the SCSI hot line. Proper handling of returned equipment can only be guaranteed with an RMA number. Return all equipment in either its original factory packaging material or containers that provide equivalent protection. Mark the outside of all containers "FRAGILE INSTRUMENT ENCLOSED."

Table of Contents

List of Figures

Preface

This manual is a guide for users of Adaptec's SDS-310F SCSI Bus Analyzer. It is assumed that the reader is already familiar with basic SCSI concepts (commands and phases) and basic logic analyzer concepts (data capture and data display).

This manual contains four chapters and two appendices organized as follows:

Chapter 1, Installation, is a tutorial chapter that details proper installation of the analyzer hardware and software. Please follow the documented procedures to ensure correct and safe operation of the analyzer.

Chapter 2, Key Concepts, is a tutorial chapter on the concepts that must be understood for effective use of the analyzer. This analyzer has significant differences from traditional logic analyzers. Even if you have extensive experience with logic analyzers, you will need to study the information in this chapter to use your analyzer to full effect.

Chapter 3, Interactive Menu, provides a reference for the analyzer's interactive user interface. The menu interface is designed to be intuitive, including context sensitive on-line help. In addition to providing interactive menu interface details, this chapter also provides important information on invoking the menu.

Chapter 4, Run Time Function Library, provides a reference for the analyzer's C Programming Language programmer's interface. This chapter describes the procedures for performing SCSI bus analysis from a 'C' program. Detailed descriptions of each function are provided.

Appendix A, Troubleshooting, provides a guide to correcting common problems with the analyzer.

Appendix B, Specifications, contains detailed electrical, timing, environmental, and physical specifications of the analyzer.

Chapter 1

Installation

In addition to this manual, the SDS-310F SCSI bus analyzer comes standard with the following components:

1. **Analyzer printed circuit board (PCB).** A PC/AT 16-bit form-factor PCB that performs single-ended SCSI bus analysis. If you purchased your analyzer in a turnkey configuration, this PCB will be shipped to you already installed in a computer. If you purchased your analyzer in an integration kit configuration, you must install this PCB in your own computer.

2. **Terminating resistors.** Four 10-pin single in-line package (SIP) resistor packs that are used to terminate the analyzer PCB's 68-pin connector for applications where the analyzer is connected to a physical end of the SCSI bus cable.

3. **SCSI 'P' connector translator.** A translator that is used to connect an 8-bit 50-pin single-ended SCSI 'A'-type bus to the analyzer PCB's standard 68-pin SCSI 'P' connector.

4. **External probe pod.** A pod with:

   - Four input wires that can be connected to non-SCSI signals for acquisition and triggering by the analyzer.
   - One output wire that will assert itself whenever the analyzer triggers.
   - One wire to attach to ground.
   - One 10-pin flat ribbon cable to connect the pod to the analyzer.

5. **Internal Connection Cable.** A 50-pin flat ribbon cable used to connect the analyzer PCB to a test adapter or differential adapter PCB inside the computer.

6. **Interactive menu diskettes.** Found in the back of this manual.

7. **Run time function library diskettes.** Found in the back of this manual.

These components together with a PC-compatible computer are everything you need to perform passive 20 nanosecond resolution, 8- or 16-bit single-ended SCSI bus analysis.

You can also perform passive 8- or 16-bit differential SCSI bus analysis with the above components if you add an optional SDS-311F SCSI differential adapter. You can also perform concurrent 8-bit SCSI bus emulations with the above components if you add an optional SDS-3F SCSI test adapter.

NOTE

For more information on Adaptec's SDS-311F differential adapter and SDS-3F test adapter product offerings, contact your local Adaptec sales representative.

Installing the Hardware

There are several configurations of the analyzer hardware depending on:

1. The type of SCSI bus you are analyzing.
2. If you want to perform simultaneous SCSI bus emulations.
3. If you want to include non-SCSI signals in your analysis.

The installation of each configuration consists of three steps:

1. Installing the appropriate PCB components inside the computer.
2. Connecting the SCSI bus to the computer.
3. Connecting the external probe pod to the analyzer (Optional).

Installing PCB Components

The SDS-310F SCSI bus analyzer is shipped in one of two basic configurations: turnkey or integration kit. In the turnkey configuration, all PCBs have been shipped to you already installed in the Adaptec-supplied computer. If you purchased a turnkey configuration, you can skip ahead to the **Connecting the SCSI Bus** section of this chapter.

In the integration kit configuration, you must install the appropriate PCB(s) in your computer by following the procedures detailed in the following subsections.

The ***minimum*** computer requirements are:

- PC/AT-compatible.
- CGA video controller.
- i286 CPU.
- 640 KBytes RAM.
- 1.2 MByte 5 1/4" or 1.44 MByte 3 1/2" floppy drive.
- Hard drive with at least 1 MByte available.
- 1, 2, or 3 free backplane slots (two 16-bit, one 8- or 16-bit), depending on your configuration.

- One of the following hexadecimal PC I/O address ranges available:
  310-31F, 330-33F, 360-36F, 380-38F.
- DOS 3.0.

After you have installed the appropriate PCBs inside your computer with the appropriate I/O addressing, SCSI bus termination, and internal connections, you may close your computer. All further configuration will be performed outside of the computer. Whatever type of SCSI bus you are analyzing, you can configure the analyzer, differential adapter, and test adapter for that bus without re-opening the computer, as explained in the **Connecting the SCSI Bus** section of this chapter.

Installing the Analyzer PCB

1. If you are installing the analyzer PCB as the only hardware option in your computer (no test adapter, differential adapter, or other vendors' PCBs), skip ahead to Step 2.

   If you are installing other hardware options, check their I/O addressing requirements. If there is a conflict with the analyzer PCB's pre-set address range of 310-31F, you must resolve this conflict before installing the analyzer PCB. To change the address of the analyzer PCB, select the desired address with dual in-line package (DIP) switches 1 and 2 on the analyzer PCB as follows:

1	2	I/O Address Range
OFF	ON	310-31F *
ON	OFF	330-33F
OFF	OFF	360-36F
ON	ON	380-38F

*Factory setting.

   DIP switches 3 and 4 are reserved and should be left OFF.

2. If you are connecting the analyzer to a physical end of the SCSI bus cable, install the four 10-pin terminating resistor SIP's into sockets RP2, RP3, RP4, and RP5. If you are connecting the analyzer to the middle of the cable, remove them. ***Never remove the 6-pin resistor SIP in socket RP1.***

NOTE

Try to be consistent with your termination requirements. Each time you change termination requirements (by moving the analyzer to or from a physical end of the SCSI cable) you will need to re-open your computer to install or remove these resistors.

3. Install the analyzer PCB in your computer's backplane. See your computer's documentation for instructions on installing hardware option PCBs. Make sure that you install the analyzer PCB into a PC/AT-compatible backplane slot. Also make sure that the 68-pin 'P' SCSI connector can be easily accessed from the outside of the computer. Lastly, make sure the PCB is securely attached to the computer to minimize the chance of malfunction or damage.

See Figure 1-1.

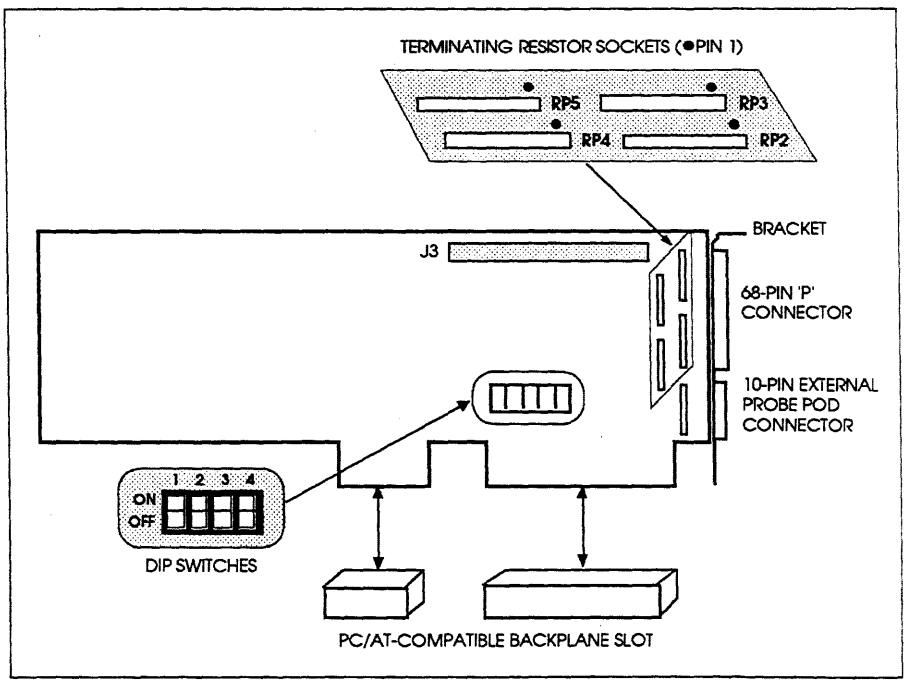

Figure 1-1.
Installing the Analyzer PCB

Adding Only a Differential Adapter PCB

To analyze a differential SCSI bus, you will need to install an SDS-311F differential adapter PCB. You should install the differential adapter PCB in a backplane slot adjacent to the analyzer PCB. The differential adapter PCB only requires a PC/XT-compatible slot, but it will also function properly in a PC/AT-compatible slot. You will need to connect the analyzer and differential adapter PCBs with a cable provided with the differential adapter. See the differential adapter user's manual for details on its installation.

See Figure 1-2.

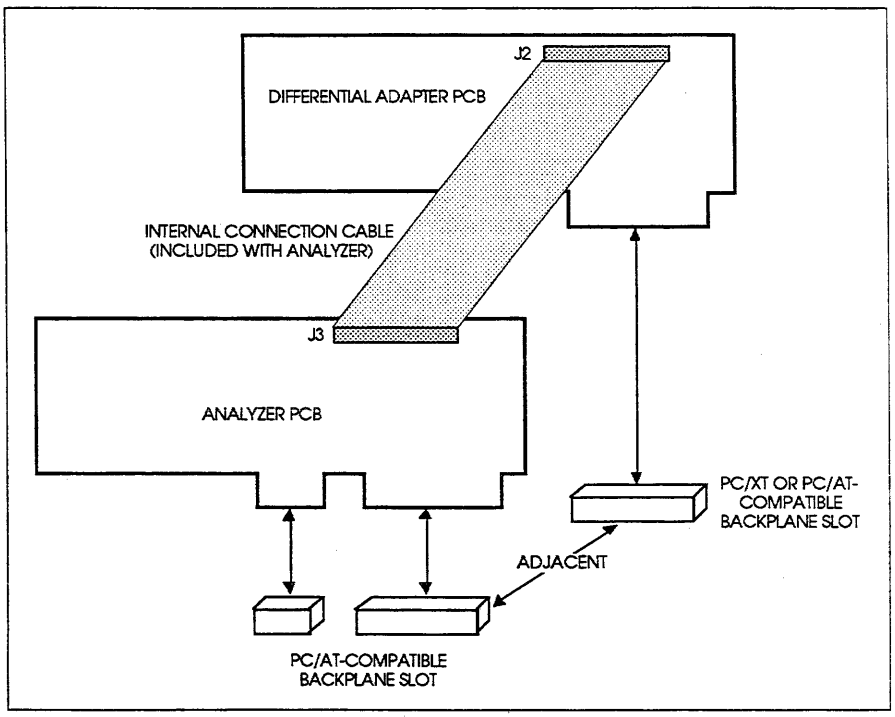

Figure 1-2.
Adding a Differential Adapter PCB

> *NOTE*
>
> *The SDS-310F will function correctly with the older model SDS-311 differential adapter. The only difference will be that differential analysis will be restricted to 8-bit 'A' cable SCSI buses. It is recommended that you purchase an SDS-311F to replace your SDS-311.*

Adding Only a Test Adapter PCB

To perform simultaneous SCSI bus emulation and analysis with the same computer, you will need to install an SDS-3F test adapter PCB. See the test adapter user's manual for details on its installation. Once the test adapter PCB is installed in the computer, there are two possible cabling configurations.

In the first configuration, the two PCBs operate independent of each other. Each connects to the SCSI bus through its own external SCSI bus connector. In this case, there are no internal connections between the two PCBs. You may install the test adapter PCB in any backplane slot, not just a slot adjacent to the analyzer PCB.

See Figure 1-3.

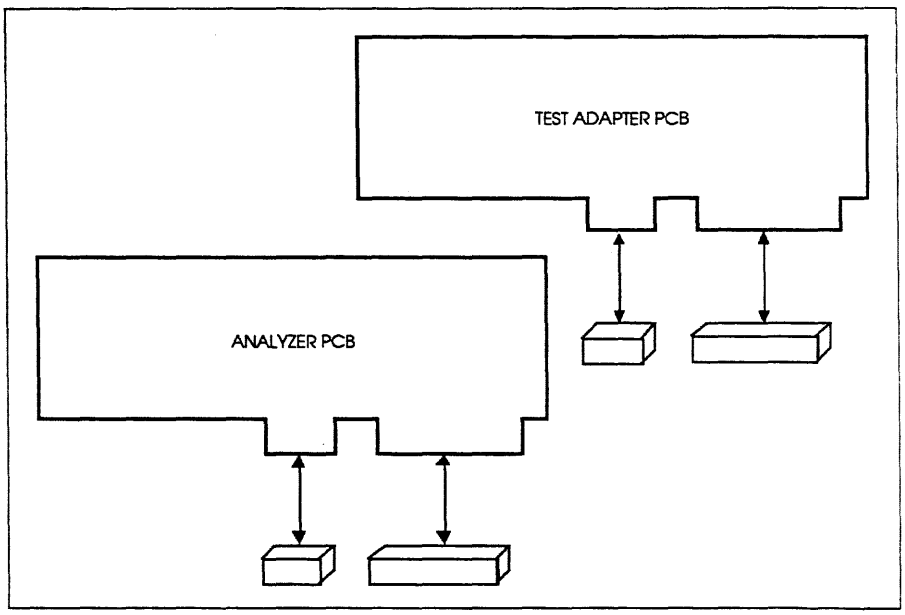

Figure 1-3.
Adding a Test Adapter PCB — First Configuration

In the second configuration, the analyzer PCB is connected internally to the test adapter PCB using the internal connection cable included with your analyzer, and the SCSI bus is connected only to the test adapter PCBs external SCSI bus connector. In this case, the test adapter PCB should be installed in a backplane slot adjacent to the analyzer PCB.

See Figure 1-4.

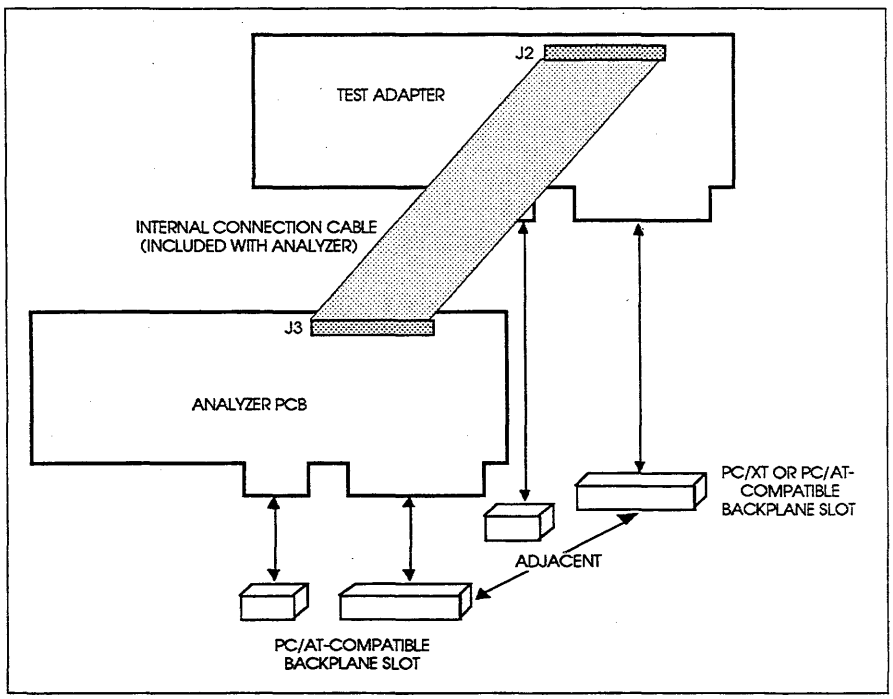

Figure 1-4.
Adding a Differential Adapter PCB — Second Configuration

The configuration you choose depends on your application. The first has the advantage that it allows analysis of 8 or 16-bit SCSI buses, even though the test adapter is limited to 8-bit emulations. The second configuration has the advantage that is only requires one connection to the SCSI bus. If all analysis will be performed on 8-bit SCSI buses, the second configuration is recommended.

Adding Both Differential Adapter and Test Adapter PCBs

For simultaneous analysis and emulation of differential SCSI buses, you will need to install both an SDS-3F test adapter PCB and an SDS-311F differential adapter PCB.

1. Install the differential adapter PCB as described in the differential adapter user's manual. Use a backplane slot adjacent to the analyzer PCB.

2. Install the test adapter PCB as described in the test adapter user's manual. Use a backplane slot adjacent to the differential adapter PCB.

3. Connect the differential adapter PCB to the test adapter and analyzer PCBs with the cables provided with your differential adapter.

See Figure 1-5.

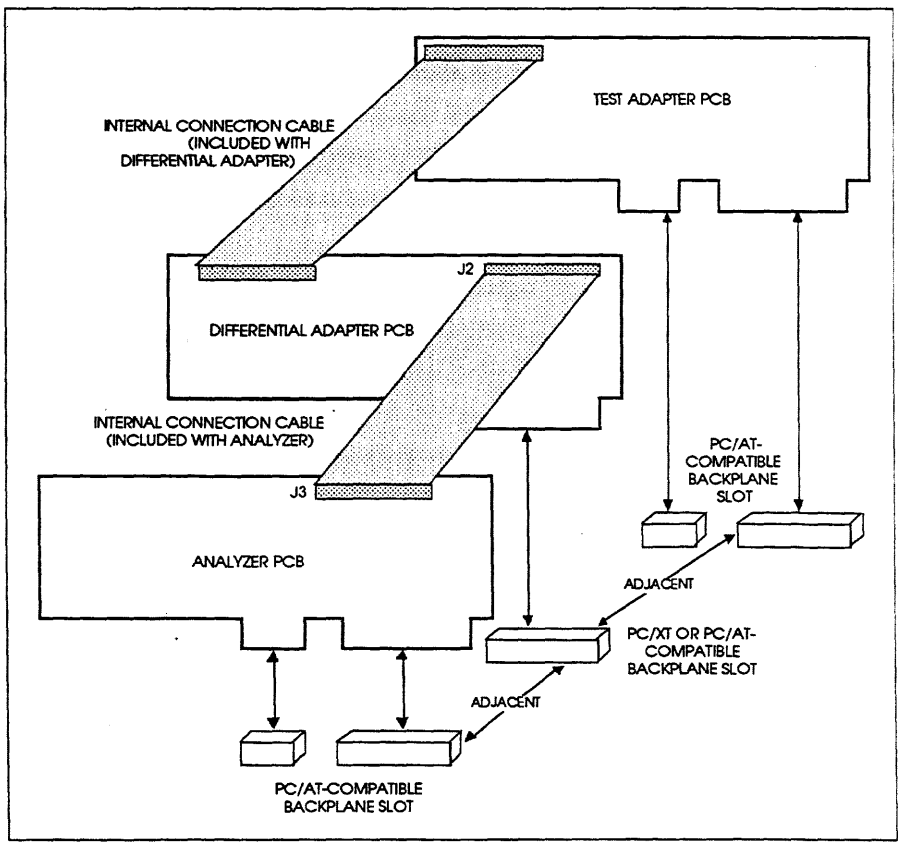

Figure 1-5.
Adding Both Differential Adapter and Test Adapter PCBs

Connecting the SCSI Bus

> *NOTE*
>
> *Before following the procedures in this section, make sure the appropriate PCBs have been installed in your computer. See the* ***Installing PCB Components*** *section of this chapter for details.*

There are 13 possible configurations when connecting a SCSI bus to your computer depending on the PCBs that are installed and the type of SCSI bus that is being connected. To find your configuration, use the following table:

AnalyzerPCB Installed?	Diff. Adapter PCB Installed?	Test Adapter PCB Installed?	SCSI Bus Width	SCSI Bus Type	Config. Number
Yes	No	No	8	SE[1]	1
Yes	No	No	16	SE	2
Yes	Yes	No	8	SE	3
Yes	Yes	No	16	SE	4
Yes	Yes	No	8	Diff	5
Yes	Yes	No	16	Diff	6[2]
Yes	No	Yes[3]	8	SE	7
Yes	No	Yes[3]	16	SE	8
Yes	No	Yes[4]	8	SE	9
Yes	Yes	Yes	8	SE	10
Yes	Yes	Yes	16	SE	11
Yes	Yes	Yes	8	Diff	12
Yes	Yes	Yes	16	Diff	13[2]

[1]SE = Single-ended.
[2]Not supported with SDS-311. Must use SDS-311F.
[3]Test adapter PCB and analyzer PCB not connected internally.
[4]Test adapter PCB connected internally to analyzer PCB.

The procedures for installing these configurations are detailed in the following subsections.

> *NOTE*
>
> *For all 13 configurations, it is assumed that proper termination is already set for your SCSI connector. See Step 2 in the* ***Installing the Analyzer PCB*** *section of this chapter for details.*

Only the Analyzer PCB Installed

Configuration 1: 8-bit Single-Ended 'A' Cable SCSI Analysis

1. On the SCSI 'P' connector translator, turn switch 5 'ON'. If the analyzer PCB's 68-pin 'P' connector is terminated, turn switches 1-4 'ON'; otherwise, turn them 'OFF'.

2. Plug the translator into the analyzer PCB's 68-pin 'P' connector.

3. Plug the SCSI bus cable into the translator's 50-pin 'A' connector.

See Figure 1-6.

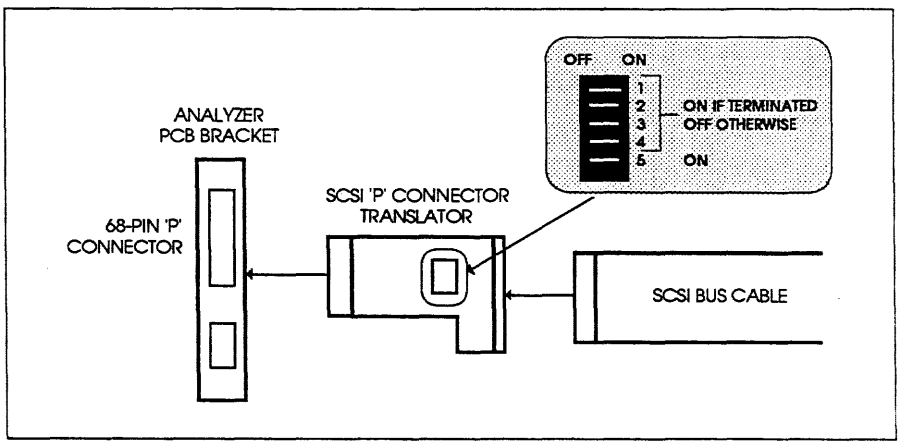

Figure 1-6. Configuration 1.

Configuration 2: 16-bit Single-Ended 'P' Cable SCSI Analysis

Plug the SCSI bus cable into the analyzer PCB's 68-pin 'P' connector.

See Figure 1-7.

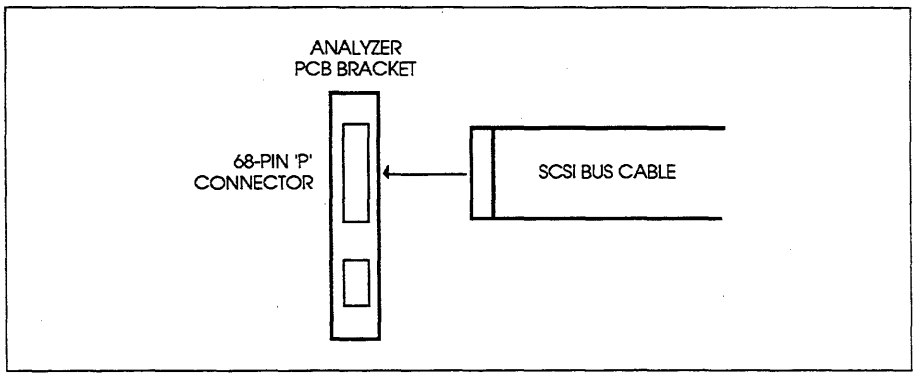

Figure 1-7. Configuration 2.

Only the Analyzer and Differential Adapter PCBs Installed

Configuration 3: 8-bit Single-Ended 'A' Cable SCSI Analysis

1. Unplug any SCSI bus cable or connector translator from the differential adapter PCB's 68-pin 'P' connector.

2. Disable the differential adapter by choosing the appropriate switch setting on the differential adapter PCB's bracket. See the differential adapter user's manual for this switch setting.

3. On the SCSI 'P' connector translator, turn switch 5 'ON'. If the analyzer PCB's 68-pin 'P' connector is terminated, turn switches 1-4 'ON'; otherwise, turn them 'OFF'.

4. Plug the translator into the analyzer PCB's 68-pin 'P' connector.

5. Plug the SCSI bus cable into the translator's 50-pin 'A' connector.

See Figure 1-8.

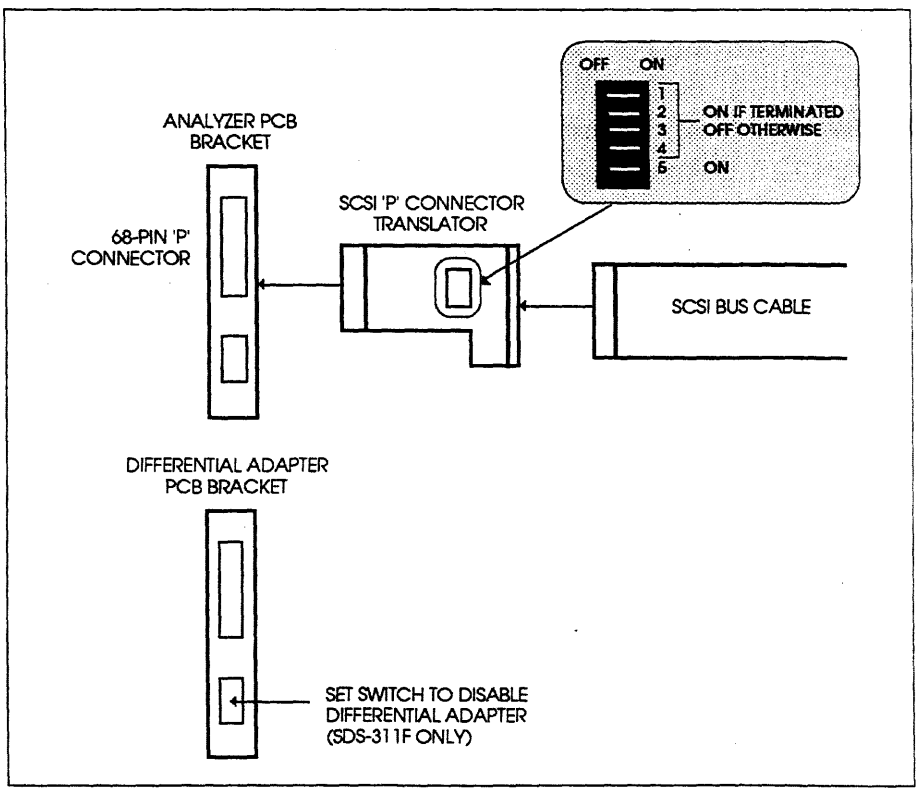

Figure 1-8. Configuration 3.

Configuration 4: 16-bit Single-Ended 'P' Cable SCSI Analysis

1. Unplug any SCSI bus cable or connector translator from the differential adapter PCB's 68-pin 'P' connector.

2. Disable the differential adapter by choosing the appropriate switch setting on the differential adapter PCB's bracket. See the differential adapter user's manual for this switch setting.

3. Plug the SCSI bus cable into the analyzer PCB's 68-pin 'P' connector.

See Figure 1-9.

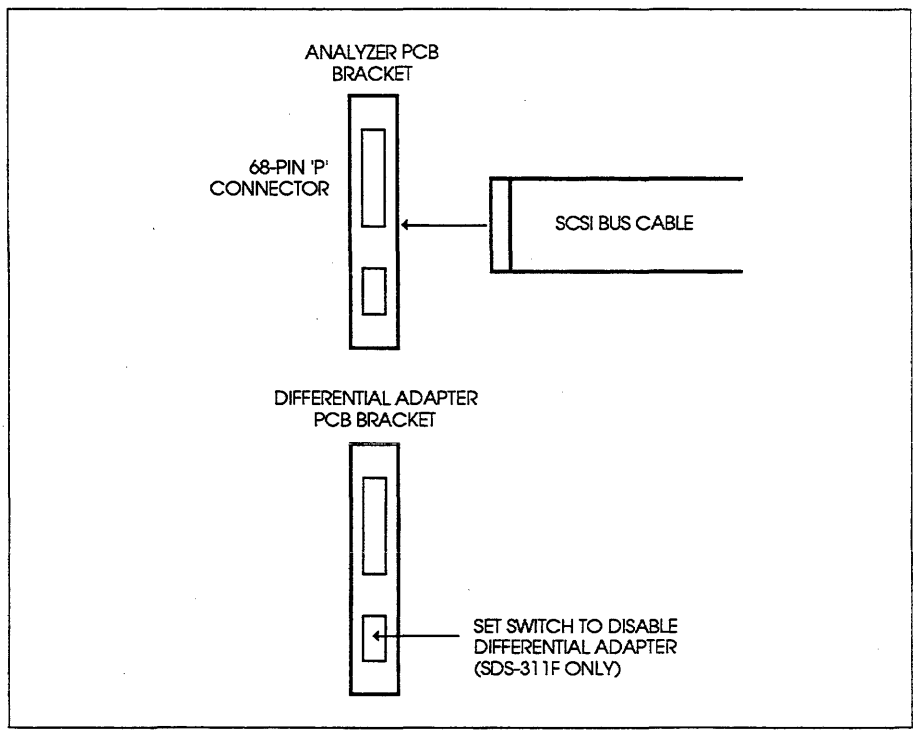

Figure 1-9. Configuration 4.

Configuration 5: 8-bit Differential 'A' Cable SCSI Analysis

1. Unplug any SCSI bus cable or connector translator from the analyzer PCB's 68-pin 'P' connector.

2. Enable the differential adapter by choosing the appropriate switch setting on the differential adapter PCB's bracket. See the differential adapter user's manual for this switch setting.

3. On the SCSI 'P' connector translator, turn switch 5 'OFF'. If the differential adapter PCB's 68-pin 'P' connector is terminated, turn switches 1-4 'ON'; otherwise, turn them 'OFF'.

4. Plug the translator into the differential adapter PCB's 68-pin 'P' connector.

5. Plug the SCSI bus cable into the translator's 50-pin 'A' connector.

See Figure 1-10.

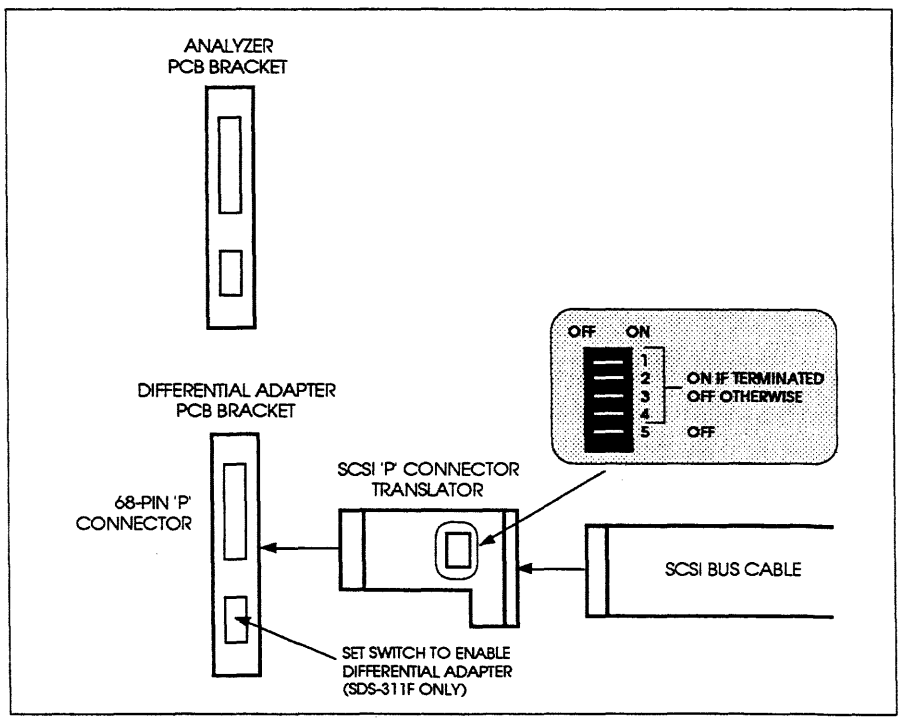

Figure 1-10. Configuration 5.

NOTE

If you are using an SDS-311, the 50-pin SCSI 'A' cable will plug directly into the differential adapter PCB. Do not use a 'P' connector translator.

Configuration 6: 16-bit Differential 'P' Cable SCSI Analysis

1. Unplug any SCSI bus cable or connector translator from the analyzer PCB's 68-pin 'P' connector.

2. Enable the differential adapter by choosing the appropriate switch setting on the differential adapter PCB's bracket. See the differential adapter user's manual for this switch setting.

3. Plug the SCSI bus cable into the differential adapter PCB's 68-pin 'P' connector.

See Figure 1-11.

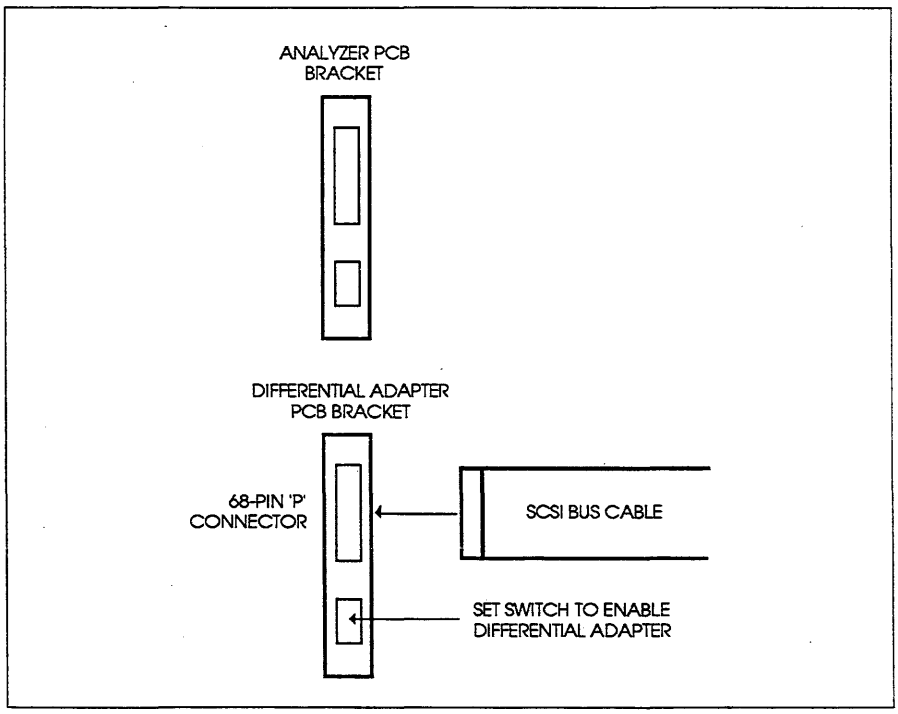

Figure 1-11. Configuration 6.

> *NOTE*
>
> *The SDS-311 does not support this configuration.*

Only the Analyzer and Test Adapter PCBs Installed

Configuration 7: 8-bit Single-Ended 'A' Cable SCSI Analysis and Optional Emulation

> *NOTE*
>
> *For simultaneous analysis and emulation, this configuration requires two 50-pin SCSI bus cable connections.*

1. On the SCSI 'P' connector translator, turn switch 5 'ON'. If the analyzer PCB's 68-pin 'P' connector is terminated, turn switches 1-4 'ON'; otherwise, turn them 'OFF'.
2. Plug the translator into the analyzer PCB's 68-pin 'P' connector.
3. Plug the SCSI bus cable into the translator's 50-pin 'A' connector.
4. Plug the SCSI bus cable into the test adapter PCB's 50-pin 'A' connector (Optional).

See Figure 1-12.

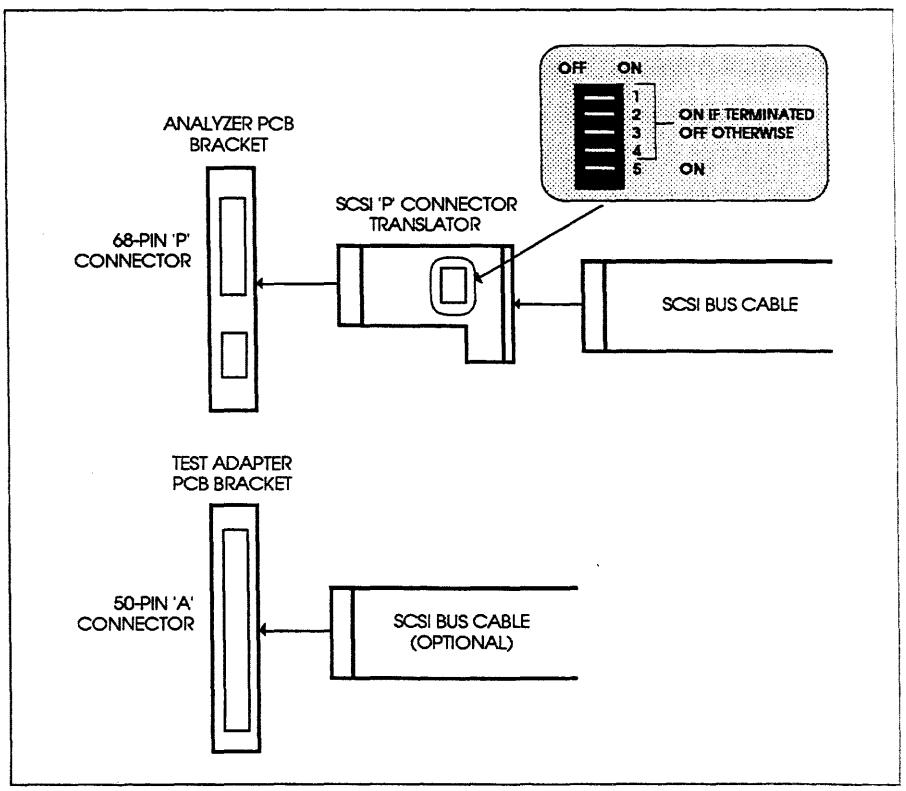

Figure 1-12. Configuration 7.

Configuration 8: 16-bit Single-Ended 'P' Cable SCSI Analysis and Optional 8-bit Single-Ended 'A' Cable SCSI Emulation

> *NOTE*
>
> *In this case, the analyzer can perform 16-bit analysis, but the test adapter can only perform 8-bit emulation. Also, for simultaneous analysis and emulation, this configuration requires two SCSI bus cable connections: one 68-pin and one 50-pin.*

1. Plug the SCSI bus cable into the analyzer PCB's 68-pin 'P' connector.

2. Plug the SCSI bus cable into the test adapter PCB's 50-pin 'A' connector (Optional).

See Figure 1-13.

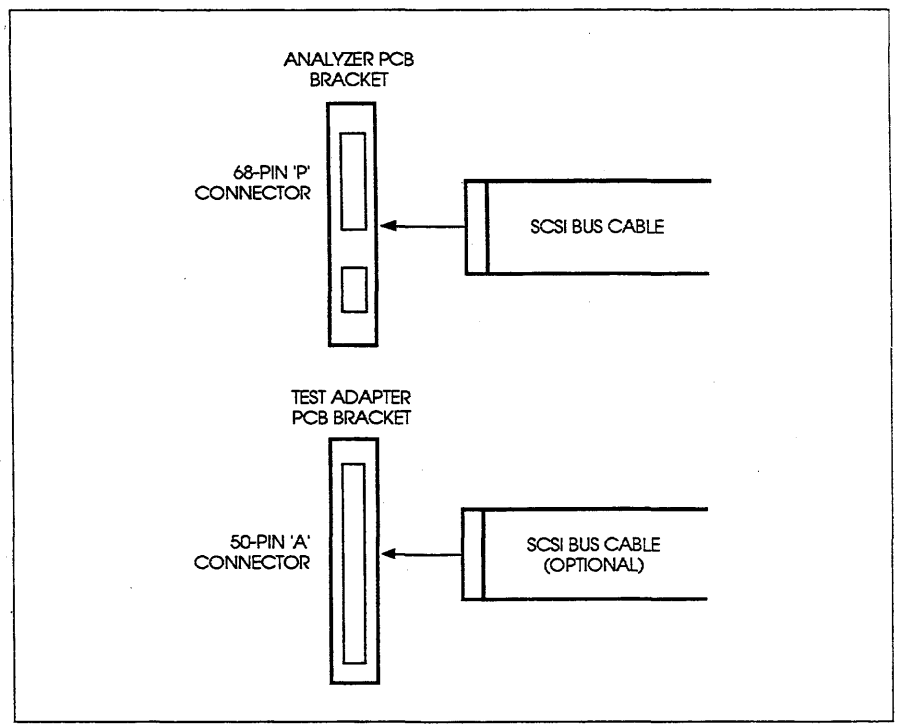

Figure 1-13. Configuration 8.

Configuration 9: 8-bit Single-Ended 'A' Cable SCSI Analysis and Optional Emulation

Plug the SCSI bus cable into the test adapter PCB's 50-pin 'A' connector.

See Figure 1-14.

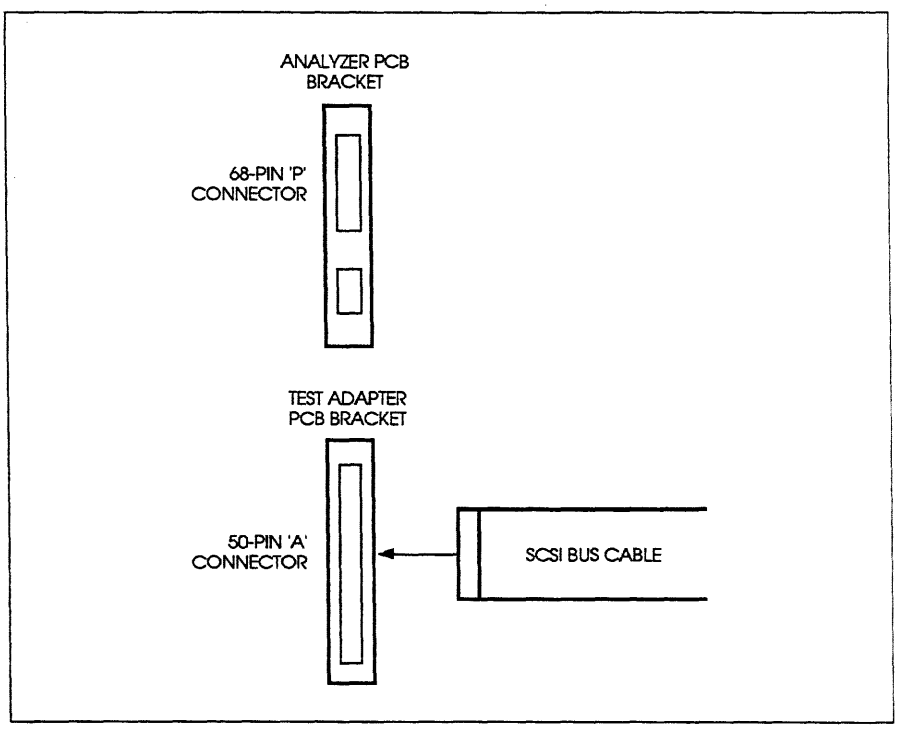

Figure 1-14. Configuration 9.

Analyzer, Differential Adapter, and Test Adapter PCBs Installed

Configuration 10: 8-bit Single-Ended 'A' Cable SCSI Analysis and Emulation

1. Unplug any SCSI bus cable or connector translator from the differential adapter PCB's 68-pin 'P' connector.

2. Enable the differential adapter by choosing the appropriate switch setting on the differential adapter PCB's bracket. See the differential adapter user's manual for this switch setting.

3. Plug the SCSI bus cable into the test adapter PCB's 50-pin 'A' connector.

See Figure 1-15.

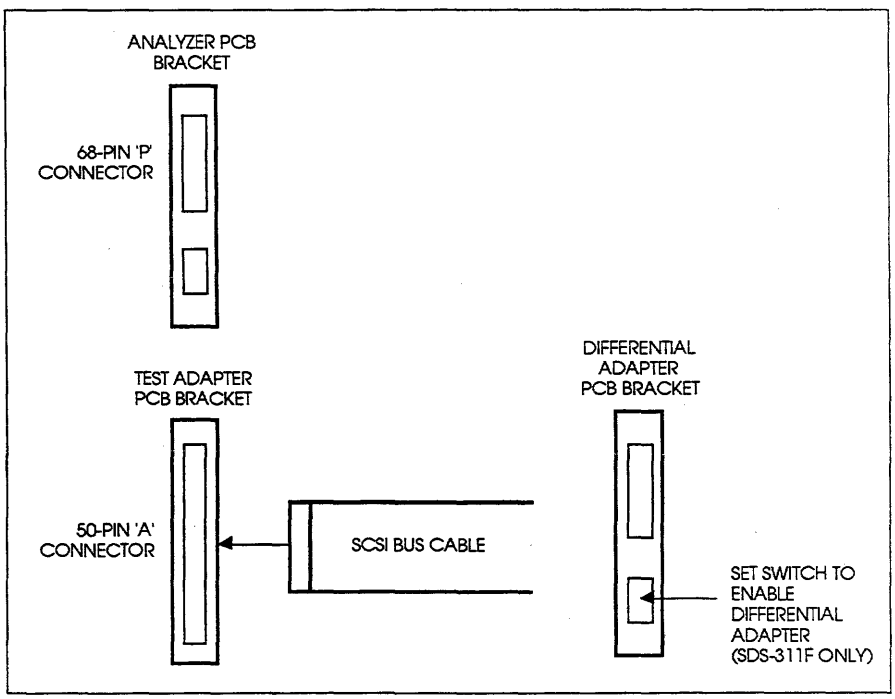

Figure 1-15. Configuration 10.

Configuration 11: 16-bit Single-Ended 'P' Cable SCSI Analysis and Optional 8-bit Single-Ended 'A' Cable SCSI Emulation

> NOTE
>
> *In this case, the analyzer can perform 16-bit analysis, but the test adapter can only perform 8-bit emulation. Also, for simultaneous analysis and emulation, this configuration requires two SCSI bus cable connections: one 68-pin and one 50-pin.*

1. Unplug any SCSI bus cable or connector translator from the differential adapter PCB's 68-pin 'P' connector.

2. Disable the differential adapter by choosing the appropriate switch setting on the differential adapter PCB's bracket. See the differential adapter user's manual for this switch setting.

3. Plug the SCSI bus cable into the analyzer PCB's 68-pin 'P' connector.

4. Plug the SCSI bus cable into the test adapter PCB's 50-pin 'A' connector (Optional).

See Figure 1-16.

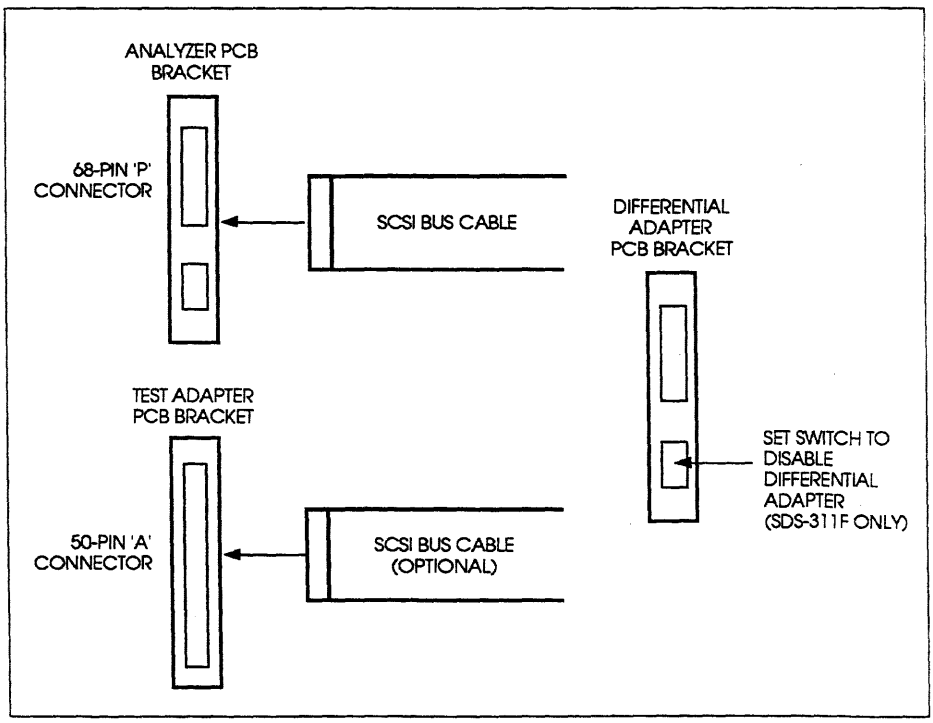

Figure 1-16. Configuration 11.

Configuration 12: 8-bit Differential 'A' Cable SCSI Analysis and Emulation

1. Unplug any SCSI bus cable or connector translator from the analyzer PCB's 68-pin 'P' connector.
2. Unplug any SCSI bus cable from the test adapter PCB's SCSI connector.
3. Enable the differential adapter by choosing the appropriate switch setting on the differential adapter PCB's bracket. See the differential adapter user's manual for this switch setting.
4. On the SCSI 'P' connector translator, turn switch 5 'OFF'. If the differential adapter PCB's 68-pin 'P' connector is terminated, turn switches 1-4 'ON'; otherwise, turn them 'OFF'.
5. Plug the translator into the differential adapter PCB's 68-pin 'P' connector.
6. Plug the SCSI bus cable into the translator's 50-pin 'A' connector.

See Figure 1-17.

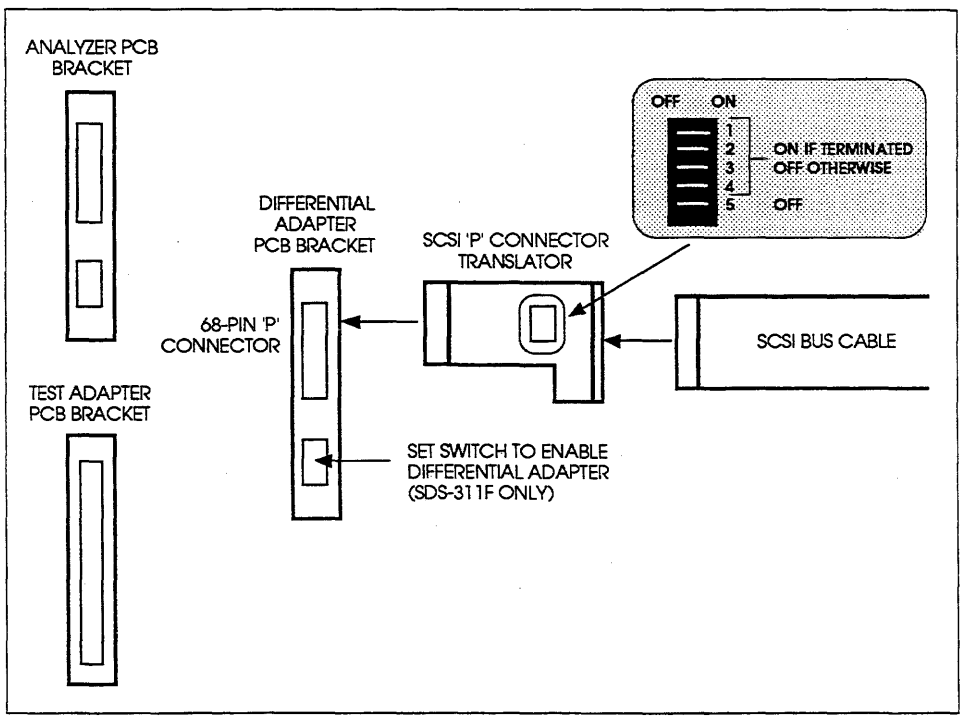

Figure 1-17. Configuration 12.

NOTE

If you are using an SDS-311, the 50-pin SCSI 'A' cable will plug directly into the differential adapter PCB. Do not use a 'P' connector translator.

Configuration 13: 16-bit Differential 'P' Cable SCSI Analysis and 8-bit Differential 'A' Cable SCSI Emulation

> *NOTE*
>
> *In this case, the analyzer can perform 16-bit analysis, but the test adapter can only perform 8-bit emulation.*

1. Unplug any SCSI bus cable or connector translator from the analyzer PCB's 68-pin 'P' connector.
2. Unplug any SCSI bus cable from the test adapter PCB's SCSI connector.
3. Enable the differential adapter by choosing the appropriate switch setting on the differential adapter PCB's bracket. See the differential adapter user's manual for this switch setting.
4. Plug the SCSI bus cable into the analyzer PCB's 68-pin 'P' connector.

See Figure 1-18.

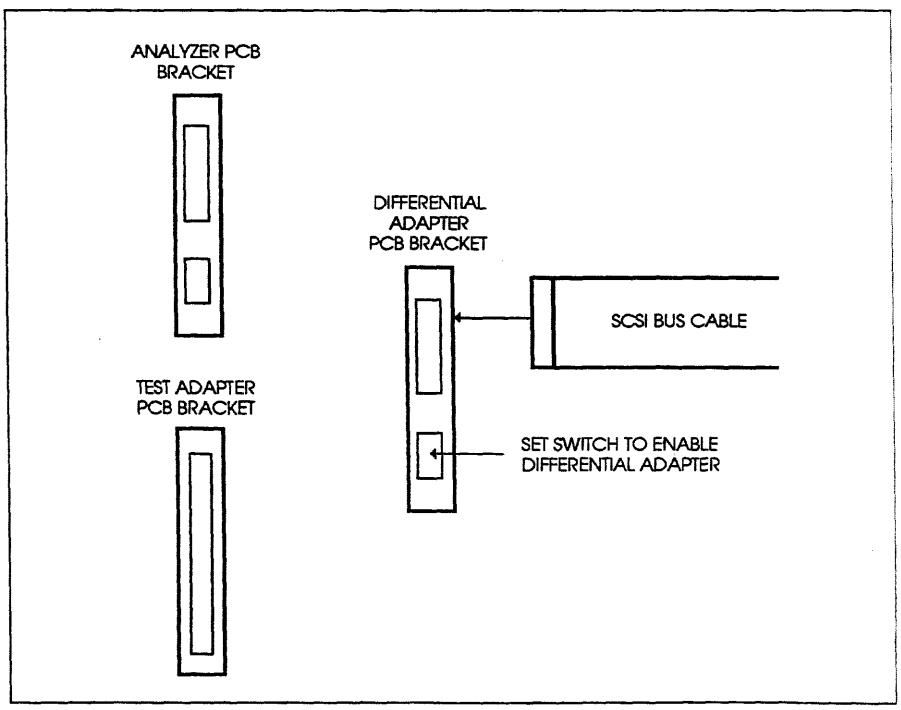

Figure 1-18. Configuration 13.

> *NOTE*
>
> *The SDS-311 does not support this configuration.*

Connecting the External Probe Pod (Optional)

> *NOTE*
>
> *Before connecting the external probe pod, make sure the appropriate PCBs have already been installed in your computer. See the* ***Installing PCB Components*** *section of this chapter for details.*

If you want to analyze non-SCSI signals along with SCSI signals, you will need to install the external probe pod. This pod always connects to the analyzer PCB, even if a differential or test adapter PCB is also installed.

To connect the pod:

1. Plug one end of the 10-pin flat ribbon cable into the analyzer PCB's 10-pin connector. The analyzer connector is keyed to prevent plugging the cable in backwards.

2. Plug the other end of the cable into the pod. The pod connector is keyed to prevent plugging the cable in backwards.

To connect the pod to non-SCSI signals:

1. Always connect the pod wire labeled GND to signal ground.

2. If want to acquire or trigger on non-SCSI TTL-class signals, connect the pod wires labeled EXT 0, EXT 1, EXT 2, and EXT 3 to the desired signals.

3. If you want to trigger an external device, connect the pod wire labeled TOUT to the desired TTL-class input pin.

See Figure 1-19.

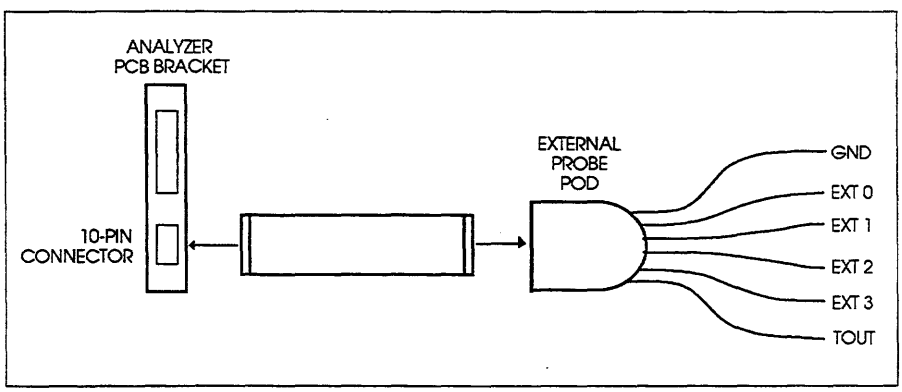

Figure 1-19. Connecting the External Probe Pod

Installing the Software

There are two independent analyzer software modules: the interactive menu and the run time function library. You may install either or both of these modules depending on your application. For more information on these modules, see the **Interactive Menu** and **Run Time Function Library** chapters of this manual.

> *NOTE*
>
> *Both modules will be installed into directory \SDS\SDS310F. After installation, it is recommended that you treat this directory as read only. Do not create any user files in this directory or any of its subdirectories. This will make the loading of future software revisions easier because, prior to loading a new revision, you can delete the entire contents of this directory to remove the old revision without regard for the files deleted.*

In the turnkey configuration, all software has already been installed on hard drive **C:** of the Adaptec-supplied computer. If you purchased your analyzer in a turnkey configuration, you can skip ahead to the **The SDS310F Configuration Utility** section of this chapter.

Installing the Interactive Menu

The analyzer interactive menu diskettes are located at the back of this manual. Two diskette formats are provided: 5 1/4" 1.2 MBytes and 3 1/2" 1.44 MBytes.

To install the menu:

1. Determine the hard drive (**C:**, **D:**, etc.) that you want to hold the menu files and make that drive current by typing *specifier*<**ENTER**> where *specifier* is the drive specifier (**C:**, **D:**, etc.).

2. Insert the diskette labeled **SDS-310F Bus Analyzer Menu, Disk 1** in floppy drive **A:** and type **A:LOAD**<**ENTER**>. The installation procedure will prompt you for all required information and any other diskettes.

> *NOTE*
>
> *If you need to load from floppy B:, before typing A:LOAD<ENTER>, execute the DOS ASSIGN A=B<ENTER> command to map floppy B: logically to floppy A:. It is recommended that after the installation is complete you re-boot your system to undo the logical assignment. See your DOS documentation for details on the ASSIGN command.*

This installation will load the menu files into subdirectories of directory **\SDS\SDS310F** of the current hard drive. The installed files will require approximately 500 KBytes.

After you have performed this installation, set up your DOS environment for analyzer use as described in the **The SDS310F Configuration Utility** section of this chapter. You will then be able to use your analyzer with the interactive menu. See the **Interactive Menu** chapter of this manual for details on invoking and operating the menu.

Installing the Run Time Function Library

The analyzer run time function library is an optional library addition to the SDS-3F Development System's SCSI C Library. You must install the SCSI C Library before installing the analyzer run time function library. See the SCSI C Library user's manual for details on its installation.

> *NOTE*
>
> *For more information on the SDS-3F Development System's SCSI C Library, contact your local Adaptec sales representative.*

The analyzer run time function library diskettes are located at the back of this manual. Two diskette formats are provided: 5 1/4" 1.2 MBytes and 3 1/2" 1.44 MBytes.

To install the library:

1. Determine the hard drive (**C:**, **D:**, etc.) that you want to hold the library files and make that drive current by typing *specifier*<**ENTER**> where *specifier* is the drive specifier (**C:**, **D:**, etc.).

2. Insert the diskette labeled **SDS-310F Bus Analyzer Run Time Function Library, Disk 1** in floppy drive **A:** and type **A:LOAD**<**ENTER**>. The installation procedure will prompt you for all required information and any other diskettes.

NOTE

*If you need to load from floppy **B:**, before typing **A:LOAD<ENTER>**, execute the DOS **ASSIGN A=B<ENTER>** command to map floppy **B:** logically to floppy **A:**. It is recommended that after the installation is complete you re-boot your system to undo the logical assignment. See your DOS documentation for details on the **ASSIGN** command.*

This installation will load the library files into subdirectories of directory **\SDS\SDS310F** of the current hard drive. The installed files require approximately 500 KBytes.

After you have performed this installation, set up your DOS environment for analyzer use as described in the **The SDS310F Configuration Utility** section of this chapter. Assuming you have previously installed the SDS-3F Development System's SCSI C Library, you will be able to create and execute user tests for the analyzer. See the **Run Time Function Library** chapter of this manual for details on creating and executing analyzer user tests.

The SDS310F Configuration Utility

NOTE

*Before following the procedures in this section, make sure the appropriate hardware and software have been installed on your computer. See the **Installing the Hardware** and **Installing the Software** sections of this chapter for details.*

The interactive menu and run time function library installations both copy a utility named **SDS310F** to the root directory of the current hard drive. After you boot your computer and before your first use of the analyzer, you must execute this utility to configure the DOS environment for analyzer use.

It is recommended that you move the two files **SDS310F.BAT** and **SDSENV.EXE** from the root to a directory in your DOS **PATH** for easier execution. If your system will be dedicated to analyzer use, you may execute **SDS310F** automatically from your system **AUTOEXEC.BAT** file. See your DOS documentation for details on **PATH** and **AUTOEXEC.BAT**.

If you installed your analyzer software on hard drive **C:** and did not change the I/O address of your analyzer PCB from the default 310, you can configure the DOS environment by simply typing **SDS310F<ENTER>** with no parameters. In this case, you can skip the remainder of this section.

If you installed your analyzer software on a hard drive other than **C:** or you changed the analyzer I/O address (or both), you must specify parameters on the **SDS310F** command line. The remainder of this section describes how to specify these parameters.

SDS310F sets the following DOS environment variables: **SBADRV**, **PATH**, and **SBAADR**. See your DOS manual for details on the DOS environment.

SBADRV

SDS310F creates a DOS environment variable called **SBADRV** and sets its value to the specifier (without a **:**) of the hard drive containing the analyzer files. The default value is **C**, but if you have loaded the software on a different hard drive you must override this value. To do this, specify a different drive letter as the first parameter of **SDS310F**.

For example,

SDS310F D<ENTER>

will set **SBADRV** to **D** for hard drive **D:**.

PATH

SDS310F sets the DOS **PATH** environment variable to reference the directory containing the analyzer executable files. (The name of this directory is *SBADRV*:**\SDS\SDS310F\BIN**, where *SBADRV* is the current value of the **SBADRV** environment variable.) If the **PATH** variable already exists when the utility is executed, the name of the analyzer executable file directory will be prepended to the current **PATH** value. If the **PATH** variable does not exist, it will be created with a value equal to the analyzer directory only.

SBAADR

SDS310F creates a DOS environment variable called **SBAADR** and sets its value to the hexadecimal I/O address of the analyzer PCB. The default value is **310**. If you have changed the address of your analyzer PCB, you must specify the new address as the second parameter of **SDS310F**. Valid values for this parameter are **310, 330, 360,** and **380**. If you specify an address parameter, you must also specify a drive parameter, even if you have loaded the software on the default drive, **C:**. See the **Installing the Analyzer PCB** section of this chapter for details on changing the analyzer PCB I/O address.

For example,

SDS310F C 330<ENTER>

will set **SBAADR** to **330**.

The 310FDIAG Diagnostic Utility

NOTE

Before following the procedures in this section, make sure the appropriate hardware and software have been installed on your computer and your DOS environment has been properly configured for analyzer operation. See the ***Installing the Hardware, Installing the Software,*** *and* ***The SDS310F Configuration Utility*** *sections of this chapter for details.*

The interactive menu and run time function library installations both copy a utility named **310FDIAG** to directory **\SDS\SDS310F\BIN**. You can use **310FDIAG** to check and report the configurations of both the analyzer and the DOS operating environment.

To run this utility, type **310FDIAG<ENTER>**. **310FDIAG** reports the following:

1. The directory that will be searched for analyzer files.
2. The current value of the **SBAADR** DOS environment variable.
3. The current analyzer PCB address switch setting.
4. Whether or not the analyzer PCB is properly configured.

See the **Troubleshooting** appendix of this manual if any problems are reported by **310FDIAG**.

Chapter 2

Key Concepts

There are two basic steps to using any logic analyzer:

1. Capture data.

2. Analyze the captured data.

The SDS-310F SCSI bus analyzer has SCSI-specific features designed into both of these steps to make SCSI bus analysis more efficient and effective. This chapter describes the key concepts that must be understood to make full use of these SCSI-specific features.

Capturing Data

SCSI bus activity is generally characterized as short bursts of very fast activity interspersed within long periods of little or no activity. Signal pulse widths can be as short as 30 nanoseconds; but times with no signal pulses can last milliseconds or longer. Effective analysis of this behavior requires resolution of less than 30 nanoseconds coupled with a means of capturing data over extended time spans.

Traditional logic analyzers typically capture data on each cycle of a fixed clock. The resolution and length of the capture data are completely determined by the clock cycle time. A 20 nanosecond cycle time will fill a 32K buffer in 655.36 microseconds [$(2^{15} \bullet 20)/1000$]. This is sufficient resolution to capture all SCSI signal pulses but an insufficient time span for effective SCSI analysis.

The SDS-310F SCSI bus analyzer samples the bus with 20 nanosecond resolution, but, unlike traditional logic analyzers, it only processes a sample if there has been a transition of at least one signal from the last processed sample. Each sample that requires processing is called an event. Each event includes a time stamp so that the timing of events can be reproduced with 20 nanosecond resolution. This method of data capture is called transition analysis and provides both the resolution and the time span required for effective SCSI analysis.

You can configure the analyzer to save all events in the capture buffer, or you can use special features of the SDS-310F's data capture hardware to control what events are actually saved. These features are: the acquisition qualifier, the trigger, the ID qualifier, and the event filter.

The following figure shows a functional block diagram of the SDS-310F's data capture hardware.

See Figure 2-1.

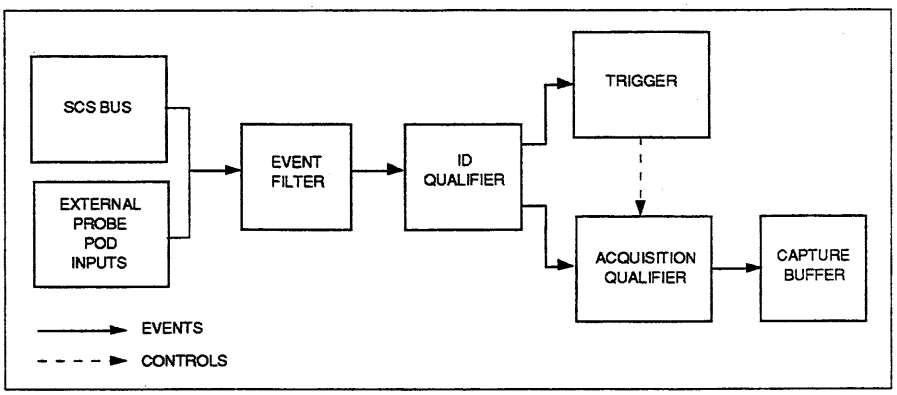

Figure 2-1. SDS-310F Data Capture Functional Block Diagram

The Capture Buffer

Each event saved in the capture buffer includes the state of the 9 SCSI control signals (BSY, SEL, ATN, REQ, ACK, C/D, I/O, MSG, RST), the 16 data lines (D0-D15), the two parity lines (DP0, DP1), the four external probe pod input wires (EXT0-3), and a 40-bit time stamp. The analyzer's capture buffer has a capacity of 32K events, each event consisting of 71 bits [9 + 16 + 2 + 4 + 40].

Event time stamps are relative to the time you started the analyzer running. The time between any two events in the capture buffer is called the delta time. The analyzer software performs delta time calculations automatically.

The time stamp clock will roll over approximately once every 12 hours. With each rollover, the time stamp of the first event following the rollover will be incorrect. Also, the delta time for any two events where one precedes a rollover event and the other follows the rollover event will also be incorrect. In most analyses, these limitations should not be a concern; 12 hours is a long time, even by SCSI standards.

The Acquisition Qualifier

Each SCSI command is executed as a sequence of bus phases: arbitration, selection, reselection, message out, command, data out, data in, status, and message in. Often, you do not need to capture the events of every phase type. For example, if you just want to analyze who's using the bus without regard for the actual commands that are being executed, you only need to capture the events of arbitration, selection, and reselection phases. If you could limit data capture to only those events of the desired phase types, you could effectively create more capacity in the capture buffer. The analyzer's acquisition qualifier gives you this control.

You specify one of four modes for the acquisition qualifier:

All State	Save the events of all bus phases: arbitration, selection, reselection, message out, command, data out, data in, status, and message in.
Arb Sel	Save only the events of arbitration, selection, and reselection phases.
Arb Sel Cmd	Save only the events of arbitration, selection, reselection, and command phases.
Arb Sel Cmd Stat	Save only the events of arbitration, selection, reselection, command, and status phases.

The biggest consumers of the capture buffer are the events of data phases. Each SCSI command typically has 10-20 message bytes, 6, 10, or 12 command bytes, and one status byte; but thousands of data bytes or words. Often it is sufficient to capture only a subset of each data phase. In addition to phase control, the analyzer's acquisition qualifier gives you control over the amount of data to capture.

The acquisition qualifier's **"All State"** option is the only option that allows capture of data phases, and with this option you also specify:

1. The offset of the first byte or word in each data phase to save.
2. The maximum number of consecutive bytes or words in each data phase to save.

> *NOTE*
>
> *Setting the maximum data byte/word count to **0** disables the saving of data phases. In this case, the events of all non-data phases, including messages, will be saved.*

The Trigger

When you tell the analyzer to run, it will immediately start saving qualified events in the capture buffer. This capture will continue until either you stop the analyzer or a trigger occurs. If more than 32K qualified events occur before the analyzer stops, the capture buffer will wrap, and earlier events will be overwritten with more recent events. This means that after the buffer fills, you will always be able to see the most recent 32K saved events but will lose any preceding events.

If you are interested in analyzing the SCSI bus activity around a particular event, and there is the possibility that enough events will follow to overwrite the desired data, you will need to use the analyzer's trigger to stop event capture before the desired data is overwritten.

You specify one of four modes for the trigger:

No Trigger Do not trigger. This will guarantee that you will always see the the last 32K events, regardless of what preceded them. This mode is especially useful if you are trying to analyze the events that lead up to a hang condition.

Immediate Trigger as soon as data capture starts. This will guarantee that you will always be able to see the first 32K qualified events regardless of what follows. For this mode, the trigger event is always at the beginning of the buffer.

Phase Trigger when a SCSI phase of the specified type is detected. Any of the following SCSI phase types may be specified: arbitration, selection, reselection, message out, command, data out, data in, status, or message in. You specify the phase type by name; the software automatically specifies the required state of the SCSI control signals for the trigger as follows (1 asserted, 0 deasserted, X any state, P transition from deasserted to asserted):

Phase Type	BSY	SEL	ATN	REQ	ACK	C/D	I/O	MSG	RST
Arbitration	1	0	0	0	0	0	0	0	0
Selection	0	1	X	X	X	X	0	X	0
Reselection	0	1	X	X	X	X	1	X	0
Message Out Byte	1	0	X	0	P	1	0	1	0
Command Byte	1	0	X	0	P	1	0	0	0
Data Out Byte	1	0	X	P	P	0	0	0	0
Data In Byte	1	0	X	P	P	0	1	0	0
Status Byte	1	0	X	P	0	1	1	0	0
Message In Byte	1	0	X	P	0	1	1	1	0

In addition to the phase type, you also specify the required state of the SCSI data lines (D0-D15), SCSI parity lines (DP0-DP1), and external probe pod input signals (EXT0-3), plus an occurrence count (up to 4095).

> *NOTE*
>
> *When specifying the required states of signals for the trigger, specify 1 for asserted, 0 for deasserted, and X for any state. P is not an option for user specification.*

A trigger occurrence is detected only when the states of an event's SCSI bus and external probe pod input signals match all of the required states specified for the trigger. The analyzer's data capture hardware stops running when the number of trigger occurrences equals the specified occurrence count.

Specify Bits This mode is the same as Phase except that you specify the required state of each SCSI control signal directly (1, 0, or X). As with the Phase mode, you also specify the required state of the SCSI data, SCSI parity, and external probe pod input signals, plus an occurrence count (up to 4095). In this mode, any SCSI bus state may be specified including illegal states.

For the Phase and Specify Bits modes, you also specify one of three trigger positions in the capture buffer. These positions assume the buffer is full. If the buffer is not full, all events and the trigger are in the buffer and available for analysis.

Before Stop the analyzer approximately 30400 events after the trigger. This will place 5% of the capture data before the trigger and 95% after.

Center Stop the analyzer approximately 16000 events after the trigger. This will place 50% of the capture data before the trigger and 50% after.

After Stop the analyzer approximately 1600 events after the trigger. This will place 95% of the capture data before the trigger and 5% after.

The ID Qualifier

SCSI buses have evolved from single initiator, single target configurations to multiple initiator, multiple target configurations. In these complex bus configurations, it is often desirable to analyze the activities of one SCSI initiator-target combination and ignore the activities of all other combinations. The ID qualifier lets you focus both the acquisition qualifier and the trigger on particular initiator-target combinations. Qualified events will 'pass through' to the acquisition qualifier and trigger; unqualified events will be suppressed (see Figure 2-1).

You specify one of four modes for the ID qualifier:

Off Disabled.

Let all events of all target and initiator ID combinations 'pass through' to both the trigger and acquisition qualifier.

> *NOTE*
>
> *If you do not plan to use the ID qualifier, select mode Off and skip ahead to the next section.*

For the remaining three ID qualifier modes, you must specify the ID's of the initiator-target combination that you are interested in. For both the initiator and target ID's, you specify one of 0 through 7 or X. Specifying X for the initiator ID allows you to focus on a particular target's activity with all initiators. Specifying X for the target ID allows you to focus on a particular initiator's activity with all targets.

SCSI commands may disconnect from the SCSI bus to allow other commands to execute before the original command reconnects to the bus to continue and/or complete its execution. The first time a command connects to the SCSI bus is called a selection. All subsequent times that the command reconnects to the bus are called reselections. The remaining three ID qualifier modes enable the ID qualifier and allow you to focus on command selections, command reselections, or both.

Sel Resel Enabled for selections and reselections.

All command selections and reselections for the specified initiator-target combination will be qualified and 'pass through' subject to the trigger and acquisition qualification options described below. All other selections and reselections will be suppressed.

Sel Enabled for selections only.

All command selections for the specified initiator-target combination will be qualified and 'pass through' subject to the trigger and acquisition qualifications options described below. All other selections and reselections, including reselections for the specified initiator-target combination, will be suppressed.

Resel Enabled for reselections only.

All command reselections for the specified initiator-target combination will be qualified and 'pass through' subject to the trigger and acquisition qualification options described below. All other selections and reselections, including selections for the specified initiator-target combination, will be suppressed.

If you select one of the above three modes, you also specify one of the following options for both the trigger and the acquisition qualifier (a different option may be chosen for the trigger and acquisition qualifier):

Off Disabled.

Let all events of all target and initiator ID combinations pass through to the trigger or acquisition qualifier.

> *NOTE*
>
> *Selecting the Off option for both the trigger and acquisition qualifier is a second way of disabling the ID qualifier.*

The remaining two options enable the ID qualifier for the acquisition qualifier. They also allow you to specify whether you want to capture the events of arbitration, selection, or reselection phases.

In order to make qualifications based on initiator-target ID combinations, the analyzer must inspect all arbitration, selection, and reselection phases. The decision of whether or not to save the events of these phases cannot be made as these phases are being inspected. Instead, you make the decision by specifying one of the following options:

Allow Enabled without arbitration-selection-reselection suppression.

Only let events of the specified initiator ID-target ID combination pass through to the acquisition qualifier. With this option, the events of arbitration, selection, and reselection phases of all initiator-target combinations will pass through, including those for other than the specified initiator-target combination.

Suppress Enabled with arbitration-selection-reselection suppression.

Only let events of the specified initiator ID-target ID combination pass through to the acquisition qualifier. With this option, no events of arbitration, selection, or reselection phases of any initiator-target combination will pass through, even for the specified initiator-target combination. This effectively limits events to those of information transfer phases.

The Event Filter

If you are performing analysis of an 8-bit bus, the event filter gives you the ability to suppress events that contain only transitions of the upper 8 data bits of the SCSI bus. In theory, these transitions should not occur, but may arise as a result of special environmental condition. For this option of the event filter, you specify one of the following two bus widths:

8 The SCSI bus is 8-bits wide. Suppress all events that contain only transitions of the upper 8 SCSI data bits D8 to D15 or DP1.

16 The SCSI bus is 16-bits wide. Do not perform any suppression based on the upper 8 SCSI data bits D8 to D15 or DP1.

During information transfers on the SCSI bus, initiators and targets qualify each byte or word with the transition of either REQ (when the SCSI I/O signal is 1) or ACK (when I/O is 0) from deasserted to asserted. Often, the SCSI data lines will go through transitions with no qualifying transitions of REQ or ACK. Ordinarily, these unqualified transitions will generate events to be processed by the analyzer. Often, these transitions are of no interest to the analysis. The event filter also gives you the control over these transitions.

You specify one of three modes for the event filter:

Off

All signal transitions (SCSI control, SCSI data, SCSI parity, and external probe pod input) generate events.

Four Events Per Transfer

Transitions of the SCSI data and parity signals by themselves will not generate events. Only transitions of the 9 SCSI control and four external probe pod input signals will generate events.

This mode limits each information transfer to at most 4 events: REQ assert, REQ deassert, ACK assert, and ACK deassert.

NOTE

In the Four Events Per Transfer mode, you can still determine the values of SCSI information bytes and words. If I/O is 1, they are valid with REQ assertion. If I/O is 0, they are valid with ACK assertion.

One Event Per Transfer

Transitions of the SCSI data and parity signals by themselves will not generate events. Transitions of any of the following 7 SCSI control signals will generate events: BUSY, SEL, ATN, C/D, I/O, MSG, and RST. In addition, if I/O is 1, a transition of REQ from deasserted to asserted will generate an event. If I/O is 0, a transition of ACK from deasserted to asserted will generate an event. Any other transitions of REQ or ACK will be suppressed.

This mode limits each information transfer to 1 event: REQ asserted or ACK asserted, depending on when the data is valid. The information byte or word saved with this event will be valid.

NOTE

Selecting this mode, while minimizing capture data, limits other features of the analyzer. Triggering in the ***Phase*** *mode is restricted to arbitration, selection, and reselection. The six predefined information transfer triggers will not be detected. Also, capture data analysis (see the next section) is limited to timing mode. State mode displays will be incorrect.*

Analyzing the Captured Data

The SDS-310F software supports two modes for analyzing the captured data: state mode and timing mode. In state mode, the analyzer software automatically translates the captured data into SCSI terms. Figure 2-2 (at the end of this section) shows a SCSI reset, test unit ready command, request sense command, and read data command sequence displayed in the state mode.

While state mode simplifies analysis of SCSI phase transitions, specific signal transition details are suppressed.

In timing mode, the captured data is presented as events without interpretation exactly as they were captured. Figure 2-3 shows captured data of a subset of Figure 2-2's SCSI activity displayed in the timing mode.

In timing mode, you can analyze individual signal transitions with 20 nanosecond resolution. If you need the detail, this is important. Note, however, that more lines are required to display this detail. In fact, Figures 2-2 and 2-3 have the same number of display lines. However, the timing display only shows activity up to the transfer of the third command byte of the test unit ready command.

You choose the data analysis mode after the data has been captured, and you can move back and forth between state and timing modes with the same captured data as your analysis requires.

State Mode

State mode data is presented as a sequence of state lines, and each line consists of three parts (left to right):

Time stamp Each state mode line includes a time stamp with 20 nanosecond resolution.

SCSI state The state of the SCSI bus for that line interpreted as follows:

State	Interpretation
Bus Free	The SCSI bus is idle.
Arb Start	At least one device is arbitrating for the bus.
Arb Win	A device has won arbitration.
Sel Start	A device is selecting another device.
Sel End	A device has selected another device.
Sel TO	Selection time out.
Resel Start	A device is reselecting another device.
Resel End	A device has reselected another device.
Resel TO	Reselection time out.
Msg Out	Message out information byte transfer(s).
Command	Command information byte transfer(s).
Data Out	Data out information byte or word transfer(s).
Data In	Data in information byte or word transfer(s).
Status	Status information byte transfer(s).
Msg In	Message in information byte transfer(s).
(Atn Assertion)	ATN changed from deassert to assert.
(Atn Deassert)	ATN changed from assert to deassert.
(Rst Assertion)	RST changed from deassert to assert.
(Rst Deassert)	RST changed from assert to deassert.
Trigger	The trigger event.

Lines of the following state types include a numeric field:

State	Numeric Field
Arb Start	ID's on the SCSI bus.
Arb Win	ID's on the SCSI bus.
Sel Start	ID's on the SCSI bus.
Resel Start	ID's on the SCSI bus.
Msg Out	Message out transfer byte(s).
Command	Command transfer byte(s).
Data Out	Data out transfer byte(s)/word(s).
Data In	Data in transfer byte(s)/word(s).
Status	Status transfer byte(s).
Msg In	Message in transfer byte(s).

Each line of information transfer phases includes up to six transfers. If an information transfer phase has more than six transfers, that phase will generate multiple state mode lines with up to six transfers per line.

If any state line includes the assertion of the SCSI ATN or RST signals, an A or R, respectively, will be displayed to the right of the state description.

Lastly, an ambiguous State will have an asterisk (*) to the far right.

Line number The number of the state mode line. Lines are numbered sequentially starting with 0.

Timing Mode

Timing mode data is presented as a sequence of events, and each event consists of three parts (left to right):

Time stamp Each timing mode line includes a time stamp with 20 nanosecond resolution.

Signal states The state of the 9 SCSI control signals (BSY, SEL, ATN, REQ, ACK, C/D, I/O, MSG, RST), the SCSI data and parity lines, and the four external probe pod input signals (EXT0-3).

Line number The number of the event in the capture buffer. Lines are numbered sequentially starting with 0.

Time Stamps

There are two choices for time stamps: real time and delta time.

Real time time stamps give the elapsed time between the start of the analyzer data capture and the time stamped event. These are the actual time stamps saved with the events during data capture. The format is sss.mmm_uuu_nnn where sss is seconds, mmm is milliseconds, uuu is microseconds, and nnn is nanoseconds. The time stamps in Figures 2-2 and 2-3 are real time.

Delta time time stamps give the elapsed time between the immediately preceding event and the time stamped event. In this case, the saved time stamps of two successive events are subtracted. The format is the same as for real time time stamps except that leading zeros are suppressed. The time stamps in Figure 2-4 are delta time.

You choose the time stamp display after the data has been captured, and you can move back and forth between real time and delta time displays with the same captured data as your analysis requires.

> *NOTE*
>
> *In state mode, the time stamp for any line is the time stamp of the first event of that state line. For example, the time stamp of line 11 of Figure 2-2 is the time that the SCSI ACK signal was first asserted in the Command phase (line 26 of Figure 2-3).*

Data Phases

Information transferred during SCSI data in or data out phases may be 8 or 16 bits wide. You may specify the width of the data displayed as either 8 or 16 bits. Figures 2-2 and 2-3 show 8-bit data displays. Figures 2-5 and 2-6 show 16-bit displays for both state and timing modes.

You choose between these two modes after the data has been captured, and you can move back and forth between 8 and 16-bit displays with the same captured data as your analysis requires.

> *NOTE*
>
> *For analysis of 8-bit SCSI buses in timing mode, the 8-bit mode is recommended because the data is displayed in both binary and hexadecimal representations. In 16-bit mode, only a hexadecimal representation is available.*

Displaying Capture Data

Capture data can be displayed to the printer, to a disk file, or to the computer's screen.

Displaying to the Printer Or a Disk File

If the data is to be displayed to the printer or a disk file, you specify the format (Timing/State, Real Time/Delta Time, 8-bit/16-bit), the first line to display, and the number of lines to display. These lines will be printed or written to the disk file as requested. This display will be static. To get another format or range of lines, you will have to make another display request. In the case of the printer, the second request will be printed after the first request. In the case of a disk file, the second request will be concatenated to the end of the first request. Figures 2-2 through 2-6 are examples of data displays to the printer.

Displaying To The Computer's Screen

If the data is to be displayed to the computer's screen, you may interactively change the data format and the range of lines displayed. Figure 2-7 shows a display to the computer's screen in the state, real time, and 8-bit modes. There is a cursor which highlights one of the lines in reverse video. Initially, the cursor line is line 0.

Interactive Display Control

This display is dynamic. Whether the display was requested from the menu or a user test, you can alter it with the following keystrokes:

F1	Toggle between state and timing modes.
F2	Toggle between 8-bit and 16-bit modes.
F3	Toggle between real time and delta time modes.
down arrow	Move the cursor down one line. If the cursor is at the bottom of the screen, scroll the data up one line.
up arrow	Move the cursor up one line. If the cursor is at the top of the screen, scroll the data down one line.
Page Up	Scroll the data down 21 lines.
Page Down	Scroll the data up 21 lines.
Return	Scroll the data up one line.
Backspace	Scroll the data down one line.
Home	Move the cursor to line 0.
End	Move the cursor to the last line.
T	Move the cursor to the trigger line.
G	Move the cursor to any line. You will be prompted for the line number.
ESC	Exit the display and return to either the menu or the user test.

In addition, in state or timing mode you may perform delta time calculations between any two lines. To do this, move the cursor to one of the lines and type x, move the cursor to the other line and type o. The delta time will be displayed in the lower left corner of the display. If x precedes o, the time will be positive; otherwise, it will be negative. Figure 2-7 shows a delta time between lines 4 and 14 of +0.002_413_200. If x or o is marking a line, then typing x or o will return that marker to the cursor line.

Searching In Timing Mode

In timing mode only, you can specify search conditions. To search forward, type F, and, to search backward, type B. You will then be prompted for the desired state of each SCSI and external input signal.

The following keys move between fields:

Tab	Move one field to the right.
down arrow	Move one field to the right.
Back Tab	Move one field to the left.
up arrow	Move one field to the left.
Home	Move to the leftmost field.
End	Move to the rightmost field.

For each SCSI control line and external input, you may specify the following:

X	Any state
0	Deasserted
P	Transition from deasserted to asserted
1	Asserted
N	Transition from asserted to deasserted

You may either type the values directly or you may toggle between them with the **space bar, +, -, right arrow, or left arrow** keys. When you type the values directly, the cursor will automatically move to the next signal to the right. When you toggle, the cursor does not move.

For each data nibble, you may either type the value directly (**0-F** or **X** for any state) or you may toggle between the values the same as for control lines. Typing directly will move the cursor to the next signal to the right; toggling will not move the cursor.

Once the desired search condition has been defined, typing **Enter** will start the search. If the search condition is found, the cursor will move to that line. You may then repeat the search by typing **N**. If the search condition is not found, an error message will be displayed, and the cursor will not move.

Typing **Esc** will exit the search mode without starting a search.

You may define up to five different search conditions. To move between the different conditions during search condition specification use the **Page Up** and **Page Down** keys.

Transporting Data

Capture data can be transported in two forms: formatted or unformatted.

Formatted data is transported by specifying that data is to be displayed to a file. This file may either be on a floppy or copied from the hard disk to a floppy using the DOS Copy command. In the case of formatted data, the data that is displayed is static. What you save is what you get. This is useful for reporting problems to people who may not own the SDS-310F.

Unformatted data allows dynamic analysis on remote SDS-310F systems. Unformatted data files are created automatically by the analyzer software each time capture data is first analyzed. The name of the file used is specified with either the interactive menu **Data File** command or the run time function library **sba_data_file** function. The extension of this file will always be .TIM. For state mode, the software will also automatically create a file with the same name, but an extension of **.STT**. This **.STT** file is not necessary for transporting unformatted data. It can be generated from the **.TIM** file. The same is not true of the **.TIM** file. It cannot be generated from the .STT. Therefore, you should be sure to transport the **.TIM** file.

If the recipient of this **.TIM** file has an SDS-310F, they can analyze the data with either the interactive menu or a user test the same as if they had captured the data on their SDS-310F.

```
-------------------------------------------------------------------------
  0.000_000_000  Bus Free                                            00000
174.032_330_780  -------------------Trigger---------------        R  00001
174.032_330_780                     (Reset Assertion)              R  00002
174.032_555_420                     (Reset Deassert)                  00003
175.308_236_980    Arb Start        0                                 00004
175.308_238_380    Arb Win          0                                 00005
175.308_238_980                     (Atn Assertion)              A    00006
175.308_239_040      Sel Start      0 5                          A    00007
175.308_240_600      Sel End                                     A    00008
175.308_830_820                     (Atn Deassert)                    00009
175.308_836_460        Msg Out      80                                00010
175.309_418_140          Command    00 00 00 00 00 00                 00011
175.310_382_620          Status     02                                00012
175.310_550_620        Msg In       00                                00013
175.310_650_180  Bus Free                                             00014
175.312_047_980    Arb Start        0                                 00015
175.312_049_380    Arb Win          0                                 00016
175.312_049_980                     (Atn Assertion)              A    00017
175.312_050_020      Sel Start      0 5                          A    00018
175.312_051_580      Sel End                                     A    00019
175.312_634_600                     (Atn Deassert)                    00020
175.312_640_480        Msg Out      80                                00021
175.313_180_420          Command    03 00 00 00 FF 00                 00022
175.313_750_840            Data In  70 00 06 00 00 00                 00023
175.314_479_100                     00 0A 00 00 00 00                 00024
175.315_020_480                     29 00 00 00                       00025
175.315_678_440          Status     00                                00026
175.316_070_940        Msg In       00                                00027
175.316_167_260  Bus Free                                             00028
180.178_472_920    Arb Start        0                                 00029
180.178_474_320    Arb Win          0                                 00030
180.178_474_920                     (Atn Assertion)              A    00031
180.178_474_960      Sel Start      0 5                          A    00032
180.178_476_520      Sel End                                     A    00033
180.179_065_280                     (Atn Deassert)                    00034
180.179_072_540        Msg Out      80                                00035
180.179_558_720          Command    28 00 00 00 00 00                 00036
180.179_610_060                     00 00 01 00                       00037
180.188_331_020            Data In  00 00 01 01 7D 00                 00038
180.188_648_640                     00 00 51 55 41 4E                 00039
180.188_651_640                     54 55 4D 20                       00040
180.189_116_120          Status     00                                00041
180.189_348_380        Msg In       00                                00042
180.189_452_560  Bus Free                                             00043
```

Figure 2-2. State Mode

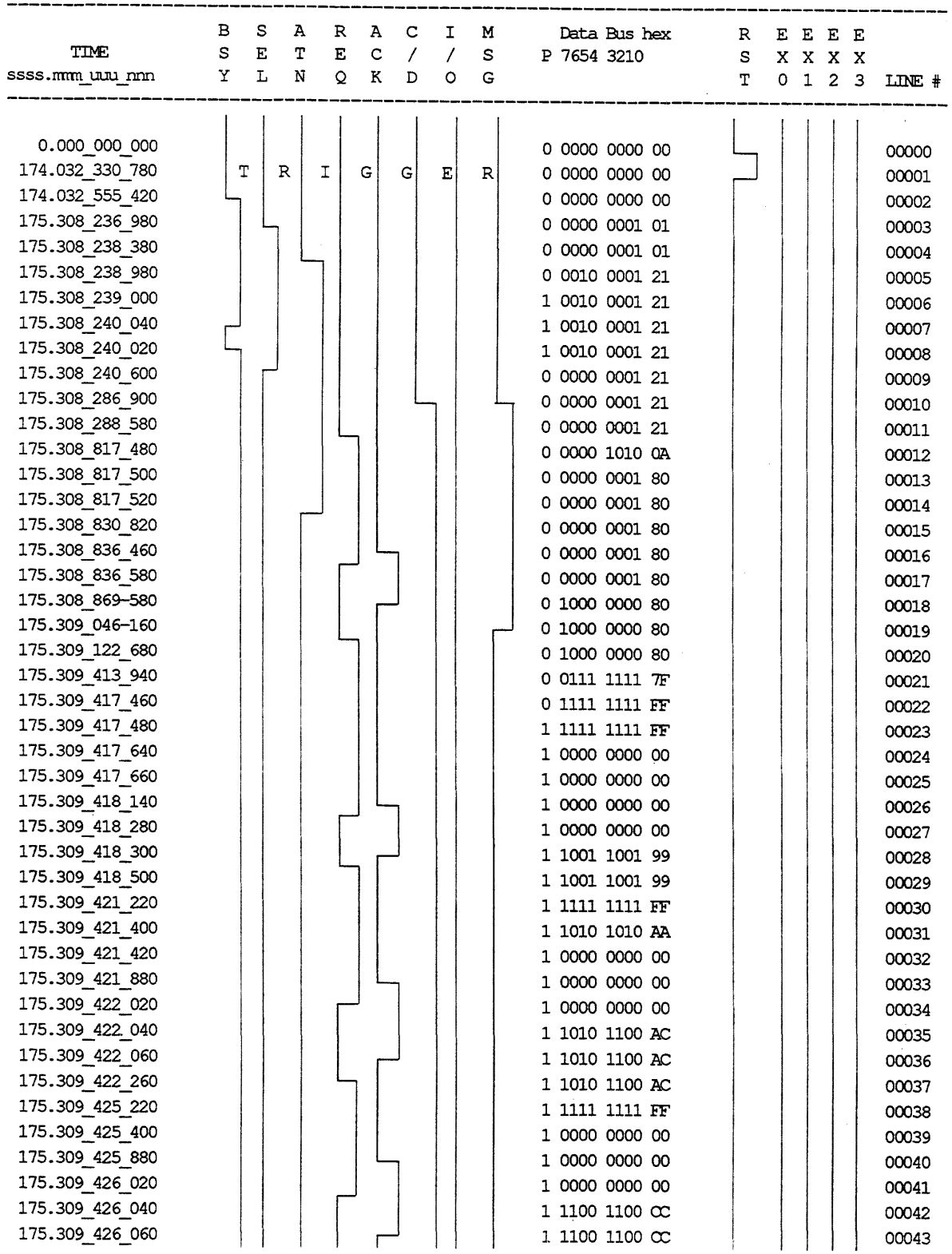

Figure 2-3. Timing Mode

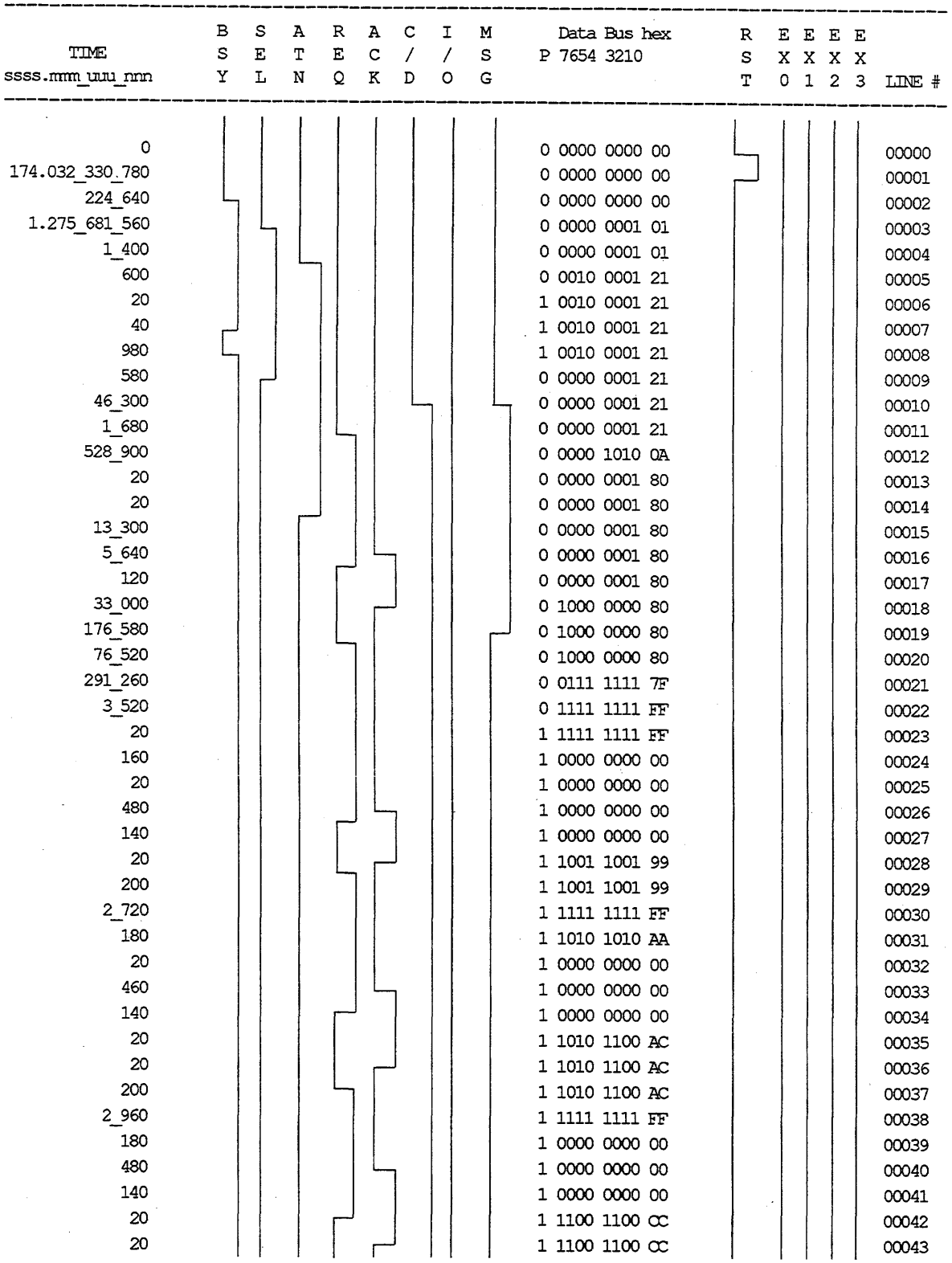

Figure 2-4. Delta Time Time Stamps

```
-------------------------------------------------------------------------
  0.000_000_000 Bus Free                                            00000
174.032_330_780 --------------------Trigger---------------        R  00001
174.032_330_780                     (Reset Assertion)             R  00002
174.032_555_420                     (Reset Deassert)                 00003
175.308_236_980    Arb Start        0                                00004
175.308_238_380    Arb Win          0                                00005
175.308_238_980                     (Atn Assertion)             A    00006
175.308_239_040      Sel Start      0 5                         A    00007
175.308_240_600      Sel End                                    A    00008
175.308_830_820                     (Atn Deassert)                   00009
175.308_836_460        Msg Out      80                               00010
175.309_418_140          Command    00 00 00 00 00 00                00011
175.310_382_620          Status     02                               00012
175.310_550_620        Msg In       00                               00013
175.310_650_180 Bus Free                                            00014
175.312_047_980    Arb Start        0                                00015
175.312_049_380    Arb Win          0                                00016
175.312_049_980                     (Atn Assertion)             A    00017
175.312_050_020      Sel Start      0 5                         A    00018
175.312_051_580      Sel End                                    A    00019
175.312_634_600                     (Atn Deassert)                   00020
175.312_640_480        Msg Out      80                               00021
175.313_180_420          Command    03 00 00 00 FF 00                00022
175.313_750_840            Data In  70 00 06 00 00 00                00023
175.314_479_100                     00 0A 00 00 00 00                00024
175.315_020_480                     29 00 00 00                      00025
175.315_678_440          Status     00                               00026
175.316_070_940        Msg In       00                               00027
175.316_167_260 Bus Free                                            00028
180.178_472_920    Arb Start        0                                00029
180.178_474_320    Arb Win          0                                00030
180.178_474_920                     (Atn Assertion)             A    00031
180.178_474_960      Sel Start      0 5                         A    00032
180.178_476_520      Sel End                                    A    00033
180.179_065_280                     (Atn Deassert)                   00034
180.179_072_540        Msg Out      80                               00035
180.179_558_720          Command    28 00 00 00 00 00                00036
180.179_610_060                     00 00 01 00                      00037
180.188_331_020            Data In  0000 0000 0001 0001 007D 0000    00038
180.188_648_640                     0000 0000 0051 0055 0041 004E    00039
180.188_651_640                     0054 0055 004D 0020              00040
180.189_116_120          Status     00                               00041
180.189_348_380        Msg In       00                               00042
180.189_452_560 Bus Free                                            00043
```

Figure 2-5. State Mode 16 Bits

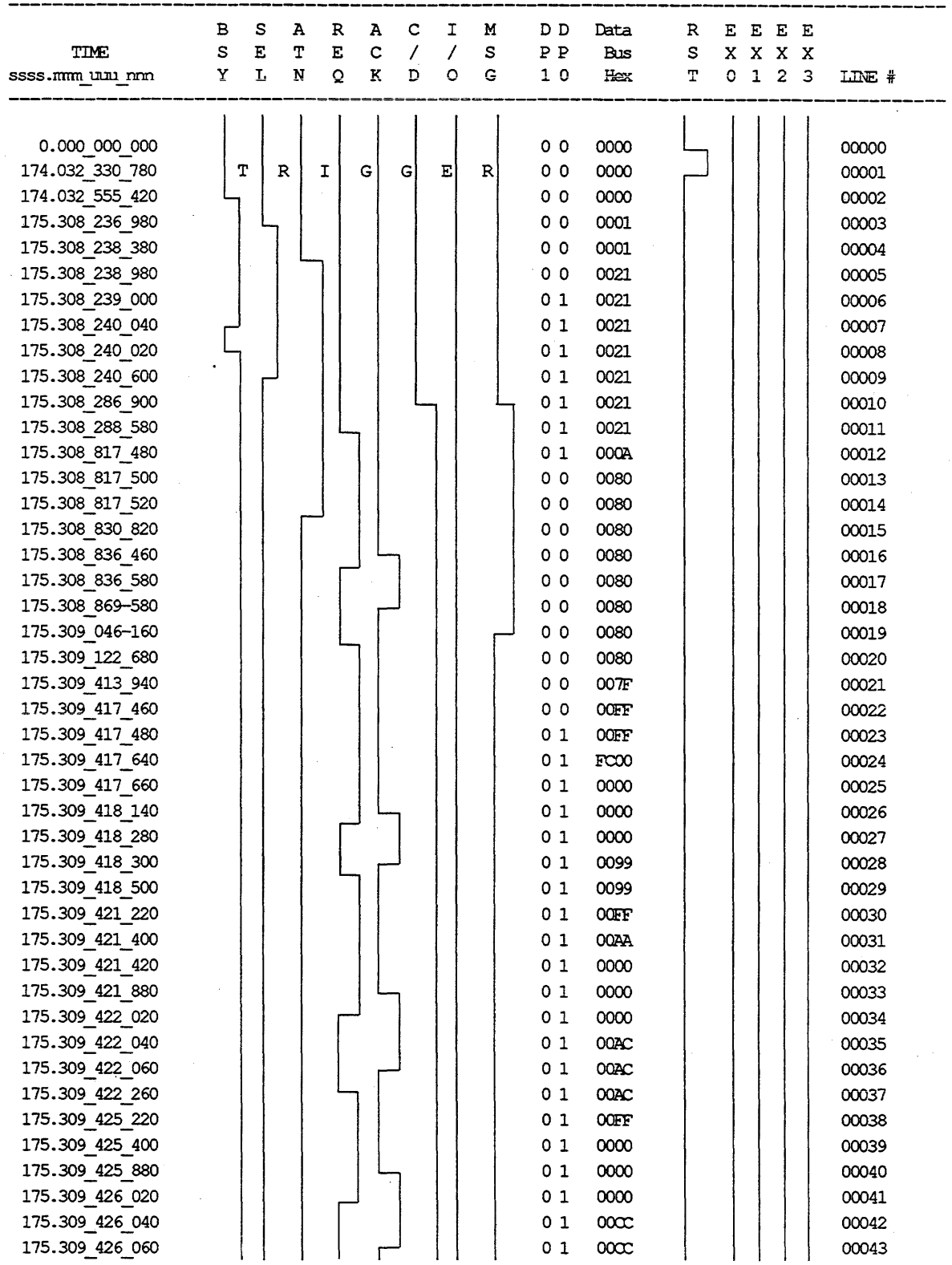

Figure 2-6. Timing Mode 16 Bits

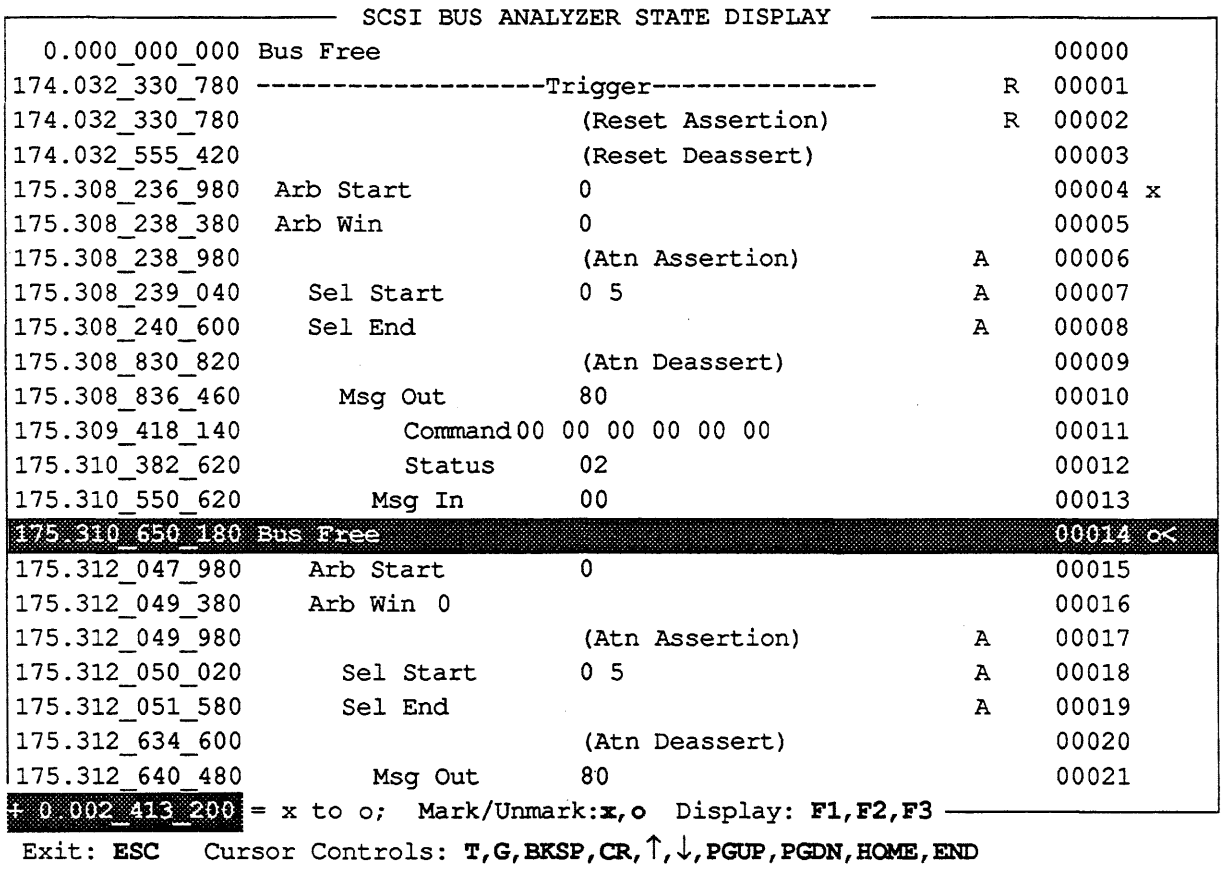

Figure 2-7. Interactive Display

Chapter 3

Interactive Menu

> *NOTE*
>
> *This chapter assumes you have already installed the analyzer hardware and software on your computer. See the* ***Installation*** *chapter of this manual for details.*

Invoking the Menu

There are two ways to invoke the menu:

1. Invoke the menu directly from the DOS command line.
2. Set up your computer to invoke the menu anytime that **ALT-A** is typed from the keyboard.

With either method, if you have installed your analyzer hardware and software correctly, and you have configured your DOS environment properly for analyzer operation, the menu's **Bus Analyzer Functions** page will be displayed:

```
                    SDS-310F SCSI Bus Analyzer (Version 2.0)
                         Copyright 1991, Adaptec, Inc.

STATUS:  IDLE
MESSAGE:

                              BUS ANALYZER FUNCTIONS
    Trigger           [Immediate   |XX X XX XXX X XX|XXXX|0x001|Before]
    Trigger Data      [XXXX XXXX XXXX XXXX]
    Event Filter      [ 8|0ff     ]
    Acq Qualifier     [All State       |0x00000000|0x000000010]
    ID Qualifier      [0|x|x|0|0]
    Config Mode       [Direct  ]
    Output Device     [Printer|DISPLAY.DAT ]
    Data File         [SBADATA ]

 1 Run
 2 Monitor
 3 Display Data       [  0000|  49|Screen |Timing 8 |Real Time ]
 C Clear
                      ANZ                        EXIT
 <-,->,Z,E,^E,
```

Figure 3-1. Bus Analyzer Functions Menu Page

Invoking the Menu Directly From the DOS Command Line

To invoke the menu directly from the DOS command line, type **ANALYZER<Enter>**.

You may specify a parameter file name as a parameter to the **ANALYZER** command. In this case, the menu will automatically configure itself according to the information in this file. If you do not specify an extension with this file name, the extension **.CFG** will be automatically assumed. See the **Parameter File** command description in this chapter for details.

Invoking the Menu With SBATSR and ALT-A

To configure your system to invoke the menu anytime **ALT-A** is typed, type **SBATSR<Enter>** from the DOS command line. Each time you boot your computer, you only need to execute this command once.

SBATSR will load a small (approximately 11 KByte) terminate and stay resident (TSR) program in your computer's memory. After this TSR is loaded, each time you type **ALT-A**, regardless of the application that is running when you type it, the TSR will:

1. Swap out the current application to your hard disk.
2. Invoke the menu.
3. When you exit the menu, reload and resume the application that was executing when you invoked the menu.

This method allows you to invoke the menu from any DOS application, including SCSI emulation utilities such as the SDS-3F Development System's Menu, Interpreter, Shell, or one of your own SCSI C Library or Power User Package user tests.

NOTE

For more information on the SDS-3F Development System and its SCSI emulation utilities, contact your local Adaptec sales representative.

Each time you invoke the menu with **ALT-A**, it will automatically configure itself according to the last Parameter File used. See the **Parameter File** command description in this chapter for details.

> *NOTE*
>
> *If you always save the current parameters each time you exit the menu, the next time you enter the menu it will be configured exactly as you left it. See the the* ***Save and Exit*** *command description for a convenient way to save parameters before exiting.*

Except for the **DOSEDIT** utility, **SBATSR** is not guaranteed to operate correctly when other TSR programs are loaded in memory.

Menu Pages

The menu has two pages: **Bus Analyzer Functions** and **Parameter Save and Exit**. To move between the two pages, use the **right arrow** and **left arrow** keys or:

1. Type **Z** to go to **Bus Analyzer Functions (ANZ)**.
2. Type **E** to go to **Parameter Save and Exit (EXIT)**.

```
                    SDS-310F SCSI Bus Analyzer (Version 2.0)
                         Copyright 1991, Adaptec, Inc.

STATUS:  IDLE
MESSAGE:

                              PARAMETER SAVE AND EXIT

  Working Directory [C:\
  Parameter File    [SBAPARMS.CFG]

P Save Parameters
L Load Parameters
V Save and Exit
I Exit

                         ANZ                        EXIT
<-,->,Z,E,^E,
```

Figure 3-2. Parameter Save and Exit Page

Each menu page has two windows. The top window shows analyzer status, any error messages, and on-line help. Analyzer status is interpreted as follows:

IDLE	The analyzer is not running and the capture buffer is empty.
RUNNING	The analyzer is running but the trigger has not occurred.
TRIGGERED	The analyzer is running, the trigger has occurred, but the capture buffer is not full.
COMPLETE	The analyzer is not running, the trigger has occurred, and the capture buffer is full.

The bottom window shows the analyzer commands and their current parameter values.

If a command is preceded by a highlighted character, that command is executed by typing the key corresponding to that character. That key is called the hot key for the function. If a command is not preceded by a highlighted character, it will execute automatically after its parameters are edited.

Editing Parameters

To enter or exit the parameter edit mode, type **Ctrl-E**. You may also exit the parameter edit mode by typing the **Esc** key.

Use the following keys to move between parameters (on line help is displayed in the top window of the menu page for each parameter you select):

down arrow	Move to the next parameter to the right. If the current parameter is the last parameter of the current command, move to the first parameter of the next command.
Tab	Same as **down arrow**.
Enter	Same as **down arrow**.
up arrow	Move to the next parameter to the left. If the current parameter is the first parameter of the current command, move to the last parameter of the preceding command.
Back Tab	Same as **up arrow**.

PgUp Move to the first parameter of the preceding command.

PgDn Move to the first parameter of the next command.

Home Move to the first parameter of the first command.

End Move to the first parameter of the last command.

There are three types of parameters:

Toggle The values for toggle parameters are predefined. There are two methods of choosing from among these predefined values:

1. Type the **space bar, +, -, right arrow,** or **left arrow** key to toggle between the predefined values.

2. Type **Ctrl-O** to display the list of predefined values in the top menu window. Values are listed in alphabetical order with an arrow indicating the current selection value. Typing the **Enter** key will set the value of the parameter to the current selection value and exit the **Ctrl-O** display. You can move the selection arrow by typing the **up arrow, down arrow, PgUp, PgDn, Home** and **End** keys. You may also start typing the name of the value you want until enough characters have been entered to point the selection arrow to the desired value.

You may hit **ESC** at any time to leave the current parameter value unchanged and exit the **Ctrl-O** display.

Numeric The value of this parameter is a number. Numbers may be entered in decimal (**0, 10, 1054**) or hexadecimal (**0x0, 0xA, 0x41E**).

String The value of this parameter is a string of characters (typically a DOS file or path name). Type the characters you want. If a string already exists, the characters you type will overwrite the current characters starting at the current position. Use the **right arrow** and **left arrow** keys to change the current position. Use the **Del** key to delete the character at the current position and shift all following characters one position left. Use the **Ins** key to enter insert mode. In insert mode, each time you type a character all characters starting at the current position will be shifted right and the character you type will be inserted at the current position.

If you are editing a parameter, but decide you want to return it to its original value, type **Ctrl-R**.

Menu Commands

Command	Description
Acq Qualifier	Configure the acquisition qualifier.
Clear	Stop data capture and clear the capture buffer.
Config Mode	Specify the analyzer configuration.
Data File	Specify a file name for the unformatted capture data.
Display Data	Format and display the capture data.
Event Filter	Configure the event filter.
Exit	Exit the menu.
ID Qualifier	Configure the ID qualifier.
Load Parameters	Load the menu parameters.
Monitor	Monitor the SCSI bus.
Output Device	Specify the device and file for formatted display data.
Parameter File	Specify a file to hold the menu parameters.
Run	Start data capture.
Save and Exit	Save the menu parameters and exit the menu.
Save Parameters	Save the menu parameters.
Trigger	Configure the non-data part of the trigger.
Trigger Data	Configure the data part of the trigger.
Working Directory	Specify the directory to hold all data files.

NOTE

A typical menu session consists of the following steps:

1. *Configure the data capture hardware as appropriate with the* ***Trigger, Trigger Data, Event Filter, Acq Qualifier, and ID Qualifier*** *commands.*
2. *Start the data capture hardware with the* ***Run*** *command.*
3. *Wait for the data capture to complete.*
4. *View the captured data with the* ***Display Data*** *command.*

Acq Qualifier [*mode* | *start* | *count*]

Configure the acquisition qualifier. See the **Acquisition Qualifier** section of this manual for details.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **none**

mode A toggle parameter specifying the acquisition mode:

All State	Capture events of all bus phases.
Arb Sel	Capture events of arbitration, selection, and reselection phases only.
Arb Sel Cmd	Capture events of arbitration, selection, reselection, and command phases only.
Arb Sel Cmd Stat	Capture events of arbitration, selection, reselection, command, and status phases only.

Default: **All State**

start A numeric parameter (**0-4294967295/0x0-0xFFFFFFFF**) specifying the offset of the byte/word in each data in or data out phase to capture.

This parameter is only used in **All State** mode.

Default: **0x0000**

count A numeric parameter (**0-4294967295/0x0-0xFFFFFFFF**) specifying the maximum number of consecutive bytes/words in each data in or data out phase to capture.

This parameter is only used in **All State** mode.

Default: **0x00010**

NOTE

Setting **mode** *to* ***All State*** *and* **count** *to* ***0*** *disables the saving of data phases. In this case, the events of all non-data phases, including messages, will be saved.*

Clear

Stop data capture and clear the capture buffer.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **C**

Config Mode [*mode*]

Specify the analyzer configuration.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **none**

mode A toggle parameter specifying the SCSI bus connection mode:

Direct The SCSI bus is connected to the analyzer PCB's 68-bit 'P' connector either directly or through the SCSI 'P' connector translator (One of configurations 1, 2, 3, 4, 7, 8, 10, or 11 as described in the **Connecting the SCSI Bus Section** of this manual).

Indirect The SCSI bus is connected to another PCB's SCSI connector and that PCB is connected to the analyzer PCB's internal J3 connector (One of configurations 5, 6, 9, 12, or 13 as described in the **Connecting the SCSI Bus Section** of this manual).

Default: The menu software automatically sets this parameter when the menu is first invoked. If in configuration 1, 2, 3, 4, 7, 8, 10, or 11, ***mode*** will be set automatically to **Direct**; otherwise, it will be set to **Indirect**.

NOTE

Change this parameter when you change from the menu start up configuration. Be careful. If this parameter is set incorrectly, each bit of the capture data will be inverted.

Data File [*file*]

Specify a file name for the unformatted capture data. See the **Transporting** section of this manual for details.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **none**

file A string parameter (up to 8 characters) specifying a valid DOS file name without extension. The menu will use this name to create one or more internal files, depending on analyzer use. Each file will have the same file name with a unique extension. These files will be put in the directory specified by the **Working Directory** command.

Default: **SBADATA**

Display Data [*start* | *count* | *device* | *mode* | *time*]

Stop data capture (if not already stopped), and format and display the capture data. See the **Analyzing the Captured Data** section of this manual for details.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **3**

start A numeric parameter (**0-65535/0x0-0xFFFF**) specifying the number of the first line to display.

Default: **0000**

count A numeric parameter (**0-65535/0x0-0xFFFF**) specifying the number of lines to display.

This parameter is used only if the *device* parameter is set to **Out Dev**.

Default: **49**

device A toggle parameter specifying the device to receive the display data:

Screen Display the data interactively on the computer monitor.

Out Dev Write the display data to the device specified with the **Output Device** command.

Default: **Screen**

mode A toggle parameter specifying the display mode:

Timing 16 Timing mode with 16 bit data.

Timing 8 Timing mode with 8 bit data.

State 16 State mode with 16 bit data.

State 8 State mode with 8 bit data.

Default: **Timing 8**

time A toggle parameter specifying the time stamp type:

Real Time The time stamp for each event will show the time from capture start until that event.

Delta Time The time stamp for each event will show the time from the immediately preceding event until that event.

Default: **Real Time**

Event Filter [*width* | *mode*]

Configure the event filter. See the **Event Filter** section of this manual for details.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **none**

width A toggle parameter specifying the SCSI data bus width:

8	The SCSI bus is an 8-bit bus. Process transitions of SCSI data lines D0-D7 and parity bit DP0, but ignore all transitions on SCSI data lines D8-D15 and DP1.
16	The SCSI bus is a 16-bit bus. Process transitions of all SCSI data lines D0-D15 and parity bits DP0 and DP1.
Default:	**8**

NOTE

Setting this parameter to **16** *during analysis of an 8-bit bus will work. However, if any of the analyzer inputs corresponding to the upper 8 data bits is floating, inadvertent transitions may cause unwanted data captures and waste the capture buffer.*

mode A toggle parameter specifying the event filter mode for information transfers:

Off	The event filter is disabled.
Four Per	For each information transfer byte/word, capture events only on control line transitions.
One Per	For each information transfer byte/word, capture events only when data is valid.
Default:	**Off**

Exit

Exit the menu.

Menu Page: **Parameter Save and Exit (EXIT)** Hot Key: **1**

ID Qualifier [*mode* | *iid* | *tid* | *storage* | *trigger*]

Configure the ID qualifier. See the **ID Qualifier** section of this manual for details.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **none**

mode A toggle parameter specifying the ID qualifier mode:

- **0** Disabled.
- **1** Enabled for selections and reselections.
- **2** Enabled for selections only.
- **3** Enabled for reselections only.

Default: **0**

iid A toggle parameter specifying the SCSI initiator ID:

- **0** Initiator ID 0 only.
- **1** Initiator ID 1 only.
- **2** Initiator ID 2 only.
- **3** Initiator ID 3 only.
- **4** Initiator ID 4 only.
- **5** Initiator ID 5 only.
- **6** Initiator ID 6 only.
- **7** Initiator ID 7 only.
- **X** Any initiator ID.

Default: **X**

tid A toggle parameter specifying the SCSI target ID:

- **0** Target ID 0 only.
- **1** Target ID 1 only.
- **2** Target ID 2 only.
- **3** Target ID 3 only.
- **4** Target ID 4 only.
- **5** Target ID 5 only.
- **6** Target ID 6 only.
- **7** Target ID 7 only.
- **X** Any target ID.

Default: **X**

storage A toggle parameter specifying the option for the acquisition qualifier:

- **0** Disabled.
- **1** Enabled without arbitration, selection, and reselection suppression.
- **2** Enabled with arbitration, selection, and reselection suppression.

Default: **0**

trigger A toggle parameter specifying the option for the trigger:

- **0** Disabled.
- **1** Enabled.

Default: **0**

Load Parameters

Load the menu parameters from the file specified with the **Parameter File** command.

Menu Page: **Parameter Save and Exit (EXIT)** Hot Key: **L**

NOTE

If you subsequently exit the menu and reinvoke it with the ALT-A option, the menu will be automatically loaded with this parameter file.

Monitor

Monitor the SCSI bus.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **2**

This command is a toggle: if bus monitoring is disabled, executing **Monitor** will enable it; otherwise, executing **Monitor** will disable it.

When bus monitoring is enabled, the current state of the SCSI bus will be displayed on the bottom line of the computer monitor. SCSI signals that are not asserted will be displayed in normal video, and signals that are asserted will be highlighted. This monitor feature is especially useful to determine the state of a bus that is hung.

This display will be updated automatically whenever the bus state changes; however, because this is a software generated display, there may be delays in the display update.

Output Device [*type* | *file*]

Specify the device and file for formatted display data.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **none**

type A toggle parameter specifying the output device:

Printer The printer.

File A disk file.

Default: **Printer**

file A string parameter (up to 12 characters) specifying a file name with extension. This parameter is only used if the *type* is **File**. The directory specified with the **Working Directory** command will be used.

Default: **DISPLAY.DAT**

Parameter File [*file*]

Specify a file to hold the menu parameters.

Menu Page: **Parameter Save and Exit (EXIT)** Hot Key: **none**

file A string parameter (up to 12 characters) specifying a valid DOS file name with or without extension. If you do not specify a file name extension, **.CFG** will be appended automatically. The directory *SBADRV*:**\SDS\SDS310F\PRM** will be used, where *SBADRV* is the current value of the **SBADRV** DOS environment variable.

Default: **SBAPARMS.CFG**

Run

Start data capture.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **1**

Save and Exit

Save the menu parameters in the file specified with the **Parameter File** command and exit the menu.

Menu Page: **Parameter Save and Exit (EXIT)** Hot Key: **V**

NOTE

If you subsequently invoke the menu with the ALT-A option, the menu will be automatically loaded with this parameter file. This provides a convenient method for ensuring that you will reenter the menu in the same state that you left it.

Save Parameters

Save the menu parameters in the file specified with the **Parameter File** command.

Menu Page: **Parameter Save and Exit (EXIT)** Hot Key: **P**

NOTE

If you subsequently exit the menu and reinvoke it with the ALT-A option, the menu will be automatically loaded with this parameter file.

Trigger [*mode* | *control* | *extern* | *count* | *position*]

Configure the nondata part of the trigger. See the **Trigger** section of this manual for details.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **none**

mode A toggle parameter specifying the trigger mode:

No Trigger	Do not trigger.
Immediate	Trigger when the data capture starts.
Arbitration	Trigger on an arbitration phase.
Selection	Trigger on a selection phase.
Reselection	Trigger on a reselection phase.
Message Out	Trigger on a message out byte transfer.
Command	Trigger on a command byte transfer.
Data Out	Trigger on a data out byte/word transfer.
Data In	Trigger on a data in byte/word transfer.
Status	Trigger on a status byte transfer.
Message In	Trigger on a message in byte transfer.
Specify Bits	Trigger on *control* bus condition.

Default: **Immediate**

control A sequence of 11 toggle parameters (**1**, **0**, or **X**) specifying the required state of the SCSI control signals (left to right): BSY, SEL, ATN, REQ, ACK, C/D, I/O, MSG, RST, DP1, and DP0.

You may also type the value of each of these parameters directly (**1**, **0**, or **X**). In this case, the cursor will move automatically to the next parameter to the right. This method provides an efficient way to set all 11 *control* parameters with less keystrokes.

These parameters are only used in the **Specify Bits** trigger mode.

Default: XX X XX XXX X XX

extern A sequence of four toggle parameters (**1**, **0**, or **X**) specifying the required state (left to right) of external probe pod input wires 3, 2, 1, and 0.

You may also type the value of this parameter directly (**1**, **0**, or **X**). In this case, the cursor will move automatically to the next parameter to the right. This method provides an efficient way to set all four *extern* parameters with less keystrokes.

These parameters are used in every trigger mode except **No Trigger** and **Immediate**.

Default: **XXXX**

count A numeric parameter specifying the number of occurrences (**1-4095/0x1-0xFFF**) of the trigger condition before triggering.

This parameter is used in every trigger mode except **No Trigger** and **Immediate**.

Default: **0x001**

position A toggle parameter specifying the position of the trigger in the capture data:

Before 95% of the capture data follows the trigger event.

Center The trigger event is centered in the capture data.

After 95% of the capture data precedes the trigger event.

This parameter is used in every trigger mode except **No Trigger** and **Immediate**.

Default: **Before**

NOTE

*See also the **Trigger Data** command description.*

Trigger Data [*data*]

Configure the data part of the trigger. See the **Trigger** section of this manual for details.

Menu Page: **Bus Analyzer Functions (ANZ)** Hot Key: **none**

data A sequence of sixteen toggle parameters (**0**, **1**, or **X**) specifying the required state (left to right) of the SCSI D15-D0 data lines.

You may also type the value of each of these parameters directly (**0**, **1**, or **X**). In this case, the cursor will move automatically to the next parameter to the right. This method provides an efficient way to set all sixteen *data* parameters with less keystrokes.

This parameter is used in all trigger modes except **No Trigger** and **Immediate**.

Default: XXXX XXXX XXXX XXXX

NOTE

*See also the **Trigger** command description.*

Working Directory [*directory*]

Specify the directory to hold the files specified by the **Data File** and **Output Device** commands.

Menu Page: **Parameter Save and Exit (EXIT)** Hot Key: **none**

directory A string parameter (up to 52 characters) specifying the name of an existing DOS directory.

Default: The directory from which the analyzer menu was invoked.

This page intentionally left blank.

Chapter 4

Run Time Function Library

NOTE

This chapter assumes you have already installed the analyzer hardware and software on your computer, plus the SDS-3F SCSI C Library. See the ***Installation*** *chapter of this manual for details.*

SDS-310F Functions

The functions in this section are designed to provide the most efficient and effective programming interface to the SDS-310F. If you are creating new user tests that will perform analysis, it is strongly recommended that you use these functions instead of the functions detailed in the **Compatible Functions** section of this chapter, even if you have experience with those other functions.

scsisba.h

The **scsisba.h** header file prototypes the functions detailed in this section. These prototypes are necessary for proper user test compilation. The **scsisba.h** file also defines all program constant names that begin with **SBA_**. As you create your user tests, you should use the names of these constants instead of their values. This will ensure compatibility with future software releases.

NOTE

"C" is case-sensitive, so these constants must be specified in upper case.

You must include this file in your user test source if you are going to call any of these functions or use any of these constants.

Example:

```
#include    <scsisba.h>
/*  scsisba.h prototypes sba_acq_qual and defines SBA_SUCCESS
    and SBA_ALL_STATE. */
user_test()
{
    unsigned int status;
    ...
    if ((status = sba_acq_qual(SBA_ALL_STATE, 0L, 4L)) !=
    SBA_SUCCESS)
        process_sba_error(status);
    ...
}
```

Function Details

Function	Description
sba_acq_qual	Configure the acquisition qualifier.
sba_clear	Stop data capture and clear the capture buffer.
sba_config_mode	Specify the analyzer configuration.
sba_data_file	Specify a file name for the unformatted capture data.
sba_delta_time	Calculate the delta time between two lines.
sba_display_data	Format and display the captured data.
sba_event_filter	Configure the event filter.
sba_id_qual	Configure the ID qualifier.
sba_interactive_display	Display data interactively to screen.
sba_log_length	Get the number of lines in the capture buffer.
sba_resolution	Set resolution of returned time values.
sba_run	Start data capture.
sba_scsi_bus	Get timing line SCSI signal values.
sba_state_data	Get state line data value.
sba_state_line	Find a state line.
sba_status	Get the analyzer status.
sba_timing_line	Find a timing line.
sba_trigger	Configure the trigger.
sba_xlate	Translate line number from one mode to the other.

NOTE

A typical user test analysis consists of the following steps:

1. *Configure the data capture hardware as appropriate with the* ***sba_acq_qual, sba_event_filter, sba_id_qual,*** *and* ***sba_trigger*** *functions.*
2. *Start the data capture hardware with the* ***sba_run*** *function.*
3. *Wait for the data capture to complete.*
4. *Analyze the captured data with the* ***sba_delta_time sba_display_data, sba_interactive_display, sba_log_length, sba_resolution, sba_scsi_bus, sba_state_data, sba_state_line, sba_timing_line,*** *and* ***sba_xlate*** *functions.*

unsigned int sba_acq_qual(unsigned int *mode*, unsigned long *start*, unsigned long *count*);

Configure the acquisition qualifier. See the **Acquisition Qualifier** section of this manual for details.

mode The acquisition mode:

SBA_ALL_STATE	Capture events of all bus phases.
SBA_ARB_SEL	Capture events of arbitration, selection, and reselection phases only.
SBA_ARB_SEL_CMD	Capture events of arbitration, selection, reselection, and command phases only.
SBA_ARB_SEL_CMD_STAT	Capture events of arbitration, selection, reselection, command, and status phases only.

Default: **SBA_ALL_STATE.**

start The offset (**0L-4294967295/0x0L-0xFFFFFFFFL**) of the first byte or word in each data phase to capture.
This parameter is only used in **SBA_ALL_STATE** mode.
Default: **0L.**

count The maximum number (**0L-4294967295/0x0L-0xFFFFFFFFL**) of consecutive bytes/words in each data in or data out phase to capture.
This parameter is only used in **SBA_ALL_STATE** mode.
Default: **16L.**

> *NOTE*
>
> *Setting* **mode** *to SBA_ALL_STATE and* **count** *to 0L disables the saving of data phases. In this case, the events of all non-data phases, including messages, will be saved.*

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_START	Illegal *start* value. All parameters returned to default values.
SBA_BAD_COUNT	Illegal *count* value. All parameters returned to default values.
SBA_BAD_MODE	Illegal *mode* value. All parameters returned to default values.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Capture all phases, but only the second 10 bytes of each data phase. */
unsigned int status;
status = sba_acq_qual(SBA_ALL_STATE, 10L, 10L);
```

unsigned int sba_clear(void);

Stop data capture and clear the capture buffer.

Return Values:

SBA_SUCCESS	Successful execution.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Stop data capture and discard the data. */
unsigned int status;
status = sba_clear();
```

unsigned int sba_config_mode (unsigned int *mode*)

Specify the analyzer configuration.

Parameters:

mode The SCSI bus connection mode:

SBA_DIRECT The SCSI bus is connected to the analyzer PCB's 68-bit 'P' connector either directly or through the SCSI 'P' connector translator (One of configurations 1, 2, 3, 4, 7, 8, 10, or 11 as described in the **Connecting the SCSI Bus Section** of this manual).

SBA_INDIRECT The SCSI bus is connected to another PCB's SCSI connector and that PCB is connected to the analyzer PCB's internal J3 connector (One of configurations 5, 6, 9, 12, or 13 as described in the **Connecting the SCSI Bus Section** of this manual).

Default: The RTFL software automatically sets this parameter when the user test starts execution. If in configuration 1, 2, 3, 4, 7, 8, 10, or 11, ***mode*** will be set automatically to **SBA_DIRECT**; otherwise, it will be set to **SBA_INDIRECT**.

NOTE

Call this function after you change from the user test start up configuration. Be careful. If this parameter is set incorrectly, each bit of the capture data will be inverted.

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_MODE	Illegal ***mode*** value. Set to default.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Change cable mid-SAT to a differential configuration. */
unsigned int status;
status = sba_config_mode(SBA_INDIRECT);
```

unsigned int sba_data_file(char *file);

Specify a file name for the unformatted capture data. See the **Analyzing the Captured Buffer** section of this manual for details.

Parameters:

file DOS file name for unformatted captured data. Standard DOS file name conventions apply. An extension does not need to be specified, as more than one temporary file may be generated depending on analyzer use. If an extension is specified, it will be ignored. A path may be specified as part of the file name. If a path is not specified, the current directory will be used.

Default: **"SBADATA".**

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_FILE	Illegal *file* value. Returned to default.
SBA_FILE_ERR	File could not be opened. *file* returned to default.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Rename raw data file for later reference. */
unsigned int status;
status = sba_data_file("D:\USER\AD060792");
```

unsigned int sba_delta_time(unsigned int *mode*, unsigned int *line1*, unsigned int *line2*, unsigned long **time*)

Calculate the delta time between two lines. See the **Capture Buffer** section of this manual for details on delta times. See the **Analyzing the Captured Data** section of this manual for details on timing and state lines.

Parameters:

mode	The analysis mode:	
	SBA_TIMING	Calculate the delta time between two timing lines.
	SBA_STATE	Calculate the delta time between two state lines.
line1	The number of the first line to use in the delta time calculation.	
line2	The number of the second line to use in the delta time calculation.	
time	Returns with the delta time value. Units are as specified by the **sba_resolution** function.	

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_MODE	Illegal *mode* value.
SBA_NO_LINE1	Line *line1* not found.
SBA_NO_LINE2	Line *line2* not found.
SBA_NO_DATA	Capture buffer is empty.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Get time between the 1st and 10th events. */
unsigned long delta_time;
unsigned int status;
status = sba_delta_time(SBA_TIMING, 0, 9, &delta_time);
```

unsigned int sba_display_data(unsigned int *start*, unsigned int *count*, unsigned int *mode*, unsigned int *time*);

Stop data capture (if not already stopped), format, and display the captured data. See the **Analyzing the Captured Data** section of this manual for details.

Parameters:

start	The number (0-65535/0x0-0xFFFF) of the first line to display.	
count	The number (0-65535/0x0-0xFFFF) of lines to display.	
mode	The display mode:	
	SBA_TIMING_16	Timing mode with 16-bit data.
	SBA_TIMING_8	Timing mode with 8-bit data.
	SBA_STATE_16	State mode with 16-bit data.
	SBA_STATE_8	State mode with 8-bit data.
time	The time stamp type:	
	SBA_REAL_TIME	The time stamp for each event will show the time from capture start until that event.
	SBA_DELTA_TIME	The time stamp for each event will show the time from the immediately preceding event until that event.

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_MODE	Illegal ***mode*** value.
SBA_BAD_TIME	Illegal ***time*** value.
SBA_TOO_FEW_LINES	Less than ***start*** lines in capture buffer.
SBA_NO_DATA	Capture buffer is empty.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Display 16-bit timing data with elapsed time stamps to standard
    output. */
unsigned int result;
result = sba_display_data(0, 0xFFFF, SBA_TIMING_16,
    SBA_REAL_TIME);
```

unsigned int sba_event_filter(unsigned int *width*, unsigned int *mode*);

Configure the event filter. See the **Event Filter** section of this manual for details.

Parameters:

width The SCSI data bus width:

8	The SCSI bus is an 8-bit bus. Process transitions of SCSI data lines D0-D7 and parity bit DP0, but ignore all transitions on SCSI data lines D8-D15 and DP1.
16	The SCSI bus is a 16-bit bus. Process transitions of all SCSI data lines D0-D15 and parity bits DP0 and DP1.

Default: **8.**

> *NOTE*
>
> *Setting this parameter to* ***16*** *during analysis of an 8-bit bus will work. However, if any of the analyzer inputs corresponding to the upper 8 data bits is floating, inadvertent transitions may cause unwanted data captures and waste the capture buffer.*

mode Specify the event filter mode for information transfers:

SBA_OFF	The event filter is disabled.
SBA_FOUR_PER	For each information transfer byte/word, capture events only on control line transitions.
SBA_ONE_PER	For each information transfer byte/word, capture events only when data is valid.

Default: **SBA_OFF.**

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_WIDTH	Illegal ***width*** value. Both parameters returned to default values.
SBA_BAD_MODE	Illegal ***mode*** value. Both parameters returned to default values.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* On 8-bit bus, suppress all SCSI data and parity line transitions. */
unsigned int status;
status = sba_event_filter(8, SBA_FOUR_PER);
```

unsigned int sba_id_qual(unsigned int *mode*, unsigned int *iid*, unsigned int *tid*, unsigned int *storage*, unsigned int *trigger*);

Configure the ID qualifier. See the **ID Qualifier** section of this manual for details.

Parameters:

mode The ID qualifier mode:

SBA_OFF	Disabled.
SBA_SEL_RESEL	Enabled for selections and reselections.
SBA_SEL	Enabled for selections only.
SBA_RESEL	Enabled for reselections only.

Default: **SBA_OFF.**

iid The SCSI initiator ID:

0	Initiator ID 0 only.
1	Initiator ID 1 only.
2	Initiator ID 2 only.
3	Initiator ID 3 only.
4	Initiator ID 4 only.
5	Initiator ID 5 only.
6	Initiator ID 6 only.
7	Initiator ID 7 only.
SBA_ANY	Any initiator ID.

Default: **SBA_ANY.**

tid The SCSI target ID:

0	Target ID 0 only.
1	Target ID 1 only.
2	Target ID 2 only.
3	Target ID 3 only.
4	Target ID 4 only.
5	Target ID 5 only.
6	Target ID 6 only.
7	Target ID 7 only.
SBA_ANY	Any target ID.

Default: **SBA_ANY.**

storage The option for the acquisition qualifier:

SBA_OFF	Disabled.
SBA_ALLOW	Enabled without arbitration, selection, and reselection suppression.
SBA_SUPPRESS	Enabled with arbitration, selection, and reselection suppression.

Default: **SBA_OFF.**

trigger The option for the trigger:

SBA_OFF	Disabled.
SBA_ALLOW	Enabled.

Default: **SBA_OFF.**

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_MODE	Illegal ***mode*** value. All parameters returned to default values.
SBA_BAD_IID	Illegal ***iid*** value. All parameters returned to default values.
SBA_BAD_TID	Illegal ***tid*** value. All parameters returned to default values.
SBA_BAD_STOR	Illegal ***storage*** value. All parameters returned to default values.
SBA_BAD_TRIG	Illegal ***trigger*** value. All parameters returned to default values.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Capture just interactions with target ID 7. */
unsigned int status;
status = sba_id_qual(SBA_SEL_RESEL, SBA_ANY, 7, SBA_ALLOW,
    SBA_OFF);
```

unsigned int sba_interactive_display(unsigned int *start*, unsigned int *mode*, unsigned int *time*);

Stop data capture (if not already stopped), and format and display the captured data. See the **Analyzing the Captured Data** section of this manual for details.

Parameters:

start	The number (0-65535/0x0-0xFFFF) of the first line to display.	
mode	The display mode:	
	SBA_TIMING_16	Timing mode with 16-bit data.
	SBA_TIMING_8	Timing mode with 8-bit data.
	SBA_STATE_16	State mode with 16-bit data.
	SBA_STATE_8	State mode with 8-bit data.
time	The time stamp type:	
	SBA_REAL_TIME	The time stamp for each event will show the time from capture start until that event.
	SBA_DELTA_TIME	The time stamp for each event will show the time from the immediately preceding event until that event.

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_MODE	Illegal ***mode*** value.
SBA_BAD_TIME	Illegal ***time*** value.
SBA_TOO_FEW_LINES	Less than ***start*** lines in capture buffer.
SBA_NO_DATA	Capture buffer is empty.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Display 16-bit timing data with elapsed time stamps to standard
output. */
unsigned int result;
result = sba_interactive_display( 0,SBA_TIMING_16, SBA_REAL_TIME);
```

unsigned int sba_log_length(unsigned int *mode*, unsigned int **lines*);

Stop data capture (if not already stopped) and get the number of lines in the capture buffer. See the **Analyzing the Captured Data** section of this manual for details on lines.

Parameter:

mode	The analysis mode:	
	SBA_TIMING	Get the number of timing lines.
	SBA_STATE	Get the number of state lines.
lines	Returns with the number of lines.	

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_MODE	Illegal *mode* value.
SBA_NO_DATA	Capture buffer is empty.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Get the number of events in the capture buffer. */
unsigned int length;
unsigned int status;
status = sba_log_length(SBA_TIMING, &length);
```

unsigned int sba_resolution(unsigned int *res*);

Set resolution of returned time values.

Parameter:

res Resolution:

SBA_SEC	Seconds.
SBA_MSEC	Milliseconds.
SBA_USEC	Microseconds.
SBA_NSEC_100	100 nanoseconds.
SBA_NSEC_20	20 nanoseconds.
SBA_NSEC	Nanoseconds.

Default: **SBA_NSEC_20.**

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_RES	Illegal *res* value. Returned to default value.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Set time resolution to nanoseconds. */
unsigned int status;
status = sba_resolution(SBA_NSEC);
```

unsigned int sba_run(void);

Start data capture.

Return Values:

SBA_SUCCESS	Successful execution.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Start data capture. */
unsigned int status;
status = sba_run();
```

unsigned int sba_scsi_bus(unsigned int *line*, unsigned long **control*, unsigned long **data*);

Get timing line SCSI signal values. See the **Analyzing the Captured Data** section of this manual for details on timing lines.

Parameter:

line The time line number whose signal values to return.

control Returns as follows:

Bit pairs 30/31, 28/29, 26/27, 24/25, 22/23, 20/21, 18/19, 16/17, and 14/15 corresponding to the SCSI BSY, SEL, ATN, REQ, ACK, C/D, I/O, MSG, and RST signals, respectively. Bit pairs are interpreted as follows:

msb	lsb	Interpretation
1	0	Asserted
1	1	Transition from asserted to deasserted
0	0	Deasserted.
0	1	Transition from deasserted to asserted

Bit 13 corresponds to the trigger and is **1** if the specified line is the trigger event; **0** otherwise.

Bits 9 and 8 correspond to the SCSI DP0 and DP1 signals, and Bits 3 to 0 correspond to the EXT3, EXT2, EXT1, and EXT0 signals, respectively. **1** is interpreted as asserted, and **0** as deasserted.

data Returns as follows:

Bits 15 to 0 correspond to the SCSI D15 to D0 lines, respectively. **1** is interpreted as asserted, and **0** as deasserted.

Return Values:

SBA_SUCCESS	Successful execution.
SBA_TOO_FEW_LINES	Less than *line* lines in capture buffer.
SBA_NO_DATA	Capture buffer is empty.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Get the SCSI signal values for timing line 100. */
unsigned long control;
unsigned long data;
unsigned int status;
status = sba_scsi_bus(100, &control, &data);
```

unsigned int sba_state_data(unsigned int *start*, unsigned int *state*, unsigned int *count*, unsigned int *offset*, unsigned long **data*);

Get state line data value. See the **Analyzing the Captured Data** section of this manual for details on state lines.

Parameters:

start The starting location (reference point) and direction of the search for the state line to get the information transfer byte from:

SBA_START_OF_LOG	Search forward from the first captured event.
SBA_END_OF_LOG	Search backward from the last captured event.
SBA_AFTER_TRIGGER	Search forward from the trigger event.
SBA_BEFORE_TRIGGER	Search backward from the trigger event.

state The type of the state line to get the information transfer byte/word from:

SBA_ARBITRATE	At least one device is arbitrating.
SBA_ARBWIN	A device has won arbitration.
SBA_SEL_START	A device is selecting another device.
SBA_RESEL_START	A device is reselecting another device.
SBA_MSG_OUT	Message out information transfer(s).
SBA_CMD	Command information transfer(s).
SBA_DATAOUT	Data out information transfer(s).
SBA_DATAIN	Data in information transfer(s).
SBA_STAT	Status information transfer(s).
SBA_MSG_IN	Message in information transfer(s).

count The occurrence count (**1-65535/0x1-0xFFFF**) of the state line of type ***state***, starting from the event and in the direction specified by ***start***, to get the information transfer byte/word from.

offset The offset (**0-4095/0x0-0xFFF**) of the data byte/word within the selected line to return.

This parameter is only used when ***state*** is an information transfer phase. If offset is greater than 5, subsequent state lines of the same information transfer phase will be searched.

data Returns as follows:

Bits 15 to 0 correspond to the SCSI D15 to D0 lines, respectively. **1** is interpreted as asserted, and **0** as deasserted.

Return Values:

SBA_SUCCESS	Successful execution.
SBA_NO_STATE	Phase specified by ***state*** not found.
SBA_BAD_STATE	Illegal ***state*** specifier.
SBA_BAD_START	Illegal ***start*** specifier.
SBA_NO_TRIG	No trigger event.
SBA_OBJ_NOT_FOUND	Data byte/word not found.
SBA_NO_DATA	Capture buffer is empty.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Get data of first data byte of first data in phase. */
unsigned long data;
unsigned int status;
status = sba_state_data(SBA_START_OF_LOG, SBA_DATAIN, 1, 1,
    &data);
```

unsigned int sba_state_line(unsigned int *start*, unsigned int *dirc*, unsigned int *state*, unsigned int *count*, unsigned int **line*);

Find a state line. See the **Analyzing the Captured Data** section of this manual for details on state lines.

Parameters:

start Number (**0-65535/0x0-0xFFFF**) of the first state line to search.

dirc Search direction:

SBA_FORWARD Search forward.
SBA_BACKWARD Search backward.

state The type of the line to search for:

SBA_BUS_FREE	The SCSI bus is idle.
SBA_ARBITRATE	At least one device is arbitrating.
SBA_ARBWIN	A device has won arbitration.
SBA_SEL_START	A device is selecting another device.
SBA_SEL_COMPLETE	A device has selected another device.
SBA_SEL_TIMEOUT	Selection time out.
SBA_RESEL_START	A device is reselecting another device.
SBA_RESEL_COMPLETE	A device has reselected another device.
SBA_RESEL_TIMEOUT	Reselection time out.
SBA_MSG_OUT	Message out information transfer(s).
SBA_CMD	Command information transfer(s).
SBA_DATAOUT	Data out information transfer(s).
SBA_DATAIN	Data in information transfer(s).
SBA_STAT	Status information transfer(s).
SBA_MSG_IN	Message in information transfer(s).
SBA_ATTN_ASSERT	ATN changed from deassert to assert.
SBA_ATTN_DEASSERT	ATN changed from assert to deassert.
SBA_RESET_ASSERT	RST changed from deassert to assert.
SBA_RESET_DEASSERT	RST changed from assert to deassert.
SBA_TRIGGER	The trigger event.

count The number of occurrences (**1-65536/0x1-0xFFFF**) of state lines of type ***state***, starting from the line determined by ***start*** and in direction ***dirc***, to find before returning the line number.

line Returns with the requested line number.

NOTE

If you set **state** *to SBA_TRIGGER, you should also set* **count** *to 1. There is at most one trigger event.*

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_STATE	Illegal *state* value.
SBA_BAD_DIRC	Illegal *dirc* value.
SBA_NO_DATA	Capture buffer is empty.
SBA_OBJ_NOT_FOUND	Line specified by parameters not found.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Get the trigger line number. */
unsigned int line;
unsigned int status;
status = sba_state_line(0, SBA_FORWARD, SBA_TRIGGER, 1, &line);
```

unsigned int sba_status(unsigned int *status);

Get the analyzer status.

Parameters:

status Returns with the analyzer status:

SBA_IDLE	Not running.
SBA_RUN	Running but the trigger has not occurred.
SBA_TRIG	Running and the trigger has occurred, but the buffer is not full.
SBA_FULL	Not running, the trigger has occurred, and the buffer is full.

Return Values:

SBA_SUCCESS	Analyzer found.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Get analyzer status. */
unsigned int status;
unsigned int sba;
status = sba_status(&sba);
```

unsigned int sba_timing_line(unsigned int *start*, unsigned int *dirc*, char **control*, char **data*, unsigned int *trigger*, unsigned int *count*, unsigned int **line*);

Find a timing line. See the **Analyzing the Captured Data** section of this manual for details on timing lines.

Parameters:

start Number (0-65535/0x0-0xFFFF) of the first timing line to search.

dirc Search direction:

SBA_FORWARD	Search forward.
SBA_BACKWARD	Search backward.

control A 16 character string with characters 0, 1, 3, 5, 6, 8, 9, 10, and 12 corresponding to the SCSI BSY, SEL, ATN, REQ, ACK, C/D, I/O, MSG, and RST control signals, respectively. Each character specifies the required state of the corresponding signal: '1' (asserted), 'P' (deasserted to asserted transition), '0' (deasserted), 'N' (asserted to deasserted transition) or 'X' (any state).

Characters 14 and 15 correspond to the SCSI DP1 and DP0 signals, respectively. Each character specifies the required state of the corresponding signal: '1' (asserted), '0' (deasserted), or 'X' (any state).

data A 19 character string with characters 0, 1, 2, 3, 5, 6, 7, 8, 10, 11, 12, 13, 15, 16, 17, and 18 corresponding the to SCSI D15-D0 signals, respectively. Each character specifies the required state of the corresponding signal: '1' (asserted), '0' (deasserted), or 'X' (any state).

trigger Trigger search enable:

SBA_YES	Search for the trigger.
SBA_NO	Do not search for the trigger.

count The number of occurrences (0-65535/0x0-0xFFFF) of timing lines of the type matching ***control***, ***data***, and ***trigger*** to find before returning the line number.

> *NOTE*
>
> *If you want to find the trigger, you should also set* **count** *to 1. There is at most one trigger event.*

line Returns with the number of the requested line.

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_DATA	Illegal *data* value.
SBA_BAD_TRIG	Illegal *trig* value.
SBA_BAD_CNTL	Illegal *control* value.
SBA_BAD_DIRC	Illegal *dirc* specifier.
SBA_NO_DATA	Capture buffer is empty.
SBA_OBJ_NOT_FOUND	Line specified by parameters not found.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Find the 512th data out transfer. */
unsigned int line;
unsigned int status;
status = sba_timing_line(0, SBA_FORWARD, "XX X XP 000 X XX",
    "XXXX XXXX XXXX XXXX", SBA_NO, 512, &line);
```

unsigned int sba_trigger(unsigned int *mode*, char **control*, char **data*, char **external*, unsigned int *count*, unsigned int *position*);

Configure the trigger. See the **Trigger** section of this manual for details.

Parameters:

mode The trigger mode:

SBA_NO_TRIGGER	Do not trigger.
SBA_IMMEDIATE	Trigger when the data capture starts.
SBA_ARBITRATION	Trigger on an arbitration phase.
SBA_SELECTION	Trigger on a selection phase.
SBA_RESELECTION	Trigger on a reselection phase.
SBA_MESSAGE_OUT	Trigger on a message out byte transfer.
SBA_COMMAND	Trigger on a command byte transfer.
SBA_DATA_OUT	Trigger on a data out byte/word transfer.
SBA_DATA_IN	Trigger on a data in byte/word transfer.
SBA_STATUS	Trigger on a status byte transfer.
SBA_MESSAGE_IN	Trigger on a message in byte transfer.
SBA_SPECIFY_BITS	Trigger on *control* bus condition.

Default: **SBA_IMMEDIATE.**

control A 16 character string with characters 0, 1, 3, 5, 6, 8, 9, 10, 12, 14, and 15 corresponding to the SCSI BSY, SEL, ATN, REQ, ACK, C/D, I/O, MSG, RST, DP1 and DP0 control signals, respectively. Each character specifies the required state of the corresponding signal: **'1'** (asserted), **'0'** (deasserted), or **'X'** (any state).

This parameter is only used in the **SBA_SPECIFY_BITS** trigger mode.

Default: **"XX X XX XXX X XX"**

data A 19 character string with characters 0, 1, 2, 3, 5, 6, 7, 8, 10, 11, 12, 13, 15, 16, 17, and 18 corresponding the to SCSI D15-D0 signals, respectively. Each character specifies the required state of the corresponding signal: **'1'** (asserted), **'0'** (deasserted), or **'X'** (any state).

This parameter is used in all trigger modes except **SBA_NO_TRIGGER** and **SBA_IMMEDIATE.**

Default: **"XXXX XXXX XXXX XXXX".**

external A four character string with characters 0, 1, 2, and 3 corresponding to the EXT3, EXT2, EXT1, and EXT0 external probe pod input signals, respectively. Each character specifies the required state of the corresponding signal: '1' (asserted), '0' (deasserted), or 'X' (any state).

This parameter is used in every trigger mode except SBA_NO_TRIGGER and SBA_IMMEDIATE.

Default: "XXXX".

count The number of occurrences (1-4095/0x1-0xFFF) of the trigger condition before triggering.

This parameter is used in every trigger mode except SBA_NO_TRIGGER and SBA_IMMEDIATE.

Default: 1.

position The position of the trigger in the capture data:

SBA_BEFORE	95% of the capture data follows the trigger event.
SBA_CENTERED	The trigger event is centered in the capture data.
SBA_AFTER	95% of the capture data precedes the trigger event.

This parameter is used in every trigger mode except SBA_NO_TRIGGER and SBA_IMMEDIATE.

Default: SBA_BEFORE.

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_MODE	Illegal ***mode*** value. All parameters returned to default values.
SBA_BAD_CNTL	Illegal ***control*** value. All parameters returned to default values.
SBA_BAD_DATA	Illegal ***data*** value. All parameters returned to default values.
SBA_BAD_INPUTS	Illegal ***inputs*** value. All parameters returned to default values.
SBA_BAD_COUNT	Illegal ***count*** value. All parameters returned to default values.
SBA_BAD_POS	Illegal ***position*** value. All parameters returned to default values.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Trigger on an arbitration by ID 0. */
unsigned int status;
status = sba_trigger(SBA_SPECIFY_BITS, "10 0 00 000 0 XX", "XXXX
    XXXX XXXX XXX1", "XXXX", 1, SBA_AFTER);
...
/*Trigger immediate. */
Status=(SBA_IMMEDIATE," "," "," ",0,0);
```

unsigned int sba_xlate(unsigned int *mode*, unsigned int *line*, unsigned int **new*)

Translate line number from one mode to the other. See the **Analyzing the Captured Data** section of this manual for details on timing and state lines.

Parameters:

mode	The translate mode:	
	SBA_TIMING	Translate from state line number to timing line number.
	SBA_STATE	Translate from timing line number to state line number.
line	The number of the line to translate.	
new	Returns with the new line number.	

Return Values:

SBA_SUCCESS	Successful execution.
SBA_BAD_MODE	Illegal *mode* value.
SBA_OBJ_NOT_FOUND	*line* not found.
SBA_NOT_FOUND	Analyzer not found.

Example:

```
/* Find the timing line that corresponds to state line 1000. */
unsigned int line;
unsigned int status;
status = sba_xlate(SBA_TIMING, 1000, &line);
```

Compatible Functions

The following functions are included only to support compatibility with the function libraries of the SDS-210, SDS-310, and SDS-310A. These functions do not support any of the new features of the SDS-310F. It is strongly recommended that you use only the **sba**-class functions for any new user test development. Furthermore, do not mix these functions with **sba**-class functions in the same user test; otherwise, unpredictable results will occur. These functions are prototyped in the **scsila.h** include file.

Function	Description
la_acq_qual	Configure the acquisition qualifier for state mode.
la_delta_time	Calculate delta time between two state mode lines.
la_display	Display the capture data.
la_extern_input	Configure the external input trigger.
la_file_name	Specify file for the unformatted capture data.
la_find_line	Find state mode line number.
la_get_status	Get analyzer status.
la_log_length	Get number of lines in the capture buffer.
la_run	Start data capture.
la_state_data	Get state mode data value.
la_trigger	Configure the SCSI bus trigger for state mode.
lat_acq_qual	Configure the acquisition qualifier for timing mode.
lat_delta_time	Calculate delta time between two timing mode lines.
lat_find_line	Find timing mode line.
lat_line_data	Get timing mode line data value.
lat_time_stamp	Get timing mode line time stamp.
lat_trigger	Configure the SCSI bus trigger condition for timing mode.

unsigned int la_acq_qual(char *mode, unsigned int count);

Configure the acquisition qualifier for subsequent data captures with the la_run("STATE") function. See the **Acquisition Qualifier** section of this manual for details.

Parameters:

mode The acquisition mode (state):

"ALL_STATE"	Capture events of all bus phases.
"ARB_SEL"	Capture events of arbitration, selection, and reselection phases only.
"ARB_SEL_CMD"	Capture events of arbitration, selection, reselection, and command phases only.
"ARB_SEL_CMD_STAT"	Capture events of arbitration, selection, reselection, command, and status phases only.

Default: **"ALL_STATE"**.

count The maximum number (**1-61439/0x1-0xEFFF**) of consecutive bytes in each data in or data out phase, starting with byte 0, to save.

Default: **16.**

This parameter is only used in **"ALL_STATE"** mode.

Return Values:

0	Successful execution.
0xFFFE	Illegal ***count*** value. Both parameters returned to default values.
0xFFFF	Illegal ***mode*** value. Both parameters returned to default values.

unsigned long la_delta_time(char *start, char *state1, unsigned int count1, char *state2, unsigned int count2);

Calculate the delta time between two state lines in the capture buffer. This function requires you to have captured data with the **la_run("STATE")** function. See the **Capture Buffer** section of this manual for details on delta times. See the **Analyzing the Captured Data** section of this manual for details on state lines.

Parameters:

start The starting location (reference point) and direction of the search for the first state line to use in the delta time calculation:

"START_OF_LOG"	Search forward from the first captured event.
"END_OF_LOG"	Search backward from the last captured event.
"AFTER_TRIGGER"	Search forward from the trigger event.
"BEFORE_TRIGGER"	Search backward from the trigger event.

state1 The type of the first line to use in the delta time calculation:

"BUS_FREE"	The SCSI bus is idle.
"ARBITRATE"	At least one device is arbitrating.
"ARBWIN"	A device has won arbitration.
"SEL_START"	A device is selecting another device.
"SEL_COMPLETE"	A device has selected another device.
"SEL_TIMEOUT"	Selection time out.
"RESEL_START"	A device is reselecting another device.
"RESEL_COMPLETE"	A device has reselected another device.
"RESEL_TIMEOUT"	Reselection time out.
"MSG_OUT"	Message out information transfer(s).
"CMD"	Command information transfer(s).
"DATAOUT"	Data out information transfer(s).
"DATAIN"	Data in information transfer(s).
"STATUS"	Status information transfer(s).
"MSG_IN"	Message in information transfer(s).
"ATTN_ASSERT"	ATN changed from deassert to assert.
"ATTN_DEASSERT"	ATN changed from assert to deassert.
"RESET_ASSERT"	RST changed from deassert to assert.
"RESET_DEASSERT"	RST changed from assert to deassert.
"LINE"	Special mode. See description below.

count1 The occurrence count (1-65535/0x1-0xFFFF) of the state line of type ***state1***, starting from the event and in the direction specified by ***start***, to use as the first line in the delta time calculation.

state2 The type of the second state line to use in the delta time calculation. Legal values for ***state2*** are the same as those for ***state1***.

count2 The occurrence count (1-65535/0x1-0xFFFF) of state lines of type ***state2***, starting from the the line determined by ***state1*** and ***count1***, and in the forward direction, to use as the second line in the delta time calculation.

If you specify "LINE" for ***state1***, ***count1*** state lines will be counted from the event in the direction specified by ***start*** to determine the first line to use in the delta time calculation. ***count2*** state lines will be counted in the forward direction from the first line in the delta time calculation to use as the second line in the delta time calculation. ***state2*** will be ignored.

Return Values:

non-0L Successful execution. The exact value is the delta time between the two specified lines. Units are microseconds.

0L Error. Call the **get_f_status("IO")** function and interpret its return value as follows:

0x40	Line specified by ***state1/count1*** not found.
0x41	Line specified by ***state2/count2*** not found.
0x42	Illegal ***state*** specifier.
0x43	Illegal ***start*** specifier.
0x44	No trigger event.
0x50	Data not captured with **la_run("STATE")**.
0xFE	Capture buffer is empty.

unsigned int la_display(unsigned int *start*, unsigned int *length*);

Stop data capture (if not already stopped) and display the captured data to the device specified with the **output_device** function. If you captured the data with the **la_run("TIMING")** function, the output will be in the 8-bit real time timing mode format. If you captured the data with the **la_run("STATE")** function, the output will be in the 8-bit real time state mode format. See the **Analyzing the Captured Data** section of this manual for details on the timing and state mode formats.

Parameters:

start	If data was captured with the **la_run("TIMING")** function, the number (**0-65535/0x0-0xFFFF**) of the first timing line to display. If data was captured with **la_run("STATE")**, the number (**0-65535/0x0-0xFFFF**) of the first state line to display.
length	If data was captured with the **la_run("TIMING")** function, the number (**0-65535/0x0-0xFFFF**) of timing lines to display. If data was captured with **la_run("STATE")**, the number (**0-65535/0x0-0xFFFF**) of state lines to display. If *start+length* exceeds the end the buffer, data is displayed to the end of the buffer.

Return Values:

0	Successful execution.
0xFFFE	Less than *start* lines in capture buffer.
0xFFFF	Capture buffer is empty.

unsigned int la_extern_input(char *inputs);

Configure the external input trigger condition for subsequent data captures with the la_run function. See the **Trigger** section of this manual for details.

Parameters:

inputs	A seven character string with characters 0, 2, 4, and 6 corresponding to the EXT3, EXT2, EXT1, and EXT0 external probe pod input signals, respectively. Each character specifies the required state of the corresponding signal: '1' (asserted), '0' (deasserted), or 'X' (any state). Default: "X X X X".

Return Values:

0	Successful execution.
0xFFF9	Illegal *inputs* value. Set to default.

unsigned int la_file_name(char *name);

Specify the name of the file to hold unformatted captured data. See the **Analyzing the Captured Buffer** section of this manual for details on this file and its use.

Parameters:

name	Name of DOS file to hold unformatted captured data. Standard DOS file name conventions apply. A path may be specified as part of the file name. If a path is not specified, the current directory will be used. Extensions will be assigned automatically. If an extension is specified, that extension will be ignored. Default: **SBADATA.**

Return Values:

0	Successfull execution.
0xFFF7	Illegal file name.

unsigned int la_find_line(unsigned int *start*, char **dirc*, char **state*, unsigned int *count*);

Find a state line number. This function requires you to have captured data with the **la_run("STATE")** function. See the **Analyzing the Captured Data** section of this manual for details on state lines.

Parameters:

start Number (0-65535/0x0-0xFFFF) of the first state line to search.

dirc Search direction:

"FORWARD"	Search forward.
"BACKWARDS"	Search backward.

state The type of the line to search for:

"BUS_FREE"	The SCSI bus is idle.
"ARBITRATE"	At least one device is arbitrating.
"ARBWIN"	A device has won arbitration.
"SEL_START"	A device is selecting another device.
"SEL_COMPLETE"	A device has selected another device.
"SEL_TIMEOUT"	Selection time out.
"RESEL_START"	A device is reselecting another device.
"RESEL_COMPLETE"	A device has reselected another device.
"RESEL_TIMEOUT"	Reselection time out.
"MSG_OUT"	Message out information transfer(s).
"CMD"	Command information transfer(s).
"DATAOUT"	Data out information transfer(s).
"DATAIN"	Data in information transfer(s).
"STATUS"	Status information transfer(s).
"MSG_IN"	Message in information transfer(s).
"ATTN_ASSERT"	ATN changed from deassert to assert.
"ATTN_DEASSERT"	ATN changed from assert to deassert.
"RESET_ASSERT"	RST changed from deassert to assert.
"RESET_DEASSERT"	RST changed from assert to deassert.
"TRIGGER"	The trigger event.

count The number of occurrences (**1-65536/0x1-0xFFFF**) of state lines of type ***state,*** starting from the line determined by ***start*** and in direction ***dirc,*** to find before returning the line number.

> *NOTE*
>
> *If you set* **mode** *to "TRIGGER", you should also set* **count** *to 1. There is at most one trigger event.*

Return Values:

<0x8000	Successful execution. The exact value is the number of the specified state line.
0x8042	Illegal ***state*** value.
0x8043	Illegal ***start*** value.
0x8044	Illegal ***dirc*** value.
0x8050	Data not captured with **la_run("STATE")**.
0xFFFE	Capture buffer is empty.
0xFFFF	Line specified by parameters not found.

unsigned int la_get_status(void);

Get analyzer status.

Return Values:

0	Running but the trigger has not occurred.
0x0001	Running and the trigger has occurred, but the buffer is not full.
0x0002	Not running, the trigger has occurred, and the buffer is full.
0x0003	Not running.
0xFFFF	Analyzer not found.

unsigned int la_log_length(void);

Stop the data capture (if not already stopped). If data was captured with the **la_run("TIMING")** function, get the number of timing lines in the capture buffer. If data was captured with the **la_run("STATE")** function, get the number of state lines in the capture buffer. See the **Analyzing the Captured Data** section of this manual for details on timing and state lines.

Return Value:

If data was captured with the **la_run("TIMING")** function, the number timing lines in the capture buffer. If data was captured with **la_run("STATE")**, the number of state lines in the capture buffer.

unsigned int la_run(char *mode);

Start data capture and prepare for analysis in timing or state mode.

> *NOTE*
>
> *Previous generation bus analyzers had a special hardware mode custom tailored for state mode data capture. The SDS-310F does not support this special mode of data capture. Instead, it always captures data as events that translate directly into timing mode output. All state mode data is generated from translations of this timing mode capture data. Because the special state capture mode hardware made more efficient use of the capture buffer than the timing mode hardware, the state mode capture depth of the SDS-310F may be less than previous generation analyzers even though the analyzer's buffer is the same physical size. See the* ***Capturing Data*** *section of this manual for descriptions of the advanced features of the SDS-310F's data capture hardware that may be used to make more efficient use of the capture buffer.*

Parameter:

mode	Data analysis mode:	
	"STATE"	State mode.
	"TIMING"	Timing mode.

Return Values:

0x0000	Successful execution.
0xFFFE	Illegal ***mode*** value.
0xFFFF	Analyzer not found.

unsigned int la_state_data(char *start, char *state, unsigned int count, unsigned int offset);

Get state line data byte value. This function requires you to have captured data with the **la_run("STATE")** function. See the **Analyzing the Captured Data** section of this manual for details on state lines.

Parameters:

start The starting location (reference point) and direction of the search for the state line to get the information transfer byte from:

"START_OF_LOG"	Search forward from the first captured event.
"END_OF_LOG"	Search backward from the last captured event.
"AFTER_TRIGGER"	Search forward from the trigger event.
"BEFORE_TRIGGER"	Search backward from the trigger event.

state The type of the state line to get the information transfer byte from:

"ARBITRATE"	At least one device is arbitrating.
"ARBWIN"	A device has won arbitration.
"SEL_START"	A device is selecting another device.
"RESEL_START"	A device is reselecting another device.
"MSG_OUT"	Message out information transfer(s).
"CMD"	Command information transfer(s).
"DATAOUT"	Data out information transfer(s).
"DATAIN"	Data in information transfer(s).
"STATUS"	Status information transfer(s).
"MSG_IN"	Message in information transfer(s).

count The occurrence count (**1-65535/0x1-0xFFFF**) of the state line of type ***state***, starting from the event and in the direction specified by ***start***, to get the information transfer byte from.

offset The offset (**0-4095/0x0-0xFFF**) of the data byte within the selected line to return.

This parameter is only used when ***state*** is an information transfer phase. If offset is greater than 5, subsequent state lines of the same information transfer phase will be searched.

Return Value:

0x00XX	Where XX is the requested byte.

For status call the **get_f_status("IO")** function and interpret its return value as follows:

0	Successful execution.
0x40	Phase specified by ***state*** not found.
0x42	Illegal ***state*** specifier.
0x43	Illegal ***start*** specifier.
0x44	No trigger event.
0x45	Data byte not found.
0x50	Data not captured with la_run("STATE").
0xFE	Capture buffer is empty.

unsigned int la_trigger(char **state*, char **data*, unsigned int *count*, char **location*);

Configure the SCSI bus trigger condition for subsequent data captures with the la_run("STATE") function. See the Trigger section of this manual for details.

Parameters:

state Trigger state definition:

"NO_TRIGGER"	Do not trigger.
"IMMEDIATE"	Trigger as soon as data capture starts.
"ARBITRATION"	Trigger on an arbitration phase.
"SELECTION"	Trigger on a selection phase.
"RESELECTION"	Trigger on a reselection phase.
"MSG_OUT"	Trigger on a message out byte transfer.
"CMD"	Trigger on a command byte transfer.
"DATA_OUT"	Trigger on a data out byte transfer.
"DATA_IN"	Trigger on a data in byte transfer.
"STATUS"	Trigger on a status byte transfer.
"MSG_IN"	Trigger on a message in byte transfer.

Default: "IMMEDIATE".

data An eleven character string with characters 0, 2, 3, 4, 5, 7, 8, 9, and 10 corresponding to the SCSI DP0, D7, D6, D5, D4, D3, D2, D1, and D0 signals, respectively. Each character specifies the required state of the corresponding signal: '1' (asserted), '0' (deasserted), or 'X' (any state).

Default: "X XXXX XXXX".

count The number of occurrences (0-65535/0x0-0xFFFF) of the trigger condition before triggering.

Default: 1.

location The position of the trigger in the capture data:

"BEFORE"	95% of the capture data follows the trigger.
"CENTERED"	The trigger is centered in the capture data.
"AFTER"	95% of the capture data precedes the trigger.

Default: "BEFORE".

Return Values:

0	Successful execution.
0xFFFC	Illegal *location* value. All parameters returned to default values.
0xFFFD	Illegal *count* value. All parameters returned to default values.
0xFFFE	Illegal *data* value. All parameters returned to default values.
0xFFFF	Illegal *state* value. All parameters returned to default values.

unsigned int lat_acq_qual(char *mode, unsigned int count);

Configure the acquisition qualifier for subsequent data captures with the la_run("TIMING") function. See the **Acquisition Qualifier** section of this manual for details.

Parameters:

mode The acquisition mode (state):

"ALL_STATE"	Capture events of all bus phases.
"ARB_SEL"	Capture events of arbitration, selection, and reselection phases only.
"ARB_SEL_CMD"	Capture events of arbitration, selection, reselection, and command phases only.
"ARB_SEL_CMD_STAT"	Capture events of arbitration, selection, reselection, command, and status phases only.

Default: **"ALL_STATE"**.

count The maximum number (**1-61439/0x1-0xEFFF**) of consecutive bytes in each data in or data out phase, starting with byte 0, to save.

Default: **0x8000**.

This parameter is only used in **"ALL_STATE"** mode.

Return Values:

0	Successful execution.
0xFFFE	Illegal *count* value. Both parameters returned to default values.
0xFFFF	Illegal *mode* value. Both parameters returned to default values.

unsigned long lat_delta_time(unsigned int *line1*, unsigned int *line2*);

Calculate the delta time between two timing lines in the capture buffer. This function requires you to have captured data with the **la_run("TIMING")** function. See the **Capture Buffer** section of this manual for details on delta times. See the **Analyzing the Captured Data** section of this manual for details on timing lines.

Parameters:

line1	The first timing line number to use in the delta time calculation.
line2	The second timing line number to use in the delta time calculation.

Return Values:

non-0L	Successful execution. The exact value is the delta time between the two specified lines. Units are 100 nanoseconds.
0L	Error. Call the **get_f_status("IO")** function and interpret its return value as follows:

0x41	Line *line1* not found.
0x42	Line *line2* not found.
0x50	Capture data not acquired with **la_run("TIMING")**.
0xFE	Capture buffer is empty.

unsigned int lat_find_line(unsigned int *start*, char **dirc*, char **control*, char **data*, char **trigger*, unsigned int *count*);

Find a timing line number. This function requires you to have captured data with the **la_run("TIMING")** function. See the **Analyzing the Captured Data** section of this manual for details on timing lines.

Parameters:

start Number (**0-65535/0x0-0xFFFF**) of the first timing line to search.

dirc Search direction:

"FORWARD"	Search forward.
"BACKWARDS"	Search backward.

control A thirteen character string with characters 0, 1, 3, 5, 6, 8, 9, 10, and 12 corresponding to the SCSI BSY, SEL, ATN, REQ, ACK, C/D, I/O, MSG, and RST control signals, respectively. Each character specifies the required state of the corresponding signal: **'1'** (asserted), **'P'** (deasserted to asserted transition), **'0'** (deasserted), **'N'** (asserted to deasserted transition) or **'X'** (any state).

data An eleven character string with characters 0, 2, 3, 4, 5, 7, 8, 9, and 10 corresponding to the SCSI DP0, D7, D6, D5, D4, D3, D2, D1, and D0 signals, respectively. Each character specifies the required state of the corresponding signal: **'1'** (asserted), **'0'** (deasserted), or **'X'** (any state).

trigger A one character string with **"X"**, **"x"**, or **"1"** specifying a search for the trigger line.

count The number of occurrences (**0-65535/0x0-0xFFFF**) of timing lines of the type matching ***control***, ***data***, and ***trigger*** to find before returning the line number.

> *NOTE*
>
> *If you want to find the trigger, you should also set* **count** *to 1. There is at most one trigger event.*

Return Values:

<0x8000	Successful execution. The exact value is the number of the specified state line.
0x8040	Illegal ***data*** value.
0x8041	Illegal ***trig*** value.
0x8042	Illegal ***control*** value.
0x8043	Illegal ***start*** value.
0x8044	Illegal ***dirc*** specifier.
0x8050	Data not captured with **la_run("TIMING")**.
0xFFFE	Capture buffer is empty.
0xFFFF	Line specified by parameters not found.

unsigned long lat_line_data(unsigned int *line*);

Get time line data and control signal values. This function requires you to have captured data with the **la_run("TIMING")** function. See the **Analyzing the Captured Data** section of this manual for details on timing lines.

Parameter:

line The time line number whose data byte to return.

Return Value:

A 32-bit value.

Bit pairs 30/31, 28/29, 26/27, 24/25, 22/23, 20/21, 18/19, 16/17, and 14/15 corresponding to the SCSI BSY, SEL, ATN, REQ, ACK, C/D, I/O, MSG, and RST signals, respectively. Bit pairs are interpreted as follows:

msb	lsb	Interpretation
1	0	Asserted
1	1	Transition from asserted to deasserted
0	0	Deasserted
0	1	Transition from deasserted to asserted

Bit 13 corresponds to the trigger and is **1** if the specified line is the trigger event; **0** otherwise.

Bits 8 to 0 correspond to the SCSI DP0 and D7 to D0 lines, respectively. **1** is interpreted as asserted, and **0** as deasserted.

Call the **get_f_status("IO")** function and interpret its return value as follows:

0	Successful execution.
0x41	Specified line not found.
0x50	Data not captured with **la_run("TIMING")**.
0xFE	Capture buffer is empty.

unsigned long lat_time_stamp(unsigned int *line*);

Get the time stamp of a time line. This function requires you to have captured data with the **la_run("TIMING")** function. See the **Analyzing the Captured Data** section of this manual for details on timing lines.

Parameter:

line	The time line number whose time stamp to return.

Return Value:

The time stamp of the specified line in 100 nanosecond units.

Call the **get_f_status("IO")** function and interpret its return value as follows:

0	Successful execution.
0x41	Specified line not found.
0x50	Data not captured with **la_run("TIMING")**.
0xFE	Capture buffer is empty.

unsigned int lat_trigger(char *control, char *data, unsigned int count, char *location);

Configure the SCSI bus trigger condition for subsequent data captures with the **la_run("TIMING")** function. See the **Trigger** section of this manual for details.

Parameters:

control A thirteen character string with characters 0, 1, 3, 5, 6, 8, 9, 10, and 12 corresponding to the SCSI BSY, SEL, ATN, REQ, ACK, C/D, I/O, MSG, and RST control signals, respectively. Each character specifies the required state of the corresponding signal: **'1'** (asserted), **'0'** (deasserted), or **'X'** (any state).

Default: "XX X XX XXX X"

data An eleven character string with characters 0, 2, 3, 4, 5, 7, 8, 9, and 10 corresponding to the SCSI DP0, D7, D6, D5, D4, D3, D2, D1, and D0 signals, respectively. Each character specifies the required state of the corresponding signal: **'1'** (asserted), **'0'** (deasserted), or **'X'** (any state).

Default: "X XXXX XXXX"

count The number of occurrences (**0-65535/0x0-0xFFFF**) of timing lines of the type matching ***control*** and ***data*** before triggering.

Default: **1.**

location Location of Trigger within acquisition memory.

"BEFORE"	Acquire about 95% of data after the trigger.
"CENTERED"	Acquire 50% of data after the trigger.
"AFTER"	Acquire about 95% of data before the trigger.

Default: "BEFORE"

Return Values:

0xFFFC	Illegal ***location*** value. All parameters returned to default values.
0xFFFD	Illegal ***count*** value. All parameters returned to default values.
0xFFFE	Illegal ***data*** value. All parameters returned to default values.
0xFFFF	Illegal ***state*** value. All parameters returned to default values.

This page intentionally left blank.

Appendix A

Troubleshooting

SCSI Hot Line

If your problem is not included in the following descriptions, call our Development Systems Operation (DSO) hot line (Monday through Friday between 8:30 AM and 5:30 PM Pacific Coast Time):

DSO Hot Line: (408) 945-2527

Common Problems

Symptom:

The following message is displayed:

```
***  CONFIGURATION ERROR  ***
No SCSI Bus Analyzer Detected
Software Load Aborted
```

Possible Causes:

1. Analyzer PCB not installed.

   *Install the analyzer PCB. See the **Installing the Analyzer PCB** section of this manual for details.*

2. Incorrect **SBAADR** DOS Environment variable value.

   *Change **SBAADR** to the correct value. See the **Configuring the DOS Environment With SDS310F** section of this manual for details.*

3. Incorrect analyzer PCB I/O address setting.

   *Set the correct address. See the **Installing the Analyzer PCB** section of this manual for details.*

4. Analyzer PCB installed in an 8-bit XT-compatible backplane slot.

   *Re-install the analyzer PCB in a 16-bit AT-compatible slot. See the **Installing the Analyzer PCB** section of this manual for details.*

Symptom:

The following message is displayed:

```
***  CANNOT FIND BIT FILES  ***
Software Load Aborted
```

Possible Causes:

1. Incorrect **SBADRV** DOS Environment variable value.

   *Change **SBADRV** to the correct value. See the **Configuring the DOS Environment With SDS310F** section of this manual for details.*

2. Incorrect software installation.

   *Re-install the software. See the **Installing the Software** section of this manual for details.*

Symptom:

One of the following messages is displayed:

```
Cannot Allocate Memory for Control Variables
Software Load Aborted
```
or
```
Insufficient Memory
```

Possible Causes:

1. Less than 640 KBytes memory installed in your computer.

   Increase memory to 640 KBytes.

2. Terminate and stay resident (TSR) programs consuming too much memory.

   Remove all terminate and stay resident (TSR) programs from memory.

Symptom:

Signal values are inverted in the timing display.

Possible Cause:

Incorrect configuration mode set.

Change to the proper mode. If you are using the interactive menu, see the description of the ***Config Mode*** *command in the* ***Interactive Menu*** *chapter of this menu for details. If you are running a user test, see the description of the* ***sba_config_mode*** *function in the* ***Run Time Function Library*** *chapter of this manual for details.*

This page intentionally left blank.

Appendix B

Specifications

Electrical Specifications

- Analyzer PCB:
  Current: 3 amps typical, 5 amps maximum
  Voltage: 5.0 Vdc ±.25 Vdc
- Input channels:
  27 SCSI signals (BSY, SEL, ATN, REQ, ACK, C/D, I/O, MSG, RST, DP0, DP1, D0-D15): As defined in the SCSI-2 standard.
  4 external probe pod input signals (EXT0-3): Standard TTL.
  Resistor SIP RP1 pulls floating signals to an asserted state
  Voltage limitations: 0 Vdc to 5 Vdc
  Signal loading: One 74HCT load plus 20 pF
  Sample depth: 32K per channel
- Output channel (TOUT):
  High-level source current: 3 mA
  Low-level sink current: 24 mA
- External probe pod ground (GND): Signal ground

Timing Specifications

- Sampling resolution: 20 nanoseconds (nsec)
- Time stamp:
  Resolution: 20 nsec
  Time between rollovers: 12.2 hours
- Minimum trigger event duration: 22 nsec
- Minimum time between trigger occurrences: 40 nsec
- External probe pod output assertion: Within 160 nsec of trigger assertion
- External probe pod output deassertion: On capture data display or clear

Environmental Specifications

- Operating temperature: 0°C (32°F) to 55°C (131°F)
- Non-operating temperature: –40°C (–40°F) to 75°C (167°F)
- Humidity: 10% to 95%, noncondensing

Physical Specifications

- Shipping weight: 2 lbs
- PCB: Standard full-size single-width AT form factor (13.25" x 4.75").
- External probe pod dimensions: 3.75" x 2.375" x .875"
- SCSI 'P' connector translator dimensions: 3" x 1.5" x .7"

Analyzer PCB 68-pin 'P' Connector (J1)

- Connector type: Nonshielded, high-density, 68-conductor connector consisting of two rows of 34 female contacts with adjacent contacts 1.27 mm (0.05") apart.

- Pin outs:

Pin	Name	Pin	Name
1	Ground	2	D12
3	Ground	4	D13
5	Ground	6	D14
7	Ground	8	D15
9	Ground	10	DP1
11	Ground	12	D0
13	Ground	14	D1
15	Ground	16	D2
17	Ground	18	D3
19	Ground	20	D4
21	Ground	22	D5
23	Ground	24	D6
25	Ground	26	D7
27	Ground	28	DP0
29	Ground	30	Ground
31	Ground	32	Ground
33	Ground	34	Open
35	Open	36	Open
37	Ground	38	Open
39	Ground	40	Ground
41	Ground	42	ATN
43	Ground	44	Ground
45	Ground	46	BSY
47	Ground	48	ACK
49	Ground	50	RST
51	Ground	52	MSG
53	Ground	54	SEL
55	Ground	56	C/D
57	Ground	58	REQ
59	Ground	60	I/O
61	Ground	62	D8
63	Ground	64	D9
65	Ground	66	D10
67	Ground	68	D11

External Probe Pod 50-pin 'A' Connector

- Connector type: Nonshielded, low-density 50-conductor connector consisting of two rows of 25 male pins with adjacent pins 2.54 mm (0.1") apart.
- Pin outs:

Pin	Name	Pin	Name
1	Ground	2	D0
3	Ground	4	D1
5	Ground	6	D2
7	Ground	8	D3
9	Ground	10	D4
11	Ground	12	D5
13	Ground	14	D6
15	Ground	16	D7
17	Ground	18	DP0
19	Ground	20	Ground
21	Ground	22	Ground
23	Ground	24	Open
25	Open	26	Open
27	Ground	28	Open
29	Ground	30	Ground
31	Ground	32	ATN
33	Ground	34	Ground
35	Ground	36	BSY
37	Ground	38	ACK
39	Ground	40	RST
41	Ground	42	MSG
43	Ground	44	SEL
45	Ground	46	C/D
47	Ground	48	REQ
49	Ground	50	I/O

Minimum Computer Requirements

- PC/AT-compatible
- CGA video controller
- i286 CPU
- 640 KBytes RAM
- 360 KByte 5 1/4", 1.2 MByte 5 1/4", or 1.44 MByte 3 1/2" floppy drive
- Hard disk drive with at least 1 MByte available
- 1, 2, or 3 free backplane slots depending on your configuration.
- One of the following hexadecimal I/O address ranges available: 310-31F, 330-33F, 360-36F, 380-38F
- DOS 3.0

This page intentionally left blank.