



TMS320C3x ***Workstation Emulator***

*Installation
Guide*

1994

Microprocessor Development Systems



TMS320C3x Workstation Emulator Installation Guide

SPRU130
December 1994



IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales offices.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

WARNING

This equipment is intended for use in a laboratory test environment only. It generates, uses, and can radiate radio frequency energy and has not been tested for compliance with the limits of computing devices pursuant to subpart J of part 15 of FCC rules, which are designed to provide reasonable protection against radio frequency interference. Operation of this equipment in other environments may cause interference with radio communications, in which case the user at his own expense will be required to take whatever measures may be required to correct this interference.

TRADEMARKS

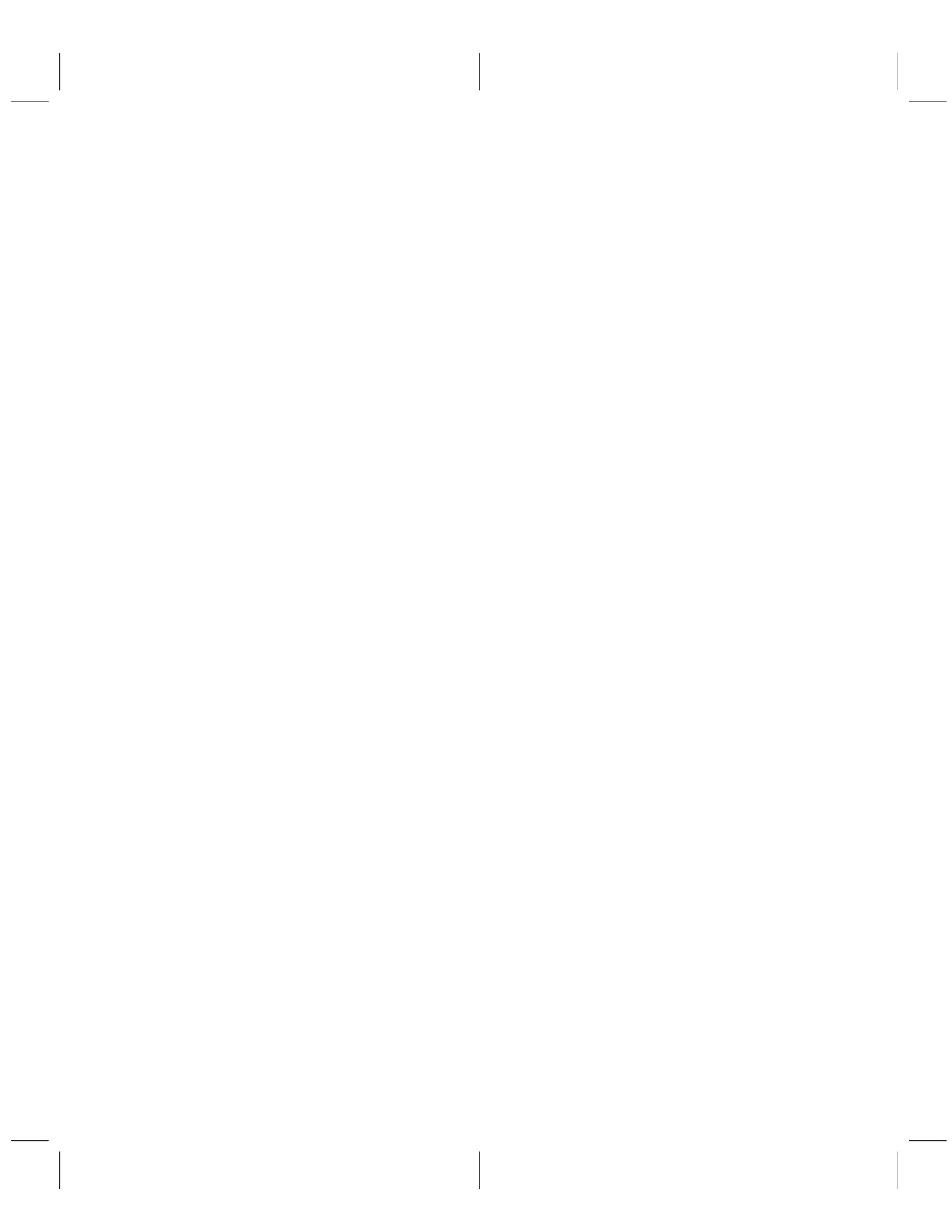
OpenWindows, Solaris, and SunOS are trademarks of Sun Microsystems, Inc.

SPARC is a trademark of SPARC International, Inc.

SPARCstation is licensed exclusively to Sun Microsystems, Inc.

UNIX is a registered trademark of Unix System Laboratories, Inc.

X Window System is a trademark of the Massachusetts Institute of Technology.



Contents

1	Installing the Emulator and C Source Debugger	1-1
	<i>Lists the hardware and software you'll need to install the workstation emulator and C source debugger; provides installation instructions for SPARC systems running SunOS.</i>	
1.1	What You'll Need	1-2
	Hardware checklist	1-2
	Software checklist	1-3
1.2	Step 1: Connecting the Emulator to a Workstation	1-4
	Locating a SCSI bus with an unused identifier	1-6
	Setting the SCSI ID on the emulator	1-7
	Adding the emulator to the SCSI bus	1-8
	Terminating the SCSI bus	1-9
1.3	Step 2: Setting Up a Workstation to Recognize the Emulator	1-10
	Modifying your workstation's configuration file	1-10
1.4	Step 3: Allowing the Debugger to Access the Emulator	1-14
1.5	Step 4: Connecting the Emulator to a Target System	1-14
1.6	Step 5: Installing the Debugger Software	1-15
1.7	Step 6: Making Sure the Emulator Supports the Debugger	1-16
1.8	Step 7: Setting Up the Debugger Environment	1-17
	Invoking the new or modified .cshrc file	1-17
	Modifying the PATH statement	1-17
	Setting up the environment variables	1-18
1.9	Step 8: Verifying the Installation	1-20
1.10	Using the Debugger With the X Window System	1-21
	Using the keyboard's special keys	1-21
	Changing the debugger font	1-22
	Color mappings on monochrome screens	1-22
2	Troubleshooting	2-1
	<i>Describes problems that you may encounter while installing and using the emulator on your workstation.</i>	
2.1	Problems When Booting Your Workstation	2-2
2.2	Problems When Resetting the Emulator	2-3
2.3	Problems When Invoking the Debugger	2-5
2.4	Additional Emulator and Debugger Problems	2-7

3	Interpreting the XDS510WS LEDs	3-1
	<i>Provides information about the eight LEDs on the workstation emulator.</i>	
3.1	XDS510WS LED Descriptions	3-2
	LED 1	3-2
	LED 2	3-3
	LED 3	3-3
	LEDs 4, 5, and 6	3-4
	LEDs 7 and 8	3-5
3.2	XDS510WS LED Reference Chart	3-6

Figures

1-1	Typical Setup of the Emulator on Your Workstation	1-4
1-2	Rear View of the XDS510WS Emulator	1-5
1-3	Front View of the XDS510WS Emulator	1-5
1-4	Connector and Switch Locations on the Front of the XDS510WS Emulator	1-7
1-5	Connecting the Emulator to Your Workstation	1-8
1-6	Connector and Switch Locations on the Rear of the XDS510WS Emulator	1-9
1-7	The External Terminator for SCSI Bus Termination	1-9
1-8	Connecting the Emulator to Your Target System	1-14
3-1	XDS510WS LEDs	3-2
3-2	Standard LED Sequences	3-6

Installing the Emulator and C Source Debugger

This chapter helps you to install the TMS320C3x XDS510WS emulator and the C source debugger on a SPARCstation™ running OpenWindows™ version 3x (or higher) under SunOS™ version 4.1.x (or higher) or Solaris™ version 2.x (or higher). After completing the installation, refer to the *TMS320C3x C Source Debugger User's Guide*.

Topic	Page
1.1 What You'll Need	1-2
1.2 Step 1: Connecting the Emulator to a Workstation	1-4
1.3 Step 2: Setting Up a Workstation to Recognize the Emulator	1-10
1.4 Step 3: Allowing the Debugger to Access the Emulator	1-14
1.5 Step 4: Connecting the Emulator to the Target System	1-14
1.6 Step 5: Installing the Debugger Software	1-15
1.7 Step 6: Making Sure the Emulator Supports the Debugger	1-16
1.8 Step 7: Setting Up the Debugger Environment	1-17
1.9 Step 8: Verifying the Installation	1-20
1.10 Using the Debugger With the X Window System	1-21

1.1 What You'll Need

The following checklists describe items that you'll need in order to use your emulator and debugger. Some of these items are shipped with your tools.

Hardware checklist

- | | | |
|--------------------------|-----------------------------------|--|
| <input type="checkbox"/> | host | A SPARCstation or 100% compatible system with a cartridge tape drive |
| <input type="checkbox"/> | display | Monochrome or color (color is recommended) |
| <input type="checkbox"/> | interface to host | A SCSI bus controller with at least one free SCSI identifier (refer to page 1-6 for more information on locating a free SCSI ID) |
| <input type="checkbox"/> | power supply | The external power supply for the emulator |
| <input type="checkbox"/> | SCSI cable | A SCSI cable used for connecting the emulator to your SPARCstation |
| <input type="checkbox"/> | SCSI terminator | A SCSI bus terminator if your emulator is at the end of the SCSI chain (refer to page 1-9 for more information on the SCSI terminator) |
| <input type="checkbox"/> | emulation cable | A cable that connects the emulator to your target system |
| <input type="checkbox"/> | target system | A board with at least one 'C3x on the emulation scan path |
| <input type="checkbox"/> | connector to target system | A 12-pin connector (two rows of six pins). Refer to Appendix A in the <i>TMS320C3x C Source Debugger User's Guide</i> for more information on the target system. |
| <input type="checkbox"/> | optional hardware | A mouse |

- 1) **To minimize the risk of electric shock and fire hazard, be sure that all major components that you interface with Texas Instruments devices are limited in energy and certified by one or more of the following agencies: UV, CSA, VDE, or TUV.**
- 2) **Turn the power off before you connect components and cables.**
- 3) **Never disconnect or reconnect any cables or other hardware devices while the emulator is turned on.**
- 4) **Be sure all devices on the SCSI bus, including your workstation, are turned off before you connect the emulator to your workstation.**

Software checklist

- operating system** OpenWindows version 3.x (or higher) running under SunOS version 4.1.x (or higher) or Solaris version 2.0 (or higher). If you're using Solaris (also known as SunOS 5.x), you must have the Binary Compatibility Package (BCP) installed; if you don't, get help from your system administrator.
- You *must* have access to root privileges to configure your SPARC; if you don't, get help from your system administrator.
- software tools** TMS320 floating-point DSP C compiler, assembler, and linker
- required files** † *c3x510ws.out*, the executable portion of the debugger that runs on the emulator
- † *emurst*, which resets the emulator and downloads *c3x510ws.out* to the emulator
- optional files** † *emuinit.cmd*, which contains debugger commands and defines a memory map. If this file does not exist when you first invoke the debugger and you don't use the `-t` option, all memory is initially invalid. This memory map should be sufficient for your needs; however, you may want to define your own memory map later. For more information on defining your own memory map, refer to the *Defining a Memory Map* chapter in the *TMS320C3x C Source Debugger User's Guide*.
- † *init.clr* is a general-purpose screen configuration file. If this file isn't present when you invoke the debugger, the debugger uses a default screen configuration.
- † *init.25*, *init.43*, and *init.50* have been provided for basic 80x25, 80x43, and 80x50 screen sizes, respectively. The *init.clr* file brings up the debugger in 80x25 mode. To bring the debugger up in another mode, copy one of the *init.xx* files to the *init.clr* file.
- For more information about these files and about setting up your own screen configuration, refer to the *Customizing the Debugger Display* chapter in the *TMS320C3x C Source Debugger User's Guide*.
- † *mono.clr* is a screen configuration file designed especially for monochrome monitors; the default is for color monitors.

† Included as part of the debugger package

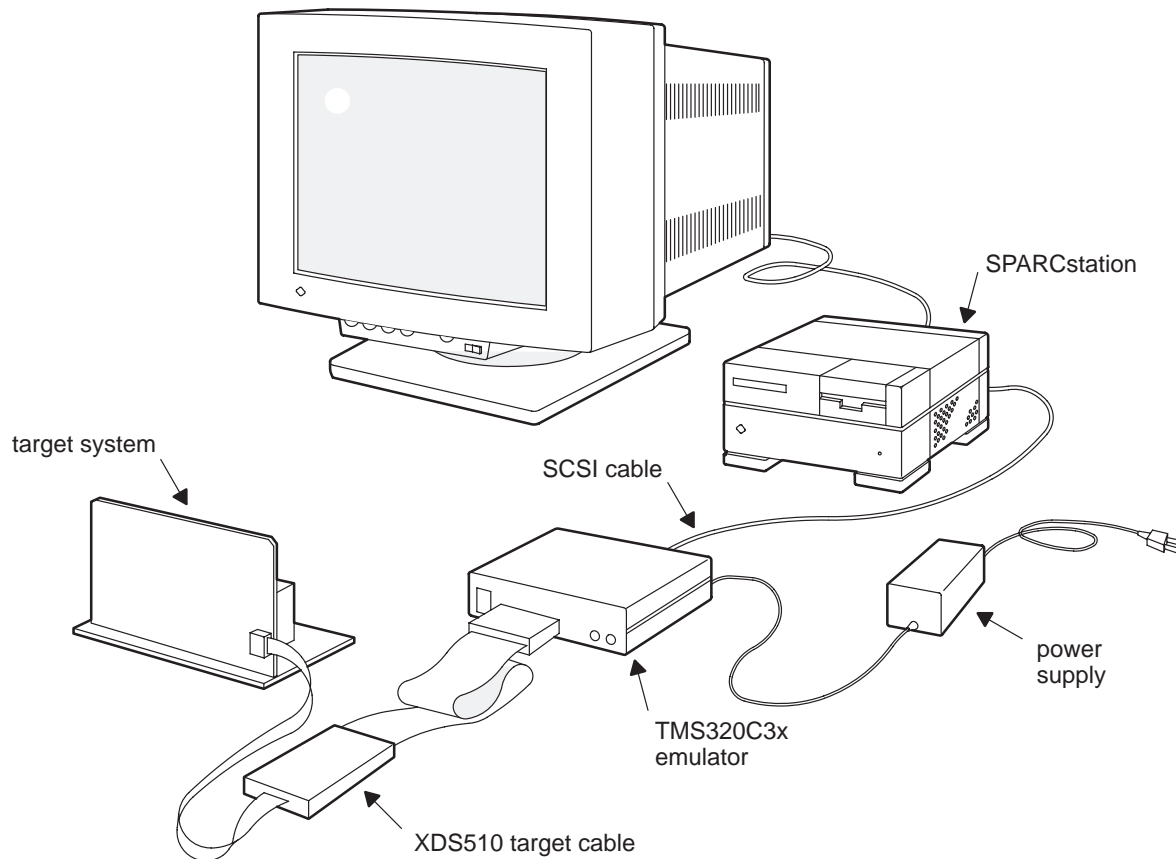
1.2 Step 1: Connecting the Emulator to a Workstation

This section contains hardware installation information for the emulator. You *must* have root access to the host machine you intend to connect to the emulator. If you do not, contact your system administrator.

Figure 1–1 shows a typical setup using the emulator, target cable, and the workstation.



Figure 1–1. Typical Setup of the Emulator on a Workstation



Before you attach the emulator to the workstation, be sure the emulator is working properly. To do this, connect the emulator to the power supply and plug in the power supply (refer to Figure 1–2).

Turn on the emulator. Refer to Figure 1–3 to locate the LEDs on the front of the emulator. When the first LED from the left is lit, the emulator power is on. If this light does not come on, check your power connections and restart the emulator.

Looking at the emulator, you should notice that the sixth LED from the left is on; this indicates that the emulator is running a self-test. The emulator is ready and running properly once the first, second, and fifth LEDs from the left are on and the sixth LED from the left is off.

If the first, second, and fifth indicator lights from the left do not come on, something is wrong with the emulator. Recheck your connections and turn the emulator off and on a second time. If the fifth indicator is still not on, shut off the emulator and contact the hotline.

Figure 1–2. Rear View of the XDS510WS Emulator

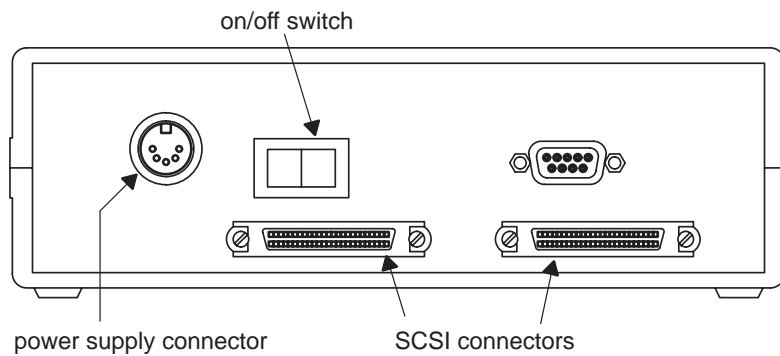
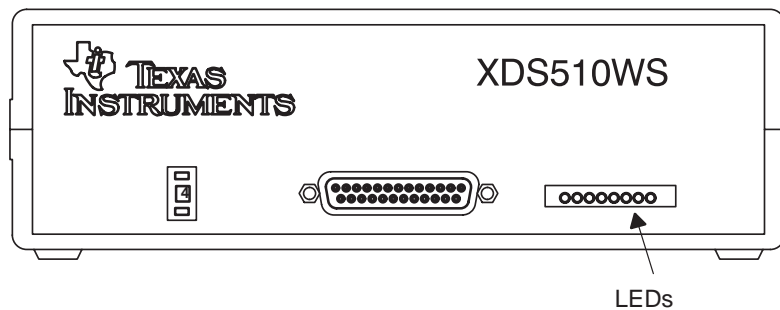


Figure 1–3. Front View of the XDS510WS Emulator




Locating a SCSI bus with an unused identifier

To continue this installation, you *must* have root access on the workstation; if you do not, contact your system administrator.

Each SCSI controller in the workstation has its own SCSI bus, and a workstation usually has only one SCSI controller (unless you have added additional controller cards). A single bus can support up to eight different devices, including the SPARCstation, each uniquely numbered 0 through 7, with the higher priority devices assigned to the larger SCSI ID numbers. A SPARCstation is SCSI ID 7 by default. CD ROMs are ID 6 by default, and cartridge tape drives are usually ID 2. The emulator uses SCSI ID 4 by default. If, however, SCSI ID 4 is already being used, you must change the emulator's ID to one that is not presently in use.


To get a list of the SCSI IDs being used on the workstation, follow these steps:

Step 1: As *root*, enter the following command to get the PROM prompt:



```
halt 
```

If you see the ok prompt, go to Step 3. If you see the > prompt, go to Step 2.

Step 2: After you receive the following message:

```
Program terminated
Type b(boot), c(continue), or n(new command mode)
>
type n .
```

Step 3: After you receive the following message:

```
Type help for more information
ok
type probe-scsi  (or, if you have multiple SCSI controllers, use
probe-scsi-all .
```

You should see a list of the SCSI IDs which are in use scroll on your screen; each device ID in the list will look like the following message:

```
Target 3
  Unit 0 disk SEAGATE ST1480 SUN Copyright (c) 1992
  Seagate all rights reserved 0000
ok
```

The number following each occurrence of the word *Target* represents the currently used SCSI IDs. In the above message, SCSI ID number 3 is used.

Note that the SPARCstation's SCSI ID is stored in the PROM environment variable *scsi_initiator_id* and can be viewed at this point when you type **printenv**.

Setting the SCSI ID on the Emulator

If the SPARCstation is already using SCSI ID 4 (see the previous section on locating SCSI IDs), then you must change the SCSI ID on the emulator.

Before resetting the emulator's SCSI ID, be sure the emulator is not turned on.

Never disconnect or reconnect any cables or other hardware devices while the emulator is turned on.

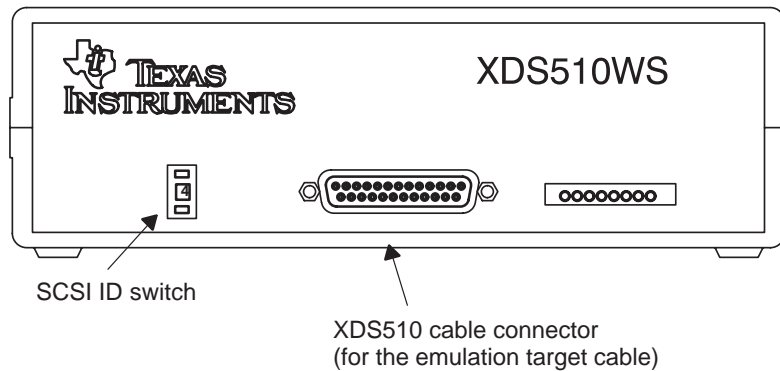
WARNING

The emulator's SCSI ID is controlled by a switch on the front panel of the emulator. Refer to Figure 1–4 for the location of this switch.

This switch has ten positions, 0 through 9; however, the emulator uses only positions 0 through 7. Do not use settings 8 and 9.

When you've finished resetting the emulator's SCSI ID, you can connect the emulation target cable to the front of the emulator.

Figure 1–4. Connector and Switch Locations on the Front of the XDS510WS Emulator



Adding the Emulator to the SCSI Bus

The SCSI bus is a chain with two distinct ends; it is not a loop. Although there may be SCSI devices within your host, the visible chain begins at the host and ends at one of the external SCSI devices. You can connect the emulator to the SCSI bus anywhere along this chain; however, it's best to place the emulator where you can easily connect it to your target system. The emulator's indicator lights should be visible and the power switch readily accessible.

Be sure all devices on the SCSI bus, including the workstation, are turned off before you connect the emulator to the workstation.

CAUTION

Connect the SCSI cable to the back of the workstation (see Figure 1-5); you can use either of the SCSI connectors. Connect the other end of the SCSI cable to the emulator (refer to Figure 1-6 for the location of the SCSI connectors on the emulator box).

Figure 1-5. Connecting the Emulator to the workstation

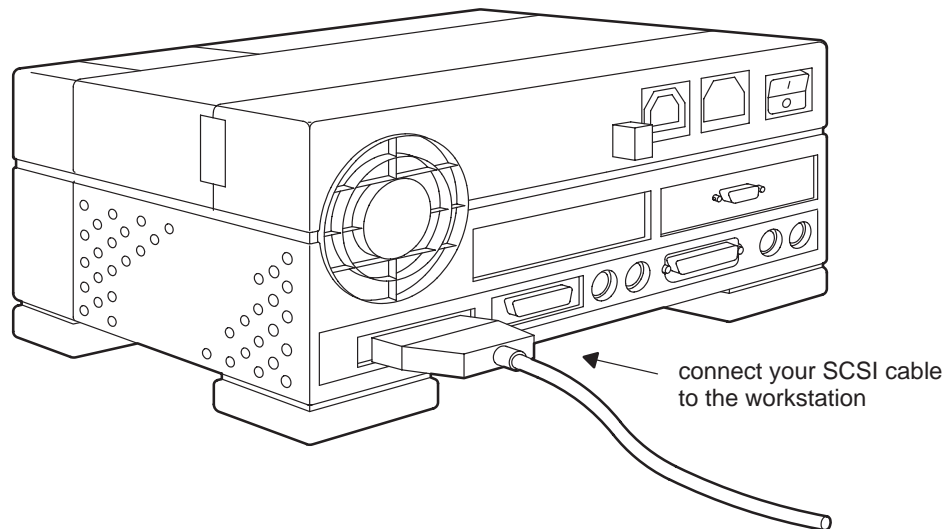
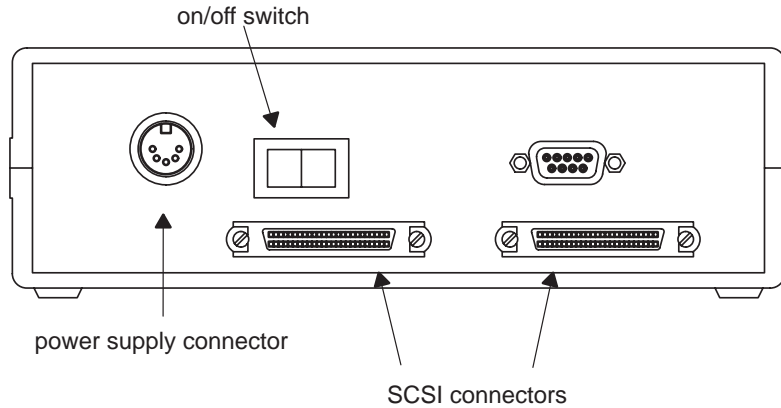


Figure 1–6. Connector and Switch Location on the Rear of the XDS510WS Emulator



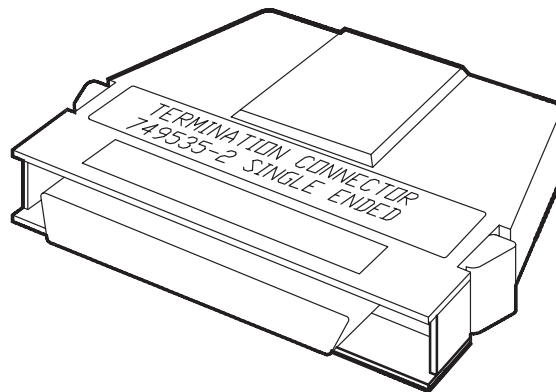
Terminating the SCSI bus

You *must* terminate the SCSI bus at each end of its chain to reduce signal noise. Only the device farthest on the chain from the host (the workstation) should be terminated; terminating intervening devices can cause intermittent errors in the SCSI bus.

If your emulator is at the end of a SCSI bus, you must terminate your emulator by connecting the external terminator (see Figure 1–7) to the unused SCSI connector on the back of your emulator.

Power up the external SCSI devices (including the emulator) before turning on the workstation.

Figure 1–7. The External Terminator for SCSI Bus Termination



1.3 Step 2: Setting Up a Workstation to Recognize the Emulator

This step varies, depending on the operating system you are using:

- If you have Solaris 2.x**, as the root user, enter:

```
halt
```

With the XDS510WS properly connected and powered up, reboot your SPARCstation with the PROM command `boot -r`. Once the system comes back up, execute the following command as the root user:

```
/usr/sbin/disks
```

Once you have entered these commands, you can skip to Section 1.4.

- If you have SunOS 4.1.x**, you must complete the instructions in this section to set up the workstation.

Modifying the workstation's configuration file

Once you have set up your emulation hardware, you must modify the workstation's configuration file to allow the debugger to access the emulator. The name of the configuration file used by the workstation normally appears in parentheses following your SunOS version number when you boot or log in to your system. In Example 1–1, the configuration file is called **GENERIC**.

Example 1–1. Locating the Name of Your Configuration File

```
Last login: Mon Mar 15 09:40:13 on console
SunOS Release 4.1.1 (GENERIC)#1: Mon Feb 1 09:00:07 CST 1993
You have mail.
```

To change this configuration file, complete these ten steps:

- 1) Switch directories to find the configuration file. To do this, enter the following command:

```
cd /usr/kvm/sys/sun4/conf
```

If this command does not work, replace **sun4** with either **sun4c** or **sun4m**, depending on the type of workstation:

Machine type	Use directory name
SPARCstation 1, 1+, or 2	sun4c
SPARCstation 10 xxx-MP (for example, 600 MP)	sun4m

Note:

If the specified directory does not exist or does not contain the specified configuration file, then your system was probably installed without modification privileges. Contact your system administrator for help.

- 2) Copy the current configuration file to a file called EMULATOR:

```
cp filename EMULATOR
```

Replace *filename* with the name of the current configuration file (see Example 1-1).

- 3) Edit the EMULATOR file. You can use any editor you are familiar with, but for the purpose of discussion, the vi editor is used. To use this editor, enter:

```
vi EMULATOR
```

Your EMULATOR file should look similar to Example 1-2. While in the editor, search for:

- *ident* and replace the string following it with the string “**EMULATOR**”.
- *options IPCSEMAPHORE* to be sure it exists in your configuration file and is not a comment; comments are preceded by the # symbol.
- *options IPCSHMEM* to be sure it exists in your configuration file and is not a comment.
- *options IPCMESSAGE* to be sure it exists in your configuration file and is not a comment.
- *target # lun 0*, where # is the SCSI ID for the emulator (4 by default; refer to Section 1.2). Be sure the entry is set up as a disk; to do this, make sure *tape st4*, for example, is changed to *disk sd4*.

Any other references to the driver that you have chosen (**sd4** is the default) should be turned into comments.

Note:

When you execute the debugger or emurst and you use the -p debugger option, you are referring to the sd# in the configuration file and to the associated rsd#a file. (This number is **not necessarily** the same as the SCSI ID number, but it can be.)

Example 1-2 shows a correctly modified EMULATOR file. Notice that modifications are highlighted and shown in bold face type. Lines preceded by # are comments and are ignored; you do not have to edit them. However, for consistency, these lines are modified.

Example 1–2. Setting Up the EMULATOR Configuration File

```
#
# @(#) GENERIC from master 1.28 90/09/21 SMI
#
# This config file describes an generic Sun-4c kernel, including all
# possible standard devices and software options.
#
# The following lines include support for all Sun-4c CPU types.
# There is little to be gained by removing support for particular
# CPUs, so you might as well leave them all in.
#
machine          "sun4c"
cpu              "SUN4C_60"   # Sun-4/60
#
# Name this kernel EMULATOR.
#
ident           "EMULATOR"
.
.
.
#
# The following options are for various System V IPC facilities.
# Most standard software does not need them, although they are
# used by SunGKS and some third-party software.
#
options IPCMESSAGE # System V IPC message facility
options IPCSEMAPHORE# System V IPC semaphore facility
options IPCSHMEM   # System V IPC shared memory facility
.
.
.
scsibus0 at esp      # declare first scsi bus
  disk sd0 at scsibus0 target 3 lun 0   # first hard SCSI disk
  disk sd1 at scsibus0 target 1 lun 0   # second hard SCSI disk
  disk sd2 at scsibus0 target 2 lun 0   # third hard SCSI disk
  disk sd3 at scsibus0 target 0 lun 0   # fourth hard SCSI disk
  disk sd4 at scsibus0 target 4 lun 0 # XDS510WS emulator
  tape st1 at scsibus0 target 5 lun 0   # second SCSI tape
  disk sr0 at scsibus0 target 6 lun 0   # CD-ROM device

scsibus1 at esp      # declare second scsi bus
  #disk sd4 at scsibus1 target 3 lun 0 # fifth hard SCSI disk
  disk sd5 at scsibus1 target 1 lun 0   # sixth hard SCSI disk
  disk sd6 at scsibus1 target 2 lun 0   # seventh hard SCSI disk
  disk sd7 at scsibus1 target 0 lun 0   # eighth hard SCSI disk
  tape st2 at scsibus1 target 4 lun 0   # third SCSI tape
  tape st3 at scsibus1 target 5 lun 0   # fourth SCSI tape
  disk sr1 at scsibus1 target 6 lun 0   # 2nd CD-ROM device
```

4) When you have finished editing your EMULATOR file, save the file and exit the vi editor by pressing (SHIFT) (Z) (Z).

5) Now, create the EMULATOR directory. Enter:

```
config EMULATOR
```

6) Once you have created the EMULATOR directory, change your current directory to your newly created directory by entering:

```
cd EMULATOR
```

7) Now you must compile the new kernel described by your configuration file. To do this, enter:

```
make
```

8) Save the old kernel file (vmunix) so that you can easily revert to it. Enter:

```
mv /vmunix /vmunix.orig
```

9) To move the new kernel file into use, enter:

```
cp vmunix /
```

10) You are now ready to reboot the workstation. Enter:

```
shutdown -r now
```

When you log onto the workstation, you should see the name EMULATOR appear in parentheses, as shown in Example 1–3. If EMULATOR does not appear in parentheses as you are rebooting the workstation, then your emulator may not be installed properly. Go back through these ten steps to be sure that you have followed each step correctly.

Example 1–3. Screen Display At Log On With the New Emulator Configuration File

```
Last login: Mon Mar 15 09:40:13 on console
SunOS Release 4.1.1 (EMULATOR)#1: Mon Feb 1 09:00:07 CST 1993
You have mail.
```

1.4 Step 3: Allowing the Debugger to Access the Emulator

The debugger accesses the emulator by reading from and writing to the device driver you defined in the EMULATOR configuration file. As a result, to execute the debugger, you must have read and write permissions on the driver file.

This step varies, depending on which version of the operating system you are using:

- ❑ **If you have Solaris 2.x**, nothing further should be required after you have performed the actions listed in the first bullet of Section 1.3. To confirm proper operation, enter the following command.

```
ls -l /dev/rsd#a
```

If rsd#a is not listed (where # is the device driver number of the emulator), return to the first bullet in Section 1.3 on page 1-10. If rsd#a is listed with permissions other than lrwxrwxrwx, change the permissions (chmod command) as shown below for SunOS 4.1.x users.

- ❑ **If you have SunOS 4.1.x**, as the root user, enter the following command, replacing # with the device driver number of the emulator (4 by default):

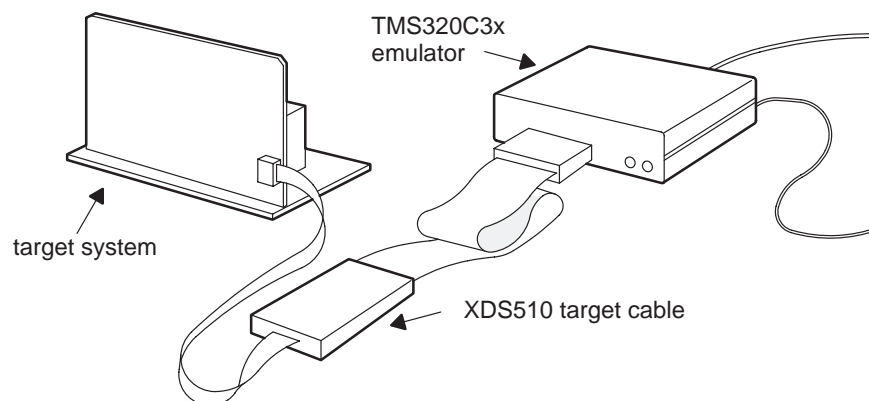
```
chmod a+rw /dev/rsd#a
```

This enables the debugger to access the emulator *without* root privileges.

1.5 Step 4: Connecting the Emulator to a Target System

In most cases, the target system is a board of your own design. It should have the appropriate emulation connector (14-pin header), as described in the hardware requirements (refer to page 1-2). To connect the target cable to your target board correctly, make sure the key is aligned properly; then, slowly and firmly, attach the cable (see Figure 1-8).

Figure 1-8. Connecting the Emulator to a Target System



1.6 Step 5: Installing the Debugger Software

This section explains the process of installing the debugger software on your hard disk system. The software package is shipped on a cartridge tape. To install the emulator software, you must restore the directory from the tape.

- 1) Insert the product tape in a cartridge tape drive.
- 2) Create a directory named *c3xhll* on your hard disk. This directory will contain the 'C3x C source debugger software. To create this directory, enter:

```
mkdir c3xhll
```

- 3) Make the emulator directory the current directory:

```
cd c3xhll
```

- 4) Copy the files from the cartridge tape to your hard disk system:

```
tar -xvf /dev/rst8
```

If the tape drive is not associated with *rst8*, replace *rst8* with the correct driver name (the *sd#* number you should have found while editing the EMULATOR configuration file); refer to substep 3 on page 1-11 for more information. If you need help, ask your system administrator.

1.7 Step 6: Making Sure the Emulator Supports the Debugger

The ROM code for the emulator does not contain the information necessary to debug a processor; that code must be downloaded from the host. This makes it easier to upgrade the emulation software. The *emurst* program downloads the necessary code for proper emulation.

To run this program, enter the *emurst* command in the following format:

emurst [-x] [-p *number*] *pathname-filename*

The *-x* option tells the *emurst* program to ignore any options specified with the *D_OPTIONS* environment variable.

Number represents the device driver number you defined in the EMULATOR configuration file (refer to Step 2 on page 1-10), and *pathname-filename* is the location and name of the *c3x510ws.out* file.

You can omit the *-p* option if the default, *rsd4a*, is the device driver for the emulator.

Note:

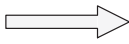
When you execute the debugger or *emurst* and you use the *-p* debugger option, you are referring to the *sd#* in the configuration file and to the associated *rsd#a* file. (This number is **not necessarily** the same as the SCSI ID number, but it can be.)

You can be sure that *emurst* succeeded when only the first and second LEDs from the left are on.

1.8 Step 7: Setting Up the Debugger Environment

To ensure that your debugger works correctly, you must:

- Modify the PATH statement to identify the c3xh11 directory.
- Define environment variables so that the debugger can find the files it needs.
- Identify any nondefault device driver used by the emulator.
- Reset the emulator.



Not only must you do these things before you invoke the debugger for the first time, *you must do them any time you power up or reboot your SPARCstation or emulator.*

You can accomplish most of these tasks by entering individual commands, but it's simpler to put the commands in your `.cshrc` file.

Example 1–4 shows a `.cshrc` file that contains the suggested modifications. The subsections following the example explain these modifications.

Example 1–4. Command Setup for the Debugger

<p>PATH statement →</p> <p>Environment variables →</p> <p>Reset the emulator →</p>	<p>→</p> <p>{</p> <p>→</p> <p>→</p>	<pre>setenv PATH "...;c3xh11;..." setenv D_SRC "path1;path2;..." setenv D_DIR "c3xh11;..." setenv D_OPTIONS "-p 4 emurst c3x510ws.out</pre>
--	-------------------------------------	---

Invoking the new or modified `.cshrc` file

If you create or modify your `.cshrc` file, you must invoke that file before invoking the debugger for the first time. To do so, enter:

```
source .cshrc
```

Modifying the PATH statement

Define a path to the debugger directory. The general format for doing this is:

```
setenv PATH "...;c3xh11;..."
```

This allows you to invoke the debugger without specifying the name of the directory that contains the debugger executable file.

Setting up the environment variables

An environment variable is a special system symbol that the debugger uses for finding or obtaining certain types of information. The debugger uses four environment variables; D_DIR, D_SRC, D_OPTIONS, and DISPLAY (X Window System™ only). The next four steps tell you how to set up these environment variables. The format for doing this is the same for both the .cshrc file and the command line.

- 1) Set up the D_DIR environment variable to identify the emulator directories:

```
setenv D_DIR "c3xh11"
```

These directories contain auxiliary files (emuinit.cmd, init.clr, etc.) that the debugger needs.

- 2) Set up the D_SRC environment variable to identify any directories that contain program source files that you'll want to look at while you're debugging source code. The general format for doing this is:

```
setenv D_SRC "path1;path2;..."
```

For example, if your programs were in a directory named *csource*, the D_SRC setup would be:

```
setenv D_SRC "csource"
```

- 3) You can use several options when you invoke the debugger. If you use the same options repeatedly, it's more convenient to specify them by using D_OPTIONS. The general format for doing this is:

```
setenv D_OPTIONS [object filename] [debugger options]
```

This tells the debugger to load the specified object file and use the selected options each time you invoke the debugger. These are the options that you can identify with D_OPTIONS:

```
-b[b]           -d machinename  -f filename     -i pathname  
-p dev drv#    -profile         -s              -t filename  
-v
```

Note that you can override D_OPTIONS by invoking the debugger or emurst with the -x option.

For more information about options, refer to the invocation section in Chapter 1, *Overview of a Code Development and Debugging System*, in the *TMS320C3x C Source Debugger User's Guide*.

- 4) If you are using the X Window System, you can use the DISPLAY environment variable to display the debugger on a different machine from the one the debugger is running on. The general format for doing this is:

```
setenv DISPLAY "machine name"
```

For example, if you are running the debugger on a machine called opie and you want the 'C3x debugger display to appear on a machine called barney, the DISPLAY setup would be:

```
setenv DISPLAY barney:0
```

You can also display the debugger on a different machine by using the `-d` option when invoking the debugger:

```
emu3x -d barney:0 
```

For more information about using the debugger under the X Window System, refer to Section 1.10 on page 1-21.

1.9 Step 8: Verifying the Installation

To ensure that you have correctly installed the emulator and debugger software, enter this command at the system prompt:

```
emu3x sample
```

You should see a display similar to this one:

The screenshot displays the emu3x debugger interface. At the top, there are menu options: Load, Brea, Watch, Memory, Color, MoDe, Run=F5, Step=F8, Next=F10. The main window is divided into three sections: DISASSEMBLY, CPU, and COMMAND.

DISASSEMBLY:

Address	Hex	Label	OpCode	Comment
80985d	00809938		ABS1	IOF,R0
80985e	08750000	c_int00:	LDI	0,ST
80985f	50700080		LDIU	128,DP
809860	0834985c		LDI	&0f0985cH, SP
809861	080b0014		LDI	SP,AR3
809862	50700080		LDIU	128,DP
809863	0828985d		LDI	&0f0985dH,AR0
809864	04e8ffff		CMPI	-1,AR0
809865	6a05000d		BZ	0809873H
809866	08402001		LDI	*AR0++(1),R0
809867	6a250009		BZD	0809873H
809868	081b0000		LDI	R0,RC
809869	08492001		LDI	*AR0++(1),AR1
80986a	08402001		LDI	*AR0++(1),R0
80986b	187b0001		SUBI	1,RC
80986c	6480986d		RPTB	080986dH
80986d	da002120		LDI	*AR0++(1),R0 STI R0,*AR
80986e	04e00000		CMPI	0,R0
80986f	6a26fff9		BNZD	080986bH
809870	081b0000		LDI	R0,RC
809871	08492001		LDI	*AR0++(1),AR1
809872	08402001		LDI	*AR0++(1),R0
809873	50700080		LDIU	128,DP
809874	62809800		CALL	main
809875	62809877		CALL	exit
809876	78000000		RETI	
809877	0f2b0000	exit:	PUSH	AR3
809878	080b0014		LDI	SP,AR3
809879	0f240000		PUSH	R4

CPU:

Register	Value
PC	0080985e
SP	00000755
R0	00000003
R1	00000005
R2	00000007
R3	00000000
R4	00000000
R5	00000000
R6	00000000
R7	00000000
AR0	0001802
AR1	00000000
AR2	00000000
AR3	00000000
AR4	00000000
AR5	00000000
AR6	00000000
AR7	00000000
IR0	00000000
IR1	00000000
ST	00000000
RC	00000000
RS	00000000
RE	00000000
DP	000000f0
BK	00000000
IE	00000000
IF	00000000
IOF	00000088

COMMAND:

```
(c) Copyright 1989, Texas Instruments
Silicon Revision 2
Emulator Revision 1
Loading sample.out
None
>>>
```

MEMORY:

Address	Value	Address	Value	Address	Value
000000	0000004b	00000040	00000041	00000042	
000004	00000043	00000044	00000045	00000046	
000008	00000047	00000048	00000049	0000004a	
00000c	00000000	00000000	00000000	00000000	
000010	00000000	00000000	00000000	00000000	
000014	00000000	00000000	00000000	00000000	
000018	00000000	00000000	00000000	00000000	
00001c	00000000	00000000	00000000	00000000	
000020	00000000	00000000	00000000	00000000	

If you see a display similar to this one, you have correctly installed your emulator and debugger.

If you don't see a display, then your debugger or board may not be installed properly. Go back through the installation instructions and be sure that you have followed each step correctly; then re-enter the command above.

1.10 Using the Debugger With the X Window System

If you're using the X Window System to run the 'C3x debugger, you need to know about the keyboard's special keys, the debugger fonts, and using the debugger on a monochrome monitor.

Using the keyboard's special keys

The debugger uses some special keys that you can map differently from the current configuration of your keyboard. Some keyboards, such as the Sun Type 5 keyboard, may have these special symbols on separate keys. Other keyboards, such as the Sun Type 4 keyboard, do not have the special keys.

The special keys that the debugger uses are shown in the following table with their corresponding keysym. A **keysym** is a label that interprets a keystroke; it allows you to modify the action of a key on the keyboard.

Key	Keysym
(F1) to (F10)	F1 to F10
(PAGE UP)	Prior
(PAGE DOWN)	Next
(HOME)	Home
(END)	End
(INSERT)	Insert
(→)	Right
(←)	Left
(↑)	Up
(↓)	Down

Use the X utility `xev` to check the keysyms that are associated with your keyboard. If you need to change the keysym definitions, use the `xmodmap` utility. For example, you could create a file that contains the following commands and use that file with `xmodmap` to change a Sun Type 4 keyboard to match the keys listed above:

```
keysym R13      = End
keysym Down     = Down
keysym F35      = Next
keysym Left     = Left
keysym Right    = Right
keysym F27      = Home
keysym Up       = Up
keysym F29      = Prior
keysym Insert   = Insert
```

Refer to your X Window System documentation for more information about using `xev` and `xmodmap`.

Changing the debugger font

You can change the font of the debugger screen by using the `xrdb` utility and modifying the `.Xdefaults` file in your root directory. For example, to change the fonts of the 'C3x debugger to Courier, add the following line to the `.Xdefaults` file:

```
emu3x*font:courier
```

For more information about using `xrdb` to change the font, refer to your X Window System documentation.

Color mappings on monochrome screens

Although a color monitor is recommended, the following table shows the color mappings for monochrome screens:

Color	Appearance on Monochrome Screen
black	black
blue	black
green	white
cyan	white
red	black
magenta	black
yellow	white
white	white

Troubleshooting

This chapter describes some common problems you may encounter while using your emulator or debugger on your workstation. You should be familiar with the procedures described in Chapter 1 before trying to troubleshoot problems with the XDS510WS and its software.

Topic	Page
2.1 Problems When Booting Your Workstation	2-2
2.2 Problems When Resetting the Emulator	2-3
2.3 Problems When Invoking the Debugger	2-5
2.4 Additional Emulator and Debugger Problems	2-7

2.1 Problems When Booting Your Workstation

After installing your emulator, the problems listed below may occur when you attempt to boot your workstation. The list includes suggestions for resolving the problems.

- Your workstation will not boot when connected to your emulator, even if your emulator is not turned on.
 - 1) Be sure that all of your SCSI cables are connected securely and that the SCSI bus is terminated properly (see *Terminating the SCSI bus* on page 1-9).
 - 2) Remove any unnecessary SCSI devices from the bus.
 - 3) Make sure that the total length of the SCSI bus is less than six meters, including the section of the bus within the SPARCstation chassis.
- Your workstation will boot when the emulator is turned off but will not boot when the emulator is turned on.

Your emulator's SCSI ID conflicts with the SCSI ID of another device on the SCSI bus. Go back through the instructions on page 1-6.

2.2 Problems When Resetting the Emulator

After you power up the emulator and the workstation, if you have the following problems while attempting to reset the emulator, implement the applicable solutions:

- When executing the emurst command, you receive this message:

```
emurst file [.out]:
```

You forgot to specify the *pathname-filename* of the C3x510ws.out file. You can specify it at this prompt, or you can re-execute emurst with *pathname-filename* specified on the command line.

- When executing the emurst command, you receive this message:

```
>> can't initialize the target system
```

- You haven't set the IPCSEMAPHORE option to allow the debugger to access the emulator. Be sure that the configuration file, EMULATOR, has the options line set correctly, without comments. Then, use the corrected configuration file to build the currently executing kernel (see Section 1.3 on page 1-10).

- There are too many current semaphores on the system. Clean up the unused semaphores by using the **ipcs -st** and **ipcrm** utilities, and try to execute emurst again.

- You may not have permission to access the driver file you specified with the **-p** debugger option. Normally, you specify the **-p** debugger option on the command line or in the D_OPTIONS environment variable. Remember, if you haven't specifically reset the driver file number to another number, the default is 4. Have the root user execute the following command, and try to execute emurst again.

```
chmod a+rw /dev/rsd#a
```

- The driver file you specified with the **-p** debugger option is not correctly associated to your emulator in your configuration file. Make sure your configuration file contains a line similar to this:

```
disk sd# at scsibus<m> target <s> lun 0
```

where # is the device driver number and <s> is the SCSI ID that you set with the switch at the front of the XDS510WS. The <m> is 0 (zero) unless the XDS510WS is connected to a second SCSI bus that you added to your SPARCstation, which would cause <m> to change. Use the corrected configuration file to build the currently executing kernel (see Section 1.3 on page 1-10).

- You haven't turned on the XDS510WS, or it hasn't completed its self-test. Turn on the XDS510WS, and wait for the self test to complete

successfully before executing emurst. The self-test has completed, once the sixth LED from the left is off and the first, second, and fifth LEDs from the left are on.

- When executing the emurst command, you receive this message:

```
>> error loading file
```

- The emurst utility can't find the C3x510ws.out file as specified. If you didn't specify the *pathname-filename* with an extension as part of the name, the emurst utility appends the default extension *.out* to the name.
- If you didn't provide path information (just the filename), emurst searches first in the current directory and then in all of the directories specified in the *D_DIR* environment variable before returning this error message. Make sure the correct file is located where emurst can find it.
- The file that you specified to emurst isn't appropriate for this use. Use the C3x510ws.out file that is included with the debugger software.

2.3 Problems When Invoking the Debugger

If you encounter these problems when you invoke the debugger, the suggested solutions may resolve the problems:

- You receive the following message when executing the emu3x command:

```
CANNOT INITIALIZE THE TARGET !!
- Check I/O configuration
- Check cabling and target power
```

- The emurst command didn't successfully execute before you tried to invoke the debugger. Execute emurst (see Section 1.7 on page 1-16). emurst has completed successfully if you see your command prompt after this message:

```
EMURST for XDS510WS loading <pathname-filename> at #
where <pathname-filename> is the location of the C3x510ws.out file
and # refers to the file /dev/rsd#a, which is associated with the emulator
in the configuration file, EMULATOR. Also, you can be sure that emurst
succeeded when only the first and second LEDs from the left are on.
```

- The `-p` debugger option that you entered on the command line or in the `D_OPTIONS` environment variable specifies a different driver file from the one used by emurst. Remember, if you haven't specifically reset the driver file number to another number, the default is 4. Use the same `-p` option that you used when you executed emurst. (Refer to Section 1.7 on page 1-16 for more information on the `-p` option.)
 - Intermittent errors cause the emurst command to fail. Make sure that your emulation cable is firmly attached both to the XDS510WS and to your target system.
 - Intermittent errors cause the emurst command to fail. Make sure that your target system is receiving sufficient power at the required voltage to allow all devices on the board to work properly.

- You receive the following message at the operating-system command line when trying to execute the emu3x command:

```
emu3x: display :0.0 doesn't know font 7x14
```

The default font file that the debugger uses (`7x14.ff`) couldn't be found by OpenWindows. OpenWindows searches for these font files in the directories specified in the `FONTPATH` environment variable. To correct the problem, do one of the following:

- Add the font file `7x14.ff` to a directory defined in the `FONTPATH` environment variable.

- Add to the .Xdefaults file in your home directory the line:
emu3x*font: *GoodFontName*
where *GoodFontName* is the name of a font that OpenWindows can find.
- Copy a valid font file onto 7x14.ff.

Note:

The operating-system window provides operating-system messages. These messages differ from the error messages that you may see in the COMMAND window of the debugger.

2.4 Additional Emulator and Debugger Problems

The operating-system window displays operating-system messages. These messages differ from the error messages that you may see in the COMMAND window of the debugger. If you receive one of these operating-system messages while executing the emurst or the debugger, refer to the following explanations.

Note:

For each of the following four bulleted items (), note that the messages are *status messages*, **not error messages**.

- In your operating-system window, you receive the following message while executing emu3x or emurst under SunOS 4.1.x:

```
<date> <time> <hostname> vmunix: sd<n>: disk not
responding to selection
```

or under Solaris 2.x:

```
WARNING: /sbus@1,f8000000/esp@0,800000/sd@<n>,0(sd<n>):
        disk not responding to selection
```

The XDS510WS didn't respond to the SPARCstation in the required amount of time. This can be caused by one of these conditions:

- The XDS510WS isn't powered.
 - The XDS510WS is executing its self-test.
 - The XDS510WS is executing a lengthy debugger command such as a large memory fill.
- In your operating-system window, you receive the following message while executing emu3x or emurst under SunOS 4.1.x:

```
<date> <time> <hostname> vmunix: sd<n>: offline
```

or under Solaris 2.x:

```
WARNING /sbus@1,f8000000/esp@0,800000/sd@<n>,0(sd<n>):
        offline
```

The SPARCstation is unable to select the XDS510WS after several attempts and therefore considers the emulator to be offline. This message can be generated during large memory-fill instructions and should **not** be considered an error by itself or in combination with the preceding message. The debugger automatically corrects for this situation unless a major error has taken place, in which case the debugger eventually returns an error message in the COMMAND window of the debugger.

- ❑ In your operating-system window, you receive the following message while executing emu3x or emurst under SunOS 4.1.x:

```
<date> <time> <hostname> vmunix: sd<n>: disk okay
```

or under Solaris 2.x:

```
WARNING: /sbus@1,f8000000/esp@0,800000/sd@<n>,0(sd<n>):  
disk okay
```

The SPARCstation has reconnected with the XDS510WS after the XDS510WS didn't respond to the selection. When the debugger recovers from the offline condition (described in the previous bulleted item), one of the two messages shown above is written to the operating-system window.

- ❑ In your operating-system window, you receive the following message while executing emu3x or emurst under SunOS 4.1.x:

```
sd<n> at esp0 target <p> lun 0  
sd<n>: Vendor 'TI-ASP', product 'XDS510-WS_Rev.*', 130  
512 byte blocks  
<date> <time> <hostname> vmunix: sd<n>: corrupt label -  
wrong magic number
```

If the emulator has been inactive on the bus since the SPARCstation's last attempt to access it, the XDS510WS returns to an active status on the bus. The above message informs you of this *new* SCSI device.

Note:

Since the SPARCstation interprets the emulator as a SCSI disk, the SPARCstation expects it to be formatted. When the SPARCstation first finds that the new device isn't formatted, it produces the corrupt label message.

Interpreting the XDS510WS LEDs

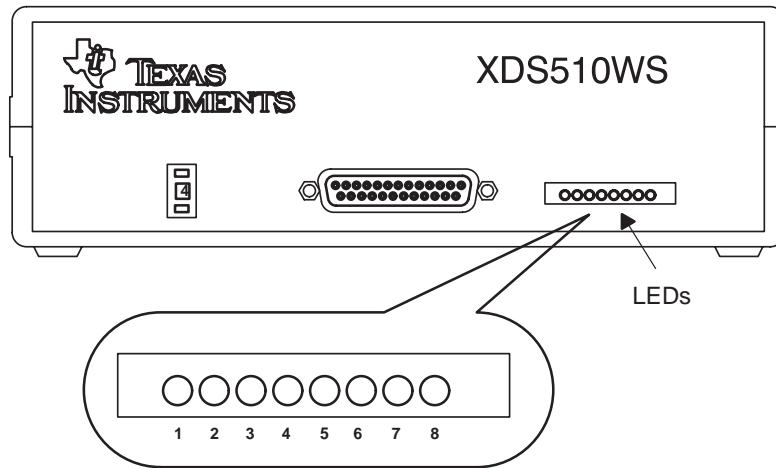
The TMS320C3x XDS510WS emulator provides status information about its operation through eight LEDs on the front panel of the emulator chassis.

Topic	Page
3.1 XDS510WS LED Descriptions	3-2
3.2 XDS510WS LED Reference Chart	3-5

3.1 XDS510WS LED Descriptions

On the front of the XDS510WS is a small panel of LEDs that provide status information during the operation of the emulator (refer to Figure 3–1).

Figure 3–1. XDS510WS LEDs



The LEDs are numbered from left to right, starting with LED 1 through LED 8. The three LED conditions are:

Meaning		LED Symbol
Off	=	○
On	=	●
Intermittent†	=	◐

† Intermittently on and off; no steady state

LED 1

LED 1 is on whenever the system is plugged in and switched on. If LED 1 doesn't come on, you should:

- 1) Ensure that the power supply is firmly plugged into a proper outlet.
- 2) Check to see that the power supply cable is firmly plugged into the XDS510WS.
- 3) Check to see that the XDS510WS is switched on.

LED 2

When LED 2 is on, the XDS510WS has detected a power loss on the target system.

Note:

After you apply power to the target, LED 2 remains on until you invoke the debugger.

When you invoke the debugger, if LED 2 fails to go off and the debugger fails to start, you should ensure that the emulation cable is firmly and correctly attached to both the XDS510WS and the target. Also, check to see that the target is turned on and has sufficient power. Additionally, check to see whether the target was designed to provide V_{CC} to the emulation header pin, PD.

Once LED 2 has gone off, if it comes on during your debugging session, the target system has lost power.

LED 3

LED 3 is on whenever the XDS510WS is executing an emulation instruction. Normally, you won't notice the sporadic on/off state of this LED.

Occasionally, when you're performing a time-consuming emulation command such as a large FILL, LED 3 will stay on for a considerable period. If LED 3 stays on for too long (longer than five minutes), this could indicate a problem. If other signs also indicate a problem (such as an unresponsive mouse or keyboard), you may continue working by exiting the debugger, cycling the power on the XDS510WS, and beginning again.

LEDs 4, 5, and 6

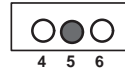
LEDs 4, 5, and 6 indicate an error has occurred and signify the state of the emulator.

When you first power up the XDS510WS and immediately after you execute an emurst command, the emulator performs a self-test. LEDs 4 and 5 will be off and LED 6 will be on to indicate that the self-test is being performed:

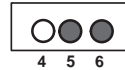


The self-test should take only a few seconds; something is wrong if these three LEDs show the pattern above for more than a minute.

If the self-test completes successfully, LEDs 4 and 6 will be off and LED 5 will be on:



If these three LEDs show this pattern:



there has been a communication error. These errors are generally not serious, but if you can't continue without intervention, cycle the power on the XDS510WS, re-execute emurst, and restart the debugger.

LEDs 7 and 8

When LEDs 7 and 8 are active (intermittently on and off), they indicate that a SCSI transfer is in progress with the emulator. If the debugger seems to hang and the LEDs become fixed (not flashing) in any pattern other than 7 and 8 off as shown below:



there is probably a problem. You can cycle the power on the XDS510WS, re-execute emurst, and restart the debugger.

3.2 XDS510WS LED Reference Chart

Figure 3–2 shows the standard LED sequences. These patterns allow you to understand quickly the operational status of the emulator and its functions.

Figure 3–2. Standard LED Sequences

On ●	Off ○	Intermittent ◐	Key to LED Status

Power-On Sequences

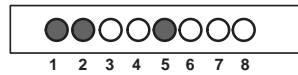


before power-on

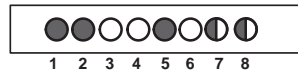


steady state

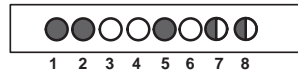
EMURST Sequences



before emurst



steady state



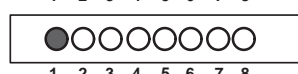
Debugging Sequences



before debugging



steady state



Index

A

arrow keys 1-22
assembler 1-3

B

-b debugger option, with D_OPTIONS environment variable 1-19
batch files
 .cshrc 1-18 to 1-20
 sample 1-18
 emuinit.cmd 1-3
 emurst 1-3
 init.clr 1-4
 initialization, emuinit.cmd 1-3
 invoking, .cshrc 1-18
 mono.clr 1-4
 screen sizes, PC systems 1-4
booting, problems 2-2
building the kernel 1-14

C

C3x510ws.out file 1-3, 1-17
C3xhll directory 1-16, 1-19
cable requirements 1-2
cd UNIX command 1-11, 1-14, 1-16
changing directories 1-11, 1-14
changing the debugger display (font) 1-23
chmod UNIX command 1-15, 2-3
colors, mapping with X Windows 1-23
config command 1-14

configuration file
 confirming 1-15
 example 1-13
 locating the name 1-11
 modifying 1-11 to 1-14
 renaming 1-11 to 1-14
copying files
 cp UNIX command 1-12
 UNIX command 1-14
cp UNIX command 1-12, 1-14
.cshrc file 1-18
 invoking 1-18
 sample 1-18
customizing the display
 changing the font 1-23
 init.clr file 1-4
 mono.clr file 1-4
 screen sizes, PC systems 1-4

D

-d debugger option, with D_OPTIONS environment variable 1-19
D_DIR environment variable 1-19
D_OPTIONS environment variable 1-19
D_SRC environment variable 1-19
debugger
 access to emulator 1-15
 changing the displayed font 1-23
 displaying on a different machine 1-19
 environment setup 1-18 to 1-20
 installation 1-1 to 1-23
 troubleshooting 2-5 to 2-6
 verifying 1-21
 troubleshooting 2-5 to 2-6, 2-7 to 2-8
 using the X Window System 1-22 to 1-23

- default
 - memory map 1-3
 - screen configuration file
 - color displays* 1-4
 - monochrome displays* 1-4
 - PC systems* 1-4
 - screen sizes* 1-4
 - SCSI ID, for the emulator 1-7
- directories
 - C3xhll directory 1-16, 1-19
 - for auxiliary files 1-19
 - for debugger software 1-16, 1-18
 - identifying additional source directories 1-19
- DISPLAY environment variable 1-19
- display requirements 1-2
- downloading code 1-17
- driver file
 - selecting 1-12, 1-17
 - troubleshooting 2-3, 2-5

E

- emu3x command
 - options, D_OPTIONS environment variable 1-19
 - troubleshooting 2-5 to 2-6, 2-7 to 2-8
 - verifying the installation 1-21
- emuinit.cmd file 1-3
- emulation cable 1-8
- emulator
 - adding to the SCSI bus 1-9 to 1-10
 - additional tools 1-3
 - assigning a SCSI ID 1-7
 - booting problems 2-2
 - communicating with your workstation 1-11
 - connecting to workstation 1-9 to 1-10
 - connecting to your target system 1-15
 - debugger environment 1-18 to 1-20
 - debugger installation 1-1 to 1-23
 - verifying* 1-21
 - default SCSI ID 1-7
 - driver file access 2-3
 - front view 1-6, 1-8
 - host system 1-2
 - installation 1-5 to 1-10
 - debugger software* 1-16
 - verifying* 1-21
 - LED lights 1-6, 3-1 to 3-6
 - locating the SCSI ID 1-7
 - memory, default map 1-3
 - operating system 1-3
 - rear view 1-6, 1-10
 - requirements
 - cable* 1-2
 - display* 1-2
 - hardware* 1-2
 - power* 1-2
 - software* 1-3
 - resetting 1-3
 - problems* 2-3 to 2-4, 2-7 to 2-8
 - screen, configuration files 1-4
 - self-test 1-6
 - setting the SCSI ID 1-8
 - state, LED lights 3-4
 - target system 1-2
 - terminating the SCSI bus 1-10
 - troubleshooting 2-1 to 2-8
 - typical setup 1-5
- EMULATOR file 1-11 to 1-14
 - creating a new kernel 1-14
 - creating the EMULATOR directory 1-14
 - example 1-13
 - modifying 1-12
 - running the new kernel 1-14
 - troubleshooting 2-3
- emurst command 1-3, 1-17
 - specifying parameters 2-3
 - troubleshooting 2-3 to 2-4 2-7 to 2-8
 - when invoking the debugger 2-5
- end key 1-22
- environment variables
 - D_DIR 1-19
 - D_OPTIONS 1-19
 - D_SRC 1-19
 - DISPLAY 1-19
 - displaying the debugger on a different machine 1-19
 - for debugger options 1-19
 - identifying auxiliary directories 1-19
 - identifying source directories 1-19
 - PATH 1-18
- errors, LED lights 3-4
- exiting your workstation 1-7

F

- f debugger option, with D_OPTIONS environment variable 1-19
- F1 key, mapping 1-22

F2 key, mapping 1-22
 F3 key, mapping 1-22
 F4 key, mapping 1-22
 F5 key, mapping 1-22
 F6 key, mapping 1-22
 F7 key, mapping 1-22
 F8 key, mapping 1-22
 F9 key, mapping 1-22
 F10 key, mapping 1-22
 font, changing the debugger display 1-23
 font file, troubleshooting 2-5

G

GENERIC file 1-11 to 1-14
 renaming 1-12

H

halt UNIX command 1-7
 hardware
 checklist 1-2
 installation 1-5 to 1-10
 home key 1-22
 host system 1-2

I

-i debugger option, with D_OPTIONS environment variable 1-19
 ident 1-12
 identifier, locating a SCSI ID 1-7
 init.clr file 1-4
 initialization batch files, emuinit.cmd 1-3
 insert key 1-22
 installation
 debugger software 1-16
 emulator 1-5 to 1-10
 troubleshooting 2-1 to 2-8
 verifying 1-21
 invoking
 .cshrc file 1-18
 debugger 1-21
 troubleshooting 2-5 to 2-6
 ipcrm UNIX command 2-3

ipcs UNIX command 2-3
 IPCSEMAPHORE option 1-12, 2-3
 IPCSHMEM option 1-12

K

kernel
 building 1-14
 confirming 1-15
 modifying 1-11 to 1-14
 running 1-14
 keyboard, mapping keys 1-22
 keys, special keys with the X Window System 1-22
 keysym label 1-22

L

labels, keysym 1-22
 LED lights 1-6, 3-1 to 3-6
 after emurst 1-17
 LED 1 3-2
 LED 2 3-3
 LED 3 3-3
 LED 4 3-4
 LED 5 3-4
 LED 6 3-4
 LED 7 3-5
 LED 8 3-5
 location 3-2
 overview 3-6
 standard sequences 3-6
 states 3-2
 lights, on the front of the emulator 1-6
 linker 1-3
 locating the SCSI ID 1-7
 ls UNIX command 1-15

M

make UNIX command 1-14
 mapping keys for use with X Windows 1-22
 memory
 default map 1-3
 mapping, emuinit.cmd file 1-3
 mkdir UNIX command 1-16
 monochrome monitors, color mapping with X Windows 1-23

mono.clr file 1-4
mv UNIX command 1-14

O

OpenWindows, finding the font file 2-5
operating system 1-3
optional files 1-3
options
 IPCMESSAGE option 1-12
 IPCMESSAGE 1-12
 IPCSEMAPHORE 1-12, 2-3
 IPCSHMEM 1-12

P

-p debugger option
 selecting the driver file 1-12, 1-17
 troubleshooting 2-3, 2-5
 using with emurst 1-17
 with D_OPTIONS environment variable 1-19
page-up/page-down keys 1-22
PATH statement 1-18
permissions
 changing 1-15
 root access 1-3
power loss, detecting 3-3
power requirements 1-2
power supply 1-6
printenv command 1-7
probe-scsi UNIX command 1-7
-profile debugger option, with D_OPTIONS environment variable 1-19

R

rebooting your workstation 1-7, 1-14
renaming files 1-12
required files 1-3
required tools 1-3
resetting
 emurst command 1-3
 emurst file, *troubleshooting* 2-3 to 2-4, 2-7 to 2-8
retrieving files from tape 1-16
root privileges 1-3

Index-4

S

-s debugger option, with D_OPTIONS environment variable 1-19
SCSI bus
 adding devices to it 1-9 to 1-10
 terminating the chain 1-10
SCSI ID
 assigning 1-7
 locating 1-7
 setting on the emulator 1-8
self-test 1-6
shutdown UNIX command 1-14
software checklist 1-3
SPARC stations 1-11
special keys, X Window System 1-22
Sun
 display requirements 1-2
 hardware requirements 1-2
 host system 1-2
 operating system 1-3
 power requirements 1-2
 setting up debugger environment 1-18 to 1-20
 software requirements 1-3
 target system 1-2
sun4 1-11
sun4c 1-11
sun4m 1-11
switching directories 1-11

T

-t debugger option, with D_OPTIONS environment variable 1-19
tape cartridge, retrieving files from 1-16
tar UNIX command 1-16
target # 1-12
target system 1-2
 connection to emulator 1-15
termination, external terminator for the SCSI chain 1-10
terminator, SCSI bus 1-2
troubleshooting 2-1 to 2-8
 booting the workstation 2-2
 invoking the debugger 2-5 to 2-6
 resetting the emulator 2-3 to 2-4, 2-7 to 2-8
 using the debugger 2-7 to 2-8

U

utilities

- emurst 1-3, 1-17, 2-4
- ipcrm 2-3
- ipcs -st 2-3
- xev 1-22
- xmodmap 1-22
- xrdb 1-23

V

- v debugger option, with D_OPTIONS environment variable 1-19

- verifying, installation 1-21
 - troubleshooting 2-5

- vi editor 1-12
 - exiting 1-14
 - invoking 1-12

W

- workstation
 - configuring 1-11

workstation (continued)

- connecting emulator 1-9 to 1-10
- modifying your configuration file 1-11 to 1-14
- modifying your kernel 1-11 to 1-14
- problems when booting 2-2
- rebooting 1-14
- typical setup 1-5

X

- x debugger option 1-19
 - using with emurst 1-17

X Window System

- changing the displayed font 1-23
- color mapping 1-23
- displaying the debugger on a different machine 1-19
- special keys 1-22
- using with the debugger 1-22 to 1-23
- xev utility 1-22
- xmodmap utility 1-22

- .Xdefaults file, changing the displayed debugger font 1-23

- xev utility 1-22

- xmodmap utility 1-22

