For more information, please call

United States	**1 800 800 7441**
United Kingdom	**0 800 440 440**
Australia	**008 802 677**

Corporate Office
2011 North Shoreline Boulevard
Mountain View, CA 94043

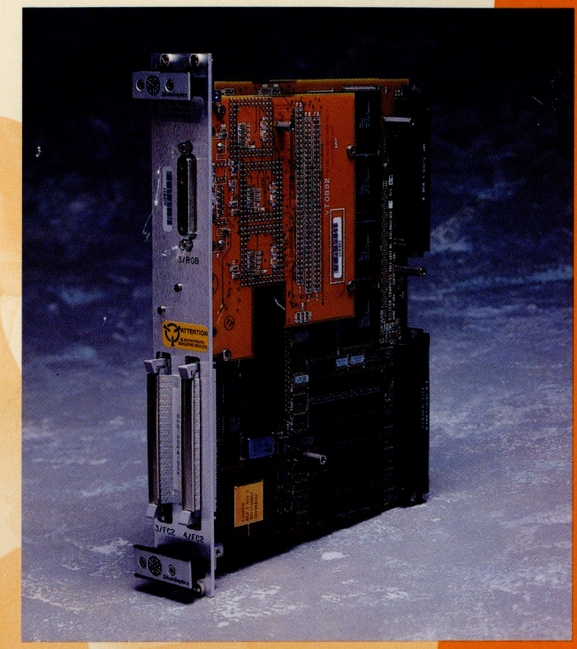
SiliconGraphics
Computer Systems

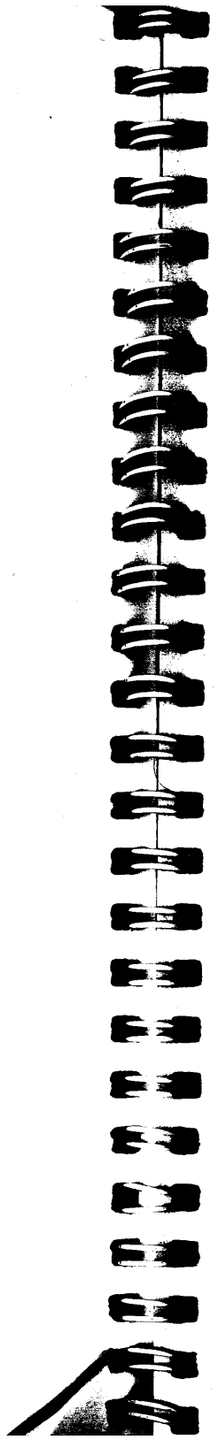
OSG - TG-V(7/92)

SiliconGraphics
Computer Systems

TG - V Graphics

System Integrator's Guide





TG - V Graphics
System Integrator's Guide

Document Number 007-5016-010

Contributors

Written by Lawrence Ertel
Edited by Loraine McCormick
Illustrated by Keith Granger
Engineering contributions by Bob Abbott, Tore Kellgren, Gordon Koo, Larry Lewis, Todd Nordland, Edward (Ted) Wilcox, and Ken Williams

TG - V Graphics System Integrator's Guide
Document Number 007-5016-010

Silicon Graphics, Inc.
Mountain View, California

Contents

Figures

Tables

Chapter 1

Introduction

This guide is for system integrators who plan to install and run the TG - V graphics subsystem with a Silicon Graphics® CPU subsystem such as the V20, V30, or V35. This guide contains the following chapters:

- Chapter 1, "Introduction," explains how to use this guide.
- Chapter 2, "Product Overview," summarizes product features.
- Chapter 3, "Hardware Installation," provides instructions for unpacking and installing the TG - V graphics subsystems.
- Chapter 4, "Software Installation," provides instructions for installing the IRIX™ operating system.
- Chapter 5, "Diagnostics," provides instructions for operating the IDE diagnostics program.
- Chapter 6, "TG - V Graphics Subsystem Architecture," offers a detailed description of the graphics subsystem.
- Appendix A, "Component Layout," illustrates the front panel and circuit boards used in the TG - V graphics subsystem.
- Appendix B, "Operating Conditions," lists the power requirements of the TG - V graphics subsystem.
- Appendix C, "DIP Switches," lists the optional TG - V graphics switch settings.
- Appendix D, "Pinouts for TG - V Graphics Video Connector," outlines the pinouts for the TG - V graphics video connector.

Note: You must have an installed 6U CPU subsystem in order to run the TG - V graphics subsystem. For information on installing the V20 or V30/V35 CPU subsystems, please refer to the integrator's guide included with those products.

Style Conventions

This *TG - V Graphics System Integrator's Guide* uses the following conventions:

- References to other documents are in *italics*.
- References to other chapters and sections within this guide are in quotation marks.
- Step-by-step instructions used to perform a task are shown as numbered sentences. These may be followed by additional instructions, marked with a small box, that further explain the step. For example:

  1. Make sure the VMEbus backplane is properly jumpered.
     - If necessary, remove any coverings or enclosures to get access to the rear of the VMEbus backplane. (Some VMEbus backplanes may be jumpered from the front.)
     - With the backplane exposed, replace the single IACK and the four BUS GRANT jumpers from each of the empty slots between the graphics subsystem and your other cards.

- Programming variables are shown in italics.
- Text that appears on screen is shown in a typewriter (courier) font.
- Individual keyboard commands are shown in boldface within angle brackets.
- Keyboard user input is shown in a boldface typewriter (courier) font.

Chapter 2

Product Overview

The TG - V graphics subsystem makes the full graphics processing power of the Silicon Graphics Personal IRIS™ workstation available for any standard 6U VMEbus card cage with P2 backplane connectors. The TG - V graphics subsystem comes prepackaged and ready to install.

In conjunction with a V20 or V30/V35 CPU subsystem, the TG - V graphics subsystem is compatible with Silicon Graphics' entire product line of reduced instruction set computing (RISC) systems, and is an ideal solution for applications that require the functionality and performance of a Personal IRIS graphics workstation in a VMEbus-based embedded system.

The TG - V graphics subsystem provides full support for the Silicon Graphics standard Graphics Library™ to calculate and display three-dimensional objects. It also provides support for the X Window™ system and Motif™ window manager.

The V20 and V30/V35 CPU subsystems can support up to four TG - V graphics subsystems, with each subsystem monitor displaying multiple windows. The window system and mouse allow you to move freely between the subsystem monitors and to cut and paste between them.

TG - V Graphics Subsystem Features

The TG - V graphics subsystem connects to a CPU subsystem through the P2 connectors via a flat ribbon cable on the back of the VMEbus backplane (rows A and C).

Each TG - V graphics subsystem can drive a single display monitor. The CPU subsystem can support up to four TG - V graphics subsystems, for a total of up to four simultaneous graphical displays per CPU.

Like the CPU subsystem, the TG - V graphics subsystem uses IEEE Std. 1014-1987 double-height boards. Depending on the options you order, each TG - V graphics subsystem can occupy between two and four adjacent slots in the VMEbus backplane.

The basic TG - V graphics subsystem occupies two slots and provides:

- a Geometry Engine™ (GE) that transforms graphics primitives in world coordinates into objects (points, lines, spans) described in screen coordinates.
- a raster engine that performs scan conversions, coupled with a frame buffer that supports 8-bit color images on a 1280 x 1024 pixel screen in the basic configuration.
- a display engine that determines the modes of the various images to display (single- or double-buffered, RGB, or color index), supports fast 32-bit RGB pixel DMA, and manages the monitor timing.

Options include:

- a turbo board that provides enhanced performance for geometric computations; this configuration includes an added double-height Eurocard board. The additional board provides four digital signal processing (DSP) chips running at 16.67 MHz, which pipeline data into the raster engine in round-robin fashion.

- a 6U form factor daughter card that extends the frame buffer to support 24-bit color images (5 MBytes of video RAM (VRAM)); when attached, this daughter card fits into an additional slot on the VMEbus.
- a daughter card with a 24-bit z-buffer with 1 sign bit and 23 integer bits for enhanced 3-D image display.

Graphics performance is summarized in Table 2-1.

Capability	Base	Turbo
2D Vectors/Sec.	90 K	200 K
3D Vectors/Sec.:		
Plain	90 K	200 K
Anti-aliased	50 K	140 K
Polygons/Sec., 4-sided, 10 X 10 ind.:		
Flat-shaded	9.1 K	32 K
Z-buffered, Gouraud shaded	5.1 K	25 K
Polygon Fill, Pixels/Sec.:		
Flat-shaded	34 M	34 M
Z-buffered	6.6 M	6.6 M
Pixel DMA, Pixels/Sec.:		
Writes	8.2 M	8.2 M
Reads	4.2 M	4.2 M

Table 2-1 Graphics Performance Ratings

Chapter 3

Hardware Installation

This chapter explains how to install the TG - V graphics subsystem into the VMEbus card cage. The steps involved include:

1. preparations
2. unpacking
3. installing the CPU subsystem
4. installing the graphics subsystem(s)
5. installing additional VMEbus devices
6. connecting peripheral devices
7. checking the installation
8. powering up the system

Be sure to read and understand the complete contents of any procedure described in this chapter before attempting it. If something is not clear to you or if you encounter problems, be sure to get help from your support provider.

Note: Your VME system configuration must conform to industry standard VME specifications in order to ensure proper operation of your subsystems.

Service and Support Information

When you purchased your system you may have purchased a support program from either Silicon Graphics, Inc., or a software vendor. Whenever you encounter any problems that you cannot solve using the methods described in this document, contact the organization from which you purchased the support program.

If you did not purchase a support program, contact your sales representative to arrange for support.

Electrostatic Discharge (ESD) Precautions

This equipment is extremely sensitive and may be susceptible to damage caused by Electrostatic Discharge (ESD). ESD is an electrical discharge (spark) caused by the build-up of electrical charge on clothing and other materials.

You must use proper ESD preventative measures:

- Connect a ground strap to your wrist and to the metal frame of the VMEbus card cage.
- You and all the electrical equipment that you handle during this installation must be at ground potential to avoid damage from ESD.
- Keep the boards in the antistatic bags provided.
- Remove a board from its antistatic bag only when you are properly grounded to the card cage with a ground strap.
- Do not use an ohmmeter on the boards.
- The card cage should be powered down and grounded until the installation is complete and ready for testing. The AC power cord provides a ground for the card cage and should be left plugged in unless some other ground is provided.

Electromagnetic Interference (EMI) Compliance

To make this system comply with FCC, VDE, and VCCI emission requirements, follow these guidelines:

- Ensure that the VMEbus chassis and power supply meet FCC, CISPR-22, and VDE specifications.
- Securely tighten screws holding the VMEbus cards.
- Install blanking panels over unused slots.
- Connect all peripheral devices with shielded cables.
- Ensure the internal and external unshielded cables that connect to Input/Output connectors have EMI suppression components such as ferrite cable, cable clamps, filters, etc.

General Guidelines for Installing Hardware

Remember the following guidelines whenever you install or handle the electronic equipment described in this guide:

- In addition to electrostatic discharge, this equipment is also susceptible to damage from physical impact, exposure to excessive temperatures, moisture, solvents, and other conditions known to damage electronic equipment. Please handle with care!
- This equipment has been carefully designed and manufactured to fit properly into the VMEbus card cage. While moderate force is necessary to seat a subsystem in the card cage, you should not have to struggle. If you experience any difficulty whatsoever, remove the subsystem from the card cage and inspect both subsystem and backplane for bent connector pins, obstructions, or other damage. *Do not apply impact or excessive force to seat a subsystem into the card cage; never use tools or other implements to seat a board into a card cage.*
- Use only AC outlets that are properly grounded.

- Make sure that the power supply is adequate for the number of slots being used; typically 30W at 5V for each slot.
- Make sure the card cage has adequate air flow for cooling purposes; use covers and airflow restrictors on all empty slots.

Warning: **Never expose the card cage power supply. Power supplies generate high voltages, *even when not plugged in!* Contact with an exposed or improperly grounded power supply can deliver a severe electrical shock. The power supply must remain grounded and shielded. If you are not certain which coverings prevent access to the power supply, refer to the documentation for your card cage before removing any of its coverings.**

Preparations

Both CPU and graphics subsystems arrive assembled and ready to install. Before you begin, make sure that you have the proper items on hand and that the operating conditions described in Appendix B, "Operating Conditions" have been met.

Before removing the board assemblies from their packaging, make sure that the ambient temperature is between 0°C and 55°C. The work area should be free of excessive dust, airborne fumes, solvents or other possible contaminants, and any other conditions that might result in damage to electronic equipment.

In addition to this guide, you will need the items listed below to install this equipment successfully:

- an electrostatic discharge grounding strap for each person who may need to handle the PC-board assemblies
- a small standard (flat-head) screwdriver
- any other tools needed to remove covers or enclosures surrounding your VMEbus card cage

Note: Before installing the CPU or graphics subsystem, check the ground pins on the P1 and P2 connectors of the slots in which you intend to place it. These pins must be present to ground the subsystem properly.

Power Supply

Be sure your power supply adequately supports the number of slots in your backplane.

The TG - V graphics subsystem has between two and four 6U VMEbus boards, depending on the configuration. When installing multiple subsystems, it is especially important to be sure that the power supply is adequate for the number of slots taken up. The VMEbus standard of 30 watts at 5 volts per slot is sufficient. (See Appendix B, "Operating Conditions," for information on power requirements for this CPU.)

The CPU Subsystem

Make sure a V20 or V30/V35 CPU subsystem is properly installed before installing the TG - V graphics subsystem. Please refer to the system integrator's guide included with your CPU for information on installing that subsystem.

Note: The V30/V35 CPU subsystems support either TG - V graphics or Elan/XS - V graphics. Be sure your CPU subsystem is properly configured to run your TG - V graphics subsystem. See Appendix D, "Switches, Jumpers and Memory Maps," in the *V30/35 System Integrator's Guide* for more information.

Unpacking

While unpacking your TG - V graphics subsystem remember to inspect each assembly and any accompanying materials for damage. When it is not in the card cage, place the new equipment on a clean, stable surface that is safely insulated from electrostatic discharge (ESD). If possible, leave the PC-board assemblies in their conductive plastic wrappings when they are not in the card cage.

Caution: Be sure to wear an approved electrostatic discharge (ESD) grounding strap whenever you handle this equipment.

Your package should contain the following items:

- the TG - V graphics subsystem similar to the one shown in Figure 3-1
- a ribbon cable to connect the graphics subsystem to the CPU
- 10 feet of video cable (optional)
- A single CD-ROM or several installation tapes. Depending on the options you ordered, the installations tapes include the following:
  - Execution Only Environment tape, Part 1 (*eoe1*)
  - Execution Only Environment tape, Part 2 (*eoe2*)
  - optional Development Environment tape (*dev*)

Note: The installation software may have been packaged with your CPU subsystem.

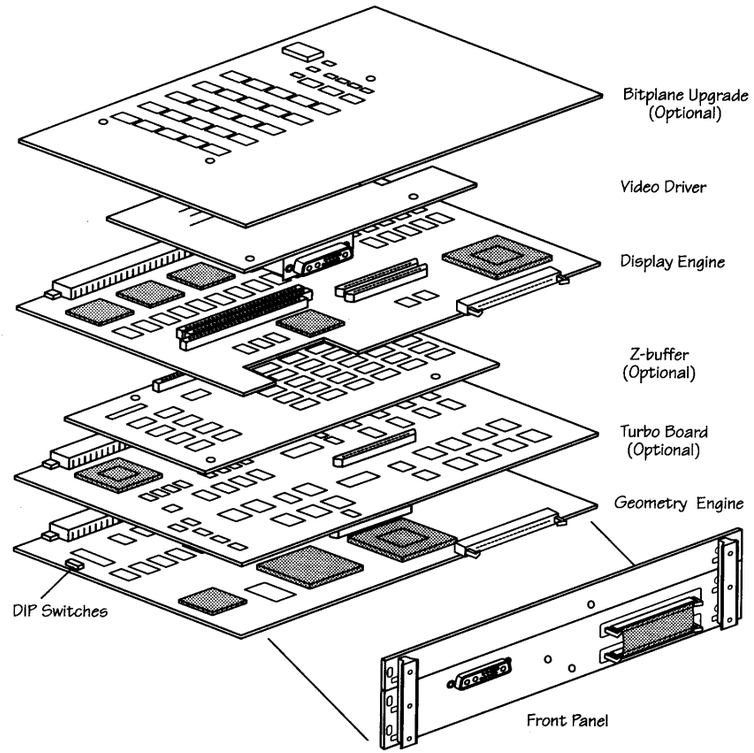


Figure 3-1 TG - V Graphics Subsystem

Installing the TG - V Graphics Subsystem

The TG - V graphics subsystem is not a VMEbus device. It uses only the power and GND pins and the user definable pins on the P2 connector (rows A and C).

The graphics subsystem can reside in any set of slots; however, it is recommended that you place it next to the CPU subsystem, and that you place each subsequent graphics subsystem in an adjacent slot.

To install one or more graphic subsystems:

1. Set the unit numbers if you're installing multiple graphics displays.

   By default, each graphics subsystem is set up as unit number zero. For systems with multiple displays (up to four), you must assign each graphics subsystem a separate unit number using the S601 switch located on the top edge of the Graphics Engine board (see Figure A-2 in Appendix A, "Component layout"). For information on setting this switch, see Appendix C, "DIP Switches."

2. Orient the graphics subsystem so that the component side (top) faces toward the right with respect to the faceplate. Make sure that each board is properly aligned in its respective slot. (See Figure 3-2.)

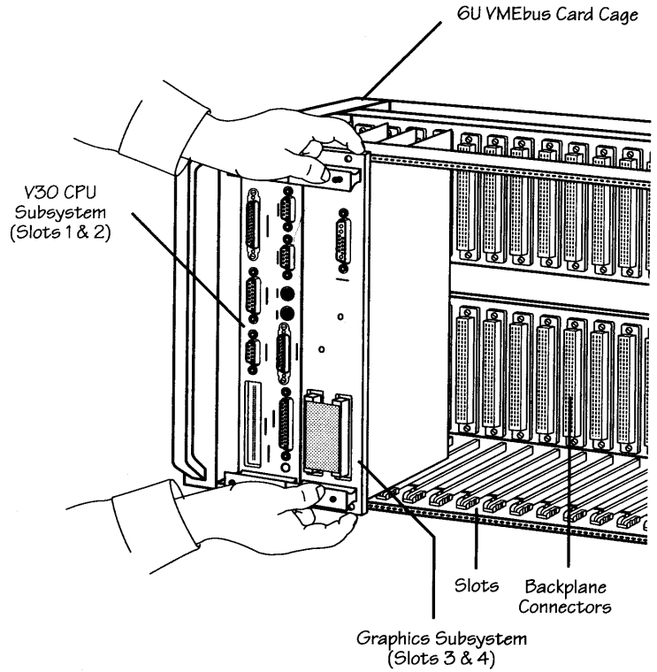


Figure 3-2 Inserting the TG - V Graphics Subsystem

3. Slide the subsystem back along the slots and seat it into the backplane with a firm, steady pressure. The subsystem should slide smoothly along the slots and fit snugly into the backplane connectors.

4. Secure the subsystem to the card cage using the captive screws located near the extractor tabs at the top and bottom of the faceplate.

5. Remove any coverings or enclosures to access the rear of the VMEbus backplane.

6. Connect the graphics subsystem(s) to the CPU subsystem.

   - If you have a V20 CPU subsystem, one end of the provided ribbon cable has two closely spaced connectors. Attach this end to the P2 connectors of slots containing the CPU subsystems' I/O and CPU boards (the two outside boards).

     If you have a V30/V35 CPU subsystem, connect one end of the ribbon cable to the P2 connector of the slot containing the I/O board.
   - Attach the other end of the ribbon cable to the P2 connector of the slot containing the Geometry Engine board (the board closest to the CPU subsystem).

   Figure 3-3 illustrates the TG - V graphics subsystem properly connected to a V20 CPU subsystem.

   Figure 3-4 illustrates the TG - V graphics subsystem properly connected to a V30/V35 CPU subsystem (without the optional memory upgrade).

   Note: If you're installing multiple graphics subsystems, you must use a special multihead ribbon cable that is ordered separately from Silicon Graphics.

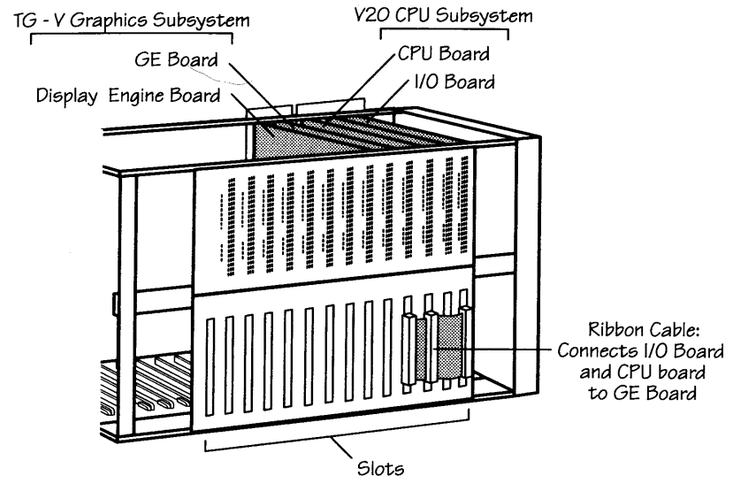


Figure 3-3 TG - V Graphics Connected to the V20 CPU Subsystem

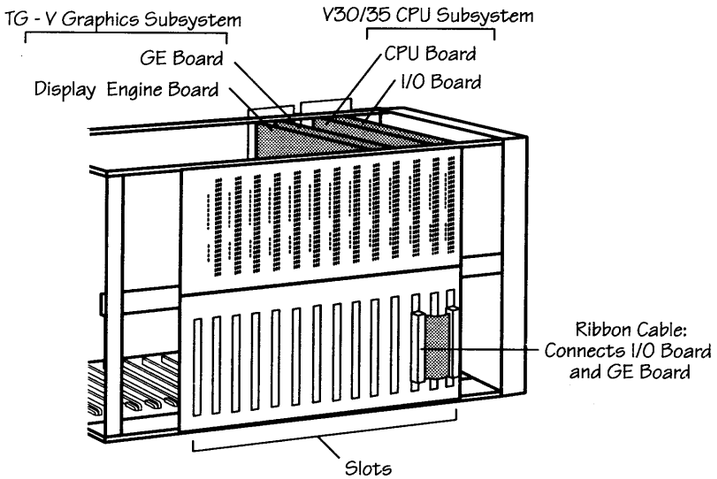


Figure 3-4 TG - V Graphics Connected to the V30/V35 CPU Subsystem

7. Replace any missing jumpers from empty slots between VMEbus cards. You do not jumper slots occupied by a graphic subsystem; these lines are connected through the boards.

   Figure 3-5 illustrates a properly jumpered empty slot between the V30/V35 CPU subsystem and the TG - V graphics subsystem; however, it is recommended that you place the graphics subsystem in the slots immediately adjacent to the CPU subsystem.

8. Use a monitor cable to connect your monitor to the video port on the TG - V graphics subsystem.

9. Repeat steps 1 through 8 for each graphics subsystem to be installed.

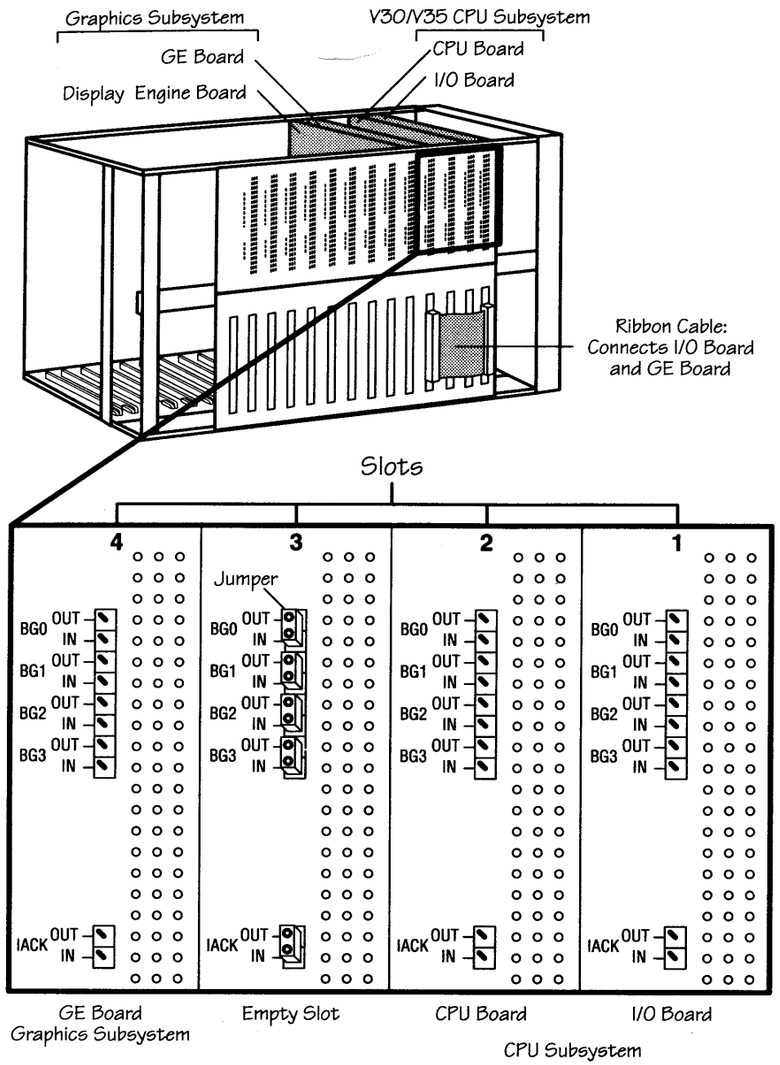


Figure 3-5 VMEbus Backplane Jumpers

Installing Additional VMEbus Devices

At this time you can install any additional VMEbus-resident devices by following the manufacturer's instructions.

Connecting Peripheral Devices

With the power to the card cage still off, you are now ready to connect peripheral devices to your CPU subsystem. Please refer to the section titled "Connecting Peripheral Devices" in your CPU subsystem integrator's guide for complete information on connecting devices to the CPU subsystem.

Checking the Installation

When all of the desired boards are installed, you are ready to inspect the card cage. After verifying that the installation is correct, you can replace any coverings or enclosures to secure the cage. (Depending on the physical layout of your VMEbus card cage, you may need to replace covers or enclosures before connecting your peripheral devices.)

When inspecting the card cage, ensure that:

- Assemblies and boards are seated properly and secured.
- Unoccupied slots between VMEbus cards are correctly jumpered.
- For each graphics subsystem, the GE board is connected to the appropriate CPU subsystem.
- Connections and jumpers are tight and solid.
- Peripheral connectors are securely attached.
- CPU jumpers and switches are properly configured for your environment.

- Air flow inhibitors are properly installed (where appropriate) in unoccupied slots.
- You follow the guidelines for maintaining EMI compliance as outlined under the heading "Electromagnetic Interference (EMI) Compliance" earlier in this chapter.

Once you verify that the hardware is properly installed, you can power up the VMEbus card cage.

Powering Up the System

Connect the card cage and any peripherals that require it to a grounded source of AC power, and switch on the card cage to power up the system.

When the system first comes up, the CPU performs a number of diagnostic tests on itself. The CPU diagnostics are displayed on the ASCII terminal if the *bootmode* remains on its default value, *bootmode=d*.

The CPU diagnostics are not displayed if the *bootmode* has been reset so that *bootmode=c*.

After the CPU diagnostic tests are completed, a series of graphics subsystem tests are run if a graphics subsystem is present. The results of the graphics diagnostic tests appear on an attached graphics monitor if the *console* value remains on its default value, *console=g*.

If no graphics monitor is attached, the results of the graphics test appear on the ASCII terminal. The test results also appear only on the ASCII terminal if the console value has been reset so that *console=d*.

Note: See the section "Preparing the Standalone Environment" in Chapter 4 for information on setting the *console* and *bootmode* variables.

These tests take a minute or two to complete. In case of error or failure, contact your support representative or the organization from which you purchased your support program. If you have not purchased a support program, contact your sales representative to arrange for support. Do not attempt to correct or repair any boards or components. Refer to Chapter 5, "Diagnostics" for an explanation of the power-on self-tests.

When the self-test completes successfully, you see the following message on the system console:

```
Starting up the system ...

To perform system maintenance instead, press <Esc>
```

Press **<Esc>** to bring up the System Maintenance menu. At this point, the hardware installation is complete; you are ready to install optional software.

Chapter 4

Software Installation

This chapter explains how to install system software onto a V20 or V30/V35 subsystem directly from tape or CD-ROM. The procedure takes about an hour to complete.

You should install the system software at the time you install the CPU subsystem. The installation instructions outlined in this chapter are the same as those provided in the CPU subsystem integrator's guide; they are included in this chapter only for your convenience. You may skip this chapter if you have already installed the software.

Note: The graphics subsystem must be properly connected to the CPU subsystem when the software is installed to ensure that the appropriate software options are installed.

Software for the V20 and V30/V35 includes the standard IRIX Execution Only Environment and the optional IRIS Development Option (ido).

The installation process involves loading the system software onto disk using the standalone *inst*(1M) program. For background information about the *inst* program and the installation process in general, refer to the *IRIS Software Installation Guide.*

Alternatively, you can install system software over a network using a tape drive or CD-ROM attached to a remote host. For multiple installations, you can copy the software onto a remote host's disk using *distcp*(1M), and then install it over the network. The *Personal System Administration Guide* describes these procedures for installing software over the network.

Installing Additional Software

You may want to install additional software that you receive from Silicon Graphics or another company that writes application software or device drivers. Always follow the instructions that come with the software to make sure you install new software correctly.

Preparations

To install the IRIX system software from tape or CD-ROM you need:

- this guide
- *IRIS Software Installation Guide*
- a tape drive or CD-ROM device that can be connected to the V30/35
- a partitioned disk drive that holds at least 380 MBytes of data (the drive is already partitioned when you receive it from Silicon Graphics)

  For information on formatting and partitioning disk drives, refer to the *IRIX Site Administrator's Guide.*
- the Execution Only Environment (eoe) CD-ROM or tapes (1 and 2)

  The CD-ROM contains the entire eoe. If you are using tapes, tape 1 includes the IDE diagnostics program, the miniroot, and the first portion of the standard system software release; tape 2 includes the remainder of the standard release.
- the (optional) IRIS Development Option (ido) CD-ROM or tape that holds the standard software development tools

Note: You may have received a disk drive that already has the eoe system software installed. If so, you need to install only the optional software you've purchased.

Preparing the Hardware

Before attempting to install system software, make sure that system hardware:

- is properly installed in the VMEbus card cage (as described in Chapter 3)
- passes the power-on self-tests

Figure 4-1 shows one possible system configuration using external SCSI disk and tape drives. Remember to power down the card cage before connecting or disconnecting the drives.

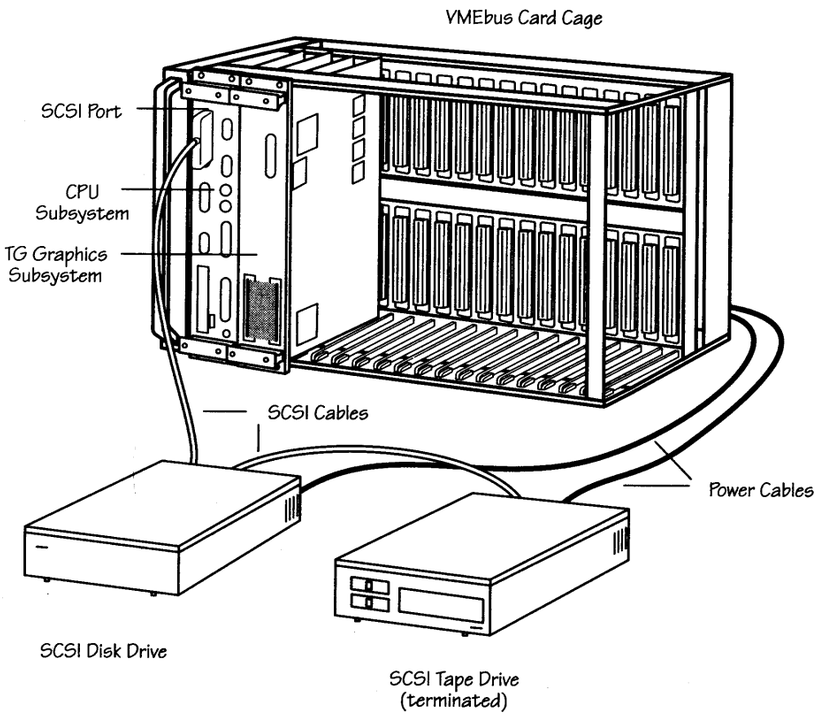


Figure 4-1 Installing Software with External SCSI Tape and Disk Drives

Preparing the Standalone Environment

With the system powered up, make sure that the standalone environment is set up properly. The system gives the message:

```
Starting up the system ...

To perform system maintenance instead, press <Esc>
```

When the message appears, do the following:

1. Press the **<Esc>** key to bring up the System Maintenance menu. The system then displays:

```
System Maintenance Menu

1 Start system

2 Install System Software

3 Run Diagnostics

4 Recover System

5 Enter Command Monitor

Option?
```

2. Enter the command monitor by choosing option 5. The command monitor issues the >> prompt.
3. Make sure that the correct standalone environment is present by giving the *printenv* command. You should see several standalone environment variables listed, including the following:

```
...
bootfile=dksc(0,1,8)sash
...
console=g
...

bootmode=d

...
```


4. The *bootfile* variable determines the device from which the system is to boot. In order to install system software using a local tape drive or CD-ROM device, this variable must have the value shown. If the *bootfile* variable is not correct, use the *setenv* command to assign the correct value to this variable:

```
setenv bootfile dksc(0,1,8)sash
```

5. The *console* environment variable indicates that the system console is either the graphics monitor (g or G) or an ASCII terminal (d); this means that the console is attached to the graphics (g) port or the diagnostics (d) port. The graphics monitor is the default setting; however, if you do not have a graphics subsystem in operation, the software automatically switches to an ASCII terminal.

   If you wish to manually set the console variable, type:

```
setenv console g
```

   for a graphics monitor, or type

```
setenv console d
```

   for an ASCII terminal.

   Note: The upper-case graphics monitor variable *G* displays a Silicon Graphics logo in the upper left corner of the screen, the lower-case *g* does not; this is the only difference between the two.

6. The *bootmode* variable determines whether the CPU diagnostics are displayed when the system is booted. The default variable, *bootmode=d*, displays the diagnostics. You can manually set this value by typing:

```
setenv bootmode d
```

   If you do not want the diagnostics displayed, type:

```
setenv bootmode c
```

7. When these variables are set to the correct values, type `init` to return to the System Maintenance menu. You are now ready to begin installing system software.

Installing the System Software

You can install the system software from a local drive attached directly to the V30/V35 subsystem or from a remote drive over a network.

Note: If you are not using a preformatted hard disk from Silicon Graphics, you must format your disk before installing software. Refer to the *IRIX Site Administrator's Guide* for instructions on formatting or partitioning a hard disk.

To install the system software:

1. Insert the eoe1 tape into the tape drive, or the CD-ROM into the CD-ROM drive. Choose option 2, "Install System Software," from the System Maintenance menu. When you do, the system displays:

```
Installing System Software ...
Press <Esc> to return to the menu.

Insert the installation tape, then press <enter>:
```

2. Press the **<Enter>** key to load the miniroot and *inst* program. This takes about 10 minutes. When *inst* is installed and ready, the system displays:

```
Inst Main Menu

1. from [source]
2. list [keywords]
3. go
4. install [keywords] [names]
5. remove [keywords] [names]
6. keep [keywords] [names]
7. step [keywords] [names]
8. versions [args]
9. help [topic]
10. admin
11. quit

Inst>
```


3. Choose option 3 to automatically install the entire eoe1 distribution. If you are installing from CD-ROM, skip steps 4 and 5.
4. If you are installing from tape, in about 20 minutes you are asked:

   ```
   Is there more software to install?
   ```

   Answer **y** to this question to proceed with the next tape.
5. You next see the prompt:

   ```
   Insert the next tape, then press <enter>:
   ```

   Replace the eoe1 tape with the eoe2 tape and press **<Enter>**. This tape takes about 15 minutes to install. Repeat this process for the dev tape (about 5 minutes).
6. When you are done, answer **n** when *inst* asks:

   ```
   Is there more software to install?
   ```

   When you answer **n** to this question, *inst* may ask if it should automatically reconfigure the kernel. Answer **y** if so prompted.

   The software is now installed.
7. Answer **y** when *inst* asks if you want to start the operating system. This reboots the system.

For additional information on installing the system software, refer to the *IRIS Software installation Guide*.

Chapter 5

Diagnostics

Your Silicon Graphics system provides extensive support for testing both CPU and graphics subsystems. The subsystems come with two sets of diagnostics:

- a basic set of power-on tests ensures that the system can bootstrap and run standalone programs
- the interactive diagnostics environment (IDE), a standalone program that runs in the PROM monitor through the standalone shell (*sash*)

This chapter describes both sets of diagnostics and the various tests they perform. If you need general information about the PROM monitor and the standalone shell, please refer to the *IRIX Site Administrator's Guide* that came with your system.

Power-on Tests

The following power-on diagnostics are performed automatically whenever the system is powered up. The "Fault" (yellow) light-emitting diode (LED) flashes while these tests are under way. Output from these tests is displayed only when an ASCII terminal is used as the system console.

In addition to the CPU power-on tests, a test of the graphics subsystem is run if one is present. In the case of multiple graphic subsystems, the test is run only on subsystem 0. Output from this test is displayed only when an ASCII terminal is used as the system console.

Test Failure

When a power-on test fails, the PROM monitor may not come up, or it may come up displaying an error message.

If the memory test detects a bad SIMM module, the failure message displays an IDE alphanumeric symbol to indicate the specific byte that needs replacing as well as the SIMM bank in which it's located. The SIMM bank can be located on either the CPU board or the optional Memory Expansion Board. Refer to the system integrator's guide included with your CPU for information on locating a bad SIMM module.

Using IDE Diagnostics

The interactive diagnostics environment includes comprehensive tests for all of the I/O interface controllers and ports, the graphics subsystem (subsystem 0 only) and its various components, main memory, and other system components.

You can use the IDE program to verify that your system hardware is performing properly or to isolate hardware malfunctions. The IDE program must be loaded into the monitor from disk, from tape, or over the network before you can run it. If you have installed the operating system as described in Chapter 4, the IDE program is on disk and ready to run.

Starting IDE in Terse or Verbose Mode

You can run the IDE program in either terse or verbose mode. To start the diagnostics program in terse mode select option 3, "Run Diagnostics," from the System Maintenance menu. While in terse mode, the program first reports on the hardware configuration and then runs silently, displaying error messages only when a diagnostic fails on an installed board or component.

When IDE runs in terse mode, you cannot interrupt the test suite. You must let it run to completion or reset the system.

You must run IDE in verbose mode if you wish to use it interactively. In verbose mode, you can interrupt the standard test suite at any time by pressing `<Ctrl-C>`; the program then issues the `ide>>` prompt and accepts the commands described in the next section, "IDE Interactive Commands."

To start IDE in verbose mode:

1. Enter the command monitor by selecting option 5 on the System Maintenance menu.
2. Type **`ide fe`** in response to the command monitor's >> prompt.

In verbose mode, the IDE program reports on each test it attempts, along with the success or failure of that test.

Initializing the Graphics Subsystem

It is a good idea to initialize the graphics subsystem by using the *initgr* (V30/V35) or *initgr1* (V20) IDE command prior to performing any graphics tests. Graphics tests that *must* be preceded by *initgr* or *initgr1* are listed in Table 5-1.

V20	V30/V35
re	gr1_re
redma	gr1_redma
gedma	gr1_redma
cursor 0|1	gr1_cursor 0|1
bitp	gr1_bitp
xmap	gr1_xmap
patterns	gr1_patterns
ge5load	n/a
gefifo	gr1_gefifo
drama2	gr1_drama2

Table 5-1 Graphics Tests That Must be Preinitialized

The tests outlined in Table 5-2 are for the TG - V graphics Turbo option. Run these tests only after running the commands *initgr1* and *dspload x* (V20) or *initgr* and *gr1_dspload x* (V30/V35).

V20	V30/V35
dspiram x	gr1_dspiram x
dspirama x	gr1_dspirama x
dspiramd x	gr1_dspiramd x
dspre	gr1_dspre

Figure 5-2 Turbo Graphics Tests That Must be Preinitialized

IDE Standard Tests

In either terse or verbose mode, IDE first performs an initial inventory of subsystems that are present (the equivalent of the *hinv* PROM monitor command). The initial output you see looks something like this:

```
Memory size:xx MBytes
Instruction cache size:xxxxx
Data cache size:xxxxx
CPU board:xxxxx
System options:xxxxx
...
Graphics: xxxxx
SCSI disk:dksc(0,1)
```

Once the program starts, it runs a standard test suite. These tests cover all major subsystems and components, taking about 25 minutes to complete.

The standard test suite has two parts. The first part tests the CPU subsystem. The second part tests the graphics subsystem if one is present. Some tests apply only to specific models or options; if IDE attempts to access a piece of hardware that is not supported on this system, IDE reports the device as missing and continues with the next test.

Note: Certain tests take over the graphics display monitor. When these tests are running, the graphics monitor may go black or display various shapes and patterns. This is no cause for alarm. When the tests finish running, the monitor resumes the console display.

CPU tests can fail with any of the following errors. If IDE reports one of these errors, contact your support provider.

```
Failure detected on CPU board.
Failure detected on FPU.
Failure detected on VME device.
Failure detected on SCSI device[s].
```

Graphics tests can fail with any of the following errors; contact your support provider if IDE reports one.

```
DMA failure detected between CPU and GRAPHICS boards.
Failure detected on GRAPHICS board.
Failure detected on BITPLANE EXPANSION board.
Failure detected on AUXILIARY PLANES board.
Failure detected on Z-BUFFER board.
Failure detected on Turbo board.
```

IDE Interactive Commands

The interactive commands available under IDE allow you to check the operation of the boards and components that make up the system.

The IDE interactive commands are classified by subsystem. General commands and those that pertain to the graphics subsystem are listed below. The IDE commands for the CPU subsystem are listed in the system integrator's guide that came with your CPU subsystem.

General Commands

The following general commands are used to modify more specific commands pertaining to either the CPU or graphic subsystem:

{cmd; cmd ... }
Command may be grouped between { }.

? Provides on-line help for each IDE command.

exit Exit IDE.

for *[expr1; expr2; expr3] cmd*
Repeat *cmd* while *expr2* is true; similar to the *for* statement in the C programming language.

if *[expr] cmdl [else cmd2] [fi]*
Execute *cmdl* if *expr* is true, *cmd2* otherwise; *fi* is optional.

repeat *n cmd*
Repeat *cmd* n times.

report *[n]* Set verbose level, where *n* represents a scale of one to four, with four being the most verbose.

wait [*"message"*]
Display the indicated message, if any, and wait for an `<Enter>` to be typed in on the keyboard.

while *[expr] cmd*
Repeat *cmd* while *expr* is true.

Graphics Subsystem Tests

The following are TG - V graphics subsystem tests.

bitp Test all bitplanes (including z-buffer). *bitp* has the following options:

bitp [0|1|2|3|4|a|b|z|?]

Bitplane tests:

0 Test bitplanes with 0x0000000
2 Test bitplanes with 0xffffffff
3 Test bitplanes with 0x5555555
4 Bitplane address uniqueness test
a Test auxiliary planes option only
b Test bitplane expansion (BP4) only
z Test z-buffer only

brd x *y* [*count*]
Read bitplane location:
x Horizontal coordinates 0-1024
y Vertical coordinates 0–1280
count Number of repetitions

bwr *x y value*
Write bitplanes:
x 0–1280
y 0–1024
value Any 32-bit value

clr Clear graphics monitor screen.

ctl1 Graphics DMA channel registers data test (with V20 only).

ctl2 Graphics strobe test (with V20 only).

cursor 0 | 1 Test cursor.

dac Test DACs.

drama Microcode data RAM address uniqueness test.

drama2 Graphics subsystem microcode RAM test.

dramd Microcode data RAM walking bit test.

drd *offset* [*count*]
Read microcode data RAM (range 0-8k):
offset Starting address
count Number of words to test

dspconv IEEE to digital signal processor (DSP) floating point conversion test.

dspfifo Test FIFO on turbo board.

dspiram [*dsp*]
Turbo board DSP internal RAM test:
dsp DSP number (0-3)
DSPs loaded as follows:
0 = 0
1 = 0, 1
2 = 0,1,2
3 = 0,1,2,3

dspload [*dsp*]
Download the DSP microcode:
dsp DSP number (0-3)
DSPs loaded as follows:
0 = 0
1 = 0, 1
2 = 0,1,2
3 = 0,1,2,3

dsprama Turbo external RAM address uniqueness test.

dspramd Turbo external RAM data test.

dsprd *addr count*
Read from DSP RAM:
addr Starting address
count Number of words to test

dspre [*dsp*]
Test data path between DSP and raster engine on turbo board:
dsp DSP number (0-3)

dspscope *re_dat re_reg*
DSP scope test:
re_dat Data word
re_reg Register number

dspwr *addr hex_val*
Write to DSP RAM:
addr Address
hex_val Data value

dwr *offset data*
Write to microcode data RAM:
offset Address
data Data value

fifo FIFO test

ge5load Download graphics diagnostic microcode.

gedma CPU to graphics and graphics to CPU DMA test.

gefifo Graphics FIFO test.

gfx_probe Probe for graphics hardware.

gr_exit Exit gracefully from graphics tests.

gr1_probe Probe for graphics hardware (V30/V35 only).

hinv Hardware inventory (same as *hinv* PROM monitor command.)

initgr Initialize graphics board—includes microcode download (V30/V35 only.)

initgr1 Initialize geometry engine board—includes microcode download (V20 only.)

lca1 LCA interrupt mask registers test (V20 only.)

lca2 LCA RAM readback test(V20 only).

lre Load raster engine register.

patterns Test spans with patterns.

pdma PIC1 DMA W/R test (V30/V35 only).

re Test of raster engine.

redma CPU to bitplanes and bitplanes to CPU DMA test.

rescope *re_dat re_reg*
Raster engine scope test:
re_dat Data word
re_reg Register number

resetcons Reset graphics console.

resetgr1 Reset graphics subsystem (V20 only).

setmon [*0*|*1*|*2*|*3*]
Set monitor timing:
0 1280 X 1024 pixel non-interlaced, 60Hz
1 1280 X 1024 pixel interlaced, 30Hz
2 645 X 485 pixel RS170, 30Hz
3 780 X 575 pixel EURO 30Hz

test *subcmd*

Memory tests; *subcmd* is one of:
data [–bhwiuf] *range*
Walking bit test
parity [–bhwiuf] *range*
Parity test
addr [–wiuf] *range*
Address uniqueness test
wire [–gmnpv] index page frame
Set up address mapping for addresses in the ranges 0x0–0x7fffffff and 0xc0000000–0x0fffffff

Arguments to the data, parity, and addr subcommands are:
–b Load and store bytes.
–h Load and store halfwords.
–w Load and store words (the default).
(The –b, –h, and –w options do not apply to address ranges for other commands such as checksum, dram, or ldram.)
–i Invert the binary sense of the test; use walking zero instead of walking one; use complemented address uniqueness instead of uncomplemented
–u Repeat test until an error occurs
–f Continue repeating (forever), despite errors
range a range of addresses taking one of two forms:
start_addr:end_addr
(Specifies a range from start_addr through,but not including, end_addr)
start_addr#count
Specifies a range from start_addr through that address plus the number of units specified by count; unit size is determined by **–b, —h, or –w**

An address can be specified in either hexadecimal or decimal, with the following scaling abbreviations:
K 1024 units
P 4096 units (page)
M 10242 units

For instance, the command:

```
test addr 0xa01M#0x1M
```

performs an address uniqueness test over the one megaword (4Mbyte) range starting at address 0xa0100000

Arguments to the **wire** subcommand are:

-g	Global mapping (don't compare TLBPIDs).
-m	Allow writes at virtual addresses given by *page*.
-n	Do not look in cache after translating an address.
-p	Purge page map of entries.
-v	Mark map entry as valid for address translations.
index	A map entry index between 0 and 63.
page	A page number (five hexadecimal digits).
frame	A page frame number (five hex. digits) into which addresses beginning with page's five digits are to be translated.

For instance, the following commands set up a walking bit test on two pages of mapped memory using pages 0 and 1:

```
test wire -pvm 0 0 0x18
test wire -vm 1 10x19
test data -b 0#2p
```

turbo_probe

Probe for graphics turbo board.

urama Microcode RAM address uniqueness test.

uramd Microcode RAM data test.

urd *offset* [*count*]

Read microcode RAM:

offset	
count	

uwr *offset t8bits b32bits* [*count*]

Write to microcode RAM:

offset	Starting address
t8bits	Top eight bits of data
b32bits	Bottom 32 bits of data
count	Number of words to write

xmap Xmaps/colormaps test

zwr *x y z value*

Write to the Z-buffer:

x	0–1279
y	0–1023
z	0–1023
value	32-bit data word

Chapter 6

TG - V Graphics Subsystem Architecture

The TG - V graphics subsystem provides hardware acceleration to support the embedded computing professional's need for sophisticated 2-D and 3-D graphics performance. The TG - V graphics subsystem (coupled with a V20 or V30/V35 CPU subsystem) offers the same binary compatible compute environment as other Silicon Graphics RISC-based workstations. This means that your development environment is also your production environment, allowing all the software that runs on the standard Silicon Graphics workstations to be used in your integrated solution.

The graphics subsystem relies heavily on VLSI technology to implement its functional components: the host interface, geometry engine, raster engine, and display engine. In addition, unique architectural innovations, such as optimized partitioning of the graphics pipeline and high-speed interleaved frame buffer access, result in a tightly integrated architecture that provides fast 2-D performance and the same 3-D performance found in Silicon Graphics' Personal IRIS line of workstations.

The CPU subsystem provides the graphics subsystem with descriptions of 2-D and 3-D objects; the graphics subsystem computes and displays their images. 3-D objects are described in terms of Graphics Library routines using world coordinate vertex data. These descriptions include each object's geometric position, color, and surface normal vectors (for lighting calculations). The graphics subsystem performs transformations and other graphics operations on these objects to calculate specific pixel values for each of the 1.3 million pixels on a 1280 x 1024 high-resolution monitor.

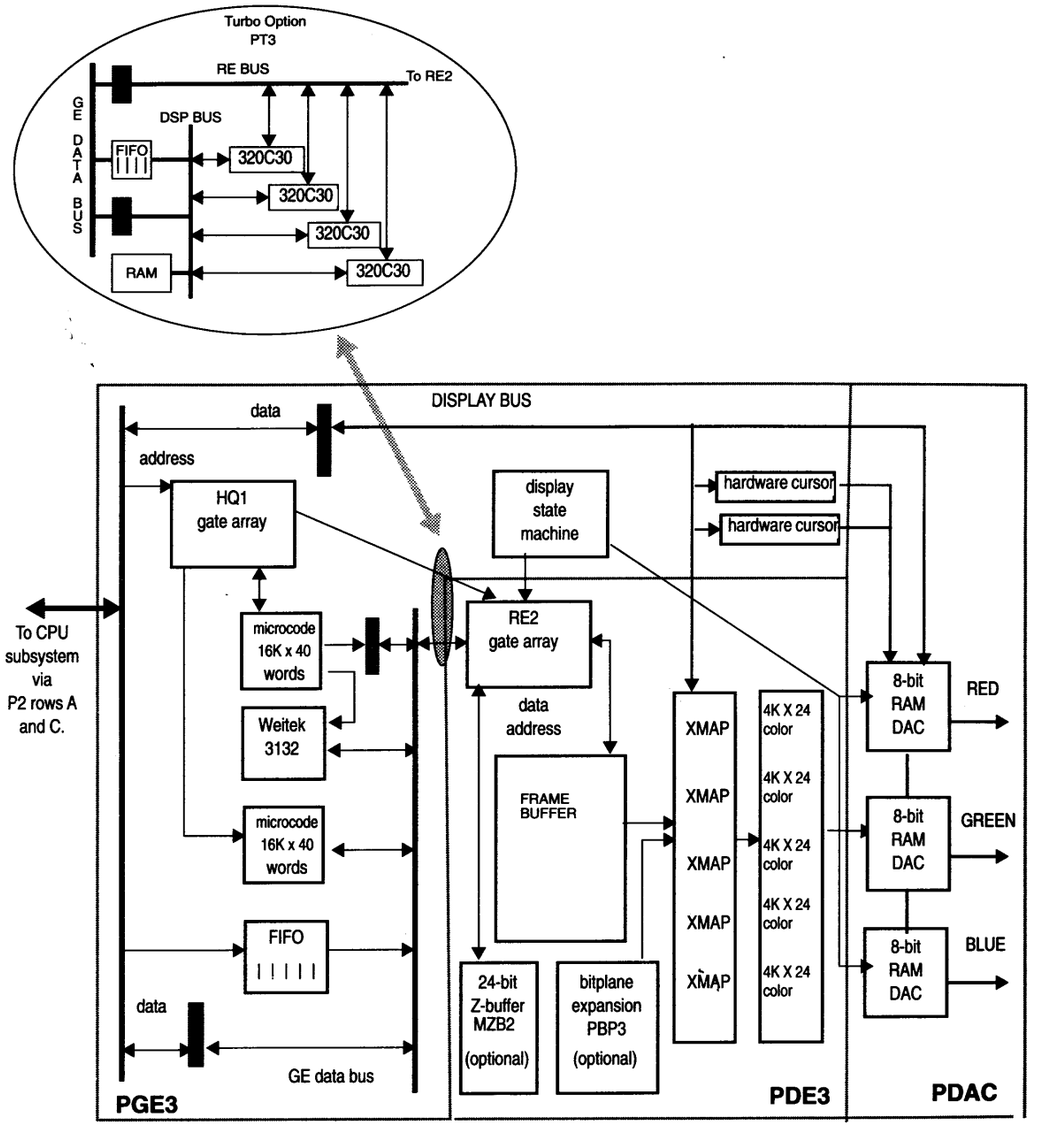


Figure 6-1 Graphics Subsystem Block Diagram

The graphics subsystem employs two internal buses: a display bus and a Geometry Engine data bus. Graphics commands and data from the CPU take one of two routes within the graphics subsystem:

- Display commands to update the color map or cursor are immediately sent out over the display bus to update the appropriate locations.
- Drawing commands are routed to the Geometry Engine through a FIFO.

Before being displayed on the screen, graphics commands from the RISC host are processed by four pipelined graphics subsystems.

Host-interface Subsystem
⇓
Geometry Subsystem
⇓
Raster Subsystem
⇓
Display Subsystem

The Host-interface Subsystem receives the commands and data from the RISC-based host via the Silicon Graphics GFX Bus (which is implemented on the user definable pins of the VMEbus) and distributes them to the appropriate component of the system. The Geometry Subsystem transforms all world coordinate data, computes slopes of line and polygon edges, and begins scan conversion. It handles graphics primitives such as points, lines, convex and concave polygons, characters, and splines, and it can calculate them with a variety of lighting models. The Raster Subsystem scan converts lines and spans, and it draws pixel data into the frame buffer and the optional z-buffer. The Raster Subsystem also performs antialiasing and dithering. Finally, the Display Subsystem reads the frame buffer and displays the image on the color monitor.

Host-Interface Subsystem

The Host-interface Subsystem contains a 32-bit Silicon Graphics GFX bus interface to the host CPU, a set of FIFOs and interface logic, and two Silicon Graphics ASICs. The host-to-graphics bus, the GFX bus, is synchronous and minimizes handshaking for maximum throughput. The GFX bus has been implemented on the user definable pins of the VMEbus P2 connector.

The interface is primarily controlled by the Silicon Graphics-designed HQ chip. This chip has the dual purpose of maintaining the interface to the host and being the Geometry Engine microcontroller. The HQ handles either single-access transfers from the host or burst DMA cycles. Single-cycle accesses pass color map and other display data to the Display Subsystem and graphics commands to the Geometry Engine Subsystem. The HQ inserts graphics commands in the FIFOs to match their flow more evenly with Geometry Engine compute time. This allows the host to continue to pass graphics commands to the HQ even when the Geometry Engine is occupied.

The HQ supports burst DMA transfers between the host and the Geometry Engine data RAM and between the host and the frame buffer. Burst transfers let the host interact rapidly with data used by the Geometry Subsystem and access the pixel data stored in the frame buffer faster. Fast pixel access lets the HQ transfer pixel images rapidly from the host to the frame buffer or retrieve frame buffer data to perform pixel operations on the host. Burst transfers to and from the Geometry Engine data RAM let the HQ switch contexts and read back Geometry Engine results quickly.

Geometry Engine (GE) Subsystem

The Geometry Engine Subsystem consists of data RAM and the Geometry Engine chip. This subsystem receives world coordinate geometric data via the 32-bit GFX bus interface, performs the specified transformations and lighting computations, and then performs any necessary calculations to reduce the subsequent vertex data into spans, lines, or points to pass to the Raster Subsystem. The Geometry Subsystem contains a single microcoded processor capable of 20 million floating point operations per second (MFLOPS), a loadable micro-store holding hand-coded routines, a data store, and the HQ microcontroller that controls the program and data flow.

The Geometry Engine processes a stream of high-level graphics commands from the FIFOs mixed with single-precision floating point data. A stack of 4 x 4 coordinate matrices rotates, translates, and scales 2-D and 3-D homogeneous coordinates. A stack of 3 x 3 matrices also transforms surface normals. The Geometry Engine also maintains position, direction, and intensity specifications for lighting calculations. It supports up to eight local or infinite point-source lights. Material specifications include coefficients for ambient, diffuse, and specular reflective aspects of an object. The color the system applies to each transformed vertex is a function of the vertex position, the normal direction, the lighting model, the lights, and the characteristics of the surface. The result of the lighting calculation is either a set of three 8-bit red, green, and blue values (RGB mode) or a single 12-bit color index (color index mode).

The Geometry Engine then clips vertex coordinates to a 6-plane bounding box. A fast accept/reject clip-checking algorithm eliminates the need for complex clipping calculations in most cases. When complex clipping is required, the Geometry Engine uses a derivative of the Cohen-Sutherland clipping algorithm to perform a perspective division and convert coordinates to screen space.

The Geometry Engine also does the first stage of scan conversion. The Raster Subsystem can draw only lines or spans, to which the Geometry Engine immediately passes the appropriate slope information. For line primitives, the Geometry Engine need only compute appropriate slopes before turning over responsibility to the Raster Subsystem. Scan conversion of polygons requires three steps:

1. Break up a polygon into trapezoids.
2. Calculate the edge slopes of each trapezoid and iterate z, R, G, B up the sides of these edges on span line boundaries (incrementing the y value by 1 and finding z, R, G, B values).
3. Calculate the span slopes and iterate between span endpoints to fill in the pixels.

The Geometry Engine performs the first two steps as illustrated in Figure 6-2.

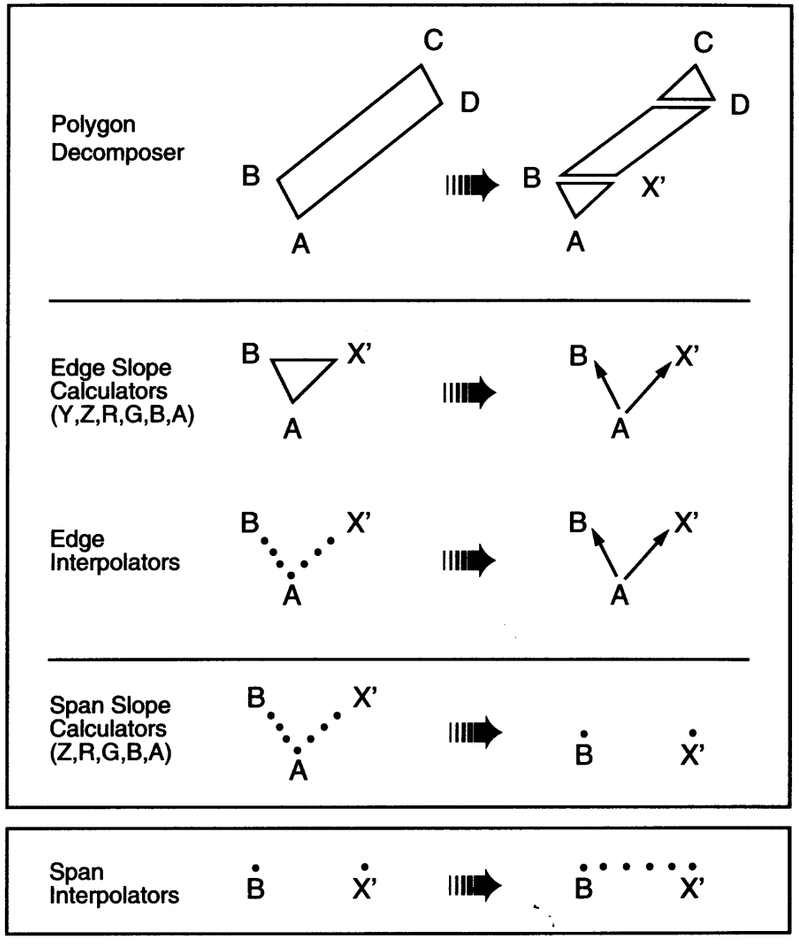


Figure 6-2 Scan Conversion Subsystem

The Geometry Engine performs scan conversions using floating-point precision to maintain coordinate integrity during iterations and slope calculations. Additionally, it computes the *x* coordinates of polygon edges to a fractional pixel tolerance. The Geometry Engine first corrects all depth and color component iterations to the nearest pixel center and then iterates them in full pixel steps. As a result, iterated color and depth values remain planar across polygonal surfaces, and subsequent z-buffer calculations result in clean intersections.

As a secondary task, the Geometry Subsystem continuously monitors the graphics pipeline so it can save the state of the pipeline when the host indicates that a new context has taken control of the CPU. The state of any context may be saved and restored quickly via the burst DMA channel.

The turbo option provides an enhanced level of Geometry Engine compute performance. This increased compute performance is provided by four TI 320C30 digital signal processing chips running at 16.67 MHz. In this configuration, graphics primitives are passed from the Geometry Engine to the DSPs, which feed drawing instructions directly to the Raster Engine. To supply these instructions in the correct order, primitives are pipelined to each of the four DSPs in round-robin fashion. The DSPs can perform single-precision (32-bit) floating point or 24-bit fixed addition and multiplication on graphics primitives, with each DSP providing approximately 1.5 times the performance of the Geometry Engine.

Raster Subsystem

The Raster Subsystem contains a single memory controller chip designed by Silicon Graphics (called the RE chip) which performs scan conversion and controls the frame buffer memory and optional z-buffer memory.

The standard frame buffer configuration has 12 bits of information at every pixel location on the screen. In this configuration, the user has 8 color bitplanes with which to work. The system uses two additional bitplanes for window ID control. The remaining two bitplanes are for overlay or underlay and are typically controlled by the Silicon Graphics Window Manager.

As an option, you can add 20 frame buffer bitplanes and 24 planes of z-buffer memory, making a total of 56 bits stored at each pixel (24-bit color, 4 bits Window ID, 4 bits overlay/underlay, and 24 bits of z-buffer). Note that the z-buffer option must be used in conjunction with the 24-bit color option.

Image Bitplanes

The image bitplanes store either 8 or 24 bits of information for each pixel of the 1280 x 1024 high-resolution display. The 8 bits/pixel configuration always uses color-index mode to draw 8-bit single-buffered, or 4-bit double-buffered images, while the 24 bits/pixel configuration allows you to choose among four modes: 24-bit single-buffered RGB operation, 12-bit double-buffered RGB, 24-bit single-buffered color index, or 12-bit double-buffered color index.

Depth Planes (Z-buffer)

When you enable hidden surface removal, the system compares the z coordinate of each incoming pixel to the current depth value already stored for the pixel. If the new z coordinate is closer to the viewpoint (and is therefore visible), the system updates the image bitplanes with the color value of the new pixel, and the z value of the new pixel is used to update the z-buffer at this location. Conversely, if the new z coordinate is farther away from the user's viewpoint, the pixel is hidden from view and is ignored. A full 24 bits of precision is maintained from the host into the z-buffer.

Overlay/Underlay Planes

The frame buffer dedicates either 2 or 4 bits per pixel to overlay and underlay operations. These bitplanes are provided for applications that use features such as pop-up menus and windowing backgrounds.

Window ID Planes

These 2 or 4 bitplanes hold the data that defines the state of the windows visible in the display. The window planes let you display multiple 3-D images in a concurrent display environment. The window planes also determine whether individual pixels need to be masked against a window on the screen. This allows the hardware to support non-rectangular windows (circular or any other shape) without imposing overhead on the window system. Additionally, they are used to determine the color mode format; color index or RGB and single or double buffered.

The RE2 Chip

The RE2 chip accepts commands, pixel location and color data, and slope data from the Geometry Subsystem. The RE2 uses this data to iterate horizontal spans or arbitrarily angled lines to produce pixel values that it writes into both the frame buffer and z-buffer. The RE2 updates only those bitplanes that are appropriate for the current drawing mode (single or double buffered, color index or RGB). The RE2 hardware supports fast screen-clear, antialiasing color index lines, and dithering shaded images via a 4 x 4 pixel randomizing scheme.

The RE2 controls all memory timing for both the frame buffer and the z-buffer. Each buffer is organized as a 5-pixel word on the screen. The RE2 uses a unique interleaving scheme to achieve the necessary bandwidth from the single chip RE2 into the 5-pixel word. The RE2 interleaves frame buffer accesses so each pixel in the 5-pixel word is rapidly updated in turn. Z-buffer accesses are done in parallel with the frame buffer access to the same address. The RE2 achieves additional bandwidth for screen clears by filling the 5-pixel word on four adjacent lines simultaneously.

The RE2 also supports fast pixel access, which lets the RE2 write a pixel per cycle on the host interface bus into the appropriate frame buffer location. A burst DMA channel between the frame buffer and the Geometry Subsystem data RAM supports fast panning and zooming. For zooming, burst pixels are read from the frame buffer into the Geometry Engine data RAM, replicated by Geometry Engine

microcode, then burst back. Panning is accomplished by reading from a back buffer and bursting the data back to the front buffer at a new address.

Display Subsystem

The Display Subsystem receives pixel information from the frame buffer, routes it through the appropriate display mode, and then sends it to the digital-to-analog converters for display.

Multimode Graphics Processors

Five multiplexed multimode graphics processors (XMAPs) concurrently read the contents of the image and window planes. Information from the window ID planes determines the color mode format that should be used to interpret the data from the image bitplanes. The TG - V graphics displays multiple images simultaneously in an overlapping window environment, in single-buffered RGB, double-buffered RGB, single-buffered color-index, and double-buffered color-index modes. Information from the overlay/underlay planes can override the standard image bitplane value when an overlay or underlay is required.

The XMAPs' flexibility lets you display both static and dynamic images at will, without having to consider your color representation scheme, and allows you to display real-time, smooth-shaded graphics within a complex windowing environment.

Digital-to-Analog Converters

High-speed digital-to-analog converters (DACs) drive the red, green and blue electron guns of the color display. When the Graphics Subsystem operates in RGB color mode, the DACs receive up to 24 bits of color information for each pixel. Eight of these bits are directly assigned to each of the red, green and blue DACs, to yield more than

16 million colors. The DACs multiplex the input from the 5 parallel pixel XMAPs. In color-index mode, the system uses pixel data packets as indices into a 12-bit-in/24-bit-out color map before sending them to the DACs. This map defines 4096 simultaneously visible colors from a palette of 16.7 million. The pixel-mapping feature of the color-index mode lets you quickly modify screen colors by simply changing the values stored in the color maps.

A user-definable graphics cursor is provided via two hardware cursor chips. You can define the cursor glyph pattern through the IRIS Graphics Library. The pattern is directly input to the DACs' overlay planes and overrides the appropriate pixel values input to the DACs. In addition, the RAMDACs used by the graphics subsystem provide gamma correction under application control for the monitor. Gamma correction converts color intensities contained in the 12-bit color index to a range of color intensities that the human eye can discern. Thus, gamma correction gives applications full use of the 12-bit color map.

Display State Machine

In addition to controlling the timing of the interleaved parallel XMAPs, the Display State Machine controls the timing of the SYNC and BLANK signals that structure the scanning pattern of the display device. Three oscillators provide the appropriate timing pulses to the output devices. The first oscillator supports both 60 Hz non interlaced and 30 Hz interlaced 1280 x 1024 display resolutions with a 107.352 MHz clock. The second oscillator provides a 12.272 MHz clock to support RS-170, which produces NTSC resolution video signals. The third oscillator supports the European video standard with a 15 MHz clock and a resolution of 780 x 480.

Appendix A

Component Layout

This appendix illustrates the front panel and all of the circuit boards used in the TG - V graphics subsystem. See Figures A-1 through A-8.

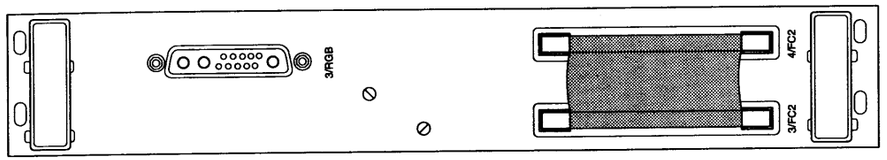


Figure A-1 TG - V Graphics Front Panel

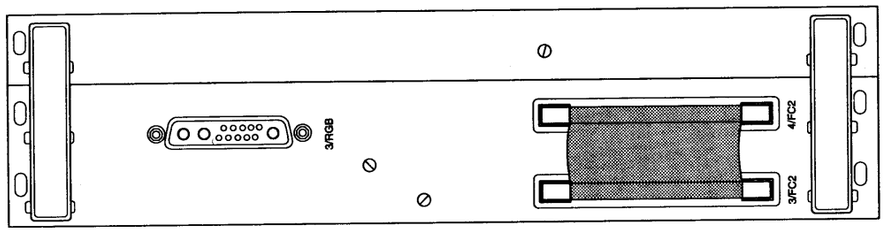


Figure A-2 TG - V Graphics (with Turbo Board) Front Panel

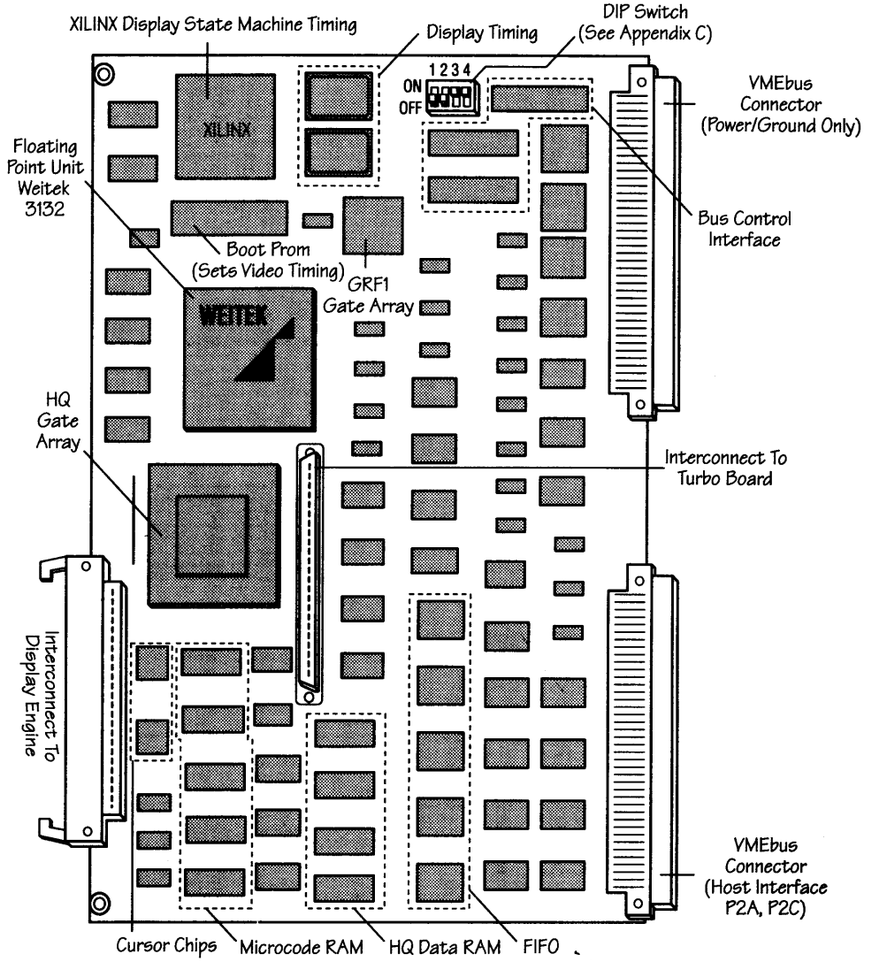


Figure A-3 Geometry Engine Board

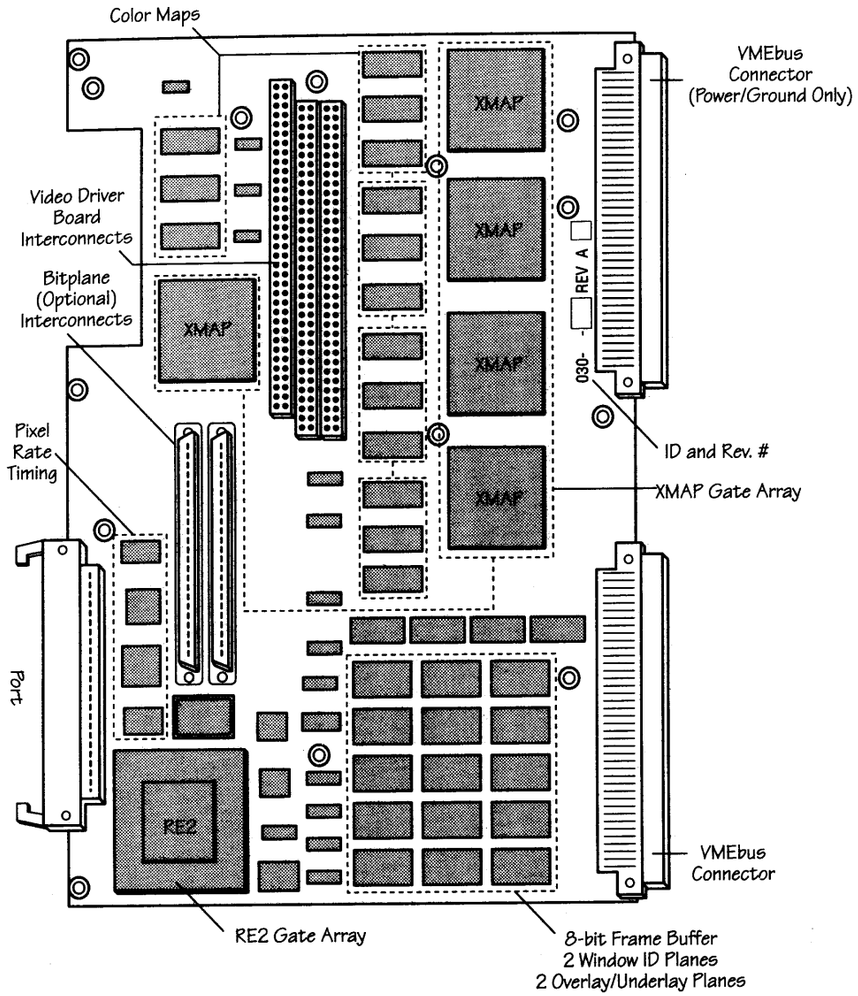


Figure A-4 Display Engine Board

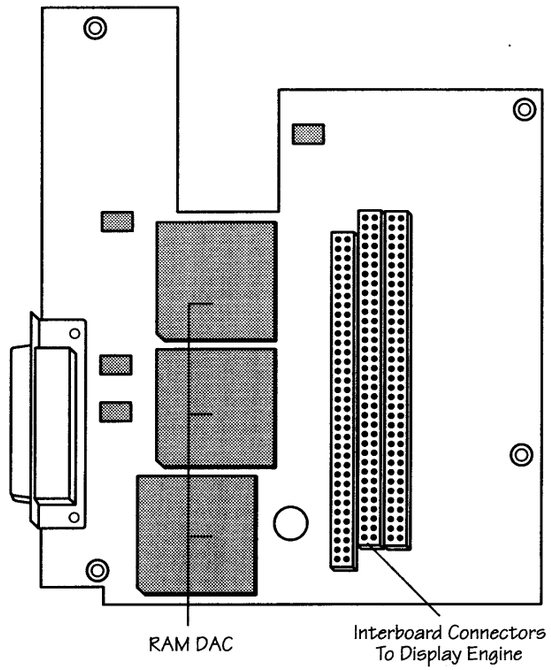


Figure A-5 Video Driver Board

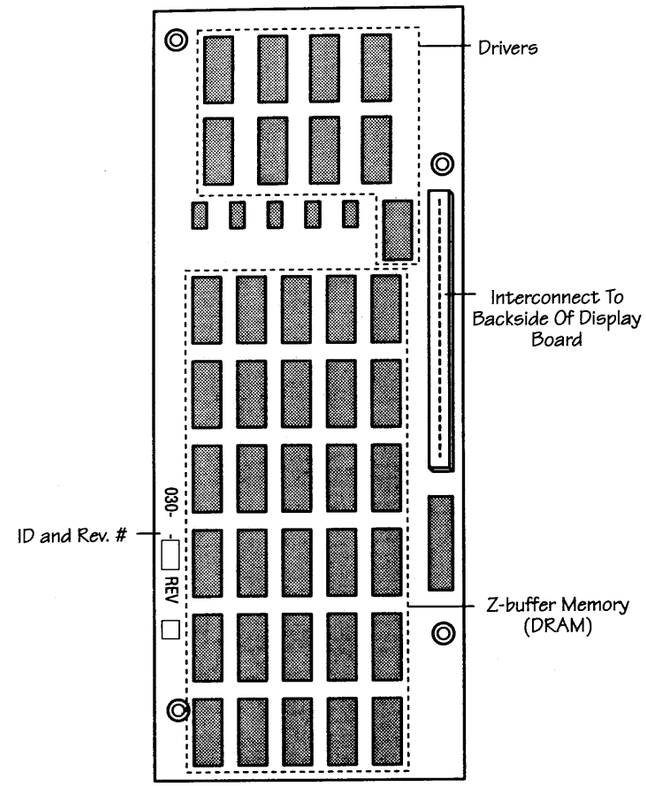


Figure A-6 Z-buffer Board (optional)

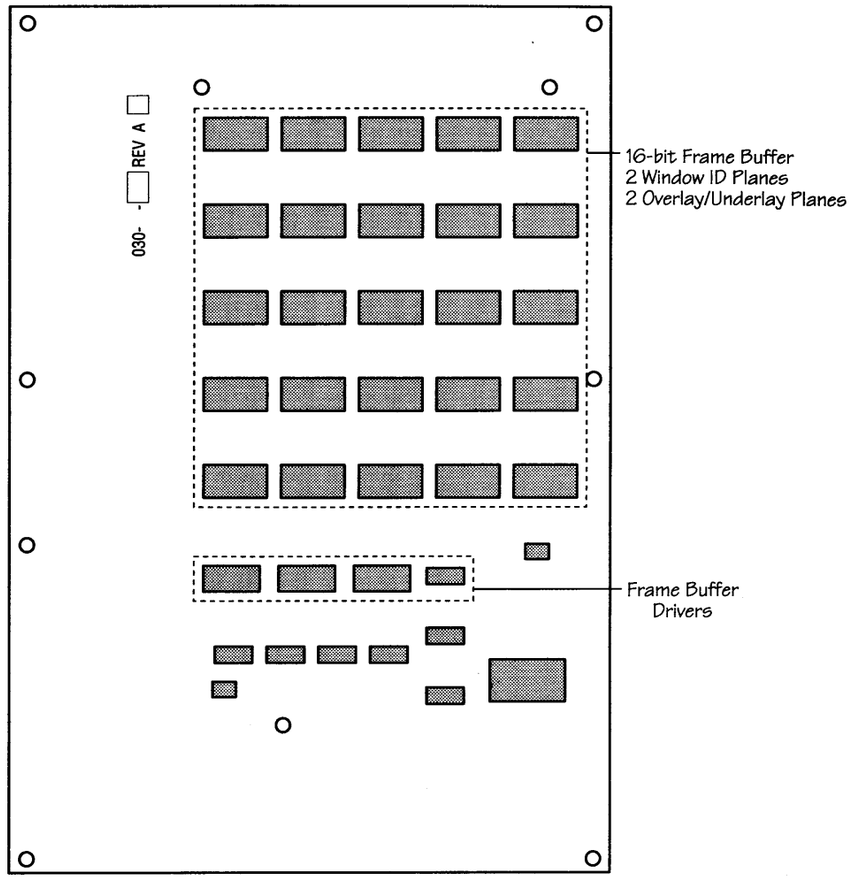


Figure A-7 Bitplane Board (optional)

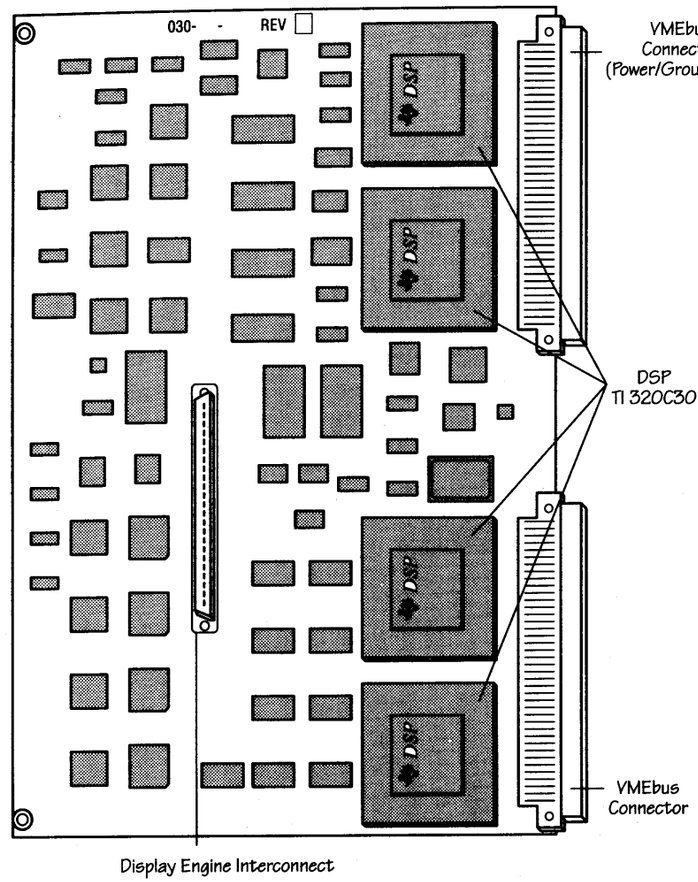


Figure A-8 Turbo Board (optional)

Appendix B

Operating Conditions

The TG - V graphics subsystem is designed to operate in accordance with the standards for any VME-based equipment. Operating temperatures must fall between 0°C to 55°C; for safe storage, between of –40°C to 70°C. Relative humidity during operation must fall within 10% and 95%, noncondensing. Requirements for typical and maximum power levels are shown in Table B-1.

Configuration	Typical	Maximum	Weight
Base TG graphics	5V/4A, +-12V/0A	5V/9A, +-12V/0A	2.2 lbs (1.0kg)
Base TG graphics + turbo + 24-bitplanes + Z-buffer	5V/10A, +-12V/0A	5V/18A, +-12V/0A	3.0 lbs (1.5kg)

Table B-1 TG - V Graphics Power Requirements

Appendix C

DIP Switches

Each TG - V graphics subsystem provides a set of switches to specify the unit number for up to four multiple graphic (multihead) displays. This appendix describes these switches and their associated options.

TG - V Graphics Subsystem Switches

The graphics subsystem uses the S601 bank of switches to set the unit number for multiple graphic displays (multiheads). It may be difficult for you to operate the switches since they are partially obscured by the board above them; you may need to use a probe to switch them. By default, the switches are set for a single display.

The switches are located on the top edge of the geometry board (see Figure A-2 in Appendix A, "Component Layout"). Table C-1 lists the optional settings for multiple graphic displays.

Switch Position				Unit Number
1	2	3	4	
off	off	on	on	0 (the default)
off	off	off	on	1
off	off	on	off	2
off	off	off	off	3
on	on	off	off	Select slot ID via external ID pins: Pins P2 and A31 selects ID 0Pins P2 and C31 selects ID 1

Table C-1 TG - V Graphics DIP Switch Settings

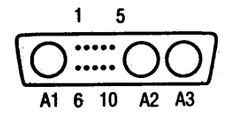


Figure D-1
Video (13W3) Connector Pin Numbering

Appendix D

Pinouts for TG - V Graphics Video Connector

Table D-1 lists pinout data for the TG - V graphics video connector illustrated in Figure D-1.

Connector	Pin	Signal	Pin	Signal
Video 13W3 connector. *See Figure D-1.*	1	NC	2	LOW RES (Logic High)
	3	CSYNC.OUT	4	HDRIVE.OUT
	5	VDRIVE.OUT	6	MONTYPE (Logic High)
	7-10	GND	A1	RED
			A2	GREEN
			A3	BLUE

Table D-1 Pinouts for TG - V Graphics Video Connector

For pinout information on peripheral devices, please refer to your CPU subsystem integrator's guide.

Index