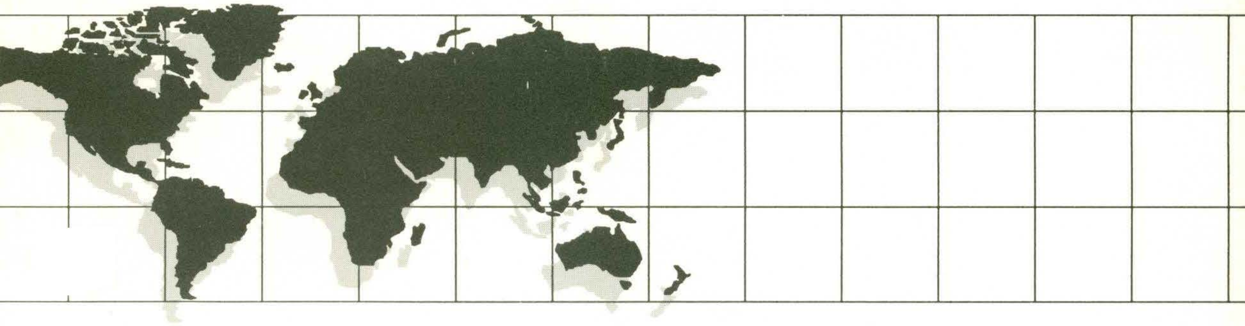


CTOS®

System Administration Guide

UNISYS

UNISYS

CTOS®

System Administration Guide

CTOS III 1.0
CTOS II 3.4
CTOS I 3.4
CTOS/XE 3.4
Standard Software 12.2
System Build Utilities 1.0/3.4
Video Access Method 4.0
Priced Item

August 1992

Printed in USA
4357 4599-200

Page Status

Page	Issue
iii	Original
iv	Blank
v through xv	Original
xvi	Blank
xvii trhough xxviii	Original
1-1 through 1-6	Original
2-1 through 2-12	Original
3-1 through 3-5	Original
3-6	Blank
4-1 through 4-16	Original
5-1 through 5-12	Original
6-1 through 6-16	Original
7-1 through 7-20	Original
8-1 through 8-14	Original
9-1 through 9-28	Original
10-1 through 10-6	Original
11-1 through 11-32	Original
12-1 through 12-6	Original
13-1 through 13-15	Original
13-16	Blank
14-1 through 14-8	Original
15-1 through 15-14	Original
16-1 through 16-37	Original
16-38	Blank
17-1 through 17-29	Original
17-30	Blank
18-1 through 18-12	Original
19-1 through 19-9	Original
19-10	Blank
20-1 through 20-39	Original
20-40	Blank
Glossary-1 through 23	Original
Glossary-24	Blank
Index-1 through 15	Original
Index-16	Blank

Contents

Section 5. Bootstrapping

Section 6. Implementing System Security

Section 10. Accessing Data Throughout the Cluster

Section 11. Adding Hard Disks

Section 15. Optimizing System Performance

Section 16. Configuring Workstation Operating Systems

Figures

Tables

About This Guide

Who This Manual Is For

This manual is for system administrators who are responsible for setting up and maintaining CTOS®-based clusters. You may be an experienced full-time system administrator, or you may administer a cluster in addition to other job duties. In either case, this manual is written with the following assumptions about your skills and knowledge:

- You are familiar with basic workstation operations, such as signing on and using the Executive.
- You understand file system concepts, such as volumes, directories, and *!Sys*.

If you need to learn about basic workstation operations or the file system, see the *CTOS Executive User's Guide* before you continue with this manual.

What This Manual Covers

This manual describes the tasks performed by system administrators and provides detailed procedures for performing those tasks. It also provides an overview of workstation and shared resource processor hardware and software components and introduces the software tools available to system administrators. In addition, it describes troubleshooting techniques for both workstations and shared resource processors.

Changes to This Edition

This edition of the *CTOS System Administration Guide* is relative to CTOS I 3.4, CTOS II 3.4, CTOS III 1.0, CTOS/XE 3.4, Standard Software 12.2, and the System Build Utilities 1.0/3.4. See "What Is New in System Software," below, for specific information about changes to the system software products.

Changes in Terminology

The following terms have been added or changed since the previous edition, and are used throughout this manual.

X-Bus workstation	Replaces the term *modular workstation*, which was formerly used to describe B26, B27, B28, and B38 workstations. Also replaces the term *integrated workstation*, which formerly described the B39 workstation.
X-Bus+ workstation	Replaces the terms *SuperGen* and *Series 5000*, which were formerly used to describe the SuperGen Series 5000 workstation.
EISA/ISA-bus workstation	Describes the SuperGen Series 3000 workstation.

Other new terms are defined in the sections to which they apply.

What This Manual Does Not Cover

This manual does not address specific hardware issues, such as installation of workstations, SRPs™, or cluster cabling. See the appropriate installation guides for information about your hardware products. Cluster hardware installation is described in the *CTOS Cluster and Network Hardware Installation Guide*.

See the *CTOS System Software Installation Planning Guide* and the Software Release Announcements for information about installing the operating system and Standard Software.

What Is New in System Software

The following features and enhancements have been added to this edition:

- The new EISA/ISA-bus workstation is briefly described in Section 2, "Understanding Hardware."
- The CTOS III virtual-memory operating system is described in Section 3, "Understanding System Software."

- Demand paging and changes to the **Partition Status** command are documented in Section 9, "Installing System Services."
- New format templates for SuperGen workstations (X-Bus + and EISA/ISA) are listed in Section 11, "Adding Hard Disks."
- Optimization of virtual memory, as it applies to Context Manager, is described in Section 15, "Optimizing System Performance."
- Many new *Config.sys* parameters are documented in Section 16, "Configuring Workstation Operating Systems."
- Changes to the system build procedure have been added to Section 18, "Building a Customized Operating System."
- Configuration of the SignOn screen is described in Section 19, "Customizing Standard Software."
- A command for creating bootable floppy diskettes is documented in Section 20, "Troubleshooting."
- The crash dump procedure and its requirements for EISA/ISA-bus workstations is described in Section 20, "Troubleshooting."

See the *CTOS Executive Reference Manual* for a more detailed list of changes to individual Standard Software commands.

How This Manual Is Organized

This manual is organized as follows:

Section 1. If You Are New to System Administration

This section provides an overview of administrative tasks for those who are new to system administration.

Section 2. Understanding Hardware

This section presents an overview of workstation and shared resource processor (SRP) hardware.

Section 3. Understanding System Software

This section presents an overview of the operating systems and Standard Software utilities.

Section 4. Using Administrative Tools

This section describes the software commands and applications you use for system administration.

Section 5. Bootstrapping

This section describes the bootstrap and system initialization sequences of workstations and SRPs.

Section 6. Implementing System Security

This section describes how to password-protect your system and prevent unauthorized persons from using it.

Section 7. Customizing User Environments

This section describes how to define the commands and applications available to users.

Section 8. Installing Applications

This section describes how to install software products from floppy diskettes or tapes.

Section 9. Installing System Services

This section describes how to install loadable system services, which supplement the operating system by providing access to additional resources, such as tape drives, modems, and printers.

Section 10. Accessing Data Throughout the Cluster

This section describes how to install and use the Cluster File Access facility, which allows files to be shared from cluster workstation to cluster workstation.

Section 11. Adding Hard Disks

This section describes how to format disks for use on workstations and shared resource processors.

Section 12. Using Tape Drives

This section describes the media and system services required to use quarter-inch cartridge (QIC), digital data storage (DDS), and half-inch tape drives.

Section 13. Backing Up and Restoring Data

This section describes how to back up and restore disks to and from tape archive media.

Section 14. Using a File System Cache

This section describes how to configure a file system cache, which increases the speed at which files are accessed.

Section 15. Optimizing System Performance

This section describes how to improve system performance by making optimal use of memory, disk space, and other system resources.

Section 16. Configuring Workstation Operating Systems

This section describes how to configure workstation operating systems to function optimally in your environment.

Section 17. Configuring Shared Resource Processor Operating Systems

This section describes how to configure SRP operating systems to function optimally in your environment.

Section 18. Building a Customized Operating System

This section describes how to further configure an operating system by building a customized version.

Section 19. Customizing Standard Software

This section describes how to customize screen messages for SignOn and the Standard Software utilities.

Section 20. Troubleshooting

This section describes troubleshooting tips and techniques.

In addition, a glossary and index are included near the end of the manual.

Conventions

The following conventions are used throughout this manual:

- New terms appear in *italics* and are defined in the glossary.
- Command names are capitalized and appear in bold type, for example, **Format Disk** and **Volume Status**.
- Names of fields appear in *italics*, for example, *[Password]* field.
- Variable information also appears in *italics*, for example,

  ...Ws*NNN*, where *NNN* stands for a three-digit number.
- Names of keys appear in small uppercase bold type, for example, **GO**.
- Volume, directory, and file names appear in *italics*, for example, *[Sys]<Sys>Config.sys*.

Where to Find More Information

The following manuals are included in the documentation set for Standard Software:

CTOS Basic Asynchronous Terminal Emulator User's Guide

This user's guide contains detailed information about the Basic Asynchronous Terminal Emulator application.

CTOS Batch Manager II Installation, Configuration, and Programming Guide

This manual describes the **Batch Foreground** command that is packaged with the Standard Software utilities. In addition, it describes the Background Batch function of Batch Manager II, which is packaged and installed separately from the Standard Software utilities.

CTOS Editor User's Guide

This user's guide contains detailed information about the Editor application, which is frequently used by system administrators for editing configuration files.

CTOS Executive Reference Manual

This manual documents the Executive command interpreter, which is a primary tool for system administration. It describes Executive commands packaged with Standard Software and is arranged alphabetically by command name. Keep it handy as a companion volume to the this manual.

CTOS Executive User's Guide

This user's guide provides step-by-step procedures for the most commonly used commands and features of the Executive. It is a good resource for people who are new to CTOS or are occasional users of the Executive.

CTOS Status Codes Reference Manual

This manual provides descriptions for CTOS status codes. It is organized numerically by status code number.

CTOS System Software Installation Planning Guide

This manual assists with the software installation process of CTOS I, CTOS II, CTOS III, CTOS/XE, Standard Software, and the Video Access Method (VAM).

Hardware Installation Guides

See the following manuals for information about workstation hardware configurations:

CTOS Media User's Guide

CTOS Workstations Planning and Owner's Maintenance Guide

SuperGen Series 5000 Installation and Maintenance Guide

CTOS Cluster and Network Hardware Installation Guide

CTOS XE-530 Shared Resource Processor Hardware Installation Guide

Administration Guides for Applications

See the following manuals for information about Context Manager, printing, networking, and electronic mail products:

CTOS Generic Print System™ Administration Guide

CTOS Context/Window Manager Installation and Configuration Guide

CTOS BNet II Installation, Configuration, and Administration Guide

CTOS OFIS™ Mail Administration Guide

Software Development Manuals

See the following manuals for information about CTOS software development products:

CTOS Debugger User's Guide

CTOS Operating System Concepts Manual

CTOS Procedural Interface Reference Manual

CTOS Programming Utilities Reference Manual: Assembler

CTOS Programming Utilities Reference Manual: Building Applications

CTOS Programming Utilities Reference Manual: Customization

Section 1
If You Are New to System Administration

Overview

If you are new to system administration or have never worked with CTOS, you may not be familiar with certain terminology or the various tasks that a system administrator performs.

This section defines some terms and briefly describes a number of administrative tasks. After reading this section, you should have a good idea of what will be required of you as a system administrator.

The sections that follow expand on the terms and concepts presented here and provide detailed procedures for performing specific tasks.

New Terms

The terms defined below are used throughout this manual and in most of the other documentation you will use with CTOS-based systems. These terms, and many others, are described in more detail in later sections.

Application	A program you interact with on your workstation. This term usually refers to a multifunctional program, such as a word processing or accounting package.
Cluster	A group of computers connected together so that they can share resources, such as files, printers, or databases.
Configuration	An arrangement of parts, such as computer hardware, or of elements, such as software programs, or a combination of both. (See also "What Does 'Configuration' Mean?," below).

CTOS	The overall name for the family of workstation and shared resource processor operating systems.
CTOS I	The real-mode workstation operating systems.
CTOS II	The protected-mode workstation operating systems without virtual memory.
CTOS III	The protected-mode workstation operating systems with virtual memory.
CTOS/XE	The shared resource processor operating systems.
Operating system	The program that controls data input and output and execution of application programs on a computer.
Server	One computer within a cluster that controls resources that are shared throughout the cluster.
Shared resource processor (SRP)	A floor-model computer that always functions as a server (also called an *XE*).
Standard Software	A set of commands and applications you use to perform basic tasks, such as copying files or backing up a disk.
System services	Optional programs that expand the capabilities of the operating system.
User file	A configuration file that defines the working environment for a specified user name.
Workstation	A desktop computer that may be configured as a server, a cluster, or a standalone workstation.

What Does "Configuration" Mean?

The word *configuration* is a commonly used term in system administration. It is used in a variety of circumstances, as described below:

Hardware configuration	The hardware devices that are present on a workstation or SRP.
Cluster configuration	The combination of different systems in a cluster.
Software configuration	The combination of the operating system and applications installed on a system.
Configuration file	A file that contains parameters for a software product or a hardware device.

CTOS systems are very configurable. That means that each individual workstation or SRP can be set up many different ways. Configurability allows systems within the cluster to meet the needs of individual users.

Administrative Duties

The actual duties you perform as a system administrator depend a great deal on the size of your cluster. In general, you are responsible for the server; sometimes you are also responsible for other workstations in the cluster. For example, if you are administering a small cluster in addition to your other job duties, it is likely that other users in the cluster administer their own workstations. On the other hand, if you are administering one or more large clusters as a full-time job, most likely you will be responsible for all aspects of system administration.

The tasks described below are typical administrative duties. In addition, many system administrators are also responsible for hardware maintenance. See the appropriate installation guides and technical reference manuals for help with your hardware requirements.

Setting Up a System

The following list briefly describes the tasks associated with setting up a workstation or SRP. See the cross-referenced sections and manuals for more detailed information.

1. Assemble the hardware.

   Setting up workstation hardware is a simple task; the only tool you need is a small, flathead screwdriver for attaching the monitor cable to the workstation. See the installation guide that accompanies the workstation for detailed instructions. SRPs are usually installed by a customer service engineer; for detailed information, however, see the *XE-530 Shared Resource Processor Hardware Installation Guide*.

2. Install system software.

   Before a workstation or SRP will function as a computer, you must install system software, which consists of the operating system and Standard Software. See the *CTOS System Software Installation Planning Guide* and the Software Release Announcements for the operating systems and Standard Software.

3. Configure the operating system.

   On a shared resource processor, you may need to configure the operating system to recognize all hardware components on the system. See Section 17, "Configuring Shared Resource Processor Operating Systems."

4. Connect workstations to the server.

   To form a cluster, workstations are connected to the server. See the *CTOS Cluster and Network Hardware Installation Guide* for detailed information about connecting the systems in a cluster.

5. Install applications.

   After you install system software, you install applications. See Section 8, "Installing Applications," for general information. See the release documentation or the software installation guide for each application for specific instructions.

6. Install system services.

   In many cases, system services are required to run applications. See Section 9, "Installing System Services," and the release documentation for each application.

7. Configure applications.

   Some applications have configuration files that affect the way they work. See the release documentation and the manual for the application for more information.

Other Administrative Tasks

In addition to setting up workstations, system administrators frequently are responsible for the following:

Assigning passwords	As system administrator, you are responsible for password-protecting the server.
Creating directories	In some cases, particularly if users share disk space on the server, you are responsible for creating directories.
Creating user files	You create user files to customize the working environment for each user.
Adding disks	In many cases, you are responsible for planning the use of disk space and adding disks when necessary.
Updating software	You install new software applications and update existing ones when new versions are released.
Configuring hardware and software	Whenever you add hardware or software, check the release documentation and application manuals for specific configuration requirements.

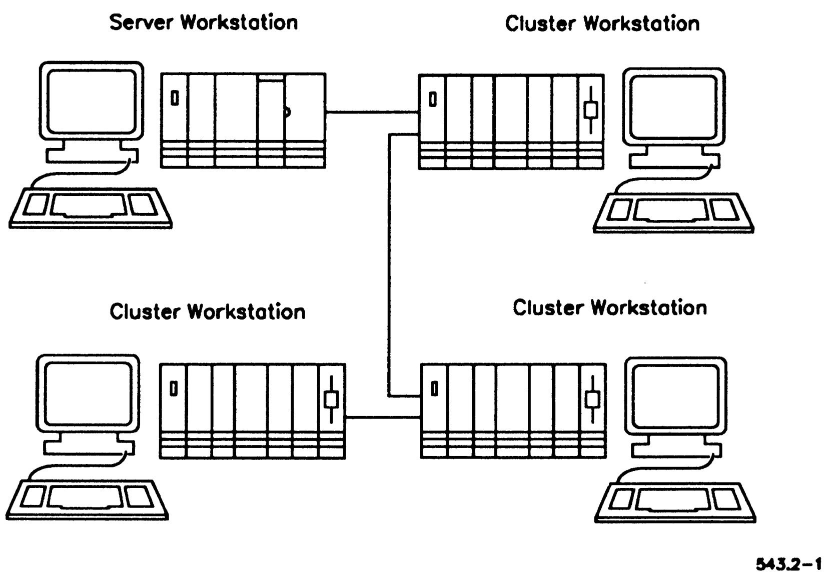
Performing backups	You are usually responsible for backing up disks on the server, and in some cases, all disks in the cluster. You do this periodically (in many cases, daily) to make duplicate copies of all data stored on the system.
Monitoring the system	As you become more experienced, you might want to keep track of system activity, for example, the number of users who are signed on at certain times or how quickly disks fill up. This helps you tailor the system to the needs of your workplace.
Correcting problems	You are responsible for diagnosing and solving problems. Troubleshooting is one of the most challenging aspects of system administration.

Section 2
Understanding Hardware

What Is a Cluster?

A *cluster* is a group of computers that are connected together so that they can share disks, printers, and other system resources. A single computer within the cluster functions as a *server*, which controls resources for the entire cluster. The server and cluster workstations are connected either with cables or with TeleCluster™, which connects them via telephone lines. Figure 2-1 shows a simple cluster configuration.

Figure 2-1. Simple Cluster Hardware Configuration



The server can be either a workstation, as shown in Figure 2-1, or a larger computer called a shared resource processor (SRP). This section provides an overview of both workstation and SRP hardware components, which are referred to throughout this manual. See the appropriate hardware installation guides for more specific information. For details about setting up and connecting systems to a cluster, see the *CTOS Cluster and Network Hardware Installation Guide.*

Workstation Hardware

A workstation is a desktop unit that functions as a server, a clustered workstation, or a standalone computer. A workstation consists of a central processing unit (CPU), a keyboard, and a monitor. Optional modules or expansion cards can be attached to most workstations to provide local disk storage space, graphics, and other enhancements.

Although there are many distinct workstation models, they can be grouped into the following categories. For more specific information, see the installation guide for your particular workstation.

X-Bus workstation

An X-Bus workstation is a collection of separately housed modules, each containing an individual hardware component, such as a processor, a disk, or a graphics controller. X-Bus workstation processors include the B26, B28, and B38 models.

A variation consists of the integrated X-Bus workstation, which contains the processor, a hard disk, and a floppy disk drive in the base unit, and can be expanded with X-Bus modules. The B39 model is an integrated X-Bus workstation.

X-Bus+ workstation

An X-Bus+ workstation consists of a base unit containing the CPU and removable cartridges, which house disk drives, graphics controllers, tape drives, and other optional equipment. In addition, an X-Bus+ workstation can be equipped with an adapter to accept X-Bus SCSI disk modules. The SuperGen Series 5000 is an X-Bus+ workstation.

EISA/ISA-bus workstation
An EISA/ISA-bus workstation consists of a base unit containing a CPU and other hardware components such as a hard disk, a floppy disk drive, and a video controller. It can be upgraded with "industry standard" hardware components. The SuperGen Series 3000 is an EISA/ISA-bus workstation.

Diskless workstation
A diskless workstation consists of a base unit containing the CPU and video controller. It uses disks on the server for data storage; disks are not connected directly to it. Some diskless workstations can be upgraded with expansion cards for graphics or extra memory. Workstations that are available only as diskless models include the B26-LCW, B28-LCW, CWS, and SuperGen Series 2000.

Processors

The latest model workstations are equipped with Intel® 80286, 80386, or 80486 central processing units (CPUs). Older workstations contain Intel 80186 CPUs. For most system administration purposes, 80286, 80386, and 80486 CPUs are grouped together as *protected-mode* processors and 80186 CPUs are called *real-mode* processors. Those terms, however, actually describe the operating system that is running on the workstation. See Section 3, "Understanding System Software," for information about real-mode and protected-mode operating systems.

Cartridges, Modules, and Expansion Cards

Many cartridges, modules, and other expansion components are available for workstations. The following list describes only a few. See your installation guide for information about installing modules and expansion cards.

Disks	Various capacity hard disks, as well as 5-1/4 and 3-1/2 inch floppy drives, are available for most workstation models.
Video controllers	A video controller other than the one that is included with the workstation processor may be required to use certain high-resolution monitors.
Tape drives	Quarter-inch cartridge (QIC) and digital data storage (DDS) tape drives are available for X-Bus and X-Bus+ workstations.
CD-ROM drives	Compact disc (CD) drives are available for X-Bus and X-Bus+ plus workstations.

Monitors

Major categories of monitors are listed below:

- Green-on-black monochrome monitors
- Color graphics monitors
- High-resolution monitors
- Video graphics array (VGA) monitors

Not all monitors can be used with every workstation model, and in many cases, an additional video controller is required. See the workstation installation guide to find out about monitor and graphics controller compatibility.

Keyboards

Several keyboard models are available for workstations. Although the position of some keys may vary, most are labeled with CTOS key names, which are used consistently throughout this manual.

Shared Resource Processor Hardware

The shared resource processor (SRP) is a floor-model computer containing multiple processors that perform different functions. For example, some processors control disk operations, while others control network communications. The processors constantly pass data back and forth and are dependent on one another to function as a system. Many processor combinations are available to provide the optimal hardware configuration for different applications.

An SRP always functions as a server. It does not have a keyboard and monitor; therefore, a special utility, called Cluster View, is required to communicate with SRP processors. (Cluster View is described in Section 4, "Using Administrative Tools.")

Hardware components of the SRP are briefly described below. For more detailed information, see the installation guide for your particular model.

Cabinets

SRP hardware components are housed in cabinets (sometimes called enclosures). An SRP consists of one primary cabinet and up to five secondary cabinets. A primary cabinet, as pictured in Figure 2-2, is equipped with a keyswitch, a reset button, a two-numeral status display, and a QIC tape drive. Both primary and secondary cabinets contain processor boards and disk drives. The various cabinet models are summarized in Table 2-1.

Figure 2-2. SRP Primary Cabinet

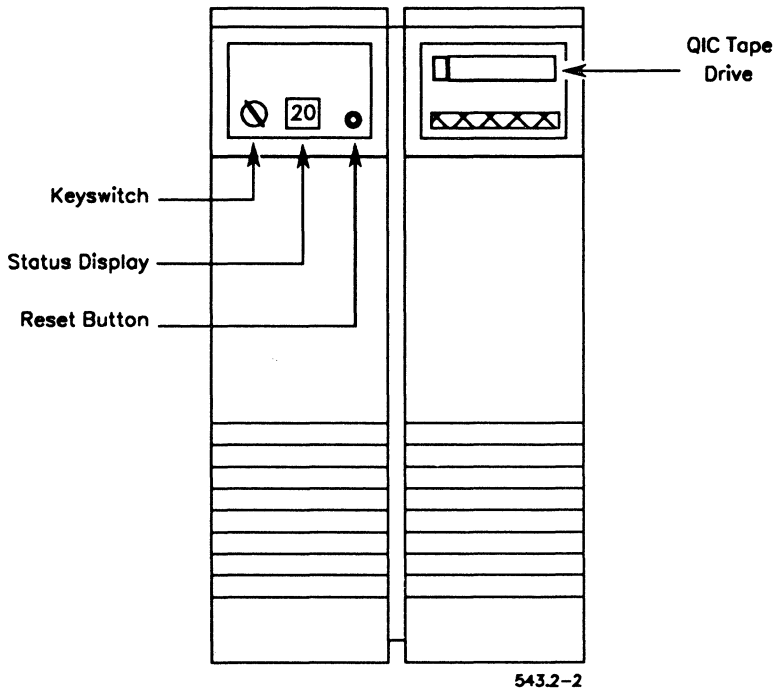


Table 2-1. SRP Cabinets

Name	Type	Description
C-Box	Primary cabinet	Contains a QIC tape drive, three 5-1/4 inch hard disk drive slots, and six processor board slots.
E-Box	Primary cabinet	Contains two 8-inch hard disk drive slots, six processor board slots, and optionally, a QIC tape drive.
B-Box	Expansion cabinet	Contains four 5-1/4 inch hard disk drive slots and six processor board slots.
X-Box	Expansion cabinet	Accommodates ten 5-1/4 or three 8-inch hard disk drives.

Processor Boards

The latest model SRP processor boards are equipped with Intel 80386 CPUs. Older processor boards contain Intel 80186 CPUs. In most cases, the terms *real mode* and *protected mode* are used throughout this manual to differentiate between the 80186 and 80386 SRP processors, respectively. See also Section 3, “Understanding System Software,” for information about real-mode and protected-mode operating systems.

Although newly installed SRPs are equipped with protected-mode processors only, upgraded systems may contain a combination of real-mode and protected-mode processor boards. See the *XE-530 Shared Resource Processor Hardware Installation Guide* for information about the compatibility of different processor models.

To identify SRP processor boards, look at the acronyms stamped on the back, which are visible when you open the rear cabinet doors. These acronyms are used throughout this manual. The different processor boards are summarized in Table 2-2. Your SRP may not contain every type of processor.

Table 2-2. SRP Processors

Name	Acronym	Mode	Description
General Processor	GP	Protected	The GP contains two RS-485 cluster channels (four ports), which can support up to 32 workstations (16 per channel), two RS-232-C ports, and one parallel printer port.
General Processor with SCSI Interface	GP	Protected	The GP+SI consists of a General Processor (see above) and a SCSI Interface board (*SCSI* stands for Small Computer Standard Interface). The expansion board contains two SCSI device controllers, which can support up to eight SCSI devices.
General Processor with Communi-cations Interface	GP	Protected	The GP+CI consists of a General Processor (see above) and a Communications Interface board. The expansion board contains six RS-232-C ports, two of which can be configured as either V.35 or X.21 ports.
Cluster Processor	CP	Real	The CP contains two RS-422 cluster channels (four ports), which can support up to 16 workstations (8 per channel); two RS-232-C ports; and one parallel printer port.
File Processor	FP	Real	The FP contains four disk inter-face ports, which can support up to four ST-506 hard disk drives.

continued

Table 2-2. SRP Processor Summary (cont.)

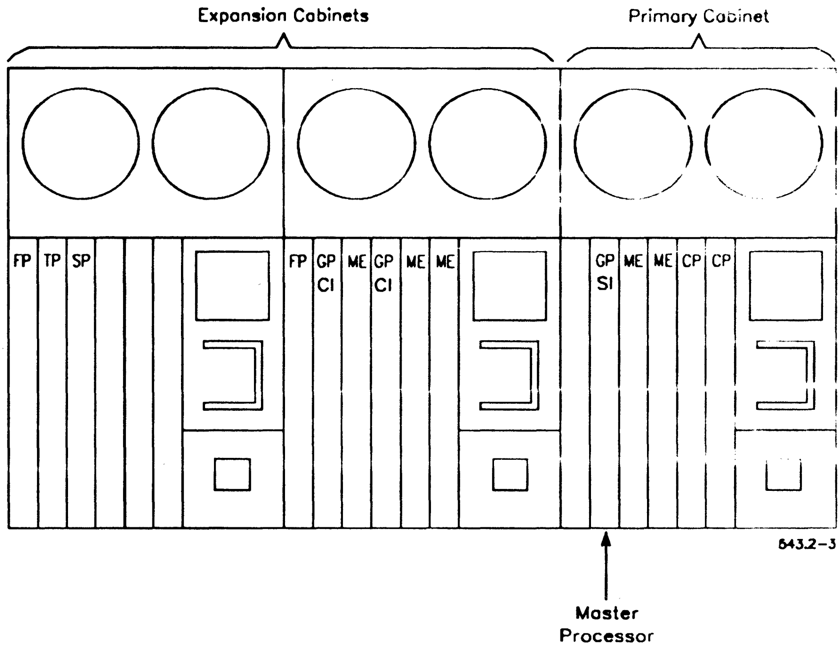
Name	Acronym	Mode	Description
Data Processor	DP	Real	The DP consists of a Storage Processor (see below) with a Storage Controller board (SC) in the adjacent slot. A DP controls up to six external SMD disk drives.
Storage Processor	SP	Real	The SP contains a tape interface that supports up to four external 9-track half-inch tape drives.
Terminal Processor	TP	Real	The TP contains ten RS-232-C channels and one parallel printer port.

Master Processor

All SRPs contain a *master processor*, which bootstraps itself first and then controls booting of the other processors. The master processor is the first processor in the primary cabinet. It must be a disk-controlling processor, that is, a General Processor with SCSI Interface, a File Processor, or a Data Processor, as described in Table 2-2.

Figure 2-3 shows the location of the master processor within a three-cabinet SRP.

Figure 2-3. SRP Master Processor



Processor Identifiers

A unique identifier (ID) is assigned to each processor. It consists of the two-letter acronym for the processor, as listed in Table 2-2 and a two-digit number identifying the processor's position within the SRP, for example, *GP00* or *CP01*.

Note: *The two-letter acronym for all General Processors, with or without a SCSI Interface or a Communications Interface, is* GP.

Processor IDs are used when configuring software; you will also encounter them when using the System Manager command or reading the system error log. Processor numbering schemes are described below.

Real-Mode Processor Numbers

Each type of real-mode processor has a unique acronym (see Table 2-2). Therefore, processors of the same type are numbered independently of other processor types. When viewing an SRP from the back, as shown in Figure 2-4, processor numbering begins from the left of the primary cabinet. For example, the first File Processor is identified as FP00, while the second is FP01; the first Cluster Processor is CP00, while the second and third are CP01, CP02, and so on. Note in Figure 2-4 that sequentially numbered processors are not necessarily adjacent to one another (for example, FP00 and FP01).

Protected-Mode Processor Numbers

All protected-mode processors are identified by the acronym GP (see Table 2-2). Therefore, protected-mode processors are numbered sequentially, regardless of whether they contain a SCSI or Communications Interface. For example, in Figure 2-4, the first GP, which has a SCSI Interface, is GP00; the second GP, which has a Communications Interface, is GP01, and so on.

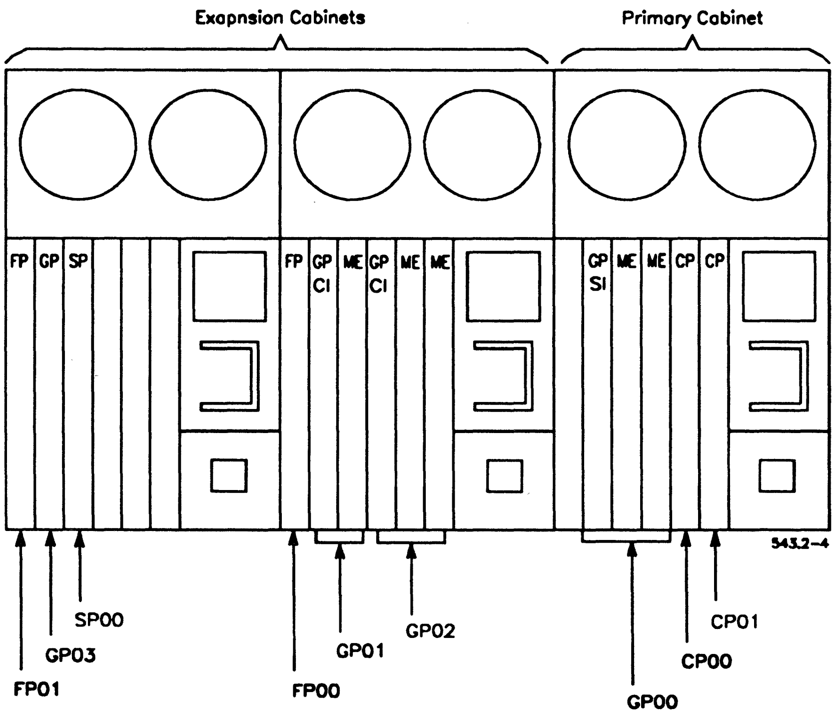
Memory Expansion Boards

Memory expansion boards (ME), as shown in Figure 2-4, are available for protected-mode processors and some real-mode processors. They expand the amount of RAM available to the processor.

Protected-mode processors can be expanded with two memory expansion boards for a total of 64M bytes of memory. See the *XE-530 Shared Resource Processor Hardware Installation Guide* for details.

Real-mode processors containing 256K bytes of memory can be expanded with one memory expansion board for a total of 768K bytes of memory. Memory expansions cannot be added to real-mode processors containing 786K bytes of memory.

Figure 2-4. SRP Processor Identifiers



Section 3
Understanding System Software

What Is System Software?

System software includes the operating system, system services, and configuration files that are required for a workstation or shared resource processor to function as a computer. This section describes the components of system software as they are installed on workstations and SRPs. See the *CTOS System Software Installation Planning Guide* and the Software Release Announcements for installation instructions.

Standard Software

Standard Software is a group of applications, utilities, and configuration files. It is distributed on floppy diskettes or QIC tape for installation on both workstations and SRPs. Many aspects of Standard Software are described in this manual. For detailed information about individual commands, however, see the *CTOS Executive Reference Manual.*

Video Access Method

The Video Access Method (VAM) software contains video services that may be required in addition to the operating system. Because a number of different video modes are supported, including bit-mapped and video graphics array (VGA), those video services are packaged and installed separately from the operating system.

Workstation Operating Systems

Workstation operating systems are divided into the following major categories:

CTOS I *(real mode)*	CTOS I is typically used on workstations containing 80186 CPUs. It can also be used, however, on certain models of 80286- and 80386-based workstations.
CTOS II *(protected mode)*	CTOS II is used on workstations containing 80286, 80386, or 80486 CPUs. It takes advantage of the faster processing speed and extended memory of those workstations.
CTOS III *(virtual memory)*	CTOS III is used on workstations containing 80386 or 80486 CPUs. It allows multiple programs to be executed simultaneously in less memory than is required on CTOS II systems.

Virtual memory operating systems are also protected-mode operating systems. Therefore, when used throughout this manual, the term *protected mode* includes virtual memory systems, unless otherwise specified.

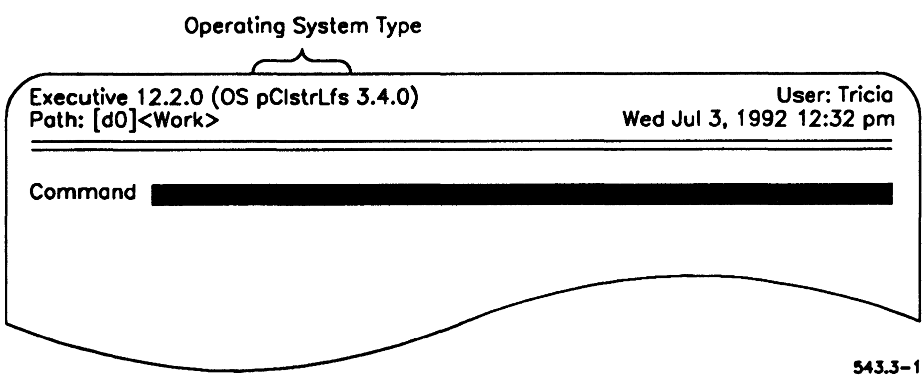
Beyond the main categories described above, the operating systems are divided into types, which determine whether a workstation functions as a server or as a cluster workstation. Workstation operating system types are listed in Table 3-1 and described later in this section.

After system software has been installed and a workstation is up and running, the operating system type is displayed on the Executive screen, as shown in Figure 3-1.

Table 3-1. Workstation Operating Systems

	Operating System Type		
Workstation Characteristic	**CTOS I** (Real Mode)	**CTOS II** (Protected Mode)	**CTOS III** (Virtual Memory)
Server	t1Srvr	pSrvr	vSrvr
Cluster with disks	t1ClstrLfs	pClstrLfs	vClstrLfs
Diskless cluster	t1Clstr	pClstr	vClstr
Cluster with disks (B27)	bAwsClstrLfs	NA	NA
Diskless cluster (B27)	bAwsClstr	NA	NA
Diskless cluster (B24)	v1Clstr	NA	NA

Figure 3-1. Operating System Identification



543.3-1

Server Workstations

The server operating system supports communications between itself and other workstations in the cluster. Any X-Bus or X-Bus+ workstation with a hard disk can function as the server when the appropriate server operating system is installed.

When possible, a protected-mode processor is the best choice for a workstation server. The server controls access to many cluster-wide system services, which are described in Section 9, "Installing System Services." These can consume a great deal of memory, so the extended memory capabilities and faster processing speed of 80286, 80386, or 80486 hardware, in combination with a protected-mode or virtual memory operating system, are often essential.

Cluster Workstations With Local File Systems

The term *local file system* (LFS) refers to a cluster workstation with a disk. Most LFS workstations are equipped with both a hard disk and a floppy disk drive; however, a workstation with a floppy disk drive only is considered to have a local file system. To function as an LFS, the workstation must be running the appropriate operating system, as shown in Table 3-1.

Local file systems can be used in different ways. For example, an LFS workstation that bootstraps from its own disk often functions as an independent system. If all applications and working files are stored locally, it is dependent on the server only for cluster-wide system services. Such a system usually remains operational when the server is not functioning.

Other workstations with local file systems may be used for storing working files only, while the operating system and application software reside on the server. In this case, applications are "downloaded" from a disk on the server to memory on the LFS; data files, however, are stored on a local disk.

Diskless Cluster Workstations

A diskless workstation does not contain a hard disk or a floppy drive. It boots from a disk on the server. Diskless workstations use a different operating system type than LFS workstations, as shown in Table 3-1.

SRP Operating Systems

The SRP uses a set of operating systems to support its various processors. All types of protected-mode processors run the same protected-mode operating system. Each type of real-mode processor, however, runs a slightly different real-mode operating system. The operating systems constantly communicate with one another to provide an integrated system. The set of SRP operating systems is named CTOS/XE.

Table 3-2 lists the prebuilt operating systems for SRP processor boards.

Table 3-2. SRP Operating Systems

Processor	Operating System	Mode
General Processor (including GP+SI and GP+CI)	pSrpGp.img	Protected
File Processor	rSrpFp.run	Real
Cluster Processor	rSrpCp.run	Real
Terminal Processor	rSrpTp.run	Real
Data Processor	rSrpDp.run	Real
Storage Processor	rSrpSp.run	Real

Section 4
Using Administrative Tools

What Tools Are Available?

As a system administrator, you routinely use certain software utilities to monitor the status of the cluster, perform troubleshooting, and update system configuration files. The following tools are commonly used by system administrators to perform daily tasks:

- The System Manager provides status information and access to frequently used commands.
- The Editor is used to customize configuration files for system software and other applications.
- Cluster View allows you to issue commands from a cluster workstation keyboard to a processor on the server.

An overview of these tools is presented in this section. More detailed instructions for using them to perform specific tasks are provided later in this manual.

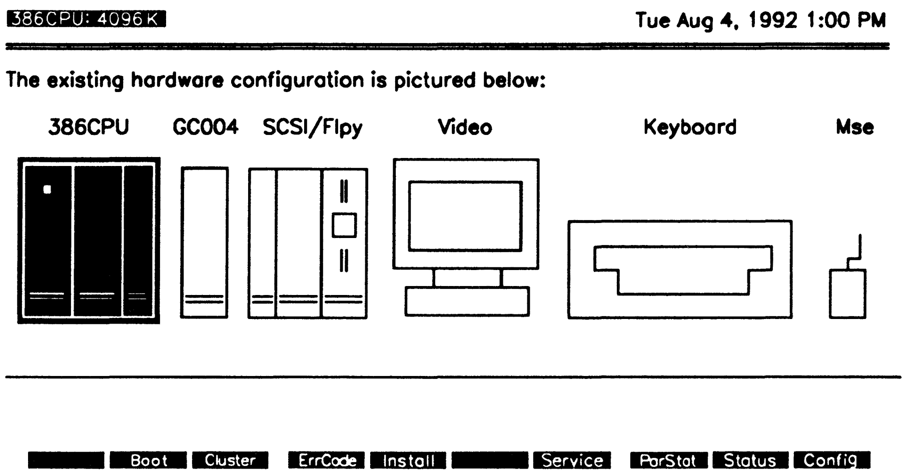
System Manager

The System Manager is an administrative control center. It combines frequently used Executive commands into a single interface that you can use to perform the following types of tasks:

- Back up disks
- Format disks
- Edit command files
- Create and modify configuration files
- Monitor memory, disk space, cluster, and network activity

Figure 4-1 shows a sample System Manager display for a workstation.

Figure 4-1. System Manager Display



543.4-1

Starting the System Manager

To start the System Manager, follow these steps:

1. On the Executive command line, type **System Manager**.
2. Press **GO**.

The System Manager display is divided into the following sections:

Status area	The status area is located at the top of the screen, above the double bar. It displays information about the selected module and the system date and time.
Hardware components	Workstation modules and other hardware components are displayed and labeled in the center of the screen. The selected module is highlighted.

Function key menu	A function key menu is displayed at the bottom of the screen. You use function keys to invoke commands that pertain to the highlighted module. Function key names change as you select different modules.

Using the System Manager

To use the System Manager, you select a hardware component and a function key for the task you want to perform. You can use the keyboard or the mouse to make selections from the System Manager display. Both methods are described below.

To exit the System Manager, press **FINISH**.

With the Keyboard

To make selections from the keyboard, follow these steps:

1. Use the arrow keys to position the highlight on the module you want to select.
2. Press the function key for the operation you want to select.

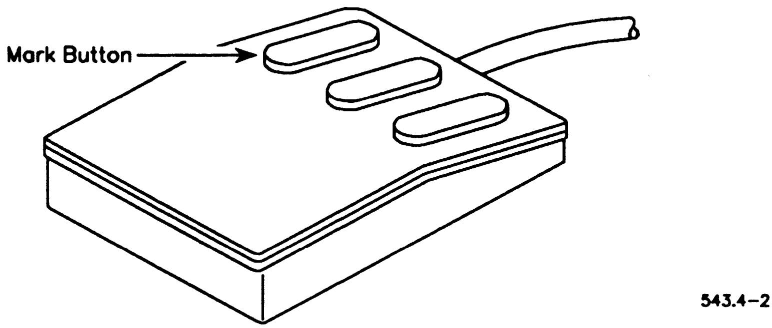
With the Mouse

To make selections with a mouse, follow these steps.

1. Move the mouse to position the highlight on the module you want to select.
2. Click the **Mark** button.
3. Position the mouse cursor (usually an arrow) on the function key you want to select.
4. Click the **Mark** button.

Figure 4-2 shows the **Mark** button on a right-handed three-button mouse. Its position is the same on a two-button mouse. You can reverse right-to-left orientation of the mouse buttons by entering **:LeftHanded:Yes** in the user configuration file. See "Mouse Options" in Section 7, "Customizing User Environments," for more information.

Figure 4-2. Mouse Mark Button (Right-Handed Configuration)



Function Keys

Each hardware component has its own set of functions. Therefore, function key labels change as you select different modules. See the System Manager command in the *CTOS Executive Reference Manual* for a complete list of function keys.

To return to the System Manager display after pressing a function key, press CANCEL.

The F1 (Remote) function key, which appears on some displays, starts the System Manager on an SRP server; it is described later in this section.

No function keys are provided for the keyboard, the Multiline Port Expander, or the PC Emulator.

Using System Manager on the SRP

You can use the System Manager on your SRP server in much the same way you use it on a workstation.

If the F1 (Remote) function key appears on your workstation display, press it to start the System Manager on the SRP server. In a few moments, the SRP System Manager is displayed, as shown in Figure 4-3.

While the System Manager is running on the SRP, the F1 function key is labeled "Local." Press it to exit System Manager on the SRP and return to the workstation display.

Note: *If your workstation display does not show the* F1 *(Remote) key, see "Starting the System Manager With Cluster View," below.*

Starting the System Manager With Cluster View

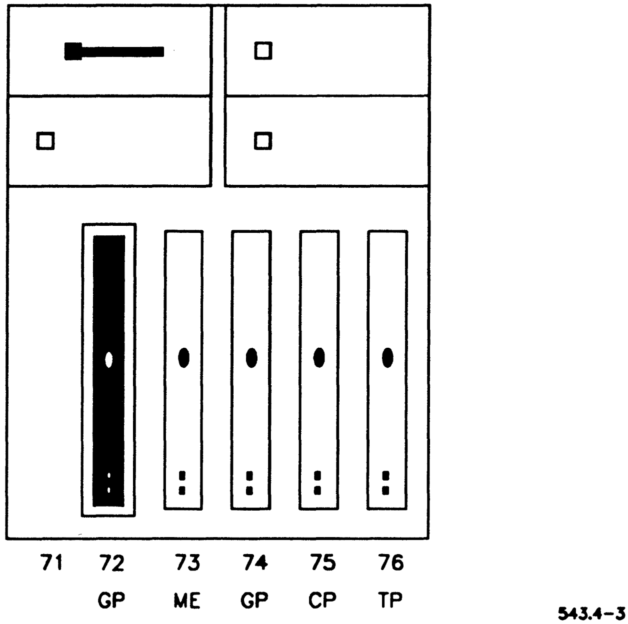
If the F1 function key does not appear on the workstation display, you first start a Cluster View session and then execute the **System Manager** command. See "Cluster View," later in this section, for information about starting Cluster View on an SRP.

System Manager Display on the SRP

On an SRP, the System Manager display resembles the system's hardware configuration, as viewed from the rear. A sample is shown in Figure 4-3.

Use the cursor keys or mouse, as described earlier in this section, to select a processor, disk, or tape drive. When you select a disk or tape drive, the processor that controls it becomes outlined in bold. If the SRP consists of more than three cabinets, press NEXT PAGE to display more cabinets. Press PREV PAGE to return to the picture of the cabinets previously displayed .

Figure 4-3. SRP System Manager Display



The Editor

The Editor is a text-editing application. You use it to make changes or additions to the following:

- Configuration files for workstations and SRPs
- System initialization files
- Format and device templates for the Format Disk command

This section provides basic instructions for using the Editor to modify configuration files. See also the *CTOS Editor User's Guide*, for detailed information about this application.

Starting the Editor

To start the Editor, follow these steps:

1. On the Executive command line, type **Editor**; then press RETURN.
2. Type the file specification of the file you want to open, as shown in the following example:

   ```
   [File name(s)]  [d1]<Memos>Crosby.memo
   ```

3. Press GO.

Using the Editor

When you open a file with the Editor, the cursor appears under the first character of the file. To make changes to the file, you position the cursor, delete existing text, and then type new text. The keys you use to move the cursor and delete text are described below. See the *CTOS Editor User's Guide* for more detailed information.

Cursor Movement Keys

Use the following keys to move the cursor:

RIGHT ARROW	One character to the right
LEFT ARROW	One character to the left
UP ARROW	Up one line
DOWN ARROW	Down one line
CODE-RIGHT ARROW	To the end of the line
CODE-LEFT ARROW	To the beginning of the line
CODE-UP ARROW	To the top of the screen
CODE-DOWN ARROW	To the bottom of the screen
SHIFT-RIGHT ARROW	Five characters to the right
SHIFT-LEFT ARROW	Five characters to the left
SHIFT-UP ARROW	Up five lines
SHIFT-DOWN ARROW	Down five lines
CODE-B	To the beginning of the file
CODE-E	To the end of the file

Deletion Keys

Use the following keys to delete or replace text:

DELETE	Deletes the character where the cursor is positioned.
BACKSPACE	Deletes the character to the left of the cursor.
OVERTYPE	Replaces existing text with new text as you type.

Cluster View

You use the Cluster View facility to issue commands from a cluster workstation keyboard to a processor on the server. Under ordinary conditions, commands are executed locally, on the workstation where you are typing. When you use Cluster View, however, your keyboard and monitor are communicating with a processor on the server. If your cluster is part of a BNet or CT-Net network, you can also use Cluster View on servers in other nodes.

Using Cluster View on a Shared Resource Processor

An SRP does not have its own keyboard and monitor; therefore, to perform any work on the SRP, you must access its processors via Cluster View. For example, suppose that you want to perform a backup of an SRP disk, using its own tape drive. If you issue the **Volume Archive** command from your cluster workstation, data first travels to your cluster workstation for processing and is then sent back along the cluster line to the tape drive on the SRP. With Cluster View, however, only input and output take place on the workstation; the command is actually executed on an SRP processor, so the backup is completed more quickly.

Using Cluster View on a Workstation Server

With Cluster View you can access the server from other workstations in the cluster. The use of Cluster View on workstation servers is optional but is often used for convenience or system security. It is particularly convenient if your server is located in a remote computer room and you want to access it from your own office. In addition, if security is paramount, you can prevent anyone from using the server by removing its keyboard.

Cluster View System Services

Cluster View consists of the following system services, which are installed on the server only:

- The Remote Keyboard Video Service provides basic Cluster View services.
- The Remote User Manager is a Cluster View option for protected mode processors.

Both system services are described in more detail below. See also Section 9, "Installing System Services."

Note: *The Remote Keyboard Video Service and the Remote User Manager are not support on EISA / ISA workstations.*

Remote Keyboard Video Service

The Remote Keyboard Video Service (RKVS) is required to use Cluster View. On SRPs, it is automatically installed on each processor, as part of the bootstrap sequence. On workstation servers, because Cluster View is optional, it is installed from the system initialization JCL file.

RKVS provides the basic Cluster View service. This allows one user at a time to execute commands on a processor via Cluster View. In addition, it runs in a "nonstop" mode; this means that after you initiate a Cluster View session, it remains running until the server is rebooted. For example, if you start a backup and then exit from Cluster View, the backup continues to run on the server. This can be convenient; however, it can be risky, too, because another user could gain access to that session and terminate the backup without your knowledge. For that reason, it is often wise to wait for a command to finish executing and then log out before you exit a Cluster View session.

Remote User Manager

The Remote User Manager (RUM) enhances Cluster View services on protected-mode processors. RUM is an optional system service that you install during system initialization.

When RUM is installed, more than one Cluster View session can run on a processor simultaneously. This means that several users can execute commands or start applications via Cluster View at the same time. Also, when using RUM, each Cluster View session is terminated when the user exits Cluster View. This ensures that other users cannot gain access to a Cluster View session from another workstation.

If you are familiar with Context Manager, think of it to understand how the Remote User Manager works. With Context Manager on a workstation, you can run several applications at the same time. It keeps applications separate from one another, so that the work you perform in one partition is not affected by other partitions. RUM is similar, but it keeps applications separate on the server, for users who are executing them from multiple workstations within the cluster.

Cluster View Commands

You start a Cluster View session from the Executive with either the **Administrator Cluster View** or **Cluster View** command. If the Remote User Manager is not installed, these commands are almost identical, so it doesn't matter which one you use. If the Remote User Manager is installed, however, these commands are used for different purposes, as described below:

Cluster View
Use this command to start multiple sessions on a processor running the Remote User Manager.

Administrator Cluster View
Use this command to prevent other users from starting a Cluster View session on a processor running the Remote User Manager. This ensures access to the primary partition, which is frequently necessary for administrative functions, such as installing system services.

If another user attempts to start a session when you are using Administrator Cluster View, the following message appears:

```
Your session cannot be started.
An administrator session is in
progress.
```

Installing Cluster View on a Shared Resource Processor

On an SRP, RKVS is installed automatically when you boot the system; therefore, no separate installation is required to use it. RUM, however, is installed during system initialization. After it is installed, no other system services can be installed on that processor.

For more information, see "Installing the Remote User Manager" in Chapter 9, "Installing System Services."

Installing Cluster View on a Workstation Server

Note: *The concept of system initialization, as discussed below, has not been introduced yet in this manual. It is included in this section, however, to complete the documentation for Cluster View. See Section 9, "Installing System Services," for detailed information about system initialization and the use of JCL files.*

On workstation servers, RKVS is optional. Therefore, it is installed as a system service from the system initialization JCL file. If you install RKVS only, the keyboard and monitor will not be usable when a Cluster View session is in progress. This is appropriate if you want to limit the use of Cluster View to a single session, or if you plan to remove the keyboard from the server.

In other cases, though, you may want to retain use of the server as a workstation. To do so, you must install the Login Service and RUM, in addition to RKVS. The Login Service allows you to start a Cluster View session on the server workstation so that you can access it from its own keyboard.

The following system initialization JCL file shows entries for RKVS, the Login Service, and RUM in boldface type:

```
Job SysInit
Run [Sys]<Sys>RKVS.run
;Other system services are installed after RKVS but
;before the Login Service and RUM.
Run  [Sys]<Sys>LoginService.run
RunNoWait  [Sys]<Sys>RUM.run
End
```

The default partition size created by the Login Service is 500K bytes, however, it can be created larger or smaller by specifying its size, in K bytes, as shown in the following example:

Run [Sys]<Sys>LoginService.run, 800

The default number of RUM sessions that can be open simultaneously is 2, however, you can specify up to 13 sessions as a parameter to RKVS, as shown in the following example:

Run [Sys]<Sys>RKVS.run, 4

Although the parameter value is placed after RKVS, it has no effect if RUM is not subsequently installed.

Starting Cluster View

The following procedure describes how to start a Cluster View session from a cluster workstation. Instructions for performing specific tasks with Cluster View are included throughout this manual. For more detailed information about the **Cluster View** command forms and parameter fields, see the *CTOS Executive Reference Manual*.

1. On the Executive command line, type **Cluster View** (or **Administrator Cluster View**); then press RETURN.

   The following command form appears:

   ```
   Cluster View
     [Processor name - XE only]     ________________
     [User name]                    ________________
     [User file password]           ________________
     [Node name]                    ________________
     [Old XE run file?]             ________________
     [Run file to invoke]           ________________
     [Partition size]               ________________
   ```

2. Fill in the command form; fields are described in Table 4-1.

3. Press GO.

   If RUM is not installed, the SignOn screen appears; sign on in the usual manner to start the Executive.

Table 4-1. Cluster View Parameter Fields

Field Name	Description
[Processor name - XE only]	Enter the four-character processor ID of the processor you want to access. The default is the processor to which your workstation's cluster line is connected.
[User name]	Enter a user name that is valid for signing on to the server. (This means that the user configuration file must be present on the server.) The default is the user name with which you are currently signed on.
[User file password]	Enter a valid password for the server. The default is the password with which you signed on.
[Node name]	Enter the name of the node with which you want to connect. The default is your own server.
[Old XE run file?]	Enter **Yes** if the program you want to execute uses obsolete methods for writing to video. Use this option if screen output is garbled or nonexistent. This option does not provide workstation-quality video, but it does provide readable output.
[Run file to invoke]	This field applies only to processors running the Remote User Manager. Enter the name of the run file you want to execute. The default is *[Sys]<Sys> Exec.run.*
[Partition size]	This field applies only to processors running the Remote User Manager. Enter the partition size, in K bytes, in which to execute the run file. The default is 400K bytes.

Working in a Cluster View Session

When you are working in a Cluster View session, your keyboard and monitor perform as though they were attached to the server, and the work you perform takes place there. Therefore, *[Sys]<Sys>* refers to the server's system volume; the exclamation point (!) is not required, because you are working on the server. While using Cluster View, you do not have access to your own local disks.

Displaying the Cluster View Menu

While you are working in a Cluster View session, you perform certain functions with the Cluster View menu, as shown in Figure 4-4. To display the Cluster View menu, press the **HELP** key.

The following keys function as described while the Cluster View menu is displayed:

FINISH	Terminates a Cluster View session.
HELP	Displays the Help facility for the application you are using.
CANCEL	Clears the Cluster View menu.
A	Starts the Debugger in Simple Mode.
B	Starts the Debugger in Multiprocess Mode.

Figure 4-4. Cluster View Menu

Key	Action	Processor:GP00
FINISH	Terminate session	
CANCEL	Exit this menu	
HELP	Remote Help	
A	Debugger (Simple Mode)	
B	Debugger (Multi–Process Mode)	

543–4.4

> *Note:* *The A and B menu items for starting the Debugger appear only during an Administrator Cluster View session. See the* CTOS Debugger User's Guide *for information about the Debugger.*

Exiting Cluster View

To exit Cluster View, follow these steps:

1. Press **HELP** to display the Cluster View menu (see Figure 4-4).
2. Press **FINISH**.

Control of the keyboard and monitor return to the cluster workstation from which the **Cluster View** command was executed.

Using Cluster View on a Workstation Server

When the Remote User Manager is installed on a workstation server, the following message appears on the server's monitor:

```
The Remote User Manager is in use.
```

To use the server's keyboard and monitor, you must start a Cluster View session on the server. To do so, press **ACTION-NEXT** to display the SignOn screen; you can then sign on to the server in the usual manner.

You cannot use Context Manager when running a Cluster View session on the server. In addition, when a Cluster View session is running on the server, an administrator session cannot be started from anywhere within the cluster.

To exit a session on the server, follow these steps:

1. On the Executive command line, type **Logout**; then press **GO**.

   The SignOn screen is displayed.
2. Press **ACTION-FINISH**.

Section 5
Bootstrapping

How a System Bootstraps

The term *bootstrap* (or just *boot*) is derived from the saying "to pull oneself up by his or her own bootstraps," meaning without any help.

Each processor contains a component called a *bootstrap ROM*, which contains the first program executed when the system is turned on or reset. (ROM stands for read-only memory.) Because this program is self-contained within the processor, the system is said to bootstrap, or to get itself started without any help. The instructions contained on the bootstrap ROM are permanently etched onto it when it is manufactured.

A major function of the bootstrap ROM is to locate and load the correct operating system (also called *System Image*) for the workstation or SRP master processor. Therefore, the operating system must be stored in a file that is recognized by the bootstrap ROM.

The bootstrap ROM recognizes the following file specifications:

- *<Sys>SysImage.sys*, for a workstation or SRP to boot from one of its own disks
- *[!Sys]<Sys>Ws*NNN>*SysImage.sys*, for a cluster workstation to boot from the server

Ws*NNN* is the workstation-type number, which varies among different processor models. Workstation-type numbers are described in "Bootstrapping a Workstation From a Server," later in this section.

Bootstrapping a Workstation

When a workstation boots from a System Image located on its own disk, it is said to boot locally or from a local disk. When you apply power to a workstation, the bootstrap ROM searches for a file named *<Sys>SysImage.sys*. It searches disks in a predetermined order, depending on the workstation model, and loads the first occurrence of *<Sys>SysImage.sys* containing a valid System Image file. At that point, the disk containing the System Image loaded by the bootstrap ROM becomes the system volume and is known as *[Sys]*.

If no disk on the workstation contains a bootable System Image, the bootstrap ROM attempts to boot from the server. The following procedure describes the workstation bootstrap sequence. These steps are illustrated in Figure 5-1.

1. The bootstrap ROM searches for the operating system. Disk devices are searched in the following order:
   a. Local floppy drives, starting with *[f0]*, then *[f1]*, and so on
   b. Local hard disks, starting with *[d0]*, then *[d1]*, and so on
   c. The *[Sys]* volume on the server
2. The bootstrap ROM loads the operating system into memory and transfers control of the workstation to the operating system. Parameters in the operating system configuration file are implemented at this time.
3. The system initialization file is executed.

Note: *On EISA/ISA workstations, the default bootstrap sequence does not proceed past the floppy drive if an unbootable floppy disk is in the floppy drive. Remove the floppy disk before rebooting the workstation.*

See Section 9, "Installing System Services," for information about system initialization file, and Section 16, "Configuring Workstation Operating Systems," for information about the operating system configuration file.

Figure 5-1. Workstation Bootstrap Sequence

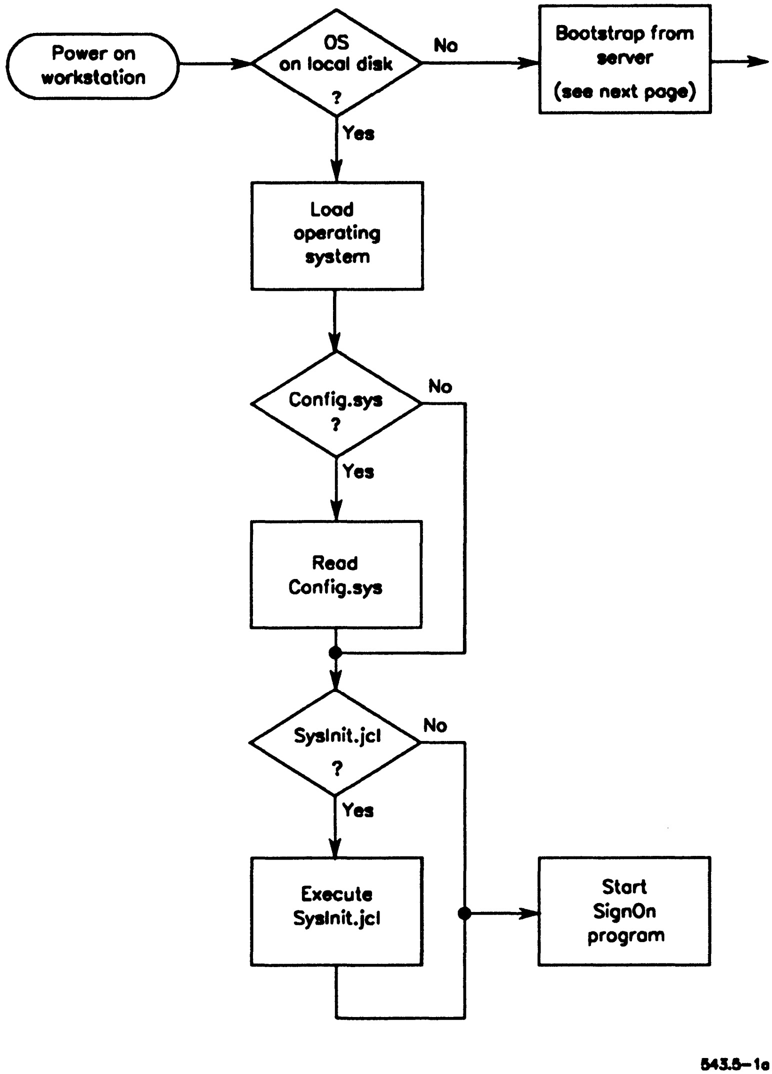
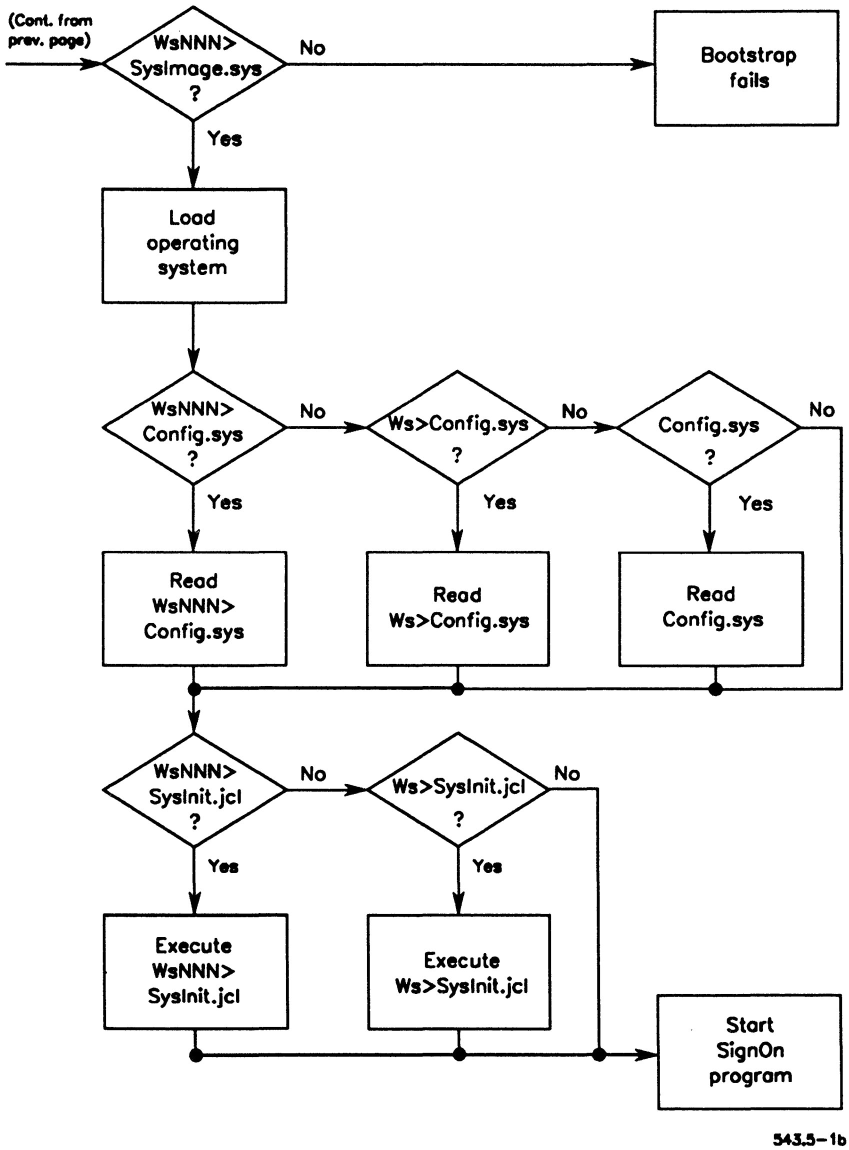


Figure 5-1. Workstation Bootstrap Sequence (cont.)



Bootstrapping a Workstation From a Server

The following workstation configurations bootstrap from a System Image located on the server:

- Cluster workstations without disks
- Cluster LFS workstations with disks initialized for data storage only (that is, not containing an operating system file)

If the bootstrap ROM does not find the System Image on a local device, it searches for it on the server. To locate the correct System Image, the bootstrap ROM searches for a file named as follows:

*[Sys]<Sys>Ws*NNN>*SysImage.sys*

where *NNN* is a three-digit workstation number, as listed in Table 5-1 and described below.

When a workstation boots from *Ws*NNN>*SysImage.sys*, a matching operating system configuration file (*Ws*NNN>*Config.sys*) and system initialization file (*Ws*NNN>*SysInit.jcl*) are executed if they exist. See the flowchart in Figure 5-1 for the exact sequence.

In addition, a SignOn text file (*Ws*NNN>*SignOn.txt*) can be created for each workstation type number; see Section 19, "Customizing Standard Software."

Workstation Type Numbers

The workstation type number *(NNN)* is derived from the following:

- A processor number etched onto the bootstrap ROM
- The devices detected by the bootstrap ROM while it attempts to bootstrap locally

Table 5-1 lists workstation type numbers. The bootstrap ROM uses the workstation type number (*NNN*) to locate the correct operating system on the server. For example, the bootstrap ROM on a diskless B26 workstation searches for *[Sys]<Sys>Ws252>SysImage.sys* on the server.

Table 5-1. Workstation Type Numbers

Workstation Number *(NNN)*	Processor	File System
090	Series 3000	Hard disk
092	Series 3000	Diskless
125	B27	Hard disk(s)
126	B27	Floppy disk(s) only
127	B27	Diskless
200	B24	Diskless
210	B38* and 386 NGEN*	Hard disk(s)
211	B38* and 386 NGEN*	Floppy disk(s) only
212	B38* and 386 NGEN*	Diskless
213	Series 5000	Hard disk(s)
219	Series 2000	Diskless
220	B39 and Series 386i	Hard disk(s)
230	Series 286i	Hard disk(s)
231	Series 286i	Floppy disk(s) only
240	B28, 286 NGEN, B38*, and 386 NGEN*	Hard disk(s)
241	B28, 286 NGEN, B38*, and 386 NGEN*	Floppy disk(s) only
242	B28, 286 NGEN, B38*, and 386 NGEN*	Diskless
250	B26 and 186 NGEN	Hard disk(s)
251	B26 and 186 NGEN	Floppy disk(s) only
252	B26, CWS, and 186 NGEN	Diskless

*Boot ROM versions 3.2 and higher boot from 210, 211, or 212; versions lower than 3.2 boot from 240, 241, or 242. The boot ROM version is displayed on the Bootstrap menu; see "Using the Bootstrap Menu," later in this section.

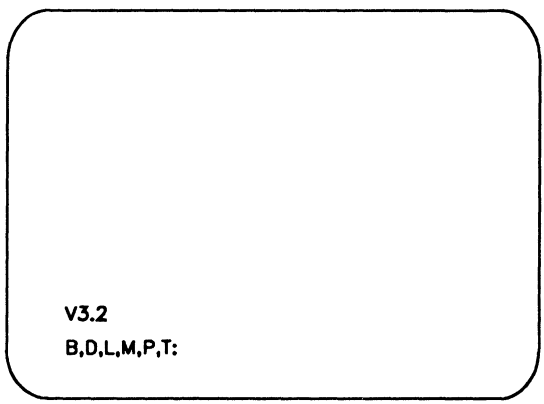
Using the Bootstrap Menu

You can force a cluster workstation to boot from the server rather than from its own disk. On some workstations, you can also bootstrap a workstation type number that is not recognized by the bootstrap ROM. To do so, you manually initiate the bootstrap sequence from the Bootstrap menu.

A representative Bootstrap menu is pictured in Figure 5-2. The bootstrap ROM version number is displayed above the Bootstrap menu. The menu options appear as single characters below the version number. Several commonly used options are described in this section.

Note: *Options on the Bootstrap menu vary among different workstation models. See your harware diagnostics manual for a complete description.*

Figure 5-2. Bootstrap Menu



543.5-2

> **Caution**
>
> The following procedure interrupts applications that are currently running on the system and may cause loss of data. Therefore, always exit applications before bootstrapping.

To bootstrap a workstation from ther server using a specific workstation number, follow these steps:

1. To invoke the Bootstrap menu (see Figure 5-2), hold down the space bar while you turn on or reset the workstation.
2. Release the space bar when the Bootstrap menu appears.
3. Type **T** (either uppercase or lowercase).
4. Type the workstation number, for example, **240**.

   Note: *On X-Bus+ workstations, a default number is displayed when you type* **T**. *It disappears when you begin typing another number.*
5. Press RETURN.
6. Type **B** (X-Bus and X-Bus + workstations) or **C** (EISA/ISA workstations).

Indirect Bootstrapping

If your server is a protected-mode workstation or an SRP, you do not need to store multiple copies of the same System Image in different files for different workstation-type numbers. Instead, each file named *[Sys]<Sys>Ws*NNN*>SysImage.sys* can contain the file specification of the actual System Image file.

For example, when you install the CTOS II operating systems, the file named *[Sys]<Sys>Ws240>SysImage.sys* contains only the following file specification:

[Sys]<Sys>pClstrLfs.img

That is the name of the file containing the protected-mode cluster LFS operating system.

Workstation Hardware IDs

On workstations equipped with appropriate hardware, you can assign a hardware ID number (Hw*NNN*) and use it instead of a workstation type number (Ws*NNN*) in the system initialization file specification, as shown below:

*[Sys]<Sys>Hw*NNN*>SysInit.jcl*

where *NNN* is the hardware ID number assigned to the workstation.

In addition, a customized SignOn text file (*Hw*NNN*>SignOn.txt*) file can be created for individual hardware ID numbers. See Section 19, "Customizing Standard Software," for more detailed information.

You assign a hardware ID number with the **Write Hardware ID** command. You can find out whether a hardware ID number is currently assigned with the **Read Hardware ID** command. See the *CTOS Executive Reference Manual* for information about using those commands.

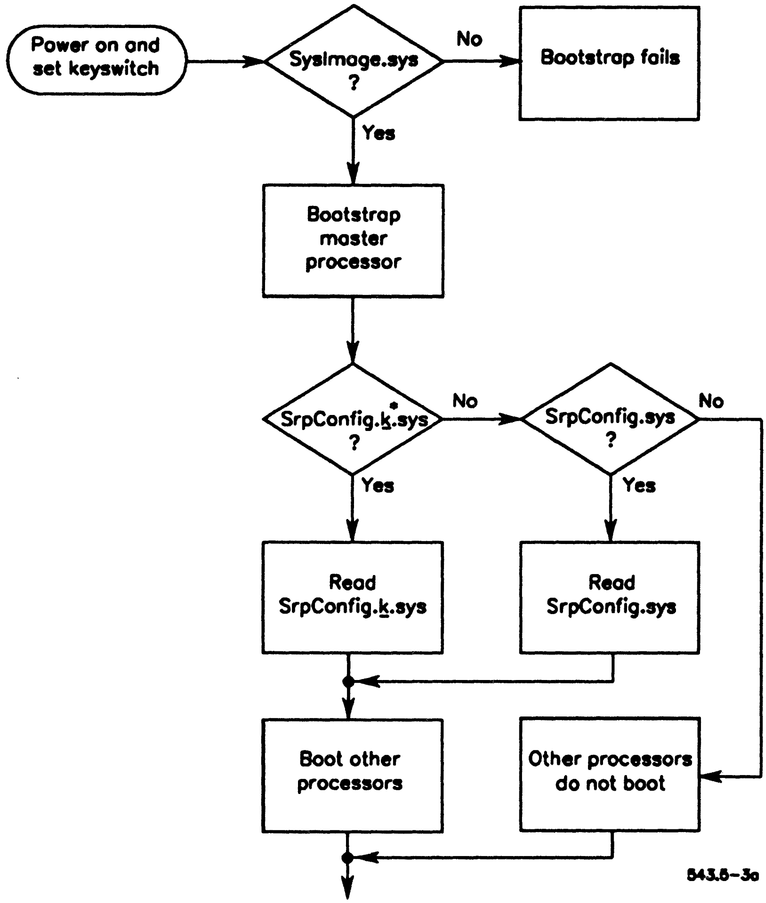
When a hardware ID number is assigned to a workstation that boots from the server, the operating system reads the hardware ID number and, if a corresponding system initialization file exists, it is used during the system initialization procedure. If not, the usual default sequence, as shown in Figure 5-1, is followed.

Bootstrapping a Shared Resource Processor

On a shared resource processor, the master processor bootstraps from the System Image located in *[Sys]<Sys>SysImage.sys*. The master processor then boots the other processors in the system. The following procedure describes the SRP bootstrap sequence. These steps are illustrated in Figure 5-3.

1. The bootstrap ROM on the master processor loads the operating system for the master processor.
2. The master processor reads the operating system configuration file and implements configuration parameters for itself.
3. The master processor boots the other processors and implements parameters, as defined in the operating system configuration file.
4. After the boot sequence, the master processor runs the system initialization sequence to install system services.

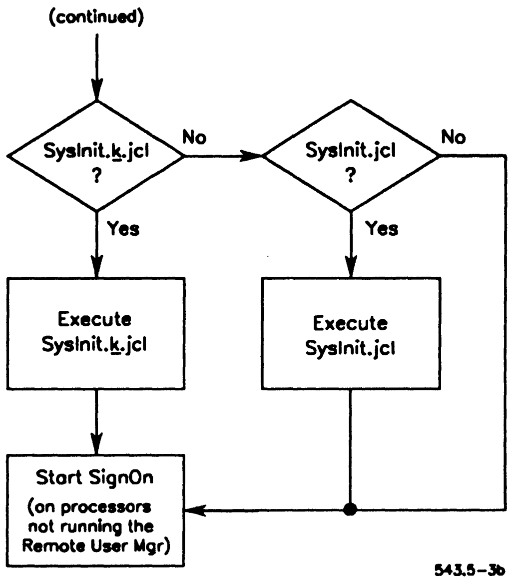
Figure 5-3. SRP Bootstrap Sequence



(continued on next page)

* k represents keyswitch position N, M, or R.

Figure 5-3. SRP Bootstrap Sequence (cont.)



Keyswitch Positions

To bootstrap an SRP, you turn the keyswitch on the front panel (see Figure 2-2) to one of three positions, as described below:

N — Stands for the *normal* keyswitch position; it is frequently used for normal working conditions. In this position, the Reset button is disabled to prevent rebooting by unauthorized persons, and the SRP reboots automatically after a system crash.

M — Stands for the *manual* keyswitch position; it is most often used in software development environments where it is necessary to debug processor crashes. In this position, the Reset button is enabled and the system does not reboot itself after a processor crash.

R — Stands for the *remote* keyswitch position; it is used to load diagnostics programs on the master processor. It can also be used to boot a minimum configuration for troubleshooting. In this position, the Reset button is disabled and the system automatically reboots after a system crash.

Keyswitch Controlled Files

In addition to the differences inherent in the keyswitch positions, these positions control loading and execution of the operating system configuration file (*SrpConfig*.k.*sys*) and the system initialization JCL file (*SysInit*.k.*jcl*). If either of those files exists with *k* corresponding to the keyswitch position from which booting takes place, that file is used. See the flowchart in Figure 5-3 for the exact sequence.

See Section 17, "Configuring Shared Resource Processor Operating Systems," for information about the operating system configuration file. See Section 9, "Installing System Services," for information about the system initialization JCL file.

Section 6
Implementing System Security

How Passwords Work

The CTOS operating system provides a multilevel approach to password protection. As system administrator, you assign passwords and protection levels to volumes, directories, and files. The correct password is then required to allow a user to access files or to execute certain commands on the system. Passwords are frequently used to protect the system in the following ways:

- To protect a volume from commands that destroy data
- To limit access to certain directories or files
- To restrict access to the entire system

Protecting Volumes

A *volume password* is the key to system security. Without a volume password, the password protection mechanism is not activated. Therefore, *you must assign a volume password to each disk that you want to protect.*

The volume password is the "master" password for the disk. It gives the user unrestricted access to the volume. It must be supplied to perform the following operations.

- To reinitialize the volume
- To change the name or password of the volume
- To back up a volume

Volume protection alone, however, does not restrict access to directories and files on the disk. To do this, you must implement additional levels of system security, which are described later in this section.

Assigning a Password to the System Volume

When you use the Standard Software Initialization Diskettes, the **Format Disk** command is included in the installation procedure. The installation pauses to display the command form for **Format Disk**, and you enter parameter values, such as a volume name and password, for the *[Sys]* volume.

> **Caution**
>
> The **Format Disk** command destroys all data on the disk. Do not use it to assign a password to a disk containing data. Instead, see "Changing a Volume Password," later in this section.

To assign a volume password when you initialize a disk, enter a password of twelve or fewer characters in the *[New volume password]* field. The password can contain letters, numbers, and punctuation, as shown in the following example:

```
Format Disk
 Device name                      d0___________________
  [Device password]               ##___________________
  [Current volume password]       #####________________
  [New volume name]               Freddie0_____________
  [New volume password]           J-123________________
  [Configuration file]            _____________________
  [Format template]               _____________________
  [Device template]               _____________________
  [Print file]                    _____________________
  [Overwrite ok?]                 _____________________
  [Bad spot file]                 _____________________
  [Recalculate defaults?]         _____________________
  [CTOS partition size in MB]     _____________________
```

Be sure to keep a record of the passwords you assign. After the disk is initialized, you will not be able to view the volume password on the screen. See also "Initializing an Unformatted Disk," in Section 11, "Adding Hard Disks."

Note: Device passwords are assigned to hard disk drives by the operating system. For the prebuilt operating systems, device passwords match device names, that is, d0, d1, *and so on.*

Assigning Passwords to Other Volumes

After you have installed system software, you initialize the other disks on the system (see Section 11, "Adding Hard Disks"). You can assign the same password to all volumes on a system, or you can make each volume password unique. If you, the system administrator, are the only user with access to volume passwords, your job will be simplified by using the same password on all disks. If, however, other users need volume-level access to certain disks, you may want to assign a different password to each disk.

Changing a Volume Password

Caution

Some applications, such as electronic mail, require reconfiguration after you change a volume name or password.

You can change or assign a volume password with the **Change Volume Name** command. To do so, follow these steps:

1. On the Executive command line, type **Change Volume Name**; then press **RETURN**.
2. Fill in the command form as shown in the following example. Parameter fields are described in Table 6-1.

```
Change Volume Name
  Device name              d0
  [Device password]
  [Old volume password]    ####
  New volume name          NewVol
  [New volume password]    efgh
```

3. Press **GO**.

Table 6-1. Change Volume Name Parameters

Field Name	Description
Device name	Enter the device name.
[Device password]	Default: None Enter the device password for the disk. (In most cases you can leave this field blank; it is required only if the disk is not a valid volume.)
[Old volume password]	Default: Default password Enter the password currently assigned to the volume.
New volume name	Enter the volume name you want to assign to the disk. If you are changing the volume password only, enter the *current volume name* (see the caution above).
[New volume password]	Default: Currently assigned password Enter the new password you want to assign to the volume.

Protecting Directories

When you create directories, you can assign a directory password and set a protection level. The protection level determines whether a volume, directory, or file password is required to gain access to a file. Protection levels are listed in Table 6-2. Protection levels also determine the type of access that is permitted to a file, as described below:

- *Read access* allows the user to view files or load programs. Read access is required for the operating system, configuration files, and application programs.
- *Modify access* allows the user to make changes to a file. Modify access is required to create and make changes to files with applications, such as OFIS Document Designer or OFIS Graphics, and to install or update software applications.

Some examples for using different protection levels are included later in this section.

Table 6-2. Protection and Access Levels

Protection Level	Password Level Required: To Read	Password Level Required: To Modify	Description
15	None	None	Unprotected. No password is required to read or modify the file.
5	None	Volume Directory	Modify protected. No password is required to read the file. A volume or directory password is required to modify the file.
0	Volume Directory	Volume Directory	Read and modify protected. A volume or directory password is required to read or modify the file.
7	None	Volume Directory File	Modify password. No password is required to read the file. A volume,directory, or file password is required to modify the file.
3	Volume Directory File	Volume Directory File	Access password. A volume, directory, or file password is required to read or modify the file.
1	Volume Directory File	Volume Directory	Read password. The file can be read with a volume, directory, or file password, but a volume or directory password is required to modify it.
23	None	Volume File	Nondirectory modify password. No password is required to read the file, but a volume or file password is required to modify it.

continued

Table 6-2. Protection and Access Levels (cont.)

Protection Level	Password Level Required: To Read	Password Level Required: To Modify	Description
19	Volume Directory File	Volume File	Nondirectory access password. The file can be read with a volume, directory, or file password, but a volume or file password is required to modify it.
51	Volume File	Volume File	Nondirectory password. A volume or file password is required to read or modify the file.

Assigning a Password to a Directory

You can assign a directory password when you create the directory or later with the **Set Directory Protection** command. To assign a password when you create a directory, follow these steps:

1. On the Executive command line, type **Create Directory**; then press **RETURN**.

2. Fill in the command form as shown in the following example. Parameter fields are described in Table 6-3.

```
Create Directory
 New directory name(s)               NewDir
  [Default protection level (15)]    0
  [Maximum number of files (75)]
  [Password for new directory]       TFS
  [Volume password]                  #####
```

3. Press **GO**.

See the **Create Directory** command in the *CTOS Executive Reference Manual* for more information.

Table 6-3. Create Directory Parameters

Parameter	Description
New directory name(s)	Enter a name for the directory you want to create.
[Default protection level (15)]	Default: 15 Enter the protection level you want assigned to new files when they are created (see Table 6-2).
[Maximum number of files (75)]	Default: 75 Enter the maximum number of files that can be created in the directory. Note that after a directory has been created, its size cannot be changed.
[Password for new directory]	Default: Active password Enter the password you want to assign to the directory.
[Volume password]	Default: Active password Enter the volume password for the volume on which you want to create the directory.

Changing a Directory Password

To change or assign a password to a directory that already exists, follow these steps:

1. On the Executive command line, type **Set Directory Protection**; then press **RETURN**.
2. Fill in the command form as shown in the following example. Parameter fields are described in Table 6-4.

```
Set Directory Protection
 Directory name(s)                          NewDir
 [Volume or directory password]             ###
 [New file protection level (current)]      5
 [New directory password]                   RAC
```

3. Press **GO**.

This command does not affect files already stored in the directory. See "Protecting Files," later in this section, to learn how to change the password and protection level of a file.

Table 6-4. Set Directory Protection Parameters

Parameter	Description
Directory name(s)	Enter the name of the directory that has the password and/or protection level you want to change.
[Volume or directory password]	Default: Current default password Enter the currently assigned directory password or the volume password.
[New file protection level (current)]	Default: Currently assigned protection level Enter the protection level you want to assign to the directory. If you leave this field blank, the protection level is not changed.
[New directory password]	Default: Currently assigned password Enter a new password for the directory. If you leave this field blank, the password is not changed.

Protecting the <*Sys*> Directory

Protection level 5 is the most appropriate choice for the <*Sys*> directory. It allows read access but does not allow modification without a password. If you "overprotect" <*Sys*> (for example, with a protection level of 0), a password is required to read any file, even files that are required to bootstrap the workstation.

To assign a password and protection level to the <*Sys*> directory, use the **Set Directory Protection** command, as shown in the following example:

```
Set Directory Protection
 Directory name(s)                      Sys
  [Volume or directory password]        #####
  [New file protection level (current)] 5
  [New directory password]              KeepOut
```

After you assign a password to <*Sys*>, you may also need to set the protection level of the files that are stored there. See "Protecting Files," later in this section.

Limiting Access to Directories

If several users share disks on the server, you may want to create an individual directory with a unique password for each user. When you create the directories, assign a protection level of 0 or 3 (see Table 6-2); this prevents access by users who do not know the correct password for the directory. Of course, if the volume password is supplied, a user can access any file or directory on the disk.

Protecting Files

File passwords and protection levels are frequently used to accomplish the following:

- To protect groups of files, such as those in the <*Sys*> directory, from being modified by unauthorized users
- To protect individual files from being either read or modified by unauthorized users

Both situations are described below.

Assigning a Protection Level to a Group of Files

The password and protection level of existing files do not change when you change the protection level of the directory in which they are stored. You assign the protection level to individual files in a separate step, as described in the following procedure.

1. On the Executive command line, type **Set Protection**; then press **RETURN**.
2. Fill in the command form as shown in the following example. Parameter fields are described in Table 6-5.

```
Set Protection
 File list                    [Sys]<Sys>*
 New file protection level    5
 [New password]
 [Confirm each?]
```

3. Press **GO**.

Table 6-5. Set Protection Parameters

Parameter	Description
File list	Enter the list of files that have protection levels and/or passwords you want to change.
New file protection level	Default: Currently assigned protection level Enter the protection level you want to assign to the files. If you leave this field blank, the protection level is not changed.
[New password]	Default: Currently assigned password Enter a new password for the files. If you leave this field blank, the password is not changed.
[Confirm each?]	Default: No If you enter **Yes**, you are prompted for confirmation before the protection level and/or password is changed for each file. If you enter **No** or leave this field blank, you are not prompted for confirmation.

Assigning a Unique Password to a File

When you create a new file, it automatically inherits the password and protection level of the directory in which it is created. You can, however, assign a unique password or change its protection level with the **Set Protection** command.

The following example shows how to assign a password to a file. In addition, by assigning protection level 23, you allow users read access without a password, but require them to enter the file password to modify the file. To truly protect a file, assign a password that is different from the directory password.

```
Set Protection
 File list                     [d1]<Work>Read.doc
 New file protection level     23
  [New password]               secret
  [Confirm each?]
```

To require a password for both read and modify access, assign protection level 51 and a unique password. This method is frequently used to assign passwords to user configuration files. See "Restricting Access to the System," below.

Restricting Access to the System

You can use file passwords to prevent users from signing on without a password. To do this, you assign a password and protection level 51 to each user configuration file (see "Assigning Passwords to User Names" in Section 7, "Customizing User Environments"). You can do this with the User File Editor command when you create the user file, or you can use the **Set Protection** command, as shown in the following example:

```
Set Protection
 File list                      [Sys]<Sys>Freddie.user
 New file protection level      51
 [New password]                 popcorn
 [Confirm each?]
```

When you protect a user file, the user must supply the file password or the volume password to sign on to the workstation.

Allowing Access to a Single Directory

You can also implement the following protection scheme, which requires a valid password for signing on and allows access to files in one directory only:

1. Create a directory with protection level 0 and a unique password.
2. Use the **Set Protection** command to assign protection level 51 and the user's unique password to the user configuration file.
3. Within the user configuration file, specify the user's unique directory password in the *:SignOnPassword:* field (see "SignOn Options" in Section 7, "Customizing User Environments").

With the above method, the user's unique password functions as both the user-file password and the directory password for the user's directory. When the password is entered in the SignOn form, it provides access to the user configuration file; when it is read again from the user configuration file, it provides access to the user's directory.

Alternatively, you could assign a different password to both the user file and the directory. The *:SignOnPassword:* entry in the user file then overrides the password the user enters in the SignOn form. This allows the user access to the directory without knowing its directory password.

Eliminating Known User Names

> **Caution**
>
> Before you delete any prepackaged user files, be sure you have created at least one working user name to sign on with. See Section 7, "Customizing User Environments," for information about creating user names.

To maximize system security, you need to prevent users from signing on with generally known user names. To do so, delete or rename the following prepackaged user files that are included with applications:

- *[Sys]<Sys>.user* is the default user file supplied with Standard Software. It allows users to sign on by pressing GO, without requiring a user name or password.
- *[Sys]<Sys>CM.user* is a sample user file supplied with Context Manager.
- *[Sys]<Sys>Gps.user* is a sample user file supplied with the Generic Print System.
- *[Sys]<Sys>Student.user* is supplied with Standard Software; it is required to use the *Getting Started With Your Workstation* training package.

Limiting Access to Certain Commands

At some workplaces, you may want to limit access to certain commands to specified users only. For example, you may want to prevent anyone but yourself from changing the system date and time.

The Command Access Service allows you to limit access to certain commands that are executed on the server. **Cluster View**, **Administrator Cluster View**, and **Set Time** are monitored by the Command Access Service.

The Command Access Service consists of a system service and a configuration file, which are described below.

Installing the Command Access Service

You install the Command Access Service on the server. To install it during system initialization, add the following entry to your system initialization JCL file:

Run [Sys]<Sys>AccessService.run, *LogFileSize*, *Pswd*

where

LogFileSize	Is the maximum number of sectors of disk space allowed for the log file. (See "Using the Command Access Service Log File," below.)
Pswd	Is a password that allows read access to the Command Access Service configuration file. (See "Protecting the Command Access Service Configuration File," below.)

On SRPs, you install the Command Access Service on one processor only.

The Command Access Service can also be installed with the **Install Command Access Service** command; see the *CTOS Executive Reference Manual.*

Configuring the Command Access Service

The Command Access Service reads a configuration file named *[!Sys]<Sys>UserCmdsConfig.sys*. This file contains entries that monitor access to the recognized commands. A sample file is shown in Figure 6-1.

Note that users from other network nodes can be specified in the configuration file, as shown in Figure 6-1. See "Allowing Access to Users on Other Nodes," below.

Figure 6-1. Command Access Service Configuration File

```
:SignOnUserName:   Tricia
:AllowedCommands:  'Cluster View'
                   'Set Time'
                   'Administrator Cluster View'

:SignOnUserName:   Jim
:AllowedCommands:  'Cluster View'

:SignOnUserName:   {Accts}Renee
:AllowedCommands:  'Cluster View'

:SignOnUserName:   Alex
```

Creating the Command Access Service Configuration File

To create the configuration file, use the Editor application, as described in the following steps:

1. On the Executive command line, type **Editor**; then press **RETURN**.
2. Type **[!Sys]<Sys>UserCmdsConfig.sys**, as shown below.

```
[File name(s)]  [!Sys]<Sys>UserCmdsConfig.sys
```

3. Press **GO** twice.

4. Type the configuration file entries, as shown in Figure 6-1.

   Valid values for *:AllowedCommands:* are **'Administrator Cluster View'**, **'Cluster View'**, and **'Set Time'**, as shown in Figure 6-1. Note that the **'Set Time'** value applies to the *Date/Time* field in the *SignOn* form, as well as the **Set Time** command.

5. Press FINISH; then press GO to save the file and exit the Editor.

After you have created the Command Access Service configuration file, you can add user names or modify entries at any time and the changes are effective immediately.

Protecting the Command Access Service Configuration File

To prevent users from modifying the Command Access Service configuration file, assign it an access level of 23 or 51 and a unique file password. See "Protecting Files," earlier in this section, for information about assigning passwords.

Using the Command Access Service Log File

When a user executes a monitored command, the Command Access Service writes an entry to the log file, *[!Sys]<Sys>Login.sys*. Each log-file entry contains the user's name, the date and time, and whether access was allowed, as shown in the following example:

```
Jim 8/13/92 10:30 AM: Cluster View - Access ALLOWED
```

If a user is not listed in *UserCmdsConfig.sys*, access is restricted, but nothing is logged when that user attempts to execute a monitored command. Therefore, to obtain information about users who attempt to execute monitored commands, you must include their user names in the configuration file. For example, in the sample configuration file, Alex's user name appears, but he is not allowed access to the monitored commands. Therefore, if he attempts to set the system date and time, the following entry is written to the log file:

```
Alex 8/13/92 10:30 AM: Set Time - Name only found
```

In the log file, new entries appear at the beginning of the file and old entries are dropped from the end of the file as it becomes full. To suppress logging, specify **0** as the log file size when you install the Command Access Service.

Allowing Access to Users on Other Nodes

To allow access to a user on another node, specify the node name before the user name, as shown in the following example:

```
:SignOnUserName: {OtherNode}Ruth
```

Note that if a local user is also named Ruth, as shown in the above example, a separate entry without a node specification must be added for the local user. Only the **Cluster View** and **Administrator Cluster View** commands are monitored for access by remote nodes.

Section 7
Customizing User Environments

What Is a User Configuration File?

A *user configuration file* (or simply a user file) defines a name to sign on with and a working environment, such as OFIS Document Designer, Context Manager™, or the Executive. It also contains information about the commands and applications that are available to the user.

You can create user files that suit your particular workplace, as demonstrated in the following examples:

- You can create user files for individual users using first names, last names, or nicknames. This method is often used when most users have their own workstations.
- You can create user names for the various tasks performed within your cluster, for example, writing or drawing. This method is sometimes used when workstations are shared among users.
- You can assign a user name to each workstation, for example, WS1, WS2, and so on. This is sometimes done to simplify signing on for large groups of users.

Creating a User File

To create a user configuration file, follow these steps:

1. On the Executive command line, type **User File Editor**; then press **RETURN**.

2. Enter a user name in the command form, as shown in the following example; then press **GO**.

```
User File Editor
 User name           Freddie
 [Old password]
 [New password]
 [Template file]
 [Command file]
```

3. Press **GO** again to create the user name. The Functions menu appears:

```
Functions
Modify Options
Select Environment
```

543.7-A

4. On the Functions menu, position the highlight on *Modify Options*; then press **GO**.

   A list of applications is displayed. This list may include applications that have not been installed on the workstation.

5. Position the highlight on the application you want to customize; then press **GO**.

   A list of options and current values is displayed, for example, options for SignOn:

```
Option category:  SignOn
Volume                        Sys
Directory                     Sys
File prefix
Password
Node
Text file
User name
Screen timeout
```

543.7-B

6. Fill in the fields as you would an Executive command form. If you leave a field blank, a default value is used.
7. When you have completed the form, press GO to save your changes.

   If you do not want to save changes, press CANCEL to dismiss the form.
8. Repeat steps 5 through 7 to modify options for other applications.
9. Press FINISH, then GO, to save the file.

The user file options for Standard Software are described later in this section. To learn about user file options for other applications, see the appropriate manual. For more detailed information about the **User File Editor** command, see the *CTOS Executive Reference Manual.*

Modifying a User File

To make changes to a user file, follow these steps:

1. On the Executive command line, type **User File Editor**; then press RETURN.
2. Enter the user name in the command form.
3. Press GO.

   The Functions menu appears.
4. Proceed with the step 5 under "Creating a User File."

Editing a User File Manually

To add an option that is not listed by the **User File Editor**, you use the Editor application. For example, you might need to add an option for a new software product.

The following subsections, "File Specifications for User Files" and "User File Format," contain information you need to open and edit a user file with the Editor.

File Specifications for User Files

To edit a user file with the Editor, you need to know its file specification. File specifications are written in the following format:

*[Sys]<Sys>*Name.*user*

where *Name* is the name the user signs on with, for example:

[Sys]<Sys>Freddie.user

User files are always stored in *[Sys]<Sys>*.

User File Format

When you open a user file with the Editor, its appearance is similar to the example shown in Figure 7-1.

Figure 7-1. User Configuration File

```
:SignOnChainFile:[sys]<sys>Exec.run
:SignOnExitFile:[sys]<sys>Exec.run
:SignOnVolume:Sys
:SignOnDirectory:Sys
:SignOnFilePrefix:
:SignOnPassword:
:ExecCmdFile:[Sys]<Sys>Sys.cmds
```

Each line of a user configuration file is written in the following format:

:Keyword:Value

where

:Keyword: — Is the name of a user file option; keywords and the placement of colons must not be changed. They correspond to the field names when you select an application from the *Options* menu in the **User File Editor**.

Value — Is the part you can change; values correspond to the entries you make in the **User File Editor**.

Adding a User File Option

The following procedure describes how to use the Editor to add options to a user file. See "Editing a User File Manually," earlier in this section, for information about file specifications and formats for user files. See the *CTOS Editor User's Guide* for more information about using the Editor.

1. On the Executive command line, type **Editor**; then press **RETURN**.
2. Type the file specification for the user file, as shown in the following example; then press **GO**.

   ```
   [File name(s)]  [Sys]<Sys>Freddie.user
   ```

3. Position the cursor at the end of the file.
4. If necessary, press **RETURN** to move the cursor to a new line.
5. Type the keyword and value as shown in the following example. User file options are listed in the release documentation and manuals for applications.

   ```
   :SignOnChainFile:[sys]<sys>Exec.run
   :SignOnExitFile:[sys]<sys>Exec.run
   :SignonVolume:dl
   :SignonDirectory:Work
   :SignonFilePrefix:
   :SignonPassword:
   :CookDinner:Yes
   ```

6. Press **FINISH**, then **GO**, to exit the Editor and save the file.

For changes to take effect, the user must log out and then sign on again.

Creating a Working Environment

An environment consists of the application and other settings, such as path, that are started when the user signs on. It can also include initial settings that affect the mouse, electronic mail, and the Executive. The keywords that are recognized by the SignOn program and, therefore, can be included in an environment, are listed in Table 7-1.

Each user file contains a default environment, and optionally, one or more additional environments. To use an optional environment, the user enters its name in the *Environments* field of the SignOn form.

Figure 7-2 shows a sample user configuration file containing a default environment and an optional environment. The subsections that follow describe how to define environments in a user configuration file.

Defining the Default Environment

The first set of entries in a user configuration file defines the default environment. In most cases, the default environment is set up to start the user's most frequently used application. To sign on to the default environment, the user leaves the *Environment* field of the SignOn form blank.

A default environment is shown at the beginning of the example user file in Figure 7-2. Note that the default environment does not need to begin with the *:Environment:* keyword. If it does, however, as in the case of some user files created with the **User File Editor**, the value for the *:Environment:* keyword is ignored by the SignOn program.

Entries for the default environment should consist only of keywords that are recognized by the SignOn program, as listed in Table 7-1. Keywords that are read by other applications should be placed at the end of the user configuration file, and should appear one time only, as shown near the end of the example file in Figure 7-2. Such keywords include those that affect the Editor, Cluster View, Generic Print System™ (GPS), and applications such as OFIS™ Document Designer™.

Defining Optional Environments

A user configuration file can contain one or more optional environments. An optional environment allows the user to sign on with his or her usual user name, but to start an application that is different from the default environment. This is particularly helpful when using an application such as OFIS Document Designer, which stores macros and phrases on a user name basis, or GPS Print Manager, which assigns administrative privileges based on a user file entry.

Figure 7-2. User Configuration File Showing Multiple Environments

A sample default environment is shown below.

```
:Environment:Default
:SignOnVolume:Mac0
:SignOnDirectory:Work
:SignOnPassword:Secret
:SignOnExitFile:[Sys]<Sys>Exec.Run
:SignOnChainFile:[Sys]<Sys>Exec.Run
:SignOnScreenTimeOut:10
```

The following sample environment starts Context Manager.

```
:Environment:ContextManager
:SignOnVolume:Sys
:SignOnDirectory:Sys
:SignOnPassword:Secret
:SignOnExitFile:[Sys]<Sys>Exec.Run
:SignOnChainFile:[Sys]<Sys>CmInstall.Run
'Install Context Manager'
[Sys]<Sys>CmConfig.sys
:SignOnScreenTimeOut:10
:MailUserName:Dooley MacMillan
:MailCenterName:PuliCenter
:MailNotification:Normal
```

The following entries are not recognized by SignOn; they should be specified one time only, at the end of the user configuration file.

```
:EditorOpenMode:Read
:EditorTabLength:4
:EditorLineMapping:LF
:GpsUserClass:Admin
:ClusterViewPartitionSizeInK:1280
```

The *:Environment:* keyword is used to introduce each optional environment, as shown near the middle of the example file in Figure 7-2. To create an optional environment, add its keywords and values with the Editor application, as described earlier in this section.

To sign on to an optional environment, the user specifies the value for the *:Environment:* keyword in the *Environment* field of the SignOn form.

Optional environments should contain only those keywords that are recognized by the SignOn program, as listed in Table 7-1. Keywords that are read by other applications should be placed at the end of the user configuration file and should appear one time only, as shown near the end of the example file in Figure 7-2. Such keywords include those that affect the Editor, Cluster View, GPS, and applications such as OFIS Document Designer.

Caution

Do not use the **User File Editor** command to modify a user file containing optional environments. Doing so deletes the keywords and values for optional environments from the user configuration file. Instead, edit such user files manually, as described earlier in this section.

Keywords Recognized by the SignOn Program

Table 7-1 lists the user-file keywords that are recognized by the SignOn program. See "User File Options for Standard Software," later in this section, for a detailed description of keywords for the Standard Software utilities. For a description of keywords that affect electronic mail, see the documentation for your mail application.

When Context Manager is defined as the environment, entries in the Context Manager configuration file may supersede some of the SignOn keyword values in the user configuration file.

The *:SignOnChainFile:* and *:SignOnExitFile:* keywords determine the applications that are started when the user signs on and exits. They are described in more detail later in this section.

Table 7-1. User Configuration File Keywords Recognized by SignOn

Keyword	Default Value
:Environment:	Default environment
:ExecCmdFile:	*[Sys]<Sys>Sys.cmds*
:MailCenterName:	None
:MailNotification:	None
:MailPassword:	None
:MailUserName:	None
:MouseAcceleration:	No change
:MouseSpeed:	No change
:SignOnChainFile:	*[Sys]<Sys>Exec.run*
:SignOnDirectory:	*Sys*
:SignOnExitFile:	*[Sys]<Sys>Exec.run*
:SignOnFilePrefix:	None
:SignOnNode:	None
:SignOnPassword:	No change
:SignOnTextFile:	None
:SignOnScreenTimeout:	Always on
:SignOnUserName:	No change
:SignOnVolume:	*Sys*

The SignOn Chain File Entry

The *:SignOnChainFile:* keyword defines the application that is started when the user signs on. The following example shows an entry that starts OFIS Document Designer:

```
:SignOnChainFile: [Sys]<Sys>OFISDocumentDesigner.run
```

The following example shows a user file entry for a Context Manager environment. To specify parameter values for the chain file, you type the Executive command name and then the parameter values as you would enter them in an Executive command form.

```
:SignOnChainFile:[Sys]<Sys>CmInstall.run
'Install Context Manager'
[Sys]<Sys>CustomCmConfig.sys
```

The SignOn Exit File Entry

The *:SignOnExitFile*: keyword defines the program that appears when the user exits an application. The following entry exits to the Executive:

```
:SignOnExitFile:[Sys]<Sys>Exec.run
```

To exit to the SignOn screen, specify *[Sys]<Sys>SignOn.run.*

The SignOn exit file can limit the user to a single application. For example, if you set up an OFIS Designer environment that exits to the SignOn screen, the user does not have access to the Executive or other applications on the system.

Limiting Access to the System

When you install Standard Software, a default user file, named *[Sys]<Sys>.user*, is created. It allows users to sign on by pressing GO, without entering a user name in the SignOn form. You can effectively limit access to the system by removing the default user file and then assigning passwords to all other user files, as described in the following sections.

Removing the Default User File

> **Caution**
>
> Make sure that you have created a user name you can sign on with before you remove the default user file. See "Creating a User File," earlier in this section.

To remove the default user file, use the **Delete** command, as shown below:

```
Delete
 File list          [Sys]<Sys>.user
 [Confirm each?]
```

Assigning Passwords to User Names

If your system is password protected, you can prevent users from signing on with each other's user names. To do this, use the **User File Editor** command to assign a password to each user file, as described below.

1. Start the **User File Editor** command, as described earlier in this section.
2. Fill in the command form, as shown in the following example:

   ```
   User File Editor
    User name          Freddie
    [Old password]
    [New password]     mypswd
    [Template file]
    [Command file]
   ```

3. Press GO.
4. Select *Modify Options*; then press GO.
5. Select *SignOn*; then press GO.
6. Enter the volume or directory password in the SignOn *Password* field.
7. Press FINISH; then press GO to save the user file and exit the User File Editor.

The password you assign with the **User File Editor** command is entered by the user in the SignOn form when the user signs on. It is a file-level password with protection level 51. See Section 6, "Implementing System Security," for information about password protection levels.

Signing On Automatically

You can set up a workstation so that it signs on automatically with a specified user name, password, and environment when it is rebooted. To use this feature, make the following entry in the system initialization JCL file:

Run [Sys]<Sys>SignonBE.run,*UserName,UserFile,Pswd,Envir*

where

UserName — Is the user's SignOn user name. The default is no user name.

UserFile — Is the file specification of the user configuration file to be read, if that is different from *[Sys]<Sys>*UserName.*user*. (See the description of the *:SignOnUserName:* entry, later in this section.) The default is *[Sys]<Sys>*UserName.*user*.

Pswd — Is a valid password for the user configuration file. The default is no password.

Envir — Is the name of the environment to be started. The default is the first environment in the specified user configuration file. (See "Creating a Working Environment," earlier in this section.)

The automatic SignOn entry is placed directly before the *End* statement, as shown in the following example:

```
Job SysInit
Command Install Mouse Service
Run [Sys]<Sys>SignonBE.run,Wink,,mypswd,OFDD
End
```

See Section 9, "Installing System Services," for detailed information about system initialization JCL files for workstations and SRPs.

Signing On With a Magnetic Card Reader

You can set up a workstation so that users sign on by inserting a card into a magnetic card reader. For a workstation to recognize a magnetic card reader, the MCR Service must be installed during system initialization. To do that, add the following entry to the system initialization JCL file:

Command Install MCR Service

See "Installing System Services on a Workstation," in Section 9, "Installing System Services," for more detailed information about the system initialization process.

The magnetic card used for signing on must contain exactly those keystrokes that would be entered in the SignOn form, including cursor movement and execution keys.

It is not advisable to included date and time information on a magnetic card. If the date and time need to be set routinely, omit the GO keystroke from the card. The user then enters the date and time in the SignOn form and presses GO manually to complete the procedure.

User File Options for Standard Software

The following Standard Software commands and applications include user file options:

- SignOn program
- Executive
- Mouse Service
- **Installation Manager** command
- Cluster View commands

The options for these programs are described below. Both the field names displayed by the **User File Editor** command and the literal keywords are listed.

See the *CTOS Editor User's Guide* for information about user file options for the Editor application.

SignOn Options

The SignOn options take effect when a user signs on. They remain in effect until changed by the user or until superseded by the Context Manager configuration file (see your Context Manager manual).

Volume

Keyword: *:SignOnVolume:*

Default: *Sys*

This entry defines a volume for the default path setting. Specify a volume or device name.

Directory

Keyword: *:SignOnDirectory:*

Default: *Sys*

This entry defines a directory for the default path setting. Specify a directory name.

File prefix

Keyword: *:SignOnFilePrefix:*

Default: None

This entry defines a file prefix for the default path setting. Specify the characters you want to use as a file prefix.

Password

Keyword: *:SignOnPassword:*

Default: See below

This entry defines the password that takes effect after the user signs on. Specify a volume or directory password.

If you specify a password, it supersedes a password entered in the *SignOn* form. If you do not specify a password, the password entered in the *SignOn* form becomes the default password. (See "Limiting Access to the System," earlier in this section, and Section 6, "Implementing System Security.")

User name

Keyword: *:SignOnUserName:*

Default: User name entered in the *SignOn* form

This entry defines the user name that takes effect after the user signs on. A user name specified here supersedes the user name entered in the *SignOn* form. It then appears as the active user name in the status area of the Executive screen, and user file options are read from the user configuration file that corresponds to it. If such a user configuration file does not exist, unpredictable behavior or errors may occur.

Text file

Keyword: *:SignOnTextFile:*

Default: None

This entry defines a file, the contents of which are displayed on the screen when the user signs on. Enter the full file specification for the file you want to display.

Screen time out

Keyword: *:SignOnScreenTimeout:*

Default: Always on

This entry defines the elapsed number of minutes before the screen is to be shut off when a workstation is not being used. Enter a number from 1 to 100.

Executive Option

The following option applies to the Executive application.

Command file

Keyword: *:ExecCmdFile:*

Default: *[Sys]<Sys>Sys.cmds*

This entry defines the command file to be used by the Executive. Enter the full file specification for a valid Executive command file.

Mouse Options

The following options apply to the mouse and are implemented only when an application reads them from the user file. To display these options with the **User File Editor** command, select *General* from the Modify Options menu.

Note: *More mouse options are available for certain applications, such as Art Designer. See the documentation for each application for more information about user file options.*

Left-handed user?

Keyword: *:LeftHanded:*

Default: No

This entry reverses the functions of the mouse buttons for a left-handed user. Specify **Yes** to reverse the mouse button functions.

Setting for speed of the mouse

Keyword: *:MouseSpeed:*

Default: 4

This entry defines the speed at which the mouse cursor moves as you move the mouse. Enter a number between **1** (slowest) and **10** (fastest).

Installation Manager Options

The following options define initial values for the **Installation Manager** command. You can change them later during the software installation procedure. See "Using the Installation Manager" in Section 8, "Installing Applications," for more information.

Install public?

Keyword: *:InstallPublic:*

Default: No

This entry defines whether software is installed on the server (public) or on the local workstation only (private). Enter **Yes** for public installations; enter **No** for private installations.

Install verbose?

Keyword: *:InstallVerbose:*

Default: No

This entry defines the type of messages displayed during software installation. Enter **Yes** to display the entire installation script (verbose). Enter **No** to display only selected status messages (silent).

Install backup?

Keyword: *:InstallBackup:*

Default: No

This entry specifies whether the current version of an application is backed up before new software is installed. Enter **Yes** to back up a current version. Enter **No** to bypass the backup operation.

Save backup?

Keyword: *:SaveBackup:*

Default: No

This entry specifies whether the backup (see *Install Backup?*, above) is saved after installation has been completed.

Use log file?

Keyword: *:InstallLogFile:*

Default: No

This entry specifies whether an installation log file is created.

Log file name

Keyword: *:InstallLogFileName:*

Default: *[Sys]<Installed>Install.log*

This entry defines a log file for command output. Enter a full file specification for the log file, including volume, directory, and file name.

Archive path (private)

Keyword: *:InstallArchivePath:*

Default: *[Sys]<Installed>*

This entry defines the volume and directory where the backup will be created during private installations. Enter a volume and directory name, including brackets, for example, *[Sys]<Installed>*.

Archive path (public)

Keyword: *:InstallPublicArchivePath:*

Default: *[!Sys]<Installed>*

This entry defines the volume and directory where the backup will be created during public installations. Enter a volume and directory name, including brackets, for example, *[!Sys]<Installed>*.

Destination volume (private)

Keyword: *:InstallVolume:*

Default: *[Sys]*

This entry defines the volume where software will be installed during private installations.

Destination volume (public)

Keyword: *:InstallPublicVolume:*

Default: *[!Sys]*

This entry defines the volume where software will be installed during public installations.

CM Config File (Private)

Keyword: *:InstallCmFile:*

Default: *[Sys]<Sys>CmConfig.sys*

This entry defines the Context Manager configuration file that will be updated during private installations. Enter a full file specification, including volume, directory, and file name.

CM config file (public)

Keyword: *:InstallCmFilePublic:*

Default: *[!Sys]<Sys>CmConfig.sys*

This entry defines the Context Manager configuration file that will be updated during public installations. Enter a full file specification, including volume, directory, and file name.

Cluster View Options

The following options define default values for the Cluster View commands. You can override defaults in the Executive command form when you execute the **Cluster View** or **Administrator Cluster View** command. See "Cluster View" in Section 4, "Using Administrative Tools," for more information.

Processor name

Keyword: *:ClusterViewProcessorName:*

Default: Processor to which the workstation is connected

This entry defines the SRP processor on which the Cluster View session is started. Enter the four-character processor ID, for example, GP00.

Run file

Keyword: *:ClusterViewRunFile:*

Default: *[Sys]<Sys>Exec.run*

This field applies only to protected-mode processors running the Remote User Manager. This entry defines a run file to be started on the specified processor. Enter the file specification of the run file.

Partition size in Kbytes

Keyword: *:ClusterViewPartitionSizeInK:*

Default: 400

This field applies only to protected mode processors running the Remote User Manager.

This entry defines a partition size for the run file named above. Enter the number of K bytes for the partition.

Default node

Keyword: *:ClusterViewPathNode:*

Default: Server

This entry defines the network node on which Cluster View will be executed. Enter the name of a network node.

Default volume

Keyword: *:ClusterViewPathVolume:*

Default: *Sys*

This entry defines a default volume for the Cluster View session. Enter the name of a volume on the server or network node.

Default directory

Keyword: *:ClusterViewPathDirectory:*

Default: *Sys*

This entry defines a default directory for the Cluster View session. Enter the name of a directory on the server or network node.

Default file prefix

Keyword: *:ClusterViewFilePrefix:*

Default: None

This entry defines a default file prefix for the Cluster View session. Enter the characters you want to use for a default file prefix.

Default password

Keyword: *:ClusterViewPathPassword:*

Default: Default password

This entry defines the default password when using Cluster View. Enter a valid password for the server or network node.

Section 8
Installing Applications

Software Packages

Applications, such as word processing programs and spreadsheets, are distributed in software packages that consist of the following:

Distribution media	Distribution media are floppy diskettes or QIC tapes that contain the programs, commands, and configuration files for an application. All applications are distributed on diskettes and some are available on QIC tape as well.
Release documentation	This document, which accompanies the distribution media, contains the most current information about the product. It is called a Software Release Announcement (SRA), Release Notes, or Release Notice. In most cases, release documentation includes installation instructions.
Manuals	A manual or set of manuals is available for every software product. Manuals contain detailed information about configuring and using applications. In some cases, manuals also include step-by-step installation procedures.

Planning the Installation

By planning the installation of software applications, you speed up the process and help to ensure that you install everything that is needed on the system. This is particularly true if you are setting up a new system.

Before you begin, decide which applications will be needed on your cluster; some of the most frequently used applications are described later in this section. Then, determine where in the cluster you need to install the software, for example, on the server only, or on all cluster workstations, as well.

Check the release documentation to be sure that the workstation or SRP meets the requirements for the product. For example, each application requires a certain amount of disk space and memory. Make sure that enough of those resources are available before you begin installing software.

What Applications Are Available?

Some frequently used applications are briefly described below. In addition, programming languages, such as Pascal, COBOL, C, and BASIC, as well as a large number of applications from other software firms, are available for CTOS workstations. See your sales representative for detailed information about software products.

Office Automation Applications

OFIS Graphics	For creating charts and drawings
OFIS Mail	For sending and receiving electronic mail messages
OFIS Document Designer	For word processing and office publishing
OFIS Imager	For scanning pictures and editing and printing scanned images
OFIS Spreadsheet	For accounting and financial planning

Communications Applications

BNet	For connecting clusters together to form a network
SNA Transport	For communicating with mainframe applications, using the IBM SNA protocol
X.25	For communicating with public data networks through X.25 circuits

Other Applications

Context Manager	For starting and running multiple applications on a workstation
Generic Print System (GPS)	For providing access to a wide variety of printing devices, including laser printers, scanners, and plotters

Using the Installation Manager

You install software applications with the **Installation Manager** command, which is packaged with Standard Software. It provides the following features for software installation:

- Installation from floppy diskettes
- Installation from QIC tape (some products only)
- Installation from the server to cluster workstations
- Public installations onto the server for use by cluster workstations
- A log file for tracking installation errors
- Removal of software packages
- Recovery from installation failure

These features are described in more detail later in this section.

Some applications were released before the **Installation Manager** command existed and do not take advantage its features. As new versions of applications are released, however, installation procedures are updated. Always check the release documentation or software installation guide for the most current installation instructions.

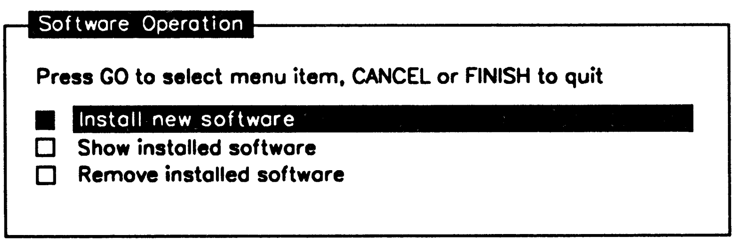
Installing Applications

Note: *The installation instructions for some older applications tell you to use the* **Install, Software Installation,** *or* **XESoftware Installation** *commands. Those commands are now obsolete, and their functionality has been replaced by the* **Installation Manager** *command.* **Installation Manager** *features, such as public installation and installing from the server, however, are not available to those applications.*

The following procedure describes how to install applications with the **Installation Manager** command.

1. On the Executive command line, type **Installation Manager**; then press GO.

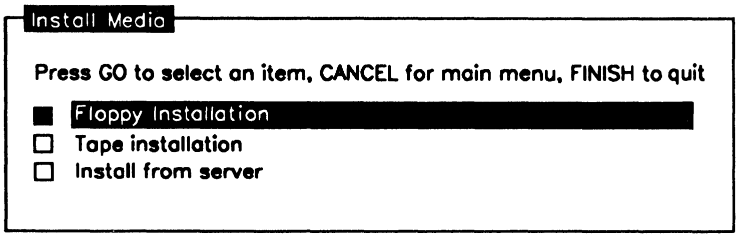
   The Software Operation menu appears:



543.8–A

2. Select *Install new software*, using one of the following methods:
   - Position the highlight with the arrow keys; then press GO.
   - Position the highlight with the mouse; then click the **Mark** button.

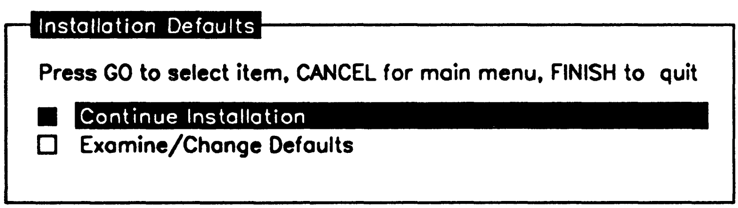
The Install Media menu appears:



543.8-B

3. Position the highlight on the distribution media you want to select; then press **GO** or click the mouse. (The *Install from server* option does not appear on standalone or server workstations.)

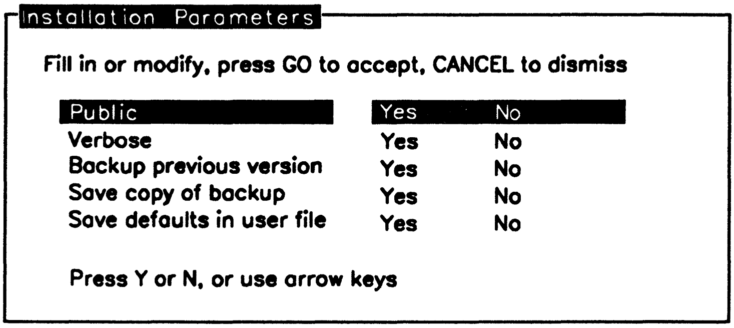
   The Installation Defaults menu appears, as shown below. From this menu, you can continue the installation with default parameters, or you can examine and change parameters. Both options are described below.



543.8-C

- To continue the installation using default parameters, position the highlight on *Continue Installation*; then press **GO** or click the mouse.

- To examine or change defaults, position the highlight on *Examine / Change Defaults*; then press **GO** or click the mouse. The Installation Parameters form appears, as shown below. Default values, which come from the user configuration file, are highlighted.



543.8-D

a. To change a default parameter, position the highlight on the line you want to change.

b. Select Yes or No by typing **Y** or **N**, moving the RIGHT ARROW and LEFT ARROW keys, or clicking your choice with the mouse. Parameters are described in Table 8-1.

4. Press GO or click the mouse to continue the installation.

Depending on the type of installation you choose, additional fields may be displayed. Fill them in as described below, pressing GO or clicking the mouse after each.

Archive path. If you entered **Yes** in *Backup previous version?*, this field appears. It is the volume and directory to which the backup will be written. Defaults are *[Sys]<Installed>* for private installation and *[!Sys]<Installed>* for public installations.

Software destination. This field shows the volume where the application will be installed. Change it if you want to install the application on a different volume. Defaults are *[Sys]* for local installations and *[!Sys]* for public installations.

Password. If a password is required to access the software destination directory, you are prompted to enter it; the installation will not continue until a valid password is supplied.

Tape spec. This field appears if you are performing a tape installation. Enter the tape specification for the application you want to install. It consists of the name of the tape drive in which the distribution media is inserted and, optionally, a tape mark number. See "Specifying a Tape Drive," in the *CTOS Executive Reference Manual*, for information about tape specifications.

User name. This field shows the user name with which you are currently signed on. During software installation, this is frequently not your usual user name. Enter a different user name if you want to update the installation database and save defaults for it, rather the user name appearing in this form.

CmConfig file. If Context Manger is installed during the installation, the name of the active CM configuration file appears in this field. If Context Manager is not installed, the default is *[Sys]<Sys>CmConfig.sys.* If you want the application added to Context Manager, enter the file specification of the CM configuration file you usually use.

Command file. If you are performing a private installation, the name of the active Executive command file appears in this field. If you want new commands placed in a different command file, enter its file specification. During public installations, new commands are always placed in *[!Sys]<Sys>Cluster.cmds* and this field is skipped.

The installation begins and you are informed when it is complete.

5. Reboot the workstation.

Table 8-1. Installation Parameters

Field Name	Description
Public	Select **No** for a *private installation*, which installs software on the local workstation. Select **Yes** for a *public installation*, which installs software on the server. Publicly installed software can be used by all workstations on the cluster.
Verbose	Select **No** for a *silent installation*. During a silent installation, only a few progress messages are displayed. Select **Yes** for a *verbose installation*. During a verbose installation, all messages and command output are displayed on the screen.
Backup previous version	Select **No** if you do not want to back up the current version of the application you are installing. Select **Yes** to back up the current version of a software product before the new version is installed. When you do this, the backed up version is automatically restored if the installation fails for any reason.
Save copy of backup	Select **No** if you do not want to save a copy of the backup. Select **Yes** if you want to save a copy of the backup. You can use this backup later to restore the previous version if necessary, for example, if you discover that the newer version isn't compatible with your hardware or software configuration.
Save defaults in user file	Select **Yes** to change your user file entries to the parameters you just selected. They become new defaults for the **Installation Manager** command. Select **No** if you do not want to change your user file.

Installation Manager Features

The **Installation Manager** includes many features, which are only briefly described on the preceding pages. The following sections contain more detailed information about tailoring software installation to the needs of your cluster.

Installing From Floppy Diskettes

Most software applications are distributed on floppy diskettes. To install an application from floppy diskettes, select *Floppy installation* from the Install Media menu.

Installing From QIC Tape

Some applications are distributed on QIC tape as well as on floppy diskettes. Applications on QIC tape can be installed from a tape drive on either the local workstation or the server.

To install an application from QIC tape, select *Tape installation* from the Install Media menu.

Installing From the Server

After you have installed an application for public use on the server, it can be installed on cluster workstations over cluster lines. This has several advantages over performing installations from floppy diskettes or tapes:

- It is convenient; users do not have to locate diskettes or tapes.
- It installs only what is necessary for cluster workstations.
- The system administrator can configure the application appropriately before other users on the cluster install it.

To install an application from the server, select *Install from server* from the Install Media menu. See also "Installing Public Software," below.

Installing Public Software

Public software is installed on the server for use by all workstations in the cluster. You can perform a public installation from a cluster workstation or on the server. When you perform a public installation, application programs are copied to the server and commands are created in a command file on the server named *[!Sys]<Sys>Cluster.cmds*.

When a user issues a command, the Executive reads the public command file if the command name is not present in the user's private command file. Therefore, all cluster workstations automatically have access to publicly installed software; no preparation or setup is required. In addition, software packages must be installed publicly to make them available to cluster workstations for installation from the server.

To install public software, specify **Yes** to *Public* on the Installation Parameters menu.

Using the Log File

In most cases, a log file named *[Sys]<Installed>Install.log* is automatically created during an installation procedure. It contains a record of command output for the installation. If an installation fails, the log file may contain error messages that can help you determine the cause of the failure.

To conserve disk space, the log file is overwritten with each installation procedure. You can, however, choose to append to the existing log file, by specifying **Yes** in the *[Append to log file?]* field, when you start the **Installation Manager** command. This can be helpful when you are troubleshooting installation problems.

The log file can also be disabled or its name changed by making an entry in the user configuration file. For more information, see "Installation Manager Options," in Section 7, "Customizing User Environments."

Removing an Application

With the **Installation Manager** command, you can remove an application from a system. That includes the programs, configuration files, and commands that are associated with the application.

The **Installation Manager** keeps track of installed software on a user name basis. Therefore, each user is allowed remove software packages that are associated with his or her user name only.

> **Caution**
>
> Do not remove the Basic System Utilities, the CTOS operating system, or the video package (VAM). They are required to boot the system.

To remove an application, start the **Installation Manager**; then follow these steps:

1. From the Software Operation menu, position the highlight on *Remove installed software*; then press GO or click the mouse.
2. Select *Public* or *Private* to display the appropriate list of software to deinstall.
3. Position the highlight on the application you want to remove; then press GO or click the mouse.

Recovering From Installation Failures

When selecting installation parameters, you can choose to back up the currently installed application. The backup takes places before any new software is installed, and it can be restored if the installation fails for any reason.

To choose this option, specify **Yes** to *Backup previous version* on the Installation Parameters menu.

To restore the older version if the installation fails, press GO when the following message is displayed:

```
Press GO to restore the backup, CANCEL to finish.
```

To determine the cause of the failure, see "Common Installation Problems," later in this section, and the release documentation for the application.

Restarting an Installation

If you do not restore the previous version after an installation failure, you might be able to restart the installation procedure at the point from which it failed. This eliminates the tedium of repeating an entire installation procedure.

Note: *The restart feature is not available for all applications. In addition, the restart feature is not available after certain nonrecoverable errors occur.*

The following procedure describes how to restart an installation. Before you restart it, though, correct the problem that previously caused the installation to fail.

1. Start the **Installation Manager** command, as described earlier in this section.

   The following message is displayed:

   ```
   The previous installation of Product Name failed.

   Press GO to restart installation, CANCEL to start new
   installation.
   ```

2. Do one of the following:

   - Press GO to restart the installation from the point at which it failed.

   - Press CANCEL to start the installation again from the beginning.

Loadable Requests

Loadable requests expand the capabilities of the operating system and may be required by certain system services and applications. When required, loadable request sets are packaged on the installation media for the software that requires them.

During software installation, loadable requests are merged into a file named *[Sys]<Sys>Request.sys*, which is loaded into memory when a workstation or SRP boots. If you bypass part of the installation procedure, however, the loadable requests may not be merged. This can result in Error 31 (No such request) when you try to install a system service or use an application.

On SRPs, you can specify a request file other than *[Sys]<Sys>Request.sys* in the operating system configuration file. If you do this, though, you may need to merge requests manually into the file you specify. See the **List Request Set**, **Make Request Set**, and **Update Request Set** commands in the *CTOS Executive Reference Manual*.

See the *:LoadableRequestFile:* parameter in Section 17, "Configuring Shared Resource Processor Operating Systems," for information about specifying a request file other than *[Sys]<Sys>Request.sys*.

Common Installation Problems

Table 8-2 describes some common problems that can occur during software installation. See also the Software Release Announcment for the product you are installing.

Table 8-2. Software Installation Errors

Error Code	Description
202	Directory full This error occurs if a directory becomes full during software installation. Check the directory named *<Installed>* (on either the private or public volume), as well as the directory into which you are installing software.
219	Access denied This error occurs if the destination directory requires a different password from *[Sys]<Sys>*. For example, if you must specify **PswdA** to access the software destination volume, but **PswdB** is required to access *[Sys]<Sys>*, the **Installation Manager** may not be able to create new commands in *[Sys]<Sys>*. Similarly, if the destination volume does not require a password, you will not be prompted to enter one; however, if a password is required for *[Sys]<Sys>*, the installation may fail. To avoid such problems, use the **Set Directory Protection** command to remove the directory password from *[Sys]<Sys>* before you begin the installation. Then, after the installation is complete, use the **Set Directory Protection** command again to reassign the original password to *[Sys]<Sys>*.
220	File in use This error occurs when the Executive command file for a workstation is in use. This can happen if the workstation is running Context Manager or if several workstations are sharing a command file on the server. If the workstation is running Context Manager, log out and after you sign on again, restart and perform the installation before you install Context Manager. For a shared command file, have all users who share the command file log out and remain logged out while you restart and perform the installation.
230	Disk full This error occurs when you do not have enough disk space for the application. You will have to make room on the disk before you restart the installation.

Section 9
Installing System Services

What Is a System Service?

A *system service* is a software program that manages or provides access to a resource that is available on an individual workstation or within the cluster. A resource is frequently a hardware device, such as a mouse, a printer, or a tape drive or it can be a software component, such as a communications gateway, a database, or an electronic mail center.

Some system services, such as those that manage the keyboard and file system, are part of the operating system, so you do not need to install them or be aware of them as system services. Because system services consume memory, however, only those that are always needed are included within the operating system itself. You must install additional system services to handle optional resources, such as printers and modems.

Installing system services is a separate task from installing software applications. When you install applications, you copy programs and commands onto a disk. When you install a system service, you load a program from a disk into memory on a processor. The programs and commands you use to install system services are placed on your disk when you install applications. Many applications require system services.

As system administrator, you determine where in the cluster system services are required, and you set up the installation procedures. This section describes many commonly used system services and includes procedures for installing them on workstations and SRPs.

What System Services Do You Need?

The system services you need to install depend on the equipment and applications being used within your cluster. Some frequently used system services are described below. A server workstation may require almost all of them, while a cluster workstation may not need any. More information about where to install system services is included later in this section.

Standard Software System Services

The following system services are packaged with Standard Software.

CD-ROM Service	Is required to use a CD-ROM drive.
Cluster File Access Services	Allow users to access files on cluster workstations, as well as on the server.
Command Access Service	Limits access to the **Administrator Cluster View, Cluster View** and **Set Time** commands to specified users only.
DataComm Service	Is required to use the DCX port expander module.
Math Service	Performs floating point calculations for applications that do not include that function.
MCR Service	Is required to use a magnetic card reader.
Mouse Service	Is required to use a mouse.
Queue Manager	Is required for spooled printing. It is also used to control message passing by other applications, such as Batch and SNA RJE. The Queue Manager is always installed on the server.
Remote Keyboard Video Service	Is required to use Cluster View. On workstation servers, it is installed as a system service. On SRPs, it is included in the operating system.

Remote User Manager	Allows multiple Cluster View sessions to be started on a protected-mode processor.
Screen Print Service	Is required to print the contents of the screen to a file.
Sequential Access Service	Is required to access digital data storage (DDS), quarter-inch cartridge (QIC), or half-inch tape drives.
Spooler	Is the pre-GPS spooled printing service. (See the appendix about pre-GPS printing in the *CTOS Generic Print System Administration Guide*.)
Statistics Service	Provides information about memory utilization and disk space.
Voice Service	Is required to send or receive voice data messages.
XBIF Service	Is required for certain workstation modules to communicate across the X-Bus™. Other module-specific system services, such as the XC-002 Service, must be installed after the XBIF Service.
XC-002 Service	Is required to use an XC-002 Port Expander module.

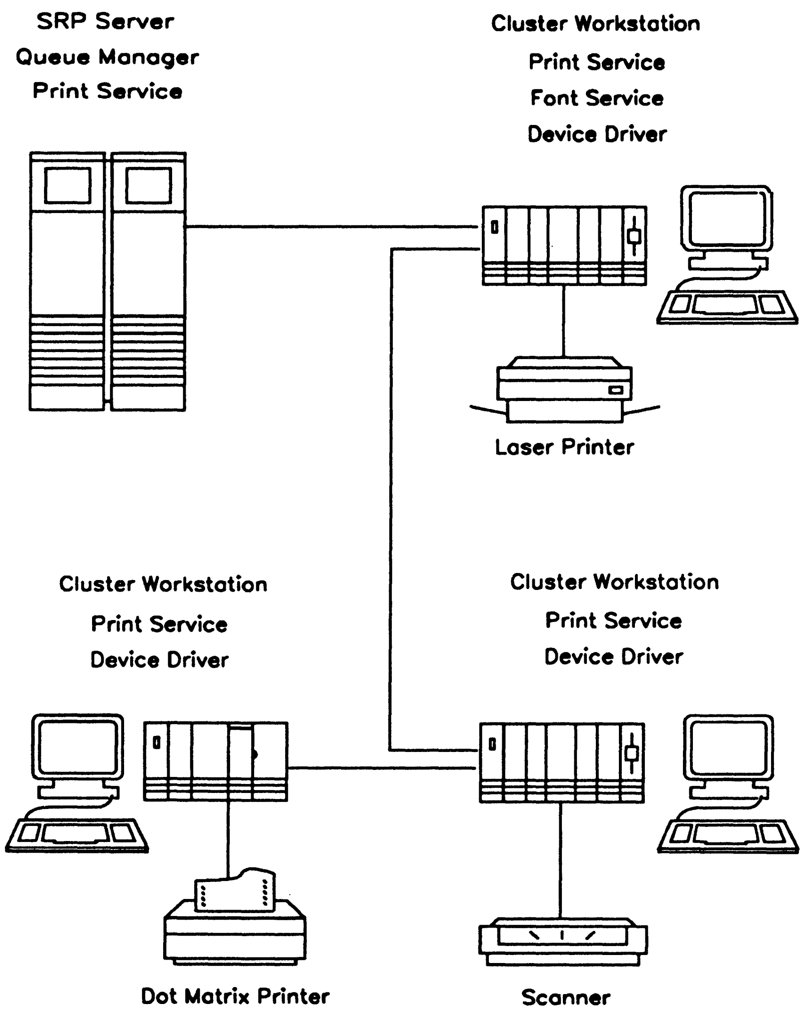
Generic Print System Services

The Generic Print System (GPS), although packaged separately from Standard Software, is usually installed and maintained by the system administrator. It includes the following system services:

Queue Manager	This is the same system service supplied with Standard Software. It is installed at the server, and only one Queue Manager per cluster is installed.
Print Service	This system service is installed at the server and on each workstation or SRP processor to which a spooled printer is attached.
Font Service	This system service can be installed at the server for use by all cluster workstations, or on each cluster workstation that is using a font data base. Performance is improved when the Font Service is installed at the cluster workstation.
Device Drivers	These system services control printers. The appropriate Device Driver is installed at each workstation or SRP processor to which a printer is attached.

Figure 9-1 shows an example of GPS system services installed within a cluster. See the *CTOS Generic Print System Administration Guide* for more detailed information.

Figure 9-1. GPS System Services in a Cluster



543.9–1

Electronic Mail Services

The following system services are required for electronic mail applications. They are frequently installed and configured by the system administrator.

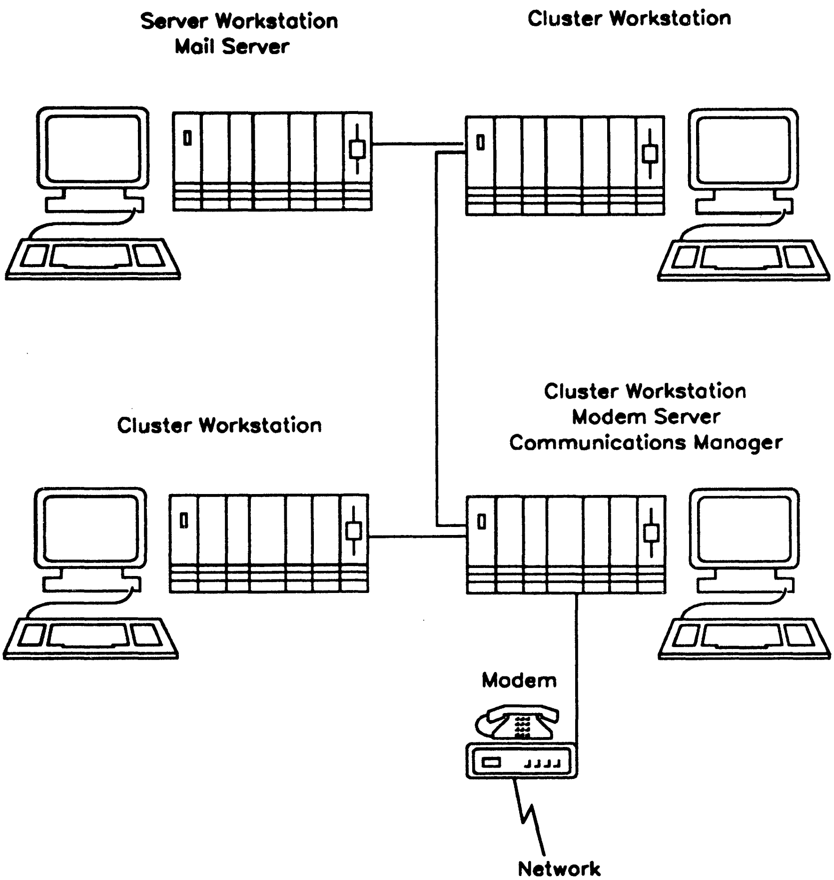
Mail Service	This system service controls distribution of incoming and outgoing mail for the entire cluster. It must be installed at the server before the Modem Service and Communications Manager are installed.
Modem Service	This system service is required to use a modem. It is installed at the system where the modem is connected. It is installed after the Mail Service but before the Communications Manager. The Modem Service is also distributed as a separate software package.
Communications Manager	This system service provides communications between mail centers. It is installed after the Modem Service on the system to which the communication line is connected. The Communications Manager is not required on mail centers communicating through a local area network.

Electronic mail applications also include the following optional system services:

- Telex/TWX Manager
- Terminal Mail Manager

Figure 9-2 shows an example of the electronic mail services installed within a cluster. See the documentation for your mail system for more detailed information.

Figure 9-2. Electronic Mail System Services in a Cluster



543.9-2

Network System Services

Network software is also installed and maintained by the system administrator. It may include a number of different system services, such as BNet, which provides communications among remote clusters, and one or more media system services, which provide data communications protocols. The types of network services you install are determined by the physical data links that connect your cluster to other nodes in the network.

See the manuals for your networking products for more detailed information.

Other System Services

Many other system services are available as separate products or as part of application software packages. These include the following:

- Clustering services, for allowing IBM PCs® and other compatible computers to share resources on a CTOS cluster
- Indexed Sequential Access Method (ISAM) Service, for database applications
- Communications services, such as SNA and X.25 gateways, for communicating with external data networks

See the product documentation for information about installing them.

Where to Install System Services

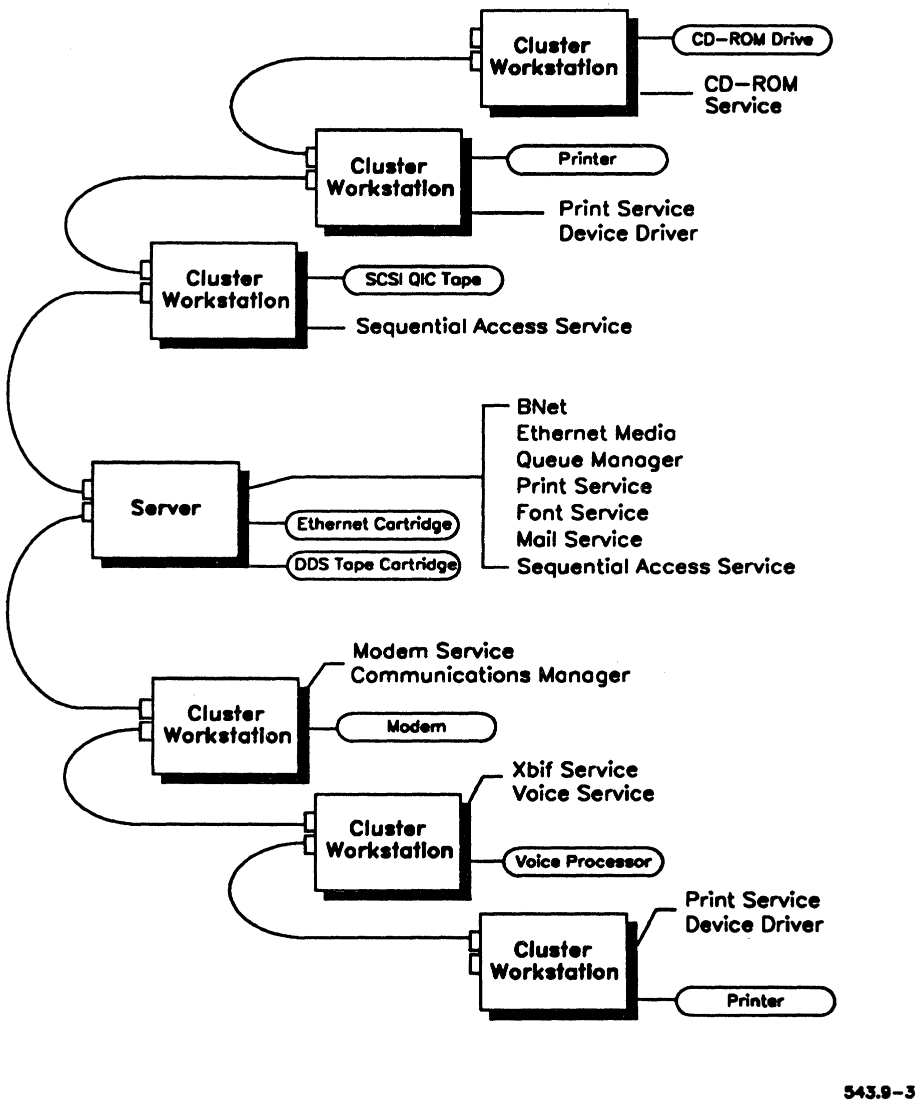
It is not unusual for many system services to be installed within a cluster, as shown in Figure 9-3. In general, system services are installed on the workstation or SRP processor to which the resource is attached. There are exceptions to this, however; for example, the Queue Manager and Mail Service are always installed on the server. See "What System Services Do You Need?," earlier in this section, for general information; always read the release documentation and operating guides for the most current information.

Where you locate resources, and subsequently install system services, depends on several factors. The layout of your work area can be a factor in where you set up equipment. For example, a modem must be installed near a telephone line and a printer requires adequate space.

For shared system services, such as printing, you need to consider the type of work for which a workstation or SRP processor is used. For example, if a workstation is often used for graphics and font applications, the printing services might decrease performance of other applications. Similarly, if an SRP processor is used extensively for communications applications, the printing services might perform more efficiently on a different processor.

The following section, "Calculating Memory Requirements," helps you determine where to install system services. See also Section 15, "Optimizing System Performance."

Figure 9-3. System Services Installed Throughout a Cluster



Calculating Memory Requirements

The term *memory* refers to random access memory (RAM), which is located on workstation and SRP processors. The term memory *does not* refer to disk storage space. All workstations, even those that are diskless, contain memory.

All system services and applications require memory in which to execute. The way you determine memory requirements for a workstaton or SRP processor depends on its operating system, as described in this section.

Operating Systems Without Virutal Memory

Real-mode and protected-mode operating systems without virtual memory use *partitioned memory management.* On such systems, system services and applications are loaded into individual memory partitions, and the amount of memory available at any given time is the total amount of processor memory minus the amount of memory allocated to those individual partitions.

Virtual Memory Operating Systems

Virtual memory operating systems use *demand paging.* On such systems, no single amount of memory is specifically available or unavailable at any given time. Instead of being loaded into partitions, applications and system services are circulated in and out of memory in 4K byte *pages.* Therefore, the number of programs and system services that can be installed and active on a virtual memory system is not static.

How Much Memory Is Available?

To determine the amount of memory available on a partitioned memory system, you use the **Partition Status** command. The **Partition Status** display is pictured in Figure 9-4; instructions for using **Partition Status** are included later in this section.

Figure 9-4. Partition Status Display

```
Partition                 Allocated        Used         Run file executing

   System                   856K           856K         pClstrLfs  3.4
2  Vdm_VGA   3.4            184K           182K
3  Mouse Server              21K            19K
4  XvtChService             337K           335K
5  Primary                  591K           230K         SystemMgr.run

Total RAM:                 4096K                        Available:  2747K
```

The **Partition Status** display lists the amount of memory allocated for each application or system service partition. Memory statistics are displayed in K bytes; each K byte is equivalent to 1024 bytes of memory.

To determine how much memory remains available on the processor, add all values in the *Allocated* column of the **Partition Status** display and subtract that total from the total amount of processor RAM. For example, in Figure 9-4, the amount of allocated memory equals 1989K bytes (856+184+21+337+591=1989). The amount of processor memory, as displayed in the *Total RAM* field, is 4096K bytes. Therefore, the amount of unallocated memory that remains available on the processor is 2107K bytes (4096-1989=2107).

On virtual memory systems, the **Partition Status** command displays the amount of memory available in frames. For the most part, you do not need to be concerned with calculating available memory on virtual memory systems; however, a small number of available frames may indicate that the system is not performing optimally. See Section 15, "Optimizing System Performance," for information about optimizing the use of virutal memory.

See the *CTOS Executive Reference Manual* for more detailed information about the **Partition Status** display.

Starting Partition Status on a Workstation

To start **Partition Status** on a workstation, follow these steps:

1. On the Executive command line, type **Partition Status**.
2. Press GO.

Starting Partition Status on an SRP

To start the **Partition Status** command on an SRP processor, use one of the following methods:

- Start a Cluster View session on the appropriate processor; then execute the **Partition Status** command from the Executive command line.
- On processors running the Remote User Manager, specify *[Sys]<Sys>SystemMgr.run* in the **Cluster View** command form; then start **Partition Status** from the System Manager display.

For more information, see "Cluster View" in Section 4, "Using Administrative Tools."

How Much Memory Is Required?

The following factors determine the amount of memory that is required to install the system services you need on a partitioned memory system. These factors must be considered separately for each workstation or SRP processor:

- What is the total amount of memory (RAM) on the workstation or SRP processor? (See "How Much Memory Is Available?," above.)
- How much memory is required for the operating system and video service partitions? (See the Software Release Announcements.)
- How much memory is required for all the system service partitions? (See the release documentation for the software packages you have installed.)
- How much memory is required for the largest application partition? (See the release documentation for each of the applications you plan to use.)

To calculate the total memory requirement, add the above factors, as shown in the following example:

```
Operating system partition                  856K
Video service partition                     184K
System-services partitions                  458K
Partition for the largest application      +800K
                                           ------
   Total                                   2298K
```

The total memory requirement for the operating system, system services, and largest application partition must not exceed the total amount of processor memory (RAM).

On virtual memory systems, the paging service moves programs and data in and out of memory as they are needed. Therefore, you can install more system services and start more applications than actually fit into memory on the processor, based on the *working set* of memory that is required by the system service or application. A working set is the amount of memory required by an application to function with reasonable performance. See Section 15, "Optimizing System Performance," for more detailed information about virtual memory management, the paging service, and working set values.

Installing System Services on a Workstation

System services can be installed on a workstation during system initialization or you can install them with Executive commands. Table 9-1 lists the Executive commands and run-file names for the system services supplied with Standard Software.

See the *CTOS Executive Reference Manual* for information about parameter values. See the release documentation for information about installing system services for other applications.

Table 9-1. System Service Commands and Run Files

System Service	Executive Command	Run File
CD-ROM Service	**Install CDROM Service**	CDROMService.run
Cluster File Access File Filter	**Install CFA File Filter**	CfaFf.run
Cluster File Access Server Service	**Install CFA Server Service**	CfaM.run
Cluster File Access Workstation Agent	**Install CFA Workstation Agent**	CfaWa.run
Command Access Service	**Install Command Access Service**	AccessService.run
DataComm Service	**Install DataComm Service**	DcxService.run
Math Service	**Install Math Service**	MathService.run
MCR Service	**Install MCR Service**	MCRService.run
Mouse Service	**Install Mouse Service**	Mouse.run
Queue Manager	**Install Queue Manager**	InstallQMgr.run
Remote Keyboard Video Service	None	RKVS.run
Remote User Manager	None	RUM.run
Screen Print Service	**Install Screen Print**	InstallScreenPrint.run
Sequential AccessService	**Install Sequential Access Service**	InstallSeqService.run
Spooler	**Install Spooler**	Spooler.run
Statistics Service	**Install Statistics Service**	Statistics.run
Voice Service	**Install Voice Service**	AudioService.run
XBIF Service	**Install XBIF Service**	Xbif.run
XC-002 Service	**Install XC002 Service**	XC002Service.run

Installing From the Executive

You can install system services with Executive commands. This method is generally used in the following circumstances:

- When you install a system service for the first time and want to make sure that it works as you expect.
- When you need a system service only at certain predictable times, for example, if you use the Sequential Access Service once a day for backups.

Note: *You must install system services before you start Context Manager on a workstation.*

To install a system service with the Executive, follow these steps:

1. Type the command name on the Executive command line; then press RETURN (see Table 9-1).
2. Fill in parameter fields as required; see the appropriate manual or release documentation.
3. Press GO.

Installing During System Initialization

System services that you use all the time, such as printing or electronic mail, can be installed during system initialization. This is convenient and automatic; whenever you reboot the workstation, system services are installed.

After the workstation bootstraps, the system initialization program searches for a file named *[Sys]<Sys>SysInit.jcl*. If it exists, its contents are read and executed. You create *SysInit.jcl* to install the appropriate system services on the workstation. Figure 9-5 shows a system initialization file for a workstation.

Figure 9-5. Workstation System Initialization File

```
Job SysInit
ContinueOnError
Run [Sys]<Sys>Xbif.run
Run [Sys]<Sys>InstallQMgr.run, y, 30
Run [Sys]<Gps>GpsInstall.run
Run [Sys]<Sys>MailServer.run
Run [Sys]<Sys>ModemServer.run
Run [Sys]<Sys>CommunicationsManager.run, Line1
Run [Sys]<Sys>InstallSeqService.run, [QIC]
Run [Sys]<Sys>MouseService.run
End
```

Creating a System Initialization File

To create a system initialization file, you use the Editor to write a simple program in Job Control Language (JCL). Table 9-2 lists the JCL statements you use in this file. See "JCL Syntax," later in this section, for rules about punctuation and spacing. See the *CTOS Editor User's Guide* for information about using the Editor.

After you create the file, reboot the workstation to install the system services.

Table 9-2. JCL Statements for Workstations

Statement	Description
Command	This statement specifies the name of an Executive command. To be executed during system initialization, the command must be present in *[Sys]<Sys>Sys.cmds.*
ContinueOnError	This statement forces continuation of the system initialization file, even if an error occurs. It affects statements that follow it until the end of the file, or until a CancelOnError statement occurs.
CancelOnError	This statement reinstates the CancelOnError condition, which terminates system initialization when an error occurs.
End	This statement defines the end of a Batch job. End your system initialization file with it, as shown in Figure 9-5.
Job	This statement defines a name for the batch job. For system initialization, specify **SysInit**, as shown in Figure 9-5.
Run	This statement specifies the name of a run file to be executed.

JCL Syntax

When you create a system initialization file, you are writing a program in Job Control Language. (JCL files are processed by the Batch facility, which can be used for jobs other than system initialization.) The following sections describe syntax rules that apply to system initialization. See the *CTOS Batch Manager II Installation, Configuration, and Programming Guide*, for more detailed information.

Specifying Run Files and Command Names

Separate the name of a run file or command from the Run or Command statement with at least one space or tab, as shown in the following examples:

Run file

```
Run [Sys]<Sys>MouseService.run
```

Command

```
Command Install Mouse Service
```

Specifying Parameters

When a run file or command requires parameters, separate them with commas, as shown in the following examples:

Run file

```
Run [Sys]<Sys>InstallQMgr.run, yes, 10
```

Command

```
Command Install Queue Manager, yes, 10
```

Each pair of commas defines one field. If you leave a parameter blank, you must enter commas as a place holder for the field. The following example shows three blank fields between the parameter values 75 and 50:

```
Run [Sys]<Sys>Net.run, SJ-Node, 67, 2, 75,,,,50
```

Commas are not required, however, for parameter values omitted at the end of a statement.

Specifying Subparameters

When a parameter consists of more than one value per field, enclose subparameters in parentheses, as shown in the following example:

```
Command Disable Cache, (File1, File2), yes
```

Entering Multiple-Line Parameters

When parameters exceed a single line, use an ampersand (&) to indicate that the statement continues on the next line, as shown in the following example:

```
Run [Sys]<Sys>CommunicationsManager.run,&
VeryLongNameIsCarriedOver
```

Entering Comments

To enter a comment, precede text with a semicolon (;), as shown in the following example:

```
;This is a comment
```

You can add comments to clarify the contents of a file (for yourself or others who might edit it) or to temporarily disable a JCL statement. The following example shows both:

```
Job SysInit
ContinueOnError
Run [Sys]<Sys>Xbif.run
;I have disabled the Seq Service.
;Run [Sys]<Sys>InstallSeqService.run, [QIC]
Run [Sys]<Sys>MouseService.run
Run [Sys]<Sys>AudioService.run
End
```

Workstations That Boot From the Server

You can create system initialization files for cluster workstations that boot from the server. Such files must be located in *[Sys]<Sys>* on the server and are named as follows:

*[Sys]<Sys>Ws*NNN*>SysInit.jcl*

where *NNN* is the three-digit workstation-type number or, if that file does not exist for a particular workstation type:

[Sys]<Sys>Ws>SysInit.jcl

If neither *Ws*NNN*>SysInit.jcl* nor *Ws>SysInit.jcl* exists, no system initialization sequence is executed on the workstation.

See also "Workstation Type Numbers" in Section 5, "Bootstrapping," for more information and diagrams of the system initialization sequence.

You can also create system initialization files that correspond to workstation hardware IDs. See Section 5, "Bootstrapping," for more detailed information.

Installing System Services on an SRP

Before you install system services on an SRP, it is important to plan the entire installation. Keep the following points in mind as you decide where to install system services on your SRP:

- Some system services must be installed on a certain type of processor.
- Some system services must be installed on the processor physically connected to the resource it is going to manage.
- Some system services must be installed before or after other system services.

See the release documentation for your applications for detailed requirements about where and in what order to install each system service.

Installing With Cluster View

You can use Cluster View to install system services with Executive commands. This is a good technique when you are setting up an SRP for the first time or adding a system service after installing a new application. After you have determined that the system services install correctly, you can add them to the system initialization file.

Note: *You must install system services on protected-mode processors before the Remote User Manager is installed.*

To install a system service with Cluster View, follow these steps:

1. On the Executive command line, type **Administrator Cluster View**; then press RETURN.
2. Fill in the command form, as shown in the following example. (See Table 4-1 for a description of parameter fields.)

```
Administrator Cluster View
  [Processor name -XE only]  GP00
  [User name]                Admin
  [User file password]       ####
  [Node name]                ____
  [Old XE run file?]         ____
  [Run file to invoke]       ____
  [Partition size]           ____
```

3. Press GO.
4. If necessary, start the Executive by signing on or exiting the application that is running.
5. Execute the command for the system service you want to install (see Table 9-1).

Installing During System Initialization

In most cases, an SRP runs many system services, so it is convenient to install them during system initialization. Figure 9-6 shows a sample system initialization file for an SRP.

To create a system initialization file for an SRP, you use the Editor application. For more information, see "The Editor" in Section 4, "Using Administrative Tools," and the *CTOS Editor User's Guide*.

Figure 9-6. SRP System Initialization File

```
Job SysInit
;Example SysInit.jcl file for an SRP

FrontPanel 30

GP00
Run [Sys]<Sys>FontService.run,[Sys]<Gps>Font.dbs, 11264
Run [Sys]<Sys>InstallQMgr.run, y, 20
Run [Sys]<Gps>GpsInstall.run

FrontPanel 31

GP01
Run [Sys]<Sys>Net.run, aNode, 59, 2, 75,,16, 1500
Run [Sys]<Sys>NetServer.run, 8, 32, 8
Run [Sys]<Sys>NetAgent.run, 8, 32, 20,,10
Run [Sys]<Sys>MailServer.run
RunNoWait [Sys]<Sys>RUM.run

FrontPanel 32

TP00
ContinueOnError
Run [Sys]<Sys>MEnet.run, 1,,64
CancelOnError
Run [Sys]<Sys>NAC.run

FrontPanel 33

GP00
Run [Sys]<Sys>InstallSeqService.run, [QIC]
RunNoWait [Sys]<Sys>RUM.run

End
```

The following sections describe important considerations for the SRP, such as keyswitch positions and special JCL statements. Be sure that you understand this information before you create system initialization files for your SRP.

Using Keyswitch Files

When an SRP is bootstrapped, the keyswitch on the front panel is set to one of three positions: N, M, or R. That keyswitch position determines which system initialization file is executed after the SRP bootstraps. For more information, see "Keyswitch Positions," in Section 5, "Bootstrapping."

The keyswitch system initialization files are named as follows:

*[Sys]<Sys>SysInit.*k*.jcl*

where *k* corresponds to the keyswitch position.

If a system initialization file for a particular keyswitch does not exist, *[Sys]<Sys>SysInit.jcl* is used instead. No default system initialization file is installed with Standard Software; you must create your own system initialization files.

JCL Statements for SRPs

Table 9-3 lists JCL statements for SRPs. Most are the same as you use for workstations. Special considerations for the processor ID, **RunNoWait** statement, and installation of the Remote User Manager are described below.

JCL syntax is the same as for workstations (see "JCL Syntax," earlier in this section). See the *CTOS Batch Manager II Installation, Configuration, and Programming Guide* for more information about using Batch processing on the SRP.

Table 9-3. JCL Statements for SRPs

Statement	Description
Command	This statement specifies the name of an Executive command. To be executed during system initialization, the command must be present in *[Sys]<Sys>Sys.cmds*.
ContinueOnError	This statement forces continuation of the system initialization file, even if an error occurs. It affects statements that follow it until the end of the file, or until a CancelOnError statement occurs.
CancelOnError	This statement reinstates the CancelOnError condition, which terminates system initialization when an error occurs.
End	This statement defines the end of a Batch job. End your system initialization file with it, as shown in Figure 9-6.
FrontPanel	This statement displays the two-digit number of your choice on the front panel. Use it to mark the progress of the system initialization sequence. This can be helpful for isolating problems.
Job	This statement defines a name for the Batch job. For system initialization, specify **SysInit**, as shown in Figure 9-6.
Run	This statement specifies the name of a run file to be executed.
RunNoWait	This statement instructs the master processor to execute the next statement in the system initialization file immediately.
xPnn	This statement specifies the processor on which to install the system service, where *xPnn* is the four-character processor ID.

Specifying a Processor

On an SRP, the master processor controls execution of the system initialization file. The master processor recognizes IDs for the other processors, so that system services can be installed anywhere on the SRP.

To specify a processor, you type the four-character ID for the processor, for example GP00, FP01, and so on. Statements that follow the processor ID are executed on that processor until another processor is identified; see Figure 9-6 for an example. For a description of processor IDs, see "Processor Boards" in Section 2, "Understanding Hardware."

Installing the Remote User Manager

The Remote User Manager is an optional system service for use on protected mode processors (see "Cluster View" in Section 4, "Using Administrative Tools"). After it is installed, no other system services can be installed on the processor. Therefore, it must be installed properly; otherwise, the system initialization sequence will stop.

On a workstation server or an SRP master processor, the Remote User Manager must be the final system service to be installed during system initialization. Figure 9-6 shows how several system services are initially installed on GP00 (the master processor) of an SRP; then, after all other system services are installed on the other processors, the Remote User Manager (*RUM.run*) is installed on the master processor.

When installing the Remote User Manager on a processor other than the master processor, it must be the last system service installed on that processor. Figure 9-6 shows how the Remote User Manager is installed as the last system service on GP01.

You must use the RunNoWait statement to install the Remote User Manager. If you inadvertently use the Run statement, the Remote User Manager will be installed, but the rest of the system initialization file will not be executed.

Using the RunNoWait JCL Statement

The RunNoWait statement works differently from the other system initialization JCL statements. Although it is not necessary to understand what it does to install the Remote User Manager, you might be interested in the difference between it and the other statements. Also, if you are using any customized system services (those that have been changed from the released versions), you may need to know when to use the RunNoWait statement.

As explained earlier, the master processor controls the system initialization procedure. When it encounters Run or Command statements, it executes them one after another, in the order they appear in the system initialization file. For example, in Figure 9-6, first the Font Service is installed on GP00, then the Queue Manager, and so on. During this procedure, the master processor waits for confirmation that a statement has finished executing before it starts the next statement. In most cases, this method works well and system initialization proceeds in an orderly fashion.

To do its job, however, the Remote User Manager must remain active on the processor on which it is installed. Because it does not finish executing, the master processor does not receive a message to continue with system initialization. Therefore, a special statement, RunNoWait, is required to install the Remote User Manager. When the master processor encounters the RunNoWait statement, it executes it and then immediately executes the next statement in the file. So, system initialization continues, even though the Remote User Manager never finishes executing. If you inadvertently install the Remote User Manager with the Run statement, the system initialization procedure waits indefinitely for the Remote User Manager to finish executing.

The Remote User Manager, it can also be used to install system services on multiple processors simultaneously. Theoretically, this can speed up system initialization, but it can also cause problems if system services must be installed in a particular order. Also, when you use the RunNoWait statement, the master processor attempts to read several files at the same time. If all the files are located on the system volume, the resulting disk activity can actually slow down the system initialization procedure. Therefore, it is usually better to reserve RunNoWait for situations where it is required, such as installing the Remote User Manager.

Common Problems With System Services

Several common errors associated with system services are described in Table 9-4.

Table 9-4. System Service Errors

Error Code	Description
31	No such request This error occurs when a loadable request (as described in Section 8) is not available to a system service. It can happen when a workstation or SRP requires rebooting after an application has been installed or updated. To correct it, try rebooting the system. This error also occurs if loadable requests were not properly merged during software installation. Try repeating the software installation procedure by using the **Installation Manager** command.
33	Service not available This error occurs when you execute a command requiring a system service that has not been installed. For example, before you can use a mouse, you must execute the **Install Mouse Service** command to install Mouse Services on your workstation. See the documentation for the application you are trying to use.
203	No such file This error can occur at the beginning of system initialization if the run file *[Sys]<Sys>Batch.run* is not present on the system (also *[Sys]<Sys>CLI.run* on SRPs only). Use the **Files** command to make sure that it is there. If it is not, copy it from the system utilities distribution media.

Section 10
Accessing Data Throughout the Cluster

Using Disks on the Server

A built-in feature of the CTOS operating system allows cluster workstation users access to disks located on the server. To do so, you simply specify an exclamation point (!) in front of the volume or device name of the disk you want to access, as shown in the following example:

[!d1]<Dept>FileOnServer.txt

Accessing Disks on Cluster Workstations

It is also possible to access files located on workstations throughout the cluster. The Cluster File Access (CFA) system services provide selective access to disks from cluster workstation to cluster workstation. A configuration file on each workstation designates the availability of its disks to other workstations in the cluster.

Note: *Cluster File Access cannot be implemented in clusters with an SRP server.*

Cluster File Access consists of the following system services:

- The CFA Server Service is installed on the server to enable Cluster File Access.
- The CFA File Filter is installed on workstations that will be accessing disks on other cluster workstations.
- The CFA Workstation Agent provides access to disks on the workstation where it is installed.

A cluster workstation does not need to run both the CFA File Filter and the CFA Workstation Agent. For example, a workstation running the CFA File Filter only can have access to other disks without providing access to its own. Only those disks on workstations running the CFA Workstation Agent are accessible throughout the cluster.

Configuring Cluster File Access

Before you install the Cluster File Access system services for the first time, you create a configuration file by using the **CFA Configure** command on the server and on each cluster workstation.

To use the **CFA Configure** command, follow these steps:

1. On the Executive command line, type **CFA Configure**; then press RETURN.
2. In the command form, specify a name for the configuration file. The default is *[d0]<Sys>CFAConfig.sys*.
3. Press GO. The CFA Cluster Workstation Configurator appears, as shown in Figure 10-1.
4. Press the LEFT ARROW or RIGHT ARROW to select an access level for each disk. (See "Protecting Directories," in Section 6, "Implementing System Security," for a description of read and modify access levels.)
5. Press FINISH, then GO, to save the configuration file.

Figure 10-1. CFA Configure Display

CFA Cluster Workstation Configurator				
Disk	Volume Name	Access		
D0	CPG-0	Read	Modify	None
D1	CPG-1	Read	Modify	None
F0	F0	Read	Modify	None

543.10-1

Installing the CFA Server Service

The Cluster File Access Server Service is installed on the server workstation only. To install it, type **Install CFA Server Service** on the Executive command line; then press GO.

Alternatively, you can install the Cluster File Access Server Service during system initialization. To do so, include the following entry in the system initialization JCL file:

```
Run [Sys]<Sys>CfaM.run
```

Installing the CFA Workstation Agent

The Cluster File Access Workstation Agent is installed on each cluster workstation that has disks that will be available to other workstations in the cluster. To install it, follow these steps:

1. On the Executive command line, type **Install CFA Workstation Agent**; then press RETURN.
2. Fill in the command form. Parameter fields are shown below and described in Table 10-1.

```
Install CFA Workstation Agent
  [Config file ([d0]<Sys>CFAConfig.sys)]        ____________
  [Maximum remote clients (4)]                  ____________
```

3. Press GO.

Alternatively, you can install the Cluster File Access Workstation Agent during system initialization. To do so, include the following entry in the system initialization JCL file:

```
Run [Sys]<Sys>CfaWA.run,ConfigFile,RemoteClients
```

Table 10-1. Install CFA Workstation Agent Parameters

Field Name	Description
[Config file ([d0]<Sys>CFAConfig.sys)]	Default: *[d0]<Sys>CFAConfig.sys* Enter the file specification of the configuration file containing the disk access information you want to use. Access levels are assigned to disks with the **CFA Configure** command.
[Maximum remote clients (32)]	Default: 32 Enter the maximum number of applications running elsewhere in the cluster that can access local disks simultaneously, via the CFA Workstation Agent.

Installing the CFA File Filter

The Cluster File Access File Filter is installed on each cluster workstation that will be accessing disks on other cluster workstations. To install it, type **Install CFA File Filter** on the Executive command line; then press **GO**.

Alternatively, you can install the Cluster File Access File Filter during system initialization. To do so, include the following entry in the system initialization JCL file:

```
Run [Sys]<Sys>CfaFF.run
```

Using Cluster File Access

To use a disk configured for Cluster File Access, specify its unique volume name (rather than its device name) in file specifications or command forms, as shown in the following example:

```
Files
  [File list (*)]  [TechPubs]<Sys>*
```

Note: *For all disks in the cluster to be accessible via Cluster File Access, each must have a unique volume name.*

Accessing Floppy Disk Drives

To access a floppy disk, specify its device specification followed by the volume name of the system volume. Separate the floppy disk drive specification and the system volume name with a colon, as shown in the following example:

```
Files
  [File list (*)]  [f0:TechPubs]<Sys>*
```

If the system volume on the workstation you want to access has not been configured for CFA modify access, you will not be able to access its floppy disk drive. In addition, if the system volume has a password, that password may be required to access the floppy disk drive. See the **CFA Configure** command in the *CTOS Executive Reference Manual* for more information.

Displaying CFA Volume Name

To display the volume names of disks available in your cluster, use the **CFA Display Volume Information** command. To do so, type **CFA Display Volume Information** on the Executive command line; then press GO.

See the *CTOS Executive Reference Manual* for more detailed information about the **CFA Display Volume Information** command.

Accessing Additional Resources

As an alternative to Cluster File Access, the BNet Cluster Access Services allow you to access files and other resources throughout the cluster and on other network nodes. BNet Cluster Access is included with CTOS as a separately installable software package. For more information, see the manual and release documentation for it.

Section 11
Adding Hard Disks

What Is a CTOS Volume?

Many systems, both workstations and SRPs, contain more than one hard disk for data storage. When you set up a new system, you initialize the system volume during the software installation procedure. After the system volume has been initialized and you have installed Standard Software, you initialize the other disks on the system. A disk must be initialized to be recognized by the CTOS operating system as a *valid volume.*

This section describes how to initialize disks with the **Format Disk** command. See also the following manuals:

- For information about initializing a system volume during software installation, see the *CTOS System Software Installation Planning Guide.*
- For information about initializing floppy diskettes, see the *CTOS Executive User's Guide.*
- For information about initializing partitioned disks for use with other operating systems, see the *CTOS Executive Reference Manual.*

Workstation Disks

For X-Bus and X-Bus+ workstations, disks are supplied in add-on modules and cartridges. The model of disk you use depends on your workstation and processor type. See the workstation hardware guides for installation instructions.

EISA/ISA-bus workstations are usually equipped with a single disk, most often called a drive.

Workstation Disk Compatibility

Different types of disks are compatible with different workstation models, as described below:

Non-SCSI disk modules	X-Bus workstations only
SCSI disk modules	X-Bus and X-Bus+ workstations
SCSI disk cartridges	X-Bus+ workstations only
IDE drives	EISA/ISA-bus workstations only

Device Names for Workstation Hard Disks

On workstations, hard disk devices are named *dn*, where *n* indicates the placement of the disk drive. Disk device names can change when you add a disk to the workstation. For example, if you add a new disk between two existing disks on an X-Bus workstation, the device name of the third disk will change accordingly. Volume names, however, which you define when you initialize disks, remain the same when physical placement of the disk is changed.

On X-Bus workstations, the first hard disk to the right of the processor is *d0*, the second, *d1*, and so on.

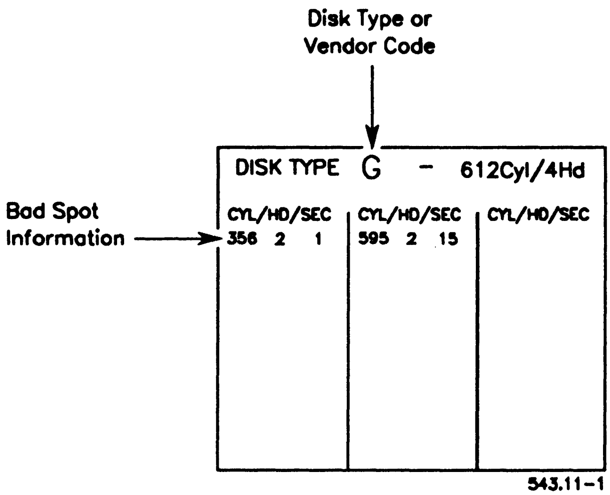
On X-Bus+ workstations, hard disks are numbered *d0* and *d1*, from top to bottom, in the base unit. Disk numbering then continues sequentially in the SCSI expansion unit to the left of the base unit, followed by the X-Bus to the right of the base unit. See the *SuperGen Series 5000 Hardware Installation and Maintenance Guide* for detailed information about device names.

On EISA/ISA-bus workstations, the hard disk device is named *d0*.

Disk Type and Bad Spot Report

Disk modules are labeled on the bottom with a disk type (sometimes called vendor code), as shown in Figure 11-1. In addition, non-SCSI disk modules are labeled with a bad spot report, also shown in Figure 11-1. In situations described later in this section, you may need to use the disk type code and bad spot report.

Figure 11-1. Workstation Disk Type and Bad Spot Report



SRP Disks

Many sizes and types of disks are available for the SRP. Unlike workstation disks, SRP disks are not packaged in modules or cartridges. They can be installed within an SRP cabinet, or they can be external to the system. In some cases, the system administrator is responsible for installing disks; in other cases, disks are installed by a field service engineer. If you need information about installing disks, see the installation guide for your SRP.

SRP Disk Compatibility

On an SRP, different types of disks are compatible with different processors, as described below:

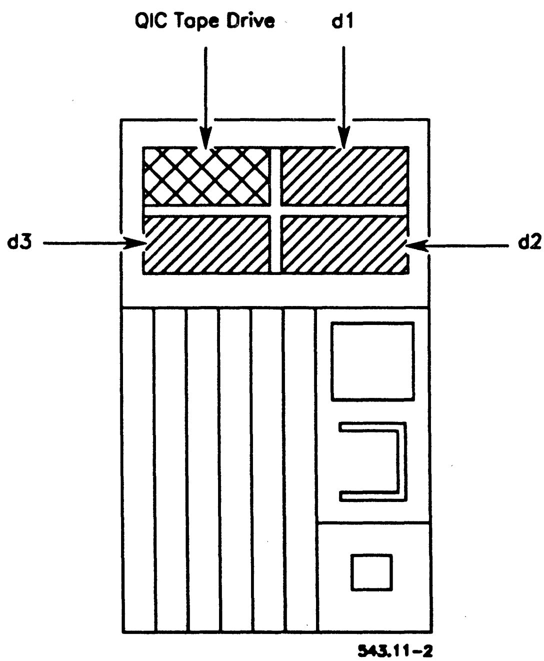
- *SCSI* disks are connected to a General Processor with SCSI Interface (GP+SI).
- Non-SCSI *ST-506* disks are connected to a File Processor (FP).
- Non-SCSI *SMD* disks are connected a Data Processor (DP).

Device Names for SRP Disk Drives

The operating system assigns default device names for disk drives controlled by the master processor only. If the master processor is a GP+SI, default disk device names are *d1* through *d15* (see the *XE-530 Shared Resource Processor Hardware Installation Guide*). On an FP master processor, defaults are *d1* through *d3*. Because a QIC tape drive occupies the first drive slot (as shown in Figure 11-2), there is no disk drive named *d0* on an SRP. If the master processor is a DP, default device names are *s0* through *s5*.

Disks connected to a processor other than the master processor must be named in the operating system configuration file. See the *:MassStorage:* parameter in Section 17, "Configuring Shared Resource Processor Operating Systems," for information about assigning device names.

Figure 11-2. Disks in an SRP Primary Cabinet (Rear View)



Initializing an Unformatted Disk

A new disk that has never been used on a CTOS system must be formatted and initialized before it is recognized as a valid volume. For some disks, that involves a simple procedure, because device-specific parameters and bad spots are already stored on the disk. For other disks, however, you must supply the device template and the bad spot information to the **Format Disk** command.

Table 11-1 lists the device template and bad spot requirements for different models of disks. If a device template or a bad spots file is required, you will need to supply that information in the *[Device template]* or *[Bad spot file]* fields of the **Format Disk** command form. See "Creating a Bad Spots File" and "Using Device Templates," later in this section, for more detailed information.

Table 11-1. Device Template and Bad Spot Requirements for Disks

Disk Model	Device Template Required?	Bad Spots File Required?
SCSI (any)	No	No*
IDE (EISA/ISA workstation)	No	No
Non-SCSI workstation module	Yes	Yes
Non-SCSI ST-506 (SRP)	Yes	Yes
Non-SCSI SMD (SRP)	Yes	Yes
Valid volume (any)	No	No

*A bad spots file is required only to add bad spots that have occurred since the disk was last initialized. See "Creating a Bad Spots File," later in this section.

The following procedure describes how to format and initialize a disk. This procedure works for most workstation disks; however, it does not initialize a system volume and it may not initialize the disk optimally for your purposes. For more detailed information, see "Using Parameter Templates," later in this section.

For information about reinitializing a valid volume, see "Reinitializing a Valid Volume," later in this section. If a disk has been corrupted, see "Reinitializing Corrupted Volumes," later in this section.

> **Caution**
>
> The following procedure destroys all data on the disk.

1. If you are formatting an SRP disk, start a Cluster View session on the processor to which the disk is connected.
2. On the Executive command line, type **Format Disk**; then press **RETURN**.
3. Fill in the command form as shown in the following example; parameter fields are described in Table 11-2.

```
Format Disk
 Device name                       dl
 [Device password]                 ##
 [Current volume password]
 [New volume name]                 Volumel
 [New volume password]             ####
 [Configuration file]
 [Format template]
 [Device template]                 (see Table 11-1)
 [Print file]
 [Overwrite ok?]
 [Bad spot file]                   (see Table 11-1)
 [Recalculate defaults?]
 [CTOS partition size in Mb]
```

4. Press **GO**.

Table 11-2. Format Disk Parameter Fields

Field Name	Description
Device name	Enter the device name of the disk you want to initialize. When formatting floppy diskettes, be sure to use the correct type of diskette for your drive; see the *CTOS Media User's Guide*.
[Device password]	Default: None Enter the device password for the disk you want to initialize. If you are using a prebuilt CTOS operating system, device passwords for hard disks match device names, and floppy drives do not have passwords. If you are using a customized operating system, however, device passwords might be different.
[Current volume password]	Default: Active password If you are reinitializing a disk, enter the volume password currently assigned to the disk. When you are initializing a new disk, there is no volume password to enter here.
[New volume name]	Default: Current volume name Enter a name, up to twelve characters long, to assign to the volume. It can contain letters, numerals, periods, and hyphens. The volume name of each disk within a cluster must be unique.
[New volume password]	Default: None Enter a password, up to twelve characters long, to assign to the volume. It can contain letters, numerals, periods, and hyphens.
[Configuration file]	Default: See below Enter the name of the configuration file containing the format and device templates for the disk. *[Sys]<Sys>FormatDiskConfig.sys* is the default. (See "Using Parameter Templates," later in this section.)

continued

Table 11-2. Format Disk Parameter Fields (cont.)

Field Name	Description
[Format template]	Default: See below Enter the name of a format template for the disk. If you leave this field blank, a default template is used. If a default template is not available, **Format Disk** calculates default values. For more information, see "Using Format Templates," later in this section.
[Device template]	Default: See below This parameter is not required when reinitializing a valid volume. For SCSI disks and IDE drives, leave this field blank. For unformatted non-SCSI disks (other than IDE), enter the name of the device template for the disk. For workstation disks, the template usually corresponds to the single-character disk type labeled on the bottom on the disk drive module. If the disk type is unknown, enter the actual device characteristics in the form of *Cylinders/Heads/Sectors*. If you leave this field blank, **Format Disk** attempts to format the disk with default parameters, which may not be optimal for the disk.
[Print file]	Default: None To record command output in a log file, enter a file specification in this field. The file must be created on a valid volume; that is, you cannot write a log file to the same disk you are formatting.
[Overwrite ok?]	Default: Ask for confirmation If you want to reinitialize a valid volume, enter **Yes**. If you do not want to overwrite a valid volume, enter **No**. If you leave this field blank, you are prompted to confirm or cancel the initialization procedure if the disk is a valid volume.

continued

Table 11-2. Format Disk Parameter Fields (cont.)

Field Name	Description
[Bad spot file]	Default: None Enter the file specification of a file containing bad spot information for the disk. This file is required if bad spot information is not available on the disk or to add bad spots that have occurred since the disk was last initialized. The bad spots file must be created on a disk other than the one you want to initialize. See Table 11-1 and "Creating a Bad Spots File," in this section.
[Recalculate defaults?]	Default: No This field applies to volumes that are already initialized. Enter **Yes** to reinitialize the disk with defaults calculated internally by the **Format Disk** command. If you enter **No** or leave this field blank, the disk is reinitialized with its current parameters. This parameter does not override parameters contained in the specified format template.
[CTOS partition size in Mb]	Default: All available disk space This parameter applies to X-Bus+ and EISA/ISA-bus workstations only. Enter a size for the CTOS partition, in M bytes. If you do not plan to install MS-DOS on the workstation, leave this field blank. See the **Format Disk** command in the *CTOS Executive Reference Manual* for more information about disk partitioning.

Creating a Bad Spots File

A bad spots file is required by the **Format Disk** command if bad spot information is not available on the disk; see Table 11-1 for a list of disks that require a bad spots file.

A bad spots file is also required when you reinitialize a valid volume if you need to add bad spots that have occurred since the last time the disk was initialized. See "Correcting Input/Output (I/O) Errors," later in this section.

If the disk requires a bad spots file, you must create it on a disk other than the one you are initializing before you begin the **Format Disk** command. A floppy diskette can be used for this purpose.

To create a bad spots file, use the Editor application, as described in the following procedure.

1. On the Executive command line, type **Editor**; then press RETURN.
2. Type a file specification for the bad spots file, as shown in the following example:

   ```
   File name(s)  [f0]<Dir>BadSpots.VolName
   ```

3. Press GO.
4. Enter the list of bad spots. (See the sections about bad spot formats, below.)
5. When you have finished entering bad spots, press FINISH, then GO, to save the file.

Bad Spot Formats for Non-SCSI Disks

Enter bad spots in one of the formats shown below, leaving a space between each bad spot entry. See the documentation for the disk to determine which format to use.

c / h / sb / bc

c / h / #sector

c / h / $sector

where

c	Is the cylinder number.
h	Is the head number.
sb	Is the starting byte number.
bc	Is the number of bits in error.
#sector	Is the sector number of a 512-byte sector.
$sector	Is the sector number of a 256-byte sector.

An example bad spots file is shown in Figure 11-3.

Figure 11-3. Bad Spots File for a Non-SCSI Disk

```
981/0/#1 877/2/#0 877/2/#1 975/4/#15 969/5/#4 757/6/#15
943/1/#5 943/1/#6 415/4/#15 1000/5/#0 24/2/#0 425/4/#9
880/5/#5 1005/5/#8 1005/5/#9 32/2/#8 737/4/#12 924/5/#8
924/5/#9 494/6/#1 849/2/#12 849/2/#13 885/4/#4 941/5/#9
941/5/#10 524/6/#7 995/4/#3 995/4/#4
```

Bad Spot Format for SCSI Disks

A bad-spots file for a SCSI disk should contain only those bad spots reported by the **PLog** command since the last time the disk was initialized. Manufacturer's bad spots and those you have previously specified during a volume initialization are already known to the SCSI disk. Specifying them a subsequent time reduces usable disk space.

To enter a bad spot, type its number, as listed in the system error log. For SCSI disks, that is a 2 to 9 digit number, for example, 12339. When entering more than one bad spot, separate each entry with a space.

Reinitializing a Valid Volume

You might need to reinitialize a valid volume before restoring a backup or reusing the disk for another purpose. When you reinitialize a valid volume, device parameters and bad spots are taken from the disk, so you do not need to specify either a device template or a bad spots file in the command form. In addition, you have the choice of reinitializing the disk using current volume parameters, such as number of files on the volume and size of the <*Sys*> directory, or specifying new ones.

The following procedure describes how to reinitialize disks on both workstation and SRPs, using current volume parameters. For information about changing volume parameters, see "Using Format Templates," later in this section.

> **Caution**
>
> The following procedure destroys all data on the disk.

1. Start the **Format Disk** command as described earlier in this section.
2. Fill in the command form as shown in the following example (see Table 11-2 for parameter descriptions).

```
Format Disk
 Device name                      d2
  [Device password]               ##
  [Current volume password]       ####
  [New volume name]               Engr2
  [New volume password]           sluggo
  [Configuration file]
  [Format template]
  [Device template]
  [Print file]
  [Overwrite ok?]                 yes
  [Bad spot file]
  [Recalculate defaults?]
  [CTOS partition size in Mb]
```

3. Press **GO**.

Using Parameter Templates

To initialize a disk, the **Format Disk** command reads parameters contained in templates from a configuration file named *[Sys]<Sys>FormatDiskConfig.sys.* That configuration file contains the following types of templates:

Format templates	Contain volume parameters, such as the maximum number of directories and files on the volume
Device templates	Contain disk hardware parameters

The configuration file supplied with Standard Software contains a variety of format templates. You can select from one of them, or you can create your own format template, as described in this section. If you do not specify a particular format template, the volume is initialized with default values.

The configuration file also contains device templates, which supply hardware parameters for specific models of disks. The configuration file contains device templates for most workstation and SRP disks; therefore, you rarely need to create new device templates. A procedure for creating device templates, however, is included later in this section.

Configuration File Format

Each line of the configuration file is written in the following format:

:Keyword:Value

where

:Keyword:	Identifies a parameter; keywords and the placement of colons must not be changed.
Value	Is a parameter value.

Keyword parameters for both format templates and device templates are described later in this section. Note that the order of format and device templates is not significant. They can be intermixed and can appear in any order within the configuration file.

Using Format Templates

The format templates supplied with Standard Software are listed in Table 11-3. To use a particular format template, specify its name in the *[Format template]* field of the **Format Disk** command form. A sample format template is shown in Figure 11-4. Instructions for creating or modifying a format template are included below.

Figure 11-4. Format Template

```
:FormatTemplate:WSSysVolume
:MaxFilesOnVolume:
:PrimaryFileHeadersOnly?:
:MaxDirectories:
:MaxFilesInSysDirectory:
:PasswordEncryption?:
:Debug?:
:SysDirectoryPassword:
:ProtectSysDirectory?:
:SystemImageSize:768
:CrashFileSize:2048
:SystemLogFileSize:48
:SuppressFormat?:No
:SurfaceTestsIfUnformatted:4
:SurfaceTestsIfFormatted:1
:OldCTOSFormat?:No
```

Table 11-3. Format Disk Standard Format Templates

Template Name	Characteristics
Defaults (if no format template is specified)	
FloppyDefault	Floppy diskette primary file headers only
DiskDefault	System volume without password protection
Floppy Volumes	
FloppyArchive	Floppy diskette suitable for backups
FloppyData	Floppy diskette with approximately 60 file headers
FloppyProtect	Floppy diskette with password protection and a write-protected *<Sys>* directory
X-Bus+ and EISA/ISA System Volumes	
SGenSysVolume	X-Bus+ or EISA/ISA system volume without password protection
SGenSysVolumeProtect	X-Bus+ or EISA/ISA system volume with password protection and a write-protected *<Sys>* directory
SGenSysVolumeEncrypt	X-Bus+ or EISA/ISA system volume with encrypted password protection and a write-protected <Sys> directory
SGenSavePartition	X-Bus+ or EISA/ISA system volume with an existing non-CTOS partition that you do not want to destroy
SGenSavePartitionProtect	X-Bus+ or EISA/ISA system volume with password protection, a write-protected *<Sys>* directory, and an existing non-CTOS partition that you do not want to destroy
SGenSavePartitionEncrypt	X-Bus+ or EISA/ISA system volume with encrypted password protection, a write-protected *<Sys>* directory, and an existing non-CTOS partition that you do not want to destroy

continued

Table 3-3. Format Disk Standard Format Templates (cont.)

Template Name	Characteristics
X-Bus Workstation System Volumes	
WSSysVolume	Workstation system volume without password protection
WSSysVolumeProtect	Workstation system volume with password protection and a write-protected *<Sys>* directory
WSSysVolumeEncrypt	Workstation system volume with encrypted password protection and a write-protected *<Sys>* directory
SRP System Volumes	
SrpSCSISysVolume	SRP SCSI system volume without password protection
SrpSCSISysVolumeProtect	SRP SCSI system volume with password protection and a write-protected *<Sys>* directory
SrpSCSISysVolumeEncrypt	SRP SCSI system volume with encrypted password protection and a write-protected *<Sys>* directory
SrpSMDSysVolume	SRP SMD system volume without password protection
SrpSMDSysVolumeProtect	SRP SMD system volume with password protection and a write-protected *<Sys>* directory
SrpSMDSysVolumeEncrypt	SRP SMD system volume with encrypted password protection and a write-protected *<Sys>* directory
SrpWinSysVolume	SRP ST-506 system volume without password protection
SrpWinSysVolumeProtect	SRP ST-506 system volume with password protection and a write-protected *<Sys>* directory
SrpWinSysVolumeEncrypt	SRP ST-506 system volume with encrypted password protection and a write-protected *<Sys>* directory

continued

Table 3-3. Format Disk Standard Format Templates (cont.)

Template Name	Characteristics
Data Storage Volumes	
DataVolume	Data storage only (a non-system volume) without password protection (workstation or SRP)
SmallDataVolume	Data storage only (a non-system volume) without password protection and approximately 500 file headers (workstation or SRP)
SrpDataVolume	Data storage only (a non-system volume) without password protection (SRP only)
Memory Disks	
MemDisk1	1-megabyte SRP memory disk
MemDisk3	3-megabytes SRP memory disk

Adding a Format Template

If none of the standard format templates is suitable, you can add one of your own. To do so, use the Editor application, as described in the following procedure:

1. On the Executive command line, type **Editor**; then press **RETURN**.
2. Type **[Sys]<Sys>FormatDiskConfig.sys**, as shown below; then press **GO**.

   ```
   [File name(s)]  [Sys]<Sys>FormatDiskConfig.sys
   ```

   (To add a format template for an SRP disk, specify **[!Sys]<Sys>FormatDiskConfig.sys**.)
3. Move the cursor the end of the file.
4. If necessary, press **RETURN** to move the cursor to a new line.
5. Type the keyword and value for each parameter (parameters are described below).
6. When you have finished editing, press **FINISH**, then **GO**, to exit the Editor and save the file.

See the *CTOS Editor User's Guide* for more detailed information about using the Editor.

Format Template Parameters

The following parameters are included in format templates. The *:FormatTemplate:* keyword must be the first parameter in a format template, and you must assign it a unique value. The other keywords can appear in any order. If a keyword is omitted or its value left blank, a default value is used. If **Yes** is specified in the *[Recalculate defaults?]* field in the command form, defaults are determined as if the disk is unformatted.

:FormatTemplate:

This must be the first keyword in each format template and a unique value must be specified. Specify a unique name for the format template.

:MaxFilesOnVolume:

Default for valid volume: Current value
Default for unformatted disk: See below

Specify the maximum number of files to be stored on the volume. The maximum number of files is based on twenty-five sectors per file; therefore, the number you specify here may be adjusted by **Format Disk** according to the size of the disk.

To specify an exact number, precede this value with a pound sign (#), for example, #5000. When the value is preceded by a pound sign, **Format Disk** attempts to create the exact number of file headers specified; in cases where secondary file headers are requested, however, **Format Disk** creates additional file headers. If the number of files requested seems too large for the size of the disk, **Format Disk** reduces the number of files to a reasonable value.

If you do not specify a value, **Format Disk** calculates a value based on the size of the disk.

:PrimaryFileHeadersOnly?:

Default: No

Enter **Yes** if you want only primary file headers on the volume. This option is usually used for floppy diskettes only. If you enter **No** or leave this field blank, secondary file headers are created. These are needed to recover data from a corrupted volume.

:MaxDirectories:

Default for valid volume: Current value
Default for unformatted disk: See below

Specify the maximum number of directories for the volume. If you do not specify a value, **Format Disk** calculates a value based on the size of the disk.

:MaxFilesInSysDirectory:

Default for valid volume: Current value
Default for unformatted disk: See below

Specify the maximum number of files to be stored in the <*Sys*> directory. For system volumes, 1000 to 2500 files are recommended. If you do not specify a value, **Format Disk** calculates a value based on the size of the disk.

:PasswordEncryption?:

Default: No

Enter **Yes** if you want the password to be encrypted. This provides security against sophisticated users who might be able to "peek" at the volume password. If you encrypt the password, though, keep careful records, because it cannot be deciphered. Enter **No** if you do not want the password to be encrypted.

:Debug?:

Default: No

Enter **Yes** to display an F for each track that is formatted and a T for each track that is surface tested. Enter **No** to suppress this information.

:SysDirectoryPassword:

Default for valid volume: Current value
Default for unformatted disk: None

Specify a password, up to twelve characters long, to assign to the <*Sys*> directory.

:ProtectSysDirectory?:

Default: No

Enter **Yes** to set the protection level of <*Sys*> to 5, which prevents users from changing or adding files without a password. (See "Protecting Directories" in Section 6, "Implementing System Security," for more information about protection levels.) Enter **No** to set <*Sys*> directory protection to 15, which does not require a password to change or create files.

:SystemImageSize:

Default for valid volume: Current value
Default for unformatted disk: 0

For nonsystem volumes, specify **0**. For system volumes, specify the number of sectors for *[Sys]<Sys>SysImage.sys*, the operating system file. The size of this file cannot be changed after you have initialized the disk. For prebuilt operating systems, 768 is the recommended value. If you are using a customized operating system, you may need a larger value.

:CrashFileSize:

Default for valid volume: Current value
Default for unformatted disk: 0

For nonsystem volumes, specify **0**. For real-mode system volumes, specify **2048**. For protected-mode system volumes, specify the number of sectors required for *[Sys]<Sys>CrashDump.sys*, the crash dump file. To calculate that number, multiply the amount of processor memory by two.

On protected mode systems, other than EISA/ISA-bus workstations, you can conserve disk space on *[Sys]*, specify **2048** to receive the first megabyte of the memory dump. You can then create an extended crash dump file on another disk to receive the entire memory dump. See "Collecting a Crash Dump" in Section 20, "Troubleshooting."

:SystemLogFileSize:

Default for valid volume: Current value
Default for unformatted disk: 0

For nonsystem volumes, specify **0**. For system volumes, enter a number of sectors for *[Sys]<Sys>Log.sys*, the system log file. To track system problems accurately, 48 sectors is recommended. If your system volume is small, however, you can specify a smaller value (for example, 10 or 20 sectors).

:SuppressFormat?:

Default: No

Enter **No** if the disk has never been formatted or if diagnostics have been run on it. If you are reinitializing a valid volume, enter **Yes**; this decreases the time required to initialize a disk. If you enter **Yes** and the disk is not a valid volume, the **Format Disk** command fails with an input/output (I/O) error (301).

:SurfaceTestsIfUnformatted:

Default: 4

Specify the number of surface tests to run on an unformatted disk (a new disk or after disk diagnostics). It is not unusual for a disk to have bad spots; however, it is important that they be detected. When a bad spot is encountered during surface testing, no data will be stored on it.

:SurfaceTestsIfFormatted:

Default: 1

Specify the number of surface tests to be run on a formatted disk. At least one surface test is recommended.

:OldCTOSFormat?:

Default: No

Enter **No** to format the disk such that it describes a logical block address (LBA) device. (For information about LBA devices, see the *CTOS Procedural Interface Reference Manual*). Such a disk is formatted optimally for its device characteristics and is for use with 3.3 CTOS I and II or 3.0 CTOS/XE operating systems (and subsequent higher versions) only. Such a disk is not usable on systems running an earlier version of the operating system.

If you enter **Yes**, the disk is formatted to be backward-compatible with earlier versions of the operating system; that is, versions earlier than 3.3 CTOS I or II and 3.0 CTOS/XE. With SCSI disks, however, such backward compatibility may result in disks that are not formatted to full capacity and that run more slowly than is optimal. For floppy disks, it is recommended that you specify **Yes** to this parameter.

:Verify?:

Default: No

Enter **Yes** to verify that the disk has been formatted and initialized correctly. Enter **No** to bypass the verification operation.

:SuppressDefaultScsiPages?:

Default: No

Enter **Yes** to use the current SCSI page settings. If you enter **No** or leave this field blank, default SCSI page parameters are used.

:SuppressVolumeStructures?:

Default: No

Enter **Yes** if you do not want CTOS volume structures to be created on this disk. This reserves the entire disk for future use as a non-CTOS partition. (After the disk has been formatted with the **Format Disk** command, use the MS-DOS **FDISK** command to create and activate a DOS partition; then use the MS-DOS **FORMAT** command to create the DOS file system. See your MS-DOS documentation for information about **FDISK** and **FORMAT** commands.)

Using Device Templates

The device templates supplied with Standard Software are listed in Table 11-4. To use a particular device template, specify its name in the *[Device template]* field of the **Format Disk** command form. A sample device template is shown in Figure 11-5, and instructions for creating a device template are included later in this section.

For SCSI disks and IDE drives, do not specify a device template; **Format Disk** obtains device parameters directly from the hardware. When formatting another type of disks or a corrupted volume, however, you may need to specify a device type. See Table 11-1, earlier in this section.

If a device template is required and you do not specify one, or if the one you specify does not exist, **Format Disk** uses the device template named "Unknown." This device template might not contain optimal parameters for the disk, or it might cause the **Format Disk** command to fail. In such a case, you must add a device template to the configuration file for the **Format Disk** command.

Figure 11-5. Device Template

```
:DeviceTemplate:Fujitsu80
:CylindersPerDisk:589
:TracksPerCylinder:7
:SectorsPerTrack:32
:BytesPerSector:512
:Removable?:No
:SeekStepRate:14
```

Adding a Device Template

You add a device template to *FormatDiskConfig.sys* in the same way you add a format template. See "Adding a Format Template," earlier in this section.

Table 11-4. Format Disk Standard Device Templates

Manufacturer and Size of Disk	Device Template Name
Workstation disk vendor codes	A to Z and A1 to Z1
Atasi, 46 megabytes	Atasi46
Ball, 100 megabytes	Ball100
Control Data, 300 megabytes	CDC300
Control Data, 340 megabytes	CDC340
Control Data, 675 megabytes	CDC675
Fujitsu, 80 megabytes	Fujitsu80
Hitachi, 51 megabytes	Hitachi51
Hitachi, 85 megabytes	Hitachi85
Maxtor, 53 megabytes	Maxtor53
Maxtor, 143 megabytes	Maxtor143
Memorex, 166 megabytes	Memorex166
Micropolis, 52 megabytes	Micropolis52
Micropolis, 85 megabytes	Micropolis85
Nortel, 350 megabytes	Nortel350
Toshiba, 85 megabytes	Toshiba85
Memory disk, 1 megabyte	MemDisk1
Memory disk, 3 megabytes	MemDisk3
Regular capacity floppy disk	FloppyType
High-capacity floppy disk	FloppyTypeHiCap

Device Template Parameters

Keywords and parameter values for device templates are listed below. They are included here in case you need to create a new device template. *Do not change parameters in the device templates supplied with Standard Software.* To determine parameter values for new device templates, see the documentation for the disk.

:DeviceTemplate:

This must be the first keyword in each device template. For its value, enter a unique device-template name.

:CylindersPerDisk:

Default: 306

Enter the number of cylinders to be allocated on the disk.

:TracksPerCylinder:

Default: 4

Enter the number of tracks to be formatted for each cylinder.

:SectorsPerTrack:

Default: 16

Enter the number of sectors to be formatted for each track. On SRP disks, you can increase usable disk space by changing this number to 17. This is the only parameter that should be changed in any of the device templates supplied with Standard Software.

:BytesPerSector:

Default: 512

Enter the number of bytes to be allocated in each sector.

:WritePreCompCyl:

Default: 0

Enter the number of the cylinder at which write-precompensation begins. Write-precompensation reduces I/O errors on inner cylinders of the disk. The default, 0, specifies that no write-precompensation and is used on disks that do not provide this feature.

:SeekStepRate:

Default: 0

Enter a number to represent the time interval, in microseconds, between successive step pulses when a seek command is issued. (See the Western Digital WD-2010 documentation.) Common values are 0 (the default) and 14.

:UtilizeEcc?:

Default: See below

Enter **Yes** to use error checking and correction (ECC) format.

The default is set according to the capabilities of the hardware. If you specify **Yes**, but ECC capability is not present, the disk is formatted in cyclic redundancy check (CRC) mode. Specify **No** if ECC capability is present but you do not want to use it.

:SpiralFactor:

Default: 0

This field applies to SMD disks only; it has no effect on other disks. Specify the sector offset from track to track on the disk.

:Removable?:

Default: No

Enter **Yes** if the disk is a removable storage medium.

:HiCapacityFloppy?:

Default: No

Enter **Yes** to format high-capacity floppy diskettes on a high-capacity drive. Enter **No** to format regular-capacity floppy diskettes on a high-capacity drive. Be sure to use the correct type of diskette for your drive; see the *CTOS Media User's Guide*. This field has no effect on regular-capacity floppy drives.

Correcting Input/Output (I/O) Errors

When you receive a disk from the manufacturer, it has been surface-tested and known bad spots are accounted for. Known bad spots are normal and do not cause loss of data if they are properly specified when the disk is initialized.

As you use the disk, however, additional bad spots may occur; these are a serious problem and can cause loss of data. New bad spots are signaled by I/O errors (Error 301) when you try to create or modify files. When I/O errors occur, you need to make bad spots known to the disk so that they will not cause loss of data.

As bad spots occur, they are reported in the system error log, which you can view or print with the **PLog** command. To make bad spots known to a disk, you enter them into a bad spots file and then reinitialize the disk, using the bad spots file as a parameter value. Procedures for creating a bad spots file and reinitializing the disk are described below.

Note: *If new bad spots occur frequently, back up the disk and replace it with a new one as soon as possible.*

Specifying Bad Spots

As bad spots occur, they are reported in the system error log, which you can view or print with the **PLog** command. You need to make those bad spots known to the disk so that data will not be stored on them. Bad spots can be defined when a disk is reinitialized, or on non-SCSI disks, with the **Bad Sector** command.

To define bad spots when you reinitialize a disk, you enter them into a text file that you create with the Editor application. You then specify the name of that file in the *[Bad spot file]* field of the **Format Disk** command form. See "Creating a Bad Spots File," earlier in this section, for a step-by-step procedure.

For information about using the **Bad Sector** command on non-SCSI disks (including IDE drives), see the documentation for it in the *CTOS Executive Reference Manual*.

Caution

A bad spots file for a SCSI disk should contain only those bad spots reported by **PLog** since the last time the disk was initialized. Respecifying bad spots that are already known to the disk reduces the amount of usable disk space.

Running Surface Tests

After creating a bad spots file, you must reinitialize the disk to make the bad spots known to the disk. When you do so, run at least four surface tests to detect additional bad spots.

To run more than one surface test (which is the default), modify the *:SurfaceTestsIfFormatted:* parameter in the appropriate format template. The following example shows you how to specify four surface tests:

:SurfaceTestsIfFormatted:4

Reinitializing the Disk

Caution

The following procedure destroys all data on the disk.

Before you reinitialize the disk, back it up so that you can restore directories and files to it later. See Section 13, "Backing Up and Restoring Data."

The following example shows you how to specify a format template and a bad spots file in the **Format Disk** command form:

```
Format Disk
 Device name                          d1
 [Device password]                    ##
 [Current volume password]            ####
 [New volume name]                    Volume1
 [New volume password]                ####
 [Configuration file]
 [Format template]                    DataVolume
 [Device template]
 [Print file]
 [Overwrite ok?]
 [Bad spot file]                      [d1]<Sys>BdSpt.d1
 [Recalculate defaults?]
 [CTOS partition size in Mb]
```

For more detailed information, see the step-by-step procedures provided earlier in this section.

Reinitializing Corrupted Volumes

A *corrupted volume* is not recognized by the operating system. The following conditions can cause a corrupted volume:

- The volume home block (VHB) becomes unreadable, usually because of bad spots on the disk.
- The **Format Disk** command is abnormally terminated, possibly because of equipment failure, or because you pressed **ACTION-FINISH**.
- Diagnostics have been run on the disk. (See the documentation for your diagnostics software product.)

In many cases, you can recover data from a corrupted volume with the **Volume Archive** command.

To reinitialize a corrupted volume, see the procedures earlier in this section for entering a device template and bad spots file in the **Format Disk** command form.

Optimizing Disk Space

In some cases, you can increase available disk space by adjusting format template parameters. When you are just beginning to use a system, it is difficult to know exactly what your needs will be. Therefore, the default parameters supplied in *[Sys]<Sys>FormatDiskConfig.sys* are a good place to start. They are based on averages of typical file lengths and number of files per directory.

As you use a disk, however, a discrepancy between available disk space (sectors) and file headers shows that the disk is not initialized optimally. To determine this, use the **Volume Status** command, as described below:

1. On the Executive command line, type **Volume Status**; then press **RETURN**.
2. Type the volume or device name of the disk you want to check; then press **GO**.

An example of the top portion of the **Volume Status** display is shown in Figure 11-6. The column labeled "Used" (far right) shows the percentage of sectors and file headers currently used on the disk. Sectors refer to disk storage space; file headers refer to slots for file names.

When a disk is initialized optimally, the percentage of used sectors and file headers should be similar. As shown in Figure 11-6, many more sectors than file headers have been used. You can optimize disk space by reinitializing the volume.

Figure 11-6. Volume Status Display

```
Status of volume TFS-1     Device dl

Initialized       Mar 13, 1992  3:14 PM
Last Modified     Jul 27, 1992  1:16 PM

                  Unused   Reserved   CTOS Volume Total   Used

Sectors            38138          0              131072    70%
File headers        2519          0                3719    32%
```

Caution

Reinitializing a volume destroys all data on it. Therefore, be sure to back up the volume before you reinitialize it. See Section 13, "Backing Up and Restoring Data," for backup and restoration procedures.

The volume shown in Figure 11-6 was initialized with too many file headers. Because file headers occupy disk space, they are taking up space that could be used for other data. The disk will be full of data long before all the file headers are used. Therefore, to optimize disk space, you could reinitialize the disk with fewer file headers. To do so, adjust the value for the *:MaxFilesOnVolume:* format template parameter.

The opposite situation occurs when you do not allocate enough file headers. In that case, you can run out of file headers while free sectors remain on the disk. This results in the following error when you try to create a file:

```
No free file headers (Error 225)
```

Another common misuse of disk space is allocating a large number of file headers in the *<Sys>* directory of a nonsystem volume. On nonsystem volumes, *<Sys>* usually contains only the mandatory files that are created when the disk is initialized. Therefore, you can specify a small number if files, for example, 25, in *<Sys>*, to make space available for other directories. To do so, adjust the value for the following format template:

:MaxFilesInSysDirectory:

Section 12
Using Tape Drives

Different Types of Tape Drives

Your workstation or SRP may be equipped with one of the following types of tape drives:

SCSI quarter-inch cartridge (QIC)	Can be found on both workstation or SRPs. They use QIC tape cartridges, which can store up to 150M bytes of data per 600-foot tape.
Non-SCSI QIC	Can also be found on both workstations or SRPs. They use QIC tape cartridges, which can store up to 60M bytes of data per 600-foot tape.
Digital data storage (DDS)	Are available as workstation modules or Series 5000 cartridges. They use DDS cartridge tapes, which can store up to 1.3G bytes (1300M bytes) of data per 60-meter tape.
Half-inch	Are available on SRPs only. They are connected to a Storage Processor and use reel-to-reel tapes. The amount of data stored on a half-inch tape varies according to tape length and recording density.

See the *CTOS Media User's Guide* for information about inserting tapes, write-protecting tapes, and tape storage specifications.

What Kind of Tapes to Use

Approved tapes for QIC and DDS tape drives are listed in Table 12-1. For half-inch tape drives, see the owner's manual for information about approved tapes.

Table 12-1. Approved Tapes for Data Storage

Drive Type	Tape Model Number	Supplier
Non-SCSI QIC	DC-600A	3M
	DC-615A	3M
	10000FTPI	Data Electronics
SCSI QIC	HD 600XTD	Unisys
DDS	DG-60M	SONY
Half-inch	See the owner's manual for your tape drive	

Hardware and Software Requirements

Before you can use a tape drive, you must install the Sequential Access Service on the workstation or SRP to which it is connected. You can do so during system initialization or with the Executive.

For non-SCSI workstation QIC drives, install the XBIF Service before you install the Sequential Access Service. See "Where to Install System Services" in Section 9, "Installing System Services."

On SRPs, you must identify tape drives in the operating system configuration file before you install the Sequential Access Service. See the *:SequentialStorage:* parameter in Section 17, "Configuring Shared Resource Processor Operating Systems." Then, install the Sequential Access Service according to the following guidelines. In most cases, installing it on the processor to which the tape drive is connected yields the best performance.

- For non-SCSI QIC drives, install the Sequential Access Service on any processor except a DP or SP.
- For half-inch tape drives, install the Sequential Access Service on the DP or SP to which the tape drive is connected.
- For SCSI QIC drives, install the Sequential Access Service on any processor, although best performance is obtained from installing it on the GP to which the tape drive is connected.

Installing the Sequential Access Service

Before you install the Sequential Access Service, exit any Context Manager or Remote User Manager sessions and, if you will be installing on an SRP, start an Administrator Cluster View session. Then, follow these steps:

1. On the Executive command line, type **Install Sequential Access Service**; then press RETURN to display the command form.
2. Fill in the command form, as shown in the following example. Parameter fields are described in Table 12-2.

```
Install Sequential Access Service
  [Device(s) ([QIC])]                                 QIC QIC1
  [Buffer pool size in Kb (64)]                       ________
  [QIC interface slot - SRP/XE only (77)]             ________
```

3. Press GO.

Alternatively, to install the Sequential Access Service during system initialization, add the following entry to the system initialization JCL file:

Run [Sys]<Sys>InstallSeqService.run, *Device*, *BuffSize*, *QicSlot*

Note that you can specify multiple tape drives by using the following syntax with the *Device* parameter:

(Device1, Device2, ..., DeviceN)

See "JCL Syntax" in Section 9, "Installing System Services," for more detailed information.

Table 12-2. Sequential Access Service Parameters

Field Name	Description
[Device(s) ([QIC])]	**Workstations.** Assign a device name for each tape drive. Defaults are *QIC* for the first tape drive and *Seq0* for the second tape drive. **SRPs.** Enter the device name assigned to each tape drive in the operating system configuration file. There is no default. See the *:SequentialStorage:* parameter in Section 17, "Configuring Shared Resource Processor Operating Systems."
[Buffer pool size in Kb (64)]	Default: 64 Enter the number of K bytes for the buffer pool for the Sequential Access Service. This buffer pool is used by all tape drives on the system. On SRPs, the buffer pool consumes memory on the processor on which you install the Sequential Access Service.
[QIC interface slot - SRP/XE only (77)]	Default: 77h This parameter applies to SRPs only. Enter the slot number, as a hexadecimal number, to which the QIC tape drive is connected.

Configuring a Tape Drive

By default, the Sequential Access Service configures tape drives to provide good performance in a variety of settings. For optimal performance, however, you may need to modify the configuration file for your tape drive. See the **Configure Sequential Access Device** command in the *CTOS Executive Reference Manual.*

Preparing Tapes for Use

Before a new QIC tape is used, it must be retensioned. If it is not correctly tensioned, it can break or malfunction in the tape drive. To retension a QIC tape, follow these steps:

1. Insert the tape cartridge into the tape drive (see the *CTOS Media User's Guide*).
2. On the Executive command line, type **Tape Retension**; then press GO.

If a tape already contains data, you can erase it before you reuse it. Erasing a tape also retensions it. Note, however, that it can take up to two hours to erase a DDS cartridge. To erase a tape, follow these steps:

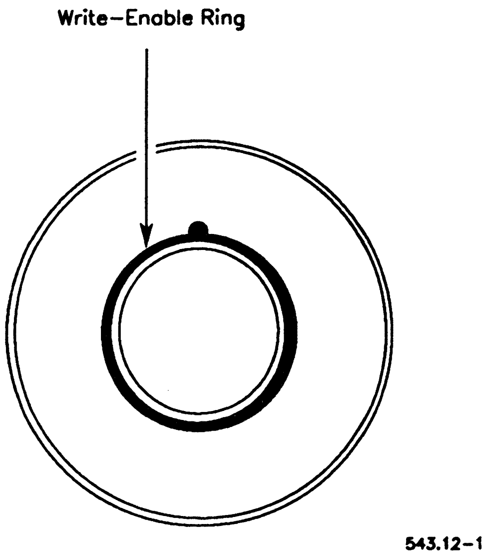
1. Insert the tape into the tape drive (see the *CTOS Media User's Guide* or the owner's manual for your tape drive).
2. On the Executive command line, type **Tape Erase**; then press GO.

Write-Enabling Half-Inch Tapes

To write-enable a half-inch tape, place a write-enable ring into the groove on the backside of the tape reel, as shown in Figure 12-1. When the ring is in place, the tape is write-enabled; when it is removed, the tape is write-protected.

See the *CTOS Media User's Guide* for information about write-enabling and write-protecting QIC and DDS tapes.

Figure 12-1. Write-Enable Ring on a Half-Inch Tape



Section 13
Backing Up and Restoring Data

Performing Routine Backups

Backups protect the data on your system against accidental loss or damage. During a backup, files are copied to an *archive dataset* on tapes or floppy diskettes. From the archive dataset, you can restore an entire volume or a single file. As a system administrator, you are most likely responsible for performing or supervising backups of the system.

This section describes how to back up data to tape. For information about backing up to floppy diskettes, see the *CTOS Executive User's Guide*.

The importance of regular backups cannot be overemphasized. After you have become familiar with the procedures, they take only minutes of your time and could save hours of work in trying to recreate the files on your system.

Although you can set up any backup schedule that is convenient for you, the following method is used by many system administrators.

1. Once a week, perform a complete backup of each disk on the system.
2. On the days between complete backups, perform an incremental backup of each disk. An incremental backup archives only those files that were created or changed since a particular date or time.

Procedures for using this method are included later in this section.

Cleaning Up Disks Before Backups

Before you begin a backup, delete files you no longer need. For example, some volumes contain a "dollar-sign" directory named <*$000*>, which contains temporary files created by applications. Delete files in that directory before you perform a backup. Doing so increases disk space and shortens backup time.

In addition, the following duplicate and temporary files are created by other applications. These files can be deleted periodically, at your discretion.

-old files	File names ending with *-old* are created by word- processing applications. They contain previous versions of files that have been edited.
.ts files	File names ending with *.ts* are typescript files produced by text-processing applications. They are required to recover a document after a system crash, power failure, or abnormal termination of the application.
.lst files	File names ending with *.lst* contain error listings created by compilers or assemblers.

Performing a Complete Volume Backup

This section describes how to back up data to tapes. Before you begin, make sure that the Sequential Access Service is properly installed and that you have enough tapes to complete the backup. See Section 12, "Using Tape Drives," for more information.

In addition, if you are backing up a disk on the server, you may want to disable the cluster so that users cannot make changes to the disk. To do so, use the **Disable Cluster** command, as described in the *CTOS Executive Reference Manual.*

To perform a complete volume backup, follow these steps:

1. Insert a tape into the tape drive (see the installation guide for your tape drive).
2. Retension or erase the tape (see "Preparing Tapes for Use," in Section 12, "Using Tape Drives").
3. On the Executive command line, type **Volume Archive**; then press **RETURN**.

4. Fill in the command form as shown in the following example. Parameter fields are described in Table 13-1. (See also the **Volume Archive** command in the *CTOS Executive Reference Manual* for more detailed parameter descriptions.)

```
Volume Archive
 Volume or device name                     d0
 [Volume or device password]               ####
 [Incremental from]
 [Suppress backup?]
 [Suppress verification?]
 [Archive dataset ([QIC])]                 [QIC]
 [Delete existing archive dataset?]
 [Print file]
 [Display structures?]
 [Verify write?]
 [Suppress user interaction?]
```

5. Press GO to begin the backup.

After each tape is complete, label it with the following information:

- Type of backup (for example, "full backup" or "complete backup")
- Volume or device name
- Date of backup
- Sequential tape number (for example, if it takes more than one tape to back up a disk, label them 1 of 2, 2 of 2, and so on)

Table 13-1. Volume Archive Parameters

Field Name	Description
Volume or device name	Enter the volume or device name of the disk you want to back up.
[Volume or device password]	Default: Active password To back up a valid volume (as is the case for routine backups), enter the volume password. To back up a corrupted volume, enter the device password. (See also "Backing Up a Corrupted Volume," later in this section.)
[Incremental from]	Default: See below To back up files created or modified since a particular date, enter a date, or a date and time, as shown in the following example: 2/4/90 8:30 AM If you leave this field blank, all files on the disk are backed up. If you enter a date only, midnight is the default time.
[Suppress backup?]	Default: No If you leave this field blank, a backup takes place when you execute **Volume Archive**. However, you can also use the command to verify the integrity of your file system. To run a file system verification only, enter **Yes**.
[Suppress verification?]	Default: No If you leave this field blank, the file system is verified after a backup. To suppress file system verification, which normally occurs after a backup, enter **Yes**. (This saves time if you are planning to reinitialize the disk immediately; however, it is not recommended for routine daily backups.)
[Archive dataset (QIC)]	Default: *[QIC]* Enter the name of the tape device you want to use. For example, **[QIC]** for a workstation tape drive or, for an SRP tape drive, the name assigned to it in the operating system configuration file.

continued

Table 13-1. Volume Archive Parameters (cont.)

Field Name	Description
[Delete existing archive dataset?]	Default: Prompt user If you leave this field blank, you are informed if the tape already contains a backup. To overwrite the tape, enter **Yes**. To prevent overwriting, enter **No**.
[Print file]	Default: Screen only If you leave this field blank, command output is written to the screen only. To write command output to a file or printer, in addition to the screen, enter a printer or file specification.
[Display structures?]	Default: No If you enter **Yes**, a detailed analysis of the volume control structures is displayed. This option is generally used only by programmers for file system error analysis.
[Verify write?]	Default: No This field applies only when you archive data to disk backup media, such as floppy diskettes. It does not apply to tape backups. Enter **Yes** if you want to verify that the data written to the archive file matches the data you are backing up. If data does not match, an I/O error (301) is reported.
[Suppress user interaction?]	Default: No If you enter **Yes**, the command exits with an error when user interaction is required. This parameter is most frequently used in software installation scripts. If you leave this field blank, **Volume Archive** pauses when user interaction is required.

Performing an Incremental Backup

Many system administrators perform a complete volume backup once a week, and an incremental backup on the days in between. This saves time by archiving only files that were created or changed since the date you specify.

You can use either of the following incremental backup methods. Procedures for restoring data are included later in this section.

Method 1

Specify the date of the last complete volume backup as the incremental date. This allows you to reuse the same tape day after day, for incremental backups. Each subsequent day, however, more and more files are backed up, so it takes longer to perform the backup.

Should you need to restore data, restore the complete volume backup first, then the incremental backup.

Method 2

Specify the current date as the incremental date. This decreases backup time, because only files created or changed on that day are backed up. However, you need more tapes, because each incremental backup must be performed on a separate tape.

Should you need to restore data, restore the complete volume backup first, then each incremental backup in chronological order, starting with the oldest.

To perform an incremental backup, follow the procedure described below for each disk on the system.

1. Insert a prepared tape into the tape drive (see "Preparing Tapes for Use" in Section 12, "Using Tape Drives"). *Do not use the same tape you used for the complete volume backup.*
2. Start the **Volume Archive** command with the Executive, as described in the preceding procedure.

3. Enter the incremental date you want to use (see "Method 1" and "Method 2," above), as shown in the following example:

```
Volume Archive
 Volume or device name                    d0
 [Volume or device password]              ####
 [Incremental from]                       2/4/91
 [Suppress backup?]
 [Suppress verification?]
 [Archive dataset ([QIC])]                [QIC]
 [Delete existing archive file?]
 [Print file]
 [Display structures?]
 [Verify write?]
 [Suppress user interaction?]
```

4. Press GO to begin the backup.

Label the tapes as described for complete volume backups, earlier in this section.

Performing Backups With Cluster View

You can perform a backup of an SRP disk from any workstation in the cluster. However, you can perform the backup more quickly if you execute it via Cluster View. If you do not use Cluster View, data travels to the cluster workstation for processing before the backup is written to tape. With Cluster View, however, data is processed on the SRP.

To perform backups with Cluster View, you must use a tape drive located on the SRP. Remember that you don't have access to local disks or tapes when using Cluster View.

To perform a backup with Cluster View, follow these steps:

1. Start a Cluster View session on the SRP (see "Cluster View" in Section 4, "Using Administrative Tools").
2. Follow the procedures for performing complete volume backups and incremental backups, earlier in this section.

Restoring Backups

You can restore all or part of an archive dataset. In a lot of cases, restoring data is a routine procedure that does not involve damage to a disk or the file system. The most common routine situations are listed below, and procedures for dealing with them are included in this section.

- You want to transfer all data to a different (perhaps larger) disk; see "Restoring a Complete Backup," below.
- You receive error messages about a disk being *fragmented*. This happens after a disk has been used for a while. It means that small blank areas, where files have been deleted, can no longer be used to store data. See "Restoring a Complete Backup," below.
- A user accidentally deletes an important file or directory; see "Restoring Portions of an Archive Dataset," below.

In other cases, you restore data because something is wrong with the disk or file system. The following situations can signal serious problems that require investigation; see "Recovering a Corrupted Volume," later in this section.

- A disk suddenly becomes unusable.
- After a weekly backup, you are informed that the file system has become corrupted.
- I/O errors (error 301) occur on a disk. (These are recorded in the system log file, which you can read with the **PLog** command; see Section 20, "Troubleshooting").

Restoring a Complete Backup

Before you restore a complete backup to the disk from which it was taken (as you would to correct fragmentation), you must reinitialize the disk. See Section 11, "Adding Hard Disks," for information about initializing disks.

> **Caution**
>
> Reinitializing destroys all data on the disk. Be sure that you have a current backup before you reinitialize a valid volume.

To restore a complete backup, follow these steps:

1. Insert the tape you want to restore into the tape drive.
2. On the Executive command line, type **Restore Archive**; then press **RETURN**.
3. Fill in the command form as shown in the following example. Parameter fields are described in Table 13-2. (See also the **Restore Archive** command in the *CTOS Executive Reference Manual* for more detailed parameter descriptions.)

```
Restore Archive
  [Archive dataset ([QIC])]           [QIC]
  [File list from (<*>*)]             <*>*
  [File list to (<*>*)]               [d1]<*>*
  [Overwrite ok?]                     ____________
  [Confirm each?]                     ____________
  [Sequence number]                   ____________
  [Merge with existing file?]         ____________
  [List files only?]                  ____________
  [Print file]                        ____________
  [Suppress user interaction?]        ____________
```

4. Press **GO**.

Table 13-2. Restore Archive Parameters

Field Name	Description
[Archive dataset ([QIC])]	Default: *[QIC]* Enter the tape device name containing the backup you want to restore.
[File list from (<>*)]*	Default: All files If you leave this field blank, all files in the archive dataset are restored. To restore selected files, enter the directory specification and file name. (See also "Restoring Portions of an Archive Dataset," later in this section.)
[File list to (<>*)]*	Default: See below If you leave this field blank, files are restored to their original directories on the default volume. To restore files to a volume other than the default, enter the volume or device name before the directory and file specification, for example, *[d1]<*>**.
[Overwrite ok?]	Default: Prompt user If you leave this field blank, you are informed when a file from the archive dataset already exists on the disk. To overwrite the disk file, enter **Yes**. To prevent overwriting, enter **No**.
[Confirm each?]	Default: No If you leave this field blank, all specified files are restored without confirmation. To be prompted to confirm restoration of each file, enter **Yes**.
[Sequence number]	Default: 1 If you leave this field blank, restoration begins with the first tape of a multiple-tape set. To specify a tape other than the first, enter its sequential number in the set (for example, **2** or **3**).

continued

Table 13-2. Restore Archive Parameters (cont.)

Field Name	Description
[Merge with existing file?]	Default: No If you leave this field blank, and the tape contains unreadable data, corresponding disk data is overwritten and destroyed. To prevent corrupted tape data from overwriting existing disk data, enter **Yes**. (See also "Recovering a Corrupted Volume," later in this section.)
[List files only?]	Default: No To list the contents of an archive dataset without restoring it, enter **Yes**. If you leave this field blank, restoration takes place in the usual manner.
[Print file]	Default: Screen only To write command output to a file or printer (in addition to the screen) enter a printer or file specification. If you leave this field blank, command output is written to the screen only.
[Suppress user interaction?]	Default: No If you enter **Yes**, **Restore Archive** exits with an error when user interaction is required. This parameter is most frequently used in software installation scripts. If you leave this field blank, **Restore Archive** pauses when user interaction is required.

Restoring Portions of an Archive Dataset

If necessary, you can restore a single file, an entire directory, or a group of files. You might need to do this, for example, if a user accidentally deletes an important file.

The following procedure describes how to restore a single file. Use wild-card characters or an at-file to restore groups of files or a directory (see “* and ? Wild-Card Characters,” in the *CTOS Executive User's Guide* or “Wild Card Characters” in the *CTOS Executive Reference Manual*).

1. Insert the archive tape into the tape drive.
2. Start the **Restore Archive** command with the Executive (see the preceding procedure, “Restoring a Complete Backup”).
3. Fill in the form, as shown in the following example. (See Table 13-2 for parameter field descriptions.)

```
Restore Archive
  [Archive dataset ([QIC])]          [QIC]
  [File list from (<*>*)]            <Dir>FleNme
  [File list to (<*>*)]              [d0]<Dir>FleNme
  [Overwrite ok?]                    ____________________
  [Confirm each?]                    ____________________
  [Sequence number]                  ____________________
  [Merge with existing file?]        ____________________
  [List files only?]                 ____________________
  [Print file]                       ____________________
  [Suppress user interaction?]       ____________________
```

4. Press GO. You are informed when the file has been restored.

Recovering a Corrupted Volume

At some point during your career as a system administrator, you will most likely encounter a *corrupted volume.* This means that data on the disk contains unreadable or unintelligible errors. Such errors can be caused by both hardware and software problems.

This section contains the following information to help you identify and correct corrupted volumes:

- How to identify a corrupted volume
- How to back up a corrupted volume
- How to troubleshoot disk problems
- How to restore data

Identifying a Corrupted Volume

Signs and symptoms of a corrupted volume vary, depending on the type and extent of damage. In some cases, the disk itself is physically damaged; in unusual cases, software errors can garble the file system.

To check for physical damage to the disk, use the **PLog** command to read the system error log. It records new bad spots that result from accidents or normal wear and tear on the disk. See Section 20, "Troubleshooting," for detailed information about the **PLog** command.

To check for software errors, a file system verification takes place when you perform volume backups. If problems are detected, you are informed that volume has been corrupted. You can also run a file system verification independently of a backup, at any time you suspect disk problems; see the **Volume Archive** command in the *CTOS Executive Reference Manual.*

In addition, error messages can occur when you execute commands. The following error message signals potential problems:

```
I/O error (Error 301)
```

Do not ignore I/O error messages. Back up the disk immediately; then repair or replace the disk.

Backing Up a Corrupted Volume

Fortunately, most corrupted volumes can be almost completely recovered with the backup and restoration procedures in this section. If, however, the volume home block (VHB) is corrupted, the operating system will not recognize the disk as a valid CTOS volume. If this should happen, you will be prompted to supply the following information during the backup procedure:

Device password	Enter the device password assigned to the disk. For prebuilt operating systems, the device password matches the device name.
Disk type	For workstation modules, enter the disk-type vendor code. For SRP disks, enter the device template listed in Table 11-4.

See Section 11, "Adding Hard Disks," for more information about device passwords and disk types. See the **Volume Archive** command in the *CTOS Executive Reference Manual* for more information about backing up corrupted volumes.

Troubleshooting Disk Problems

After you have backed up a corrupted disk, it is important to correct the problem before you restore data to it. If possible, install a spare disk and restore the backup to it while the original disk is being repaired. If this is not possible, try to prevent users from continuing to use a questionable disk by physically removing it, or warn them that they may lose data. Then, perform the following troubleshooting steps before putting the disk back into regular use.

Caution

The following procedure destroys all data on the disk.

1. Run hardware diagnostics on the disk, or follow your usual procedure to obtain service.
2. If the disk passes diagnostics, reinitialize the disk (see Section 11, "Adding Hard Disks"); be sure to run at least eight surface tests on the disk. If possible, initialize the disk overnight and run a large number of surface tests.

Restoring Data

After you have repaired or replaced a corrupted volume, you can usually recover most of the data by restoring your routine backups, as well as the backup you obtained after problems occurred. To recover the maximum amount of data, restore the archive tapes as follows:

1. Restore the most current complete volume backup you performed before disk problems occurred. (Use the procedure for restoring a complete backup, earlier in this section.)
2. Restore the incremental backups you performed after the most recent complete volume backup. (Use the procedure for restoring a backup, earlier in this section.)
3. Restore the backup you obtained from the corrupted volume. Follow the procedure for restoring a complete backup, however, specify **Yes** in the *[Merge with existing file?]* field, as shown below. This protects files that have been corrupted since you performed the routine backups.

```
Restore Archive
  [Archive dataset ([QIC])]        [QIC]
  [File list from (<*>*)]          <*>*
  [File list to (<*>*)]            [d0]<*>*
  [Overwrite ok?]                  yes
  [Confirm each?]
  [Sequence number]
  [Merge with existing file?]      yes
  [List files only?]
  [Print file]
  [Suppress user interaction?]
```

Section 14
Using a File System Cache

What Is a File System Cache?

A file system cache is an area of memory where files are stored dynamically, as they are used. When a file is cached, it is accessed from memory, rather than from disk, which increases the speed at which files are retrieved and stored.

The file system cache is not a RAM disk, although it can be used as one. In most cases, you do not specify particular files to be stored in the cache. Files circulate in and out of the cache as they are used. Therefore, the file system cache increases performance for many frequently used files, rather than a preselected few.

File system caches can be allocated on protected-mode processors only.

How Caching Works

When you access a file, sectors from it are read into the file system cache. Then, as you continue to use those sectors, they are accessed from the cache, rather than from disk. This provides faster access to the file.

Changes to cached sectors are immediately written to disk, as well as to the cache. Therefore, this is a *write-through cache*, which is equally efficient for both read and write operations.

Because the file system cache is limited in size, however, it can accommodate only a certain number of disk sectors. Therefore, sectors that are not being used can be "bumped" from the cache. The most recently used sectors remain in the cache. You can also tag certain files to be locked into or excluded from the cache. Those options are described later in this section.

A file system cache is depicted in Figure 14-1. Notice that a portion of the cache contains cache control structures. Therefore, not all the memory you allocate to the cache is available for storing files.

If the cache becomes full, you may notice a decrease in file access speed. You can check cache usage with the **Cache Status** command; see the *CTOS Executive Reference Manual* for information about using that command. By monitoring the file system cache, you can determine whether a larger cache could benefit your system.

Configuring Cache Memory

On protected-mode workstations, a default cache of 500K bytes is allocated. On SRPs, a default of cache of 500K bytes is allocated on the master processor only, if it is a General Processor.

Memory to be allocated for the cache is defined by the following entries in the operating system configuration file (that is, *[Sys]<Sys>Config.sys* on workstations or *[Sys]<Sys>SrpConfig.sys* on SRPs):

```
:FileCacheService:      (BlockSize = 4096
                        BlockCount = 128
                        MinWorkingSetBlockCount = 16)
```

The default, as shown above, allocates a 500K byte cache, as follows.

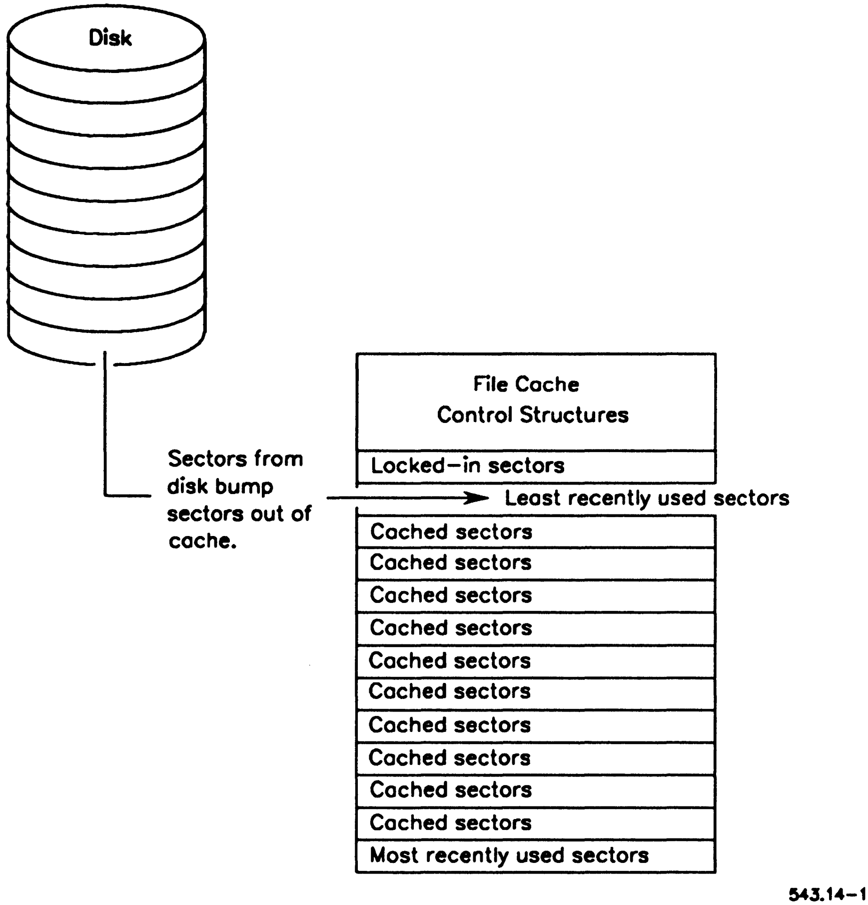
4096	(size of each block in bytes)
x 128	(number of blocks)
524,288	(total number of bytes allocated)

To change the size of the cache, increase or decrease the value of *BlockCount*. The following example allocates a 2M-byte cache.

```
:FileCacheService:      (BlockSize = 4096
                        BlockCount = 512
                        MinWorkingSetBlockCount = 16)
```

In most cases, you do not need to change the values of *BlockSize* or *MinWorkingSetBlockCount*. For more information, see the *:FileCacheService:* parameter for workstations in Section 16, "Configuring Workstation Operating Systems," and for SRPs in Section 17, "Configuring Shared Resource Processor Operating Systems."

Figure 14-1. File System Cache



Setting File Attributes for Caching

When a workstation or SRP boots, a default caching attribute is assigned to all files with the *:FileCacheDefaultEnable:* parameter in the operating system configuration file. If the default is set to **Yes**, all files are enabled for future caching until caching is specifically disabled on a per-file basis. Likewise, if the default is set to **No**, no file is enabled for caching until caching is specifically enabled, again on a per-file basis. The procedure for setting the caching attribute on individual files is described below.

Disabling Files for Caching

If the *:FileCacheDefaultEnable:* parameter is set to **Yes**, you can disable caching on individual files with the **Disable Caching** command, as shown in the following example:

```
Disable Caching
 File list     FileName________________________________
 [Print file]  ________________________________________
```

To reenable caching, use the **Enable Caching** command; see the *CTOS Executive Reference Manual.*

You can also disable caching on selected files during system initialization. To do so, use the JCL Command statement, and specify the names of the files to be excluded. Enclose file names in parentheses and separate them with commas, as shown in the following example:

```
Command Disable Caching, (FileName1, FileName2)
```

Or, you can specify an at-file containing the list of files to be excluded. To do so, surround the at-file name in single quotation marks ('), as shown in the following example:

```
Command Disable Caching, '@FileName'
```

The **Disable Caching** command does not remove a currently cached file from the file system cache. To do that, execute the **Unlock Cache** command on the file after you disable caching. See the *CTOS Executive Reference Manual* for information about using that command.

Enabling Files for Caching

If the *:FileCacheDefaultEnable:* parameter is set to **No**, you can enable caching on selected files with the **Enable Caching** command, as shown in the following example:

```
Enable Caching
 File list      FileName
 [Print file]
```

You can also enable caching on selected files during system initialization. To do so, use the JCL Command statement, and specify the names of the files to be included. Enclose file names in parentheses and separate them with commas, as shown in the following example:

```
Command Enable Caching, (FileName1, FileName2)
```

Or, you can specify an at-file containing the list of files to be included. To do so, surround the at-file name in single quotation marks ('), as shown in the following example:

```
Command Enable Caching, '@FileName'
```

Using the Cache as a RAM Disk

To emulate a RAM disk function, you can lock files into the cache. Locked-in files remain in the cache until they are specifically unlocked and removed. Remember, however, that the purpose of the cache is to improve general performance of all file-access operations; by locking many files into the cache, you could defeat that purpose.

To lock files into the cache during system initialization, use the JCL Command statement, as shown in the following example:

```
Command Lock In Cache, (FileName1, FileName2)
```

Or, specify an at-file containing the list of files to lock in, as shown in the following example:

```
Command Lock In Cache, '@FileName'
```

Alternatively, you can lock files into the cache with the **Lock In Cache** command, as shown in the following example:

```
Lock In Cache
 File list      FileName
 [Print file]
```

To unlock a file and remove it from the cache, use the **Unlock Cache** command; see the *CTOS Executive Reference Manual.*

Caching Files From the Server

The file system cache, as described above, is configured for local caching only. For a cluster workstation to cache files located on the server, the *agent cache* must be enabled on the cluster workstation. To enable the agent cache, specify **Yes** to the following *Config.sys* parameter:

```
:AgentCacheDefaultEnable: Yes
```

When the agent cache is enabled, files that you open on the server are cached in memory on the cluster workstation. This greatly improves access speed to such files.

Note: *The agent cache parameter must be enabled for caching to take place on a diskless workstation.*

The caching utilities described earlier in this section work the same with the agent cache as they do with local file system caching. Those cache utilities consist of the **Cache Status, Disable Caching, Enable Caching, Lock In Cache,** and **Unlock Cache** commands.

Remote Caching on an SRP

The term *remote caching* applies to SRPs only and refers to a disk using a cache on a processor different from the one to which it is physically connected.

Remote caching parameters are contained in the operating system configuration file, *[Sys]<Sys>SrpConfig.sys*. To configure a remote cache, add the following entries after the *:FileCacheService:* entry, in the order shown below:

:RemoteCachePool:	Designates the local cache, or sets up a new cache, for use by remote client processors.
:RemoteCacheClient:	Identifies a particular remote processor as a caching client.

Examples of remote caching parameters are included below. See also the remote caching parameters in Section 17, "Configuring Shared Resource Processor Operating Systems."

Sharing a Cache

For real-mode processors to share a cache on a protected-mode processor, they must be named as remote-cache clients. The following example shows FP00 and FP01 sharing the cache on GP00:

```
:Processor:  GP00
   :FileCacheService:             (BlockSize = 4096,
                                   BlockCount = 512)
   :RemoteCachePool:              (Name = Local)
   :RemoteCacheClient:            (Name = FP00,
                                   Pool = Local)
   :RemoteCacheClient:            (Name = FP01,
                                   Pool = Local)
```

First, the *:FileCacheService:* parameter allocates a 2M-byte cache (4096 bytes times 512 equals 2,097,152 bytes). Then, *:RemoteCachePool:* designates the local cache on GP00 to be accessible to remote processors. (Note that the local cache on a protected-mode processor is always named "Local".) Finally, *:RemoteCacheClient:* names FP00 and FP01 as remote clients of the cache.

Configuring a Remote Cache

In most cases, caching is most efficient when a real-mode processor shares a local cache on a protected-mode processor. To adjust for the increased workload imposed by cache sharing, you can make the cache size larger than the default, as shown in the preceding example. It is possible, however, to set up a remote cache for exclusive use by a specific remote processor. To do this, you allocate a cache in addition to the local cache on a protected mode processor. The following example shows a local cache on GP00 and a remote cache for use by FP00.

```
:Processor:  GP00
   :FileCacheService:           (BlockSize = 4096,
                                 BlockCount = 128)
   :RemoteCachePool:            (Name = c0
                                 BlockSize = 4096,
                                 BlockCount = 128)
   :RemoteCacheClient:          (Name = FP00,
                                 Pool = c0)
```

Note that the *:FileCacheService:* parameters allocate a 500K-byte local cache on GP00. Because the local cache will not be accessed by remote processors, it does not need to be identified with the *:RemoteCachePool:* parameter. The *:RemoteCachePool:* parameter, however, sets up another 500K-byte cache named *c0*. Finally, the *:RemoteCacheClient:* parameter names FP00 as the remote client for the cache named *c0*.

Remote cache names can be anything other than a reserved device name or "Local", which is reserved for local caches. See the *CTOS Executive Reference Manual* for a list of reserved device names. The cache name *c0*, as shown in the preceding example, is a suggestion based on CTOS device-naming conventions.

Section 15
Optimizing System Performance

Configuring Context Manager

Context Manager allows you to start and execute multiple applications concurrently on a single workstation. The way in which Context Manager is configured can significantly affect the performance of a workstation. The following sections describe Context Manager optiminzation on both and partitioned memory and virtual memory systems. See also Section 9, "Installing System Services," for information about the paging service and partitioned memory.

For more detailed information about Context Manager and its associated utilities, see the *CTOS Context / Window Manager Installation and Configuration Guide*.

Optimizing Partitioned Memory

Partitioned memory management is used by real-mode operating systems and protected-mode operating systems without virtual memory. Partitions are contiguous areas of memory that are kept separate from one another. Processing that takes place in one partition does not interfere with processing in another partition. When you use Context Manager on a partitioned memory system, each application is started in its own partition; the size and type of each partition is specified in the Context Manager configuration file.

Calculating Partition Sizes

To determine minimum and maximum partition sizes for an application, you can use the **Version** command, as follows. In the *File list* field, specify the name of the application run file and in the *[Details?]* field, specify **Yes**, as shown in the following example:

```
Version
 File list                      [Sys]<Sys>RunFileName.run
 [Details?]                     Yes
 [Print file]
 [Run file mode]
 [Min array, data, code]
 [Max array, data, code]
 [Keyboard ID]
 [Alternate NLS Style?]
```

The information displayed by the **Version** command includes the following fields:

Static memory size

Minimum dynamic memory size

Maximum dynamic memory size

To determine the smallest partition size, in bytes, in which the application can run, add the values for static memory and minimum dynamic memory. You would specify a relatively small partition size when it is important to keep a group of applications running without swapping in a small amount of memory.

To determine the largest partition size that will benefit performance, add the values for static memory and maximum dynamic memory. A partition that is larger than the static and maximum dynamic memory added together reserves memory that will never be allocated by the application and, therefore, consumes memory that could be available to other partitions.

Using Static Partitions

Static partitions are created to the size specified in the Context Manager configuration file. On real-mode workstations, Context Manager creates static partitions only. On protected-mode workstations without virtual memory, the partition type (either static or variable) is specified in the Context Manager configuration file.

The use of static partitions ensures that a particular application runs in the size of partition you specify. This is useful for applications that require a large amount of memory to perform optimally. For example, using fonts or integrating objects with OFIS Document Designer is more efficient in a large partition.

To specify static partitions in the Context Manager configuration file, type the partition size number, as shown in the following example:

```
Memory required     500 KBytes
```

Using Variable Partitions

On protected-mode workstations, you can specify either a static or a variable partition for each application. In the Context Manager configuration file, you specify a maximum size for a variable partition. Then, when Context Manager starts the application, it creates the variable partition to the maximum size specified and, if necessary, swaps other applications to obtain the contiguous memory it needs. As other applications are swapped back in, however, the size of the variable partition may be reduced so that a maximum number of applications can run simultaneously, without swapping.

To specify a variable partition in the Context Manager configuration file, add a less-than sign (<) in front the partition-size number, as shown in the following example:

```
Memory required     <500 KBytes
```

Optimal sizes for variable partitions are determined by your requirements. For example, you can minimize swapping by keeping maximum partition sizes small enough so that applications can be started with a minimum of swapping. Or, you can maximize performance of individual applications by specifying larger variable partition sizes.

Code Sharing

Certain applications, such as the Executive, are designed to share code among partitions, if more than one instance of the application is running concurrently. To take advantage of code sharing, such applications must be started in variable, rather than static, partitions.

When partitions for code-sharing applications are viewed with the **Partition Status** command, the full partition size, including the shared code, is displayed for each instance of the application. For that reason, the total amount of memory displayed in the *Allocated* fields of the main display may exceed the actual amount of physical memory available on the processor. To determine how much memory is actually being consumed by code-sharing partitions, the amount of memory displayed in the *Code* field on the Details display should be subtracted from all but one of the code-sharing partitions.

Shared code remains available to each code-sharing partition, regardless of the order in which they were started, until the last code-sharing partition is exited.

Optimizing Virtual Memory

The partition sizes specified in the Context Manager configuration file are not relevant to virtual memory operating systems. Virtual memory operating systems use *demand-paged memory management*, which loads applications into memory in 4K byte *pages* and allocates memory in 4K bytes *frames*, as it is needed.

When the total number of pages allocated to applications and system services is greater than the number of frames available, physical memory is said to be *oversubscribed*. In addition, to bring pages into physical memory, the paging service is said to process *page faults*.

Oversubscription of memory and page faults are normal and acceptable conditions. They make it possible for more applications than can be loaded into physical memory to run concurrently, usually with good performance. Problems can arise, however, if there is not enough phycial memory for all the programs to run effectively.

Suppressing Global Policy

In most cases, on virtual memory systems, swapping with Context Manager takes place as usual or is faster than on a partitioned memory system. In rare cases, however, a particular application consumes all available memory on the workstation. In such a case, an unusually long delay occurs when switching from that application back to Context Manager, and again when starting a new application.

If you experience long delays when switching or starting applications, you may be able to improve performance by suppressing the global paging policy. To do so, specify **Yes** to the *:SuppressGlobalPolicy:* parameter in *[Sys]<Sys>Config.sys*, the operating system configuration file. When the global paging policy is suppressed, memory for applications started with Context Manager is allocated according to the partition sizes specified in the Context Manager configuration file.

Correcting Global Thrashing

A condition called *global thrashing* can degrade performance on virtual memory systems. This occurs when the number of frames required by concurrently running programs exceeds the amount of physical memory. To check for global thrashing, add together the amount of memory displayed for each program in the *Working set (cur)* field of the **Partition Status** Details display. If that amount exceeds the total amount of memory, as displayed in the *Total RAM* field of the **Partition Status** main display, you can improve performance by reducing the number of programs that are running concurrently.

For more information about the **Partition Status** command, see the *CTOS Executive Reference Manual.* For more information about memory management and the paging service, see the *CTOS Operating System Concepts Manual.*

Allocating Buffers

When you are using certain applications, you may need to allocate memory for buffers. *Buffers* are portions of memory reserved for temporary storage of data. For example, when you perform tape operations, data is written to a buffer before it is transferred to the tape.

Many programs and applications use buffers. In most cases, buffers are allocated automatically and are not of concern to the system administrator. The following applications, however, may require additional or different sized buffers, depending on your system.

ISAM

Indexed Sequential Access Method (ISAM) buffers are allocated during installation of the ISAM Service, using information in the ISAM configuration file. ISAM uses buffers to store two types of data:

- ISAM data records
- Index file nodes

A different size and number of buffers can be allocated for each purpose.

Buffers for ISAM data records must be one sector larger than the minimum number of sectors required for one record.

Buffers for index file nodes must be at least as large as an index node (see your ISAM documentation). If only one ISAM file will be opened, allocate one more index buffer than the number of simultaneous users. If more than one ISAM file will be opened, try allocating approximately 10 percent more buffers.

Electronic Mail

If your mail center is communicating with many other mail centers, the performance of electronic mail can be improved by increasing the number of sector buffers in the mail center configuration file. A formula for calculating the optimal number of sector buffers is supplied in the release documentation for your electronic mail software product.

Allocating Queues

A *queue* is a portion of memory for storing a list of files or jobs awaiting processing. For example, when you initiate a spooled print request, it is stored in a queue until the file is printed. The Queue Manager maintains queues in an orderly manner, so that jobs can be processed in a particular order. The order is either on a first-come, first-served basis, or according to user-specified priorities.

There are two types of queues:

- Dynamic queues, which are created when they are needed by an application
- Static queues, which are created when the Queue Manager is installed

The type of queue is determined entirely by the application that is using it. This information is available in the release documentation and manuals for the application. When configuring an application that requires queues, you will need to know what type of queues to install.

The following sections describe both dynamic and static queues. See the *CTOS Generic Print System Administration Guide* and the **Install Queue Manager** command in the *CTOS Executive Reference Manual* for information.

Dynamic Queues

Dynamic queues are created when they are needed by an application. For example, when using the Generic Print System (GPS), queues are not created until a printer is installed. This means that you can add and remove queues without deinstalling the Queue Manager.

In addition, dynamic queues can take advantage of the Queue Manager's cache, which reads data from memory, rather than from a disk. By allocating more dynamic queues than you need, you increase the size of the cache. Although this requires more memory, it can enhance performance of the Queue Manager.

You allocate dynamic queues when you install the Queue Manager. Use the following list to determine the number of dynamic queues to allocate:

- Allocate one queue for each spooled printer.
- Allocate one queue for each application that uses the Queue Manager, such as background Batch.
- Allocate two additional queues to allow for expansion.

Optionally, if the system has enough memory, you can increase the size of the cache by allocating up to double the number of queues you need.

Static Queues

Static queues are created when the Queue Manager is installed, regardless of whether they are eventually used. Information for allocating static queues is contained in the *[Sys]<Sys>Queue.index* file. To add a static queue, you must deinstall the Queue Manager, edit *Queue.index*, and then reinstall the Queue Manager.

When you install the Queue Manager, space is automatically allocated for the static queues defined in *Queue.index*. See the *CTOS Generic Print System Administration Guide* for more detailed information about the *Queue.index* file.

Optimizing Use of Disk Space

On many systems, the majority of disk input and output activity is performed on the system volume, while other disks remain idle. On a workstation, you can easily observe this situation. If the drive light is always on, the disk is being overworked.

In many cases, you can increase efficiency by distributing frequently used files among disks, as described in the following sections.

Moving the "Scratch" Volume

Most applications create temporary files as you use them. These files are stored on a scratch volume, which is designated by the operating system. By default, both workstation and SRP operating systems define *[Sys]* as the scratch volume. On protected-mode workstations and SRPs, however, you can reduce activity on *[Sys]* by redefining the scratch volume.

To redefine the scratch volume, edit the following field in the operating system configuration file:

:ScratchVolumeName:

When you change the scratch volume, you must also create a dollar-sign directory on the new scratch volume. Name this directory *<$000>* and make it large enough to hold 750 files.

On SRPs, you can specify a scratch volume for each processor. Use this feature to assign one scratch volume per cabinet, to avoid interprocessor data transfers.

See Section 16, "Configuring Workstation Operating Systems," and Section 17, "Configuring Shared Resource Processor Operating Systems," for more detailed information about operating system configuration files.

Moving Applications

Run files for applications are installed, by default, on the system volume. To reduce the workload on *[Sys]*, you can move some of the application run files to different disks. For example, Document Designer places a particularly heavy load on the system volume. By moving its run file to a different disk, you can speed operations that require disk activity.

When you move application run files, be sure to update user files, Context Manager configuration files, and command files that contain references to the application. Also be aware that some applications, such as Art Designer and Extended Multiplan, do not work correctly if they are not located in *[Sys]<Sys>*. See the application release documentation for that type of information.

Optimizing Memory Usage on the SRP

In many cases, you can increase the efficiency of an SRP by redistributing processing among processors. You can usually do this by rearranging system services and peripheral hardware or by reallocating memory blocks.

The following sections describe some common methods of maximizing SRP processing power.

Isolating Disk-Intensive Applications

Some applications, such as ISAM or electronic mail, are disk intensive. This means that they frequently read and write disk data. To improve the performance of disk-intensive applications, isolate the appropriate system services on a dedicated disk-controlling processor (not the master processor). Then, copy all related directories and files to a disk controlled by the dedicated processor.

A disk-controlling processor is a General Processor with SCSI Interface, a File Processor, or a Data Processor.

Moving Communications Services

If communications gateways are installed on a Cluster Processor, try moving them to a General Processor. If your SRP is not equipped with a General Processor, install communications gateways on a Terminal Processor or the Cluster Processor supporting the fewest cluster workstations.

Avoiding Interprocessor Data Transfers

Whenever data is transferred between SRP processors, processing speed is reduced. This is particularly true for system services and commands executed via Cluster View. The following example demonstrates how interprocessor data transfers slow down processing.

Suppose your QIC tape drive is controlled by GP00 and you want to back up a disk also controlled by that processor. If you start a backup via Cluster View on GP00, data is processed and written to the tape by GP00. If, however, you start Cluster View on a different processor, for example, CP00, two interprocessor data transfers take place. First, data is read from the disk by GP00 and transferred to CP00 for processing; then it is transferred from CP00 back to GP00 to be written to tape.

System services that perform many disk operations perform best when installed on processors that control disks. For example, the Mail Service can be installed on any processor; installing it on a General Processor with SCSI Interface, a File Processor, or a Data Processor is the best choice, however, because disk operations are the most time consuming. In addition, the mail center should be configured to use a disk that is controlled by the processor on which the Mail Service is installed.

Adjusting Memory Blocks

Memory blocks are used for interprocessor data transfers on workstations and SRPs. They affect performance as follows:

- Too few memory blocks reduce the speed of interprocessor data transfers.
- Too many memory blocks consume memory that could be made available for processing.

This section describes different types of memory blocks and provides guidelines for adjusting them. You allocate memory blocks in the operating system configuration file, that is *[Sys]<Sys>Config.sys* on workstations and *[Sys]<Sys>SrpConfig.sys* on SRPs. For more information, see the parameters for X-blocks in Section 16, "Configuring Workstation Operating Systems," and the parameters for W-blocks, X-blocks, Y-blocks, and Z-blocks in Section 17, "Configuring Shared Resource Processor Operating Systems."

What Are Blocks?

Blocks are small portions of memory that are allocated by the operating system. There are several types of blocks:

- *X-blocks* are used during communications between the server and cluster workstations.
- *W-*, *Y-*, and *Z-blocks* are used during data transfers between SRP processors.

Adjusting the size and number of blocks can enhance system performance in certain situations.

X-Blocks

The prebuilt operating systems allocate enough X-blocks for most cluster environments. In the following situations, however, you may need to allocate additional X-blocks:

- When a processor is supporting many diskless workstations
- When a processor is running a customized operating system that supports more workstations than the prebuilt version

You can display information about X-block usage on the server. To do so, start the **Cluster Status** command. Then press **F5** (Blocks).

The *Number XBlock waits* field is incremented by one each time cluster processing must wait for a free X-block. If the system frequently waits for X-blocks, performance could most likely be improved by increasing the number of X-blocks.

To allocate more X-blocks, modify the following entries in the operating system configuration file:

`:Xblocks: (Number = n)`	(default 5)
`:XblocksSmall: (Number = n)`	(default 28)

where *n* is the number of X-blocks to be allocated. The operating system uses small X-blocks whenever possible. If data is too large for a small X-block, or if all small X-blocks are being used, the operating system uses a large X-block.

W-, Y-, and Z-Blocks

W-, Y-, and Z-blocks are used for interprocessor data transfers on SRPs only. As with X-blocks, the prebuilt SRP operating systems allocate enough W-, Y-, and Z-blocks for many environments. If, however, a processor is running many system services, you may need to allocate more blocks to optimize performance.

The operating system transfers data in the smallest available block. The size of a data transfer is determined by the application. Few applications require W-blocks; therefore, the default number of W-blocks is 0.

To determine how W-, Y-, and Z-blocks are being utilized on your SRP, use the STAT command, as described in the *CTOS Executive Reference Manual.*

To allocate more W-, Y-, and Z-blocks, modify the following parameters in the SRP operating system configuration file:

`:WBlocks:`	`(Number =` *n*`)`	(default 0)
`:YBlocks:`	`(Number =` *n*`)`	(default 4)
`:ZBlocks:`	`(Number =` *n*`)`	(default 28)

where *n* is the number of blocks to be allocated.

Using a Cache Memory Disk

A *cache memory disk* is a read/write RAM disk. It is used as a scratch volume (*[Scr]*), which can improve the performance of some applications. A cache memory disk is, however, volatile memory. That means that if the system crashes or is rebooted, all files stored in the cache memory disk are lost. Because files stored on a scratch volume are dispensable temporary files, that is not a problem. As you can imagine, though, it would be a problem if the memory disk contained your working data files.

The following example shows how a cache memory disk is configured as a scratch volume in the operating system configuration file.

```
:MassStorage: (Class = CacheMemory,
Unit = 0,
Device = m0,
Password =
Volume = CMScratch,
MaxSectors = 4096,
MaxDirectories = 5,
MaxSysFiles = 15,
MaxFiles = 750)
```

To define the memory disk as the scratch volume, you specify its volume or device name for the *:ScratchVolumeName:* parameter in the operating system configuration file, as shown in the following example:

```
:ScratchVolumeName: CMScratch
```

For more information, see also the *:MassStorage:* parameter for workstations in Section 16, "Configuring Workstation Operating Systems," and for SRPs in Section 17, "Configuring Shared Resource Processor Operating Systems."

When you use a cache memory disk as the scratch volume, you may need to create certain directories on it so that applications will work correctly. Such directories can include *<Spl>*, *<WP>*, and *<GPS>*; see the documentation for your applications for more detailed information.

The following example shows how you can set up your *SysInit.jcl* file to create directories during system initialization:

```
Command Create Directory, ([Scr]<Spl>,[Scr]<GPS>,[Scr]<WP>)
```

Section 16
Configuring Workstation Operating Systems

The Operating System Configuration File

The CTOS workstation operating systems are designed to work well in a variety of settings. In some cases, however, you need to configure an operating system to function optimally in your environment. Workstation operating systems most often require configuration in situations such as the following:

- To set up a software development environment
- To use hardware modules that are not recognized by the operating system
- To allocate memory for the file system cache

This section describes how to configure workstation operating systems. For information about configuring shared resource processor operating systems, see Section 17, "Configuring Shared Resource Processor Operating Systems."

The operating system configuration file for workstations is named *[Sys]<Sys>Config.sys*. It contains many parameters that you can modify to change certain aspects of operating system functionality.

Configuration File Format

Each line is written in the following format:

:Keyword:Value

where

:Keyword:	Is the name of a parameter.
Value	Is the configurable value of a parameter.

In addition, the following construction pertains to certain keywords that can be used on either workstations or SRPs:

:Keyword: (Subparam = Value, Subparam = Value)

where

:Keyword:	Is the name of a parameter.
Subparam	Is the name of a subparameter.
Value	Is a configurable value for a parameter or subparameter.

Configuration File Syntax

Note the following rules of syntax for parameters and subparameters:

- The colon preceding each keyword must be the first character on the line; no spaces or characters may precede it.
- A space after the colon following a keyword is optional.
- Parentheses enclose the entire set of subparameters.
- Spaces or an equal sign (=) separate subparameters from their values.

If subparameters do not fit on a single line, they may continue on the next line, as shown below:

```
:Keyword:        (subparam = value,
                  subparam = value,
                  subparam = value)
```

The amount of white space between keywords, parameters, subparameters, and values is not significant. In this manual, white space is used to enhance the readability of *Config.sys*.

A sample configuration file is shown in Figure 16-1.

Figure 16-1. Workstation Operating System Configuration File

```
:SwapFile: [Sys]<Sys>CrashDump.sys
:SwapFileSize:
:SwapFileSizeMax:
:VDMFile: [Sys]<Sys>InstallVDM.run
:ScratchVolumeName: d1
:FileCacheService: (BlockSize = 4096, BlockCount = 128)
:FileCacheDefaultEnable:Yes
:AgentCacheDefaultEnable:Yes
```

Editing the Configuration File

To make changes to *Config.sys*, follow these steps:

1. On the Executive command line, type **Editor**; then press RETURN.
2. Type *[Sys]<Sys>Config.sys*, as shown below:

   ```
   [File name(s)]  [Sys]<Sys>Config.sys
   ```

3. Press GO.

   The configuration file appears on the screen. It should look similar to the sample shown in Figure 16-1.

4. Add keywords and edit values as required.
5. When you have finished making changes, press FINISH, then GO, to save the file.
6. Reboot the workstation.

See the *CTOS Editor User's Guide* for detailed information about using the Editor.

Creating Ws*NNN*>Config.sys

You can create special versions of the operating system configuration file for cluster workstations that boot from the server. Such files must be located in *[Sys]<Sys>* on the server and are named as follows:

*Ws*NNN>*Config.sys*

where *NNN* is a three-digit workstation number.

If *WsNNN>Config.sys* does not exist for a particular workstation type number, the following file, if it exists, is used:

Ws>Config.sys

If neither *WsNNN>Config.sys* nor *Ws>Config.sys* exists, the workstation uses the server's version of *Config.sys*. If *Config.sys* does not exist (for example, on an SRP server), default parameters are used.

See also "Workstation Type Numbers" in Section 5, "Bootstrapping," for detailed information and diagrams of the bootstrap sequence.

Configurable Parameters

Configurable parameters are described in alphabetical order below. All of them are optional; if a parameter is omitted from *Config.sys*, a default value is used.

A *Config.sys* file is supplied with Standard Software. When applicable, initial values for the parameters it contains are noted in the descriptions. Parameters are also labeled as they apply to real mode, protected mode, and virtual memory operating systems.

:ActionKeySticks:

Real and protected modes

Default: No

This parameter specifies a default condition for the ACTION keystroke. Specify **Yes** if you want the keyboard to ignore the upstroke (release) of the ACTION key. This is used in cases where the user is physically incapable of holding down the ACTION key and another key simultaneously. The action key is deactivated when another key is pressed.

:AgentCacheDefaultEnable:

Protected mode only; cluster workstations only

Default: Yes

This parameter sets the default caching attribute for files opened on disks located on the server. Specify **Yes** if you want files from the server to be cached in the local file system cache on the cluster workstation. If you specify **No** or leave this field blank, files opened from the server are not cached. Note that this parameter must be set to **Yes** for caching to take place on diskless workstations.

:BeepOnToggle:

Real and protected modes

Default: No

This parameter activates an audible tone on chord keys that toggle on and off (**SHIFT LOCK** on all keyboards and **NUM LOCK** on some) in applications that read the keyboard in character mode. When set to on, one long and one short beep sound when a key toggles on, and one long and two short beeps sound when it toggles off.

:BMAttrBlinking:
:BMAttrBold:
:BMAttrHalfBright:
:BMAttrHalfReverse:
:BMAttrNormal:
:BMAttrReverse:
:BMAttrStruck:
:BMAttrUnderline:

Protected mode only; bit-mapped display modes only

Defaults: See below

These parameters define bit-mapped character attributes. You can specify **Normal, Halfbright, Underline, Reverse, Outline, Bold, Struck**, or **LowContrast**. Defaults are as indicated by the keywords except for *:BMAttrBlinking:*, the default of which is outlined text, and *:BMAttrHalfReverse:*, which is described below.

The following examples demonstrate how to change attributes with the values listed above. To change the half-bright attribute to low contrast, make the following change in *Config.sys*:

:BMAttrHalfBright: **LowContrast**

You can assign more than one attribute to a single video option. For example, to change the reverse screen attribute to half-bright reverse video, make the following entries in *Config.sys*. Notice that you must explicitly specify reverse video; it is implemented as the default only if no other entry exists.

:BMAttrReverse: **Halfbright**
:BMAttrReverse: **Reverse**

To change half-bright reverse video to low-contrast, outlined reverse video, make the following entries:

:BMAttrHalfReverse: **LowContrast**
:BMAttrHalfReverse: **Outline**
:BMAttrHalfReverse: **Reverse**

:CheckDAI:

Protected mode only

Default: No

Specify **Yes** to activate the device address identification (DAI) number. A DAI number is a physical ID that allows a program to identify a particular workstation within a workstation group.

Note that a DAI number requires special hardware and is not the same as a hardware ID. By default, hardware ID support is present in the operating system but is disabled if this parameter is set to **Yes**.

:ChordKeysStick:

Real and protected modes

Default: No

This parameter causes the keyboard to ignore the upstroke (release) of chord keys (SHIFT, CODE, and ALT). It is used in cases where the user is physically incapable of holding down a chord key and another key simultaneously. The chord key is deactivated when the next nonchord key is pressed.

Note that this parameter also controls the ACTION key, but can be overridden by positioning the *:ActionKeySticks:* parameter below it in the configuration file.

> **Caution**
>
> Set the following parameter, *:ClusterLine1:*, on servers only. If it is present on a cluster workstation and its value differs from that of the server, it can prevent the cluster workstation from communicating with the server.

:ClusterLine1: (Speed = *bps*, MaxWs = *number*)

Protected mode only

Default: See below

Note: *This parameter replaces the* :ClusterLineSpeed: *parameter of earlier versions.* :ClusterLineSpeed:, *however, is still supported in this release for migration purposes.*

This parameter defines the line speed and number of workstations supported on the cluster line. In most cases, maximum cluster line speed is dictated by the workstation hardware connected to the line. Check your workstation hardware specifications. See the *CTOS Cluster and Network Hardware Installation Guide* for information about supported cluster line speeds.

The default for server workstations is 1.8M bps. Cluster workstations use the line speed defined by the server (see the caution above). Subparameters are described below.

Subparameter	Value
Speed	Default: 1.8M bps Specify **307k**, **1.8M**, or **3.7M** ("k" means kilobits per second; "M" means megabits per second).
MaxWs	Default: 32 Specify the maximum number of workstations to be connected to the cluster line. To disable a cluster line, specify **0**.

Caution

Set the following parameter, *:ClusterLineSpeed:*, on servers only. If it is present on a cluster workstation and its value differs from that of the server, it can prevent the cluster workstation from communicating with the server.

:ClusterLineSpeed:

Real mode only

Note: *For protected-mode and virtual memory operating systems, this parameter is replaced by* :ClusterLine1:, *as described above.*

Specify **1.8Mbps**, or **307kbps**, depending on the hardware configuration of the cluster. See the *CTOS Cluster and Network Hardware Installation Guide* for information about supported cluster line speeds. Cluster workstations use the line speed defined by the server.

:ClusterTimeout:

Real and protected modes

Default: 30
Minimum: 4
Maximum: 65535

This parameter controls the amount of time that elapses before a cluster workstation terminates attempts to communicate with the server. Specify a number of seconds. If communication does not take place during the specified interval, the cluster workstation returns an error, as shown in the following example:

```
Cluster not running  (Error 6)
```

The message can vary, depending on the cause of the communication error.

:ColorMonitor:

Virtual memory only; VGA only

Default: Yes

This parameter controls color or gray scale configurations on VGA hardware that is not recognized by the operating system. Specify **Yes** if color is not displayed on a color monitor. Specify **No** if an unexpected background color appears on a gray-scale monitor.

If this parameter is set incorrectly, the monitor may yield unexpected results. Therefore, if a monitor is functioning correctly without it, it is best to omit *:ColorMonitor:* entirely from the *Config.sys* file.

:CompensateFloppy:

Real and protected modes

Default: No

Specify **Yes** if the workstation hardware includes a Mode-3 X-Bus module, such as Ethernet or a Voice Processor, which you use concurrently with a floppy disk drive. This prevents read and write errors, which can be caused by concurrent Mode-3 DMA. The compensation that prevents such errors, however, reduces the speed of access to floppy diskettes.

:ContinueLoadOnError:

Virtual memory only

Default: No

Specify **Yes** to allow loading of applications or libraries that cannot be dynamically bound. This is useful in development environments, for example, to load an application when a dynamic library it references does not exist or when a particular reference within a library does not exist. Such references are replaced by zeros at run time, which cause a general protection fault if executed.

:cParExitRunFile:

Protected mode only

Default: 8192
Minimum: 0
Maximum: 65535

This parameter specifies the minimum memory partition size required for a chain or exit operation. Specify a number of paragraphs.

:cParSpecHeap:

Protected mode only

Default: 128
Minimum: 32
Maximum: 4095

This parameter defines the size of the file-specification expansion memory heap. Specify a number of paragraphs.

:cParSysCommonHeap:

Protected mode only

Default: 16
Minimum: 10
Maximum: 4095

This parameter defines the size of the system common heap. Specify a number of paragraphs.

:cSemaphoreRecords:

Virtual memory only

Default: 300
Minimum: 0
Maximum: 1024

This parameter defines an amount of memory to reserve for use by semaphores. Specify the number of semaphore records to allocate. Each open semaphore requires one record if unnamed or two records if named. In addition, each process waiting for a noncritical semaphore requires one record. Therefore, specify approximately three times the number of semaphore records that will actually be open simultaneously.

:CrashDumpFile:

Protected mode only

Default: *[Sys]<Sys>CrashDump.sys*

Specify the file specification of the crash dump file to which memory dumps will be written. Note that on the local workstation, this file must be named *CrashDump.sys*, but it may reside on any local disk. To dump memory to the server, the crash dump file must be named *[!Sys]<Sys>Ws>CrashDump.sys*.

Note that this parameter also defines a crash dump file for the extended crash dump utility. For more information, see the *:ExtCrashDumpFile:* parameter, later in this section, and "Collecting a Crash Dump" in Section 20, "Troubleshooting," for more information.

:CreateDirectoryProtection:

Real and protected modes

Default: See below

This parameter controls whether a volume password is required to create a directory. The default is **No** for protected-mode operating systems and **Yes** for real-mode operating systems.

If you specify **Yes**, a volume password must be supplied to create a directory if the volume has a password. If you specify **No**, a password is not required, regardless of whether the volume has a password.

:CursorStart:

See *:CursorType:*.

:CursorStop:

See *:CursorType:*.

:CursorType:

Real and protected modes

Default: Underline

This parameter applies to B26, B28, B38, and B39 character-mapped workstations. Specify **Block** to change the cursor to a block-shaped character. Make the following entries to change the cursor to a double-underline character:

:CursorType: Underline
:CursorStart: 9
:CursorStop: 10

:DiskAllocationLimit:

Protected mode only

Default: All available disk space
Minimum: 512 bytes
Maximum: 4G bytes

This parameter specifies the maximum size for any one file, to prevent errant programs from consuming all disk space. Specify a number of bytes.

:DiskLogThreshold:

Protected mode only

Default: 0
Minimum: 0
Maximum: 1024

Specify the number of retries that are acceptable before an I/O error is recorded in the system error log. (You can view the system error log with the **PLog** command; see the *CTOS Executive Reference Manual*). By default, every I/O operation requiring a retry is logged. Unrecoverable errors are always logged.

:DiskRetryCount:

Protected mode only

Default: See below
Minimum: 0
Maximum: 1024

Specify the number of retries for a disk operation before it terminates with an I/O error. The default is either 4 or 8, depending on the disk.

:EnterDebuggerOnFault:

Protected mode only

Default: No

This parameter applies only when the Debugger is loaded in memory. Specify **Yes** to suspend an application and enter the Debugger when the application cannot recover from a fault. If you specify **No**, applications terminate when they cannot recover from an error. When a system service or the operating system cannot recover, a system crash occurs.

:EVBackgroundOff:

Protected mode only

Default: Yes

This parameter affects background color on VGA and enhanced-video equipped workstations. Specify **No** to enable enhanced video (EV) background color emulation.

:ExtCrashDumpFile:

Protected mode only

Default: *[Sys]<Sys>CrashDump.sys* (if large enough)

This parameter defines a file for extended crash dumping when auto dumping is enabled (see *:SuppressAutoDump:*) and if *[Sys]<Sys>CrashDump.sys* is not large enough to contain the entire dump.

Make the extended crash dump file large enough to contain the entire contents of memory. To determine the correct size, multiply the total amount of memory by 2. For example, if the workstation has 2048K bytes of memory, allocate 4096 sectors when you create the extended crash dump file.

An extended crash dump file is not created when the disk is initialized; you must create it with the **Create File** command (see the *CTOS Executive Reference Manual*).

:ExtCrashVDMFile:

Protected mode only

Default: *[Sys]<Sys>Vdm_Dmy.run*

This parameter specifies the video run file to be used when the system is performing an extended crash dump. It must be in the form of a full file specification. The default is a small video program that does not update the video hardware and, therefore, does not display the extended crash dump operation while it is executing.

In many cases, the video run file *Vdm_Ch.run* can be specified; it is small enough to be loaded during crash dumps and allows the crash dump operation to be displayed. In most cases, unless the workstation is loading a very small operating system, the video files *Vdm_VGA.run* and *Vdm_BM.run* are too large to be loaded during extended crash dump operations.

:fAllowCommLineDMAOnCPU:

Protected mode only

Default: No

This parameter affects the serial ports on B39 (Series 386i) workstations. Specify **Yes** to initialize DMA for both channel A and channel B. If you specify **No** or leave this field blank, DMA is not initialized.

:FileCacheDefaultEnable:

Protected mode only

Default: Yes

This parameter sets the default condition for file system caching. Specify **Yes** to enable caching on all files that are not specifically disabled. Specify **No** to disable caching on all files that are not specifically enabled.

This parameter alone does not configure or implement file system caching on the workstation; see the *:FileCacheService:* parameter, below. See also Section 14, "Using a File System Cache," for detailed information about file cache configuration requirements.

:FileCacheService: (BlockSize = *bytes*, BlockCount = *number*, MinWorkSetBlkCnt = *number*)

Protected mode only

Defaults: See subparameters, below

This parameter allocates memory for the file system cache. Subparameters are described below.

Subparameter	Value
BlockSize	Default: 4096 Specify the number of bytes to be allocated for each block. It must be a multiple of 4096.
BlockCount	Default: 128 on server, 0 on cluster workstations Minimum: 64K bytes divided by block size Maximum: Depends on available memory Block count times block size equals the total amount of memory required for the cache. In most cases, increase this value, rather than *BlockSize*, to increase the size of the cache. Specify the number of blocks to be allocated.
MinWorkSetBlkCnt	Default: 16 on server, 0 on cluster workstations Specify the number of blocks that cannot contain locked-in files. The default is 64K bytes, divided by the block size. For example, if block size is 4096 bytes, the default is 16. If you change the block size, the default value changes accordingly.

:FileStructureVerify:

Protected mode only

Default: No

This parameter is used for diagnosing file system or hardware problems that could result in corrupted disks. If you specify **Yes,** file headers, directory entries, and volume home blocks are verified for data integrity immediately after they are written to disk. That verification, however, degrades the speed of file system performance. If you specify **No,** a verification is not performed.

:KbdProfile: (ID = *KbdID*, AltNlsStyle = *Yes / No*)

Protected mode only

Defaults: See subparameters, below

This parameter defines the keyboard profile to be loaded at boot time. Subparameters are described below.

Subparameter	Value
ID	Default: 04h Specify the keyboard profile ID for the workstation. The default is 04h unless NLS Table 15 is present in *Nls.sys*, in which case the default is B0h. See the *CTOS Operating System Concepts Manual* for information about keyboard profiles.
AltNlsStyle	Default: No This subparameter applies only to workstations that are configured with *Nls.sys*. Specify **Yes** to override the default keyboard style the operating system reads from *Nls.sys*. The recognized keyboard style is taken from NLS Table 15 if it is present; otherwise, it is taken from NLS Table 0.

:KbdTables: (Number = *number*, Size = *bytes*)

Protected mode only

Defaults: See subparameters, below

This parameter defines the amount of memory required for loadable keyboard data blocks and the maximum number of them to be loaded. The defaults (described below) allow for one translation data block and one emulation data block to be loaded. Additional data blocks may be needed by certain applications, either because the application itself expects to load a particular keyboard table or the run file header has been set to expect one. Subparameters are described below.

Subparameter	Value
Number	Default: 1 Specify the number of translation data blocks and/or emulation data blocks that will be required.
Size	Default: See below Specify the maximum amount of memory, in bytes, required for loadable keyboard data blocks. The default is the number of bytes required for the largest translation data block and the largest emulation data block in *NlsKbd.sys.*

:LibrarySearchPath:

Virtual memory only

Default: *[Sys]<PMLib>*

This parameter defines a list of paths to be searched for dynamic link libraries. Each entry consists of a volume, directory, and optionally, a password. More than one path may be specified. Surround path entries with parentheses and separate them with commas. A period (.) denotes the user's current path. The following example illustrates multiple search-path entries:

```
:LibSearchPath: ([Sys]<Sys>, [Sys]<Lib>^pswd, .)
```

:LfsToMaster:

Real and protected modes; cluster local file systems (LFS) only

Default: No

This parameter instructs the operating system to search both *[Sys]<Sys>* on the LFS and *[!Sys]<Sys>* on the server for read-only files. Specify **Yes** to enable this option. When this option is enabled, it can be suppressed by specifying **[+Sys]<Sys>** in a file specification. That instructs the operating system to search *[Sys]<Sys>* only.

This option does not work with wild-card characters, nor does it work if the LFS is booted from the server.

:MapKeyboardID: (Source = *KbdID*, Target = *KbdID*)

Real and protected modes
(See the note for real mode, below)

Defaults: None

This parameter controls how various-style keyboards are mapped to available keyboard data blocks. When the keyboard hardware matches the ID specified as the *Source* subparameter, it is recognized as the keyboard specified as the *Target* subparameter and keyboard data blocks are loaded accordingly.

This parameter can appear multiple times within the configuration file. Subparameters are described below.

Subparameter	Value
Source	Enter the two-digit hexadecimal keyboard ID for the keyboard you want to use.
Target	Enter the two-digit hexadecimal keyboard ID for the keyboard to which you want to map.

Note: *Real-mode operating systems do not recognize subparameters. Therefore, the format of this entry consists of two 2-digit hexadecimal values, such as 040B, where the first two digits identify the source keyboard and the last two digits identify the target.*

:MassStorage: (Class = *type*,
Unit = *number*,
Device = *name*,
Password = *password*
Volume = *name*
MaxSectors = *number*
MaxDirectories = *number*
MaxSysFiles = *number*
MaxFiles = *number*)

Protected mode only

Defaults: None

This parameter defines a portion of memory as a cache memory disk. Subparameters are described below.

Subparameter	Value
Class	Specify **CacheMemory**.
Unit	Specify a number to uniquely identify the cache memory disk. In most cases, this value will be 0, because it is unusual to configure more than one cache memory disk per workstation.
Device	Specify a device name. You can assign any name, but by convention, memory disks are named M*n* (for example, m0) where *n* matches the number you assigned as the Unit subparameter.
Password	Specify a password to assign to the cache memory disk.
Volume	Specify a volume name for the cache memory disk.
MaxSectors	Specify the maximum amount of memory, in sectors, for the cache memory disk to occupy.
MaxDirectories	Specify the maximum number of directories that can be created on the cache memory disk.
MaxSysFiles	Specify the maximum number of files that can be created in the cache memory disk's *<Sys>* directory.
MaxFiles	Specify the maximum number of files that can be created on the cache memory disk.

:MaxConcurrentQuiet:

Protected mode only

Default: 30 on servers, 5 on cluster workstations

This parameter controls the number of "quiets" that can occur simultaneously. A "quiet" is the act of notifying server programs of the termination of another program. The terminated program may be local or remote. If this parameter is exceeded during operation, the system holds additional quiets until outstanding quiets are complete. Each quiet unit consumes approximately 30 bytes of memory. Too small a number can impair performance.

:MaxConcurrentTerm:

Protected mode only

Default: 15

This parameter controls the maximum number of program terminations that can occur simultaneously. If this number is exceeded during operation, the system will crash with error code 820 (Termination Heap Full). Each termination unit consumes approximately 144 bytes of memory.

:MaxHugeSg:

Virtual memory only

Default: 128
Minimum: 0
Maximum: 1024

This parameter defines the maximum number of contiguous GDT selectors that are allocated to the system and available to AllocHugeMemory (see the *CTOS Procedural Interface Reference Manual*).

:MaxInstanceSeg:

Virtual memory only

Default: 2048
Minimum: 0
Maximum: 5120

This parameter defines the maximum number of instance selectors allocated to the system. Specify a number of bytes.

Caution

Do not change the value of the *:MaxXBlocksOut:* parameter, described below, unless instructed to do so by a Technical Support engineer. An incorrect value can cause total system failure.

:MaxXBlocksOut:

Protected mode only

Default: 5
Minimum: 1
Maximum: Number of allocated X-blocks

Specify the maximum number of X-blocks that can be outstanding for each workstation.

:Mode3DMAMaster:

See *:ModuleType:*.

:ModuleType:
:XBusWindowSize:

Real and protected mode

Default: None

When a workstation bootstraps, the operating system configures the X-Bus window for each module it identifies (see "X-Bus Management" in the *CTOS Operating System Concepts Manual*). If you are using a nonstandard module, or if the documentation so instructs you, add these two parameters to the *Config.sys* file. Obtain the values from the documentation for the module.

If you specify a *:ModuleType:* parameter, it must be immediately followed by an *:XBusWindowSize:* parameter, as shown above.

In addition, the following parameters are used only in conjunction with the *:ModuleType:* and *:XBusWindowSize:* parameters.

:Mode3DMAMaster:

Protected mode only

Default: No

This parameter follows a *:ModuleType:* parameter. Specify **Yes** to indicate that the workstation is equipped with a Mode-3 DMA master module. When set to **Yes**, this parameter controls the loading of certain protected-mode programs. Those that have not been explicitly linked to load above 16M bytes of memory will be loaded somewhere within the first 16M bytes. (See the *CTOS Programming Utilities Reference Manual: Building Applications*, for information about linking programs.)

:UsedFromVirtualRealMode:

Protected mode only

Use this parameter if the workstation is equipped with a non-standard X-Bus module. It must be placed immediately after the *:ModuleType:* and *:XBusWindowSize:* parameters. Specify **Yes** if a real-mode program will be making calls to the MapXBusWindow operation (see the *CTOS Procedural Interface Reference Manual*). If you specify **No** or omit this field, Error 38 (invalid window size) occurs when a real-mode program attempts to call MapXBusWindow for the nonstandard module.

:nMaxNetUsers:

Protected mode and virtual memory; servers only

Default: See below
Minimum: See below
Maximum: 1024

This parameter defines the number of incoming requests from remote BNet nodes that can be served simultaneously on this workstation.

On protected mode servers, default values are 20, 40, and 60 for small, medium, and large operating systems, respectively. On virtual memory servers, the default is 40. Minimum values are the same as the defaults.

:NoBeep:

Virtual memory only

Default: No

This parameter defines the status of the audible system alarm. By default, it is enabled. Specify **Yes** to disable it.

:OldMaster:

Real and protected modes

Default: No

This parameter accelerates booting for workstations that boot from a real-mode server. If the server is not running a protected-mode operating system, specify **Yes**.

:PagingServiceFileCount:

Virtual memory only

Default: 50

This parameter defines the maximum number of files that can be handled by the paging service. It is provided so that the number of File Information Blocks can be increased if Error 13969 (Too Many Backing Store Files) occurs. One FIB must be allocated for each file. Specify the number of FIBs to allocate.

:PagingServiceHeapSize:

Virtual memory only

Default: See below

This parameter defines the size of the paging service heap. It is provided so that the heap size can be increased if Error 13963 (Heap Memory Not Available) occurs. The default is the number of paging service users times the size of the paging service stack. Specify a number of bytes.

:PagingServiceMaxWritesInProgress:

Virtual memory only

Default: 20

This parameter defines the maximum number of write operations that may be in progress simultaneously. It is used to limit the number of outstanding page writes to the swap file.

:PagingServicePrefaultCount:

Virtual memory only

Default: 2
Minimum: 0
Maximum: 14

This parameter defines the number of pages the paging services attempts to bring into memory in anticipation of accessing them. If it is set to 0, the paging service does not attempt to prefault any pages.

:PagingServicePrefaultCountMax:

Virtual memory only

Default: 14
Minimum: 0
Maximum: 14

This parameter defines the maximum number of pages that can be prefaulted if the *:PagingServicePrefaultCount:* parameter is enabled.

:PagingServiceStackSize:

Virtual memory only

Default: 500

This parameter defines the size of the paging service stack. It is provided so that the stack size can be increased if Error 13962 (Stack Overflow) occurs. Specify a number of bytes.

:RebootClusterOnMasterDown:

Protected mode only; cluster workstations only

Default: None

This parameter defines one or more hardware ID numbers, which if matched by the hardware ID number assigned to the workstation, cause the workstation to be rebooted following a loss of communication with the server. Specify a hardware ID number from 1 to 127. This parameter may be repeated for multiple hardware ID numbers, and it should be included in *Ws*NNN>*Config.sys* or *Ws*>*Config.sys*, if one of those files is being used instead of *Config.sys* for cluster workstations that boot from the server.

Hardware IDs are assigned with the **Write Hardware ID** command; see the *CTOS Executive Reference Manual*. In addition, hardware IDs are not supported on all workstations. See the documentation for your processor model.

:RepeatKeyFactor:

Real and protected modes

Default: 0

This parameter controls the rate at which characters are repeated while a key remains depressed in applications that read the keyboard in character mode. Enter one of the following values:

0	Normal (default) speed
1	Half speed
2	One-fourth speed
3	No repeating

:RqTracker:

Protected mode only

Default: No

This parameter enables request tracking. Specify **Yes** if you want outstanding requests copied to an exchange for tracking.

:ScratchVolumeName:

Protected mode only

Default: *Sys*

This parameter specifies the disk that is used as the "scratch" volume (*[Scr]*). The scratch volume is used for temporary storage by certain commands, such as **Floppy Copy**. Scratch files can consume a lot of disk space, and if not available, the command can fail. Therefore, you may want to specify the scratch volume to be a disk with a large amount of free disk space; such a disk is frequently not *[Sys]*.

Specify a volume or device name, with or without square brackets, for example, *[BigDaddy]* or *d1*. If you specify a volume other than *Sys* as the scratch volume, you must create a directory named <*$000*>, with a capacity of 750 files, on that disk.

:ScreenTimeout:

Real and protected modes

Default: 0 (see below)
Minimum: 0
Maximum: 109

This parameter controls the amount of time that elapses before the screen is shut off when the workstation is not being used. Specify a number of minutes. The default, 0, means that the screen always stays on.

:sEnvironment:

Protected mode and virtual memory only

Default: 1024
Minimum: 0
Maximum: 65536

This parameter defines an amount of environment memory for each user. It is provided so that the amount of environment memory can be increased if an error occurs, indicating that the environment is full. Specify a number of bytes.

:sExitProcStack:

Virtual memory only

Default: 2048
Minimum: 0
Maximum: 65536

This parameter defines the size of the stack created for ExitList procedures. Specify a number of bytes. (See SetExitList in the *CTOS Procedural Interface Reference Manual.* See also the section about dynamic link libraries in the *CTOS Operating System Concepts Manual.*)

:sLoaderHeap:

Virtual memory only

Default: 24576
Minimum: 0
Maximum: 65536

This parameter defines the size of the loader's heap. It is provided so that the size of this heap size can be increased if Error 1114 (Loader Heap Full) occurs. Specify a number of bytes.

:sMemMgrHeap:

Virtual memory only

Default: 16384
Minimum: 0
Maximum: 65536

This parameter defines the size of the memory manager's heap. It is provided so that the size of this heap can be increased if Error 477 (Memory Manager Heap Full) occurs. Specify a number of bytes.

:sNameHash:

Virtual memory only

Default: 32
Minimum: 1
Maximum: 512

This parameter defines the number of name table positions in a class. For detailed information, see "Managing Names," in the utility operations section of the *CTOS Operating System Concepts Manual*.

:sNameHeap:

Virtual memory only

Default: 61440
Minimum: 0
Maximum: 1048576

This parameter defines the size of the name heap. Specify a number of bytes.

:StdVGABlink:

Virtual memory only

Default: No

This parameter applies to EISA/ISA-bus workstations equipped with industry-standard VGA video adapters. Specify **Yes** to enable the blinking character attribute. The sixth color defined in the color palette is not displayed when the blinking option is enabled.

Note: *This option is supported only if the* :EVBackgroundOff: *parameter is also set to* **Yes**. *See the description for that parameter, earlier in this section.*

:StdVGABold:

Virtual memory only

Default: No

This parameter applies to EISA/ISA-bus workstations equipped with industry-standard VGA video adapters. If it is set to **Yes**, characters with the bold attribute are displayed in color. The color displayed is the seventh color defined in the color palette, which is programmable via the ProgramColorMapper operation; see "Using Color," in the *CTOS Programming Guide*.

If this parameter is set to **No** or left blank, characters with the bold attribute are displayed as normal text.

:StdVGAStruckThrough:

Virtual memory only

Default: No

This parameter applies to EISA/ISA-bus workstations equipped with industry-standard VGA video adapters. If it is set to **Yes**, characters with the struck-through attribute are displayed in color. The color displayed is the seventh color defined in the color palette, which is programmable via the ProgramColorMapper operation; see "Using Color," in the *CTOS Programming Guide*.

If this parameter is set to **No** or left blank, characters with the struck-through attribute are displayed as normal text.

:StdVGAUnderline:

Virtual memory only

Default: No

This parameter applies to EISA/ISA-bus workstations equipped with industry-standard VGA video adapters. If it is set to **Yes**, characters with the underline attribute are displayed in color. The color displayed is the seventh color defined in the color palette, which is programmable via the ProgramColorMapper operation; see "Using Color," in the *CTOS Programming Guide*.

If this parameter is set to **No** or left blank, characters with the underlined attribute are displayed as normal text.

:SuppressAutoDump:

Protected mode only

Default: No

This parameter implements automatic extended crash dumping. On systems that require it, an extended crash dump is performed after the first megabyte of memory has been dumped. If the crash file, *<Sys>CrashDump.sys*, is large enough to contain the extended crash dump, it will be used. If not, the file specified for *:ExtCrashDumpFile:* is used (see that parameter, above).

If you specify **Yes**, an extended crash dump does not take place. You can, however, perform it manually with the **Extended Crash Dump** command; see the *CTOS Executive Reference Manual.*

:SuppressCommDMABufferCheck:

Virtual memory only

Default: No

This parameter determines whether communications buffers are checked for contiguity before DMA is started. By default, checking is enabled. Specify **Yes** to disable checking.

:SuppressDebugger:

Protected mode only

Default: No

This parameter controls whether the Debugger is loaded into memory when the workstation boots. It applies only if the Debugger software has been installed and is present in *[Sys]<Sys>*. By default, the Debugger is loaded. If you specify **Yes**, the Debugger is not loaded.

:SuppressGlobalPolicy:

Virtual memory only

Default: No

This parameter controls the global paging policy, which allows applications to compete for 4K byte pages of memory. Specify **Yes** to suppress global paging and limit the number of pages consumed by an application to a number derived from the partition size that is specified in the Context Manager configuration file. See also Section 15, "Optimizing System Performance."

:SuppressPmDefaults:

Virtual memory only

Default: No

This parameter is useful for quickly configuring a system for use without Presentation Manager. It decreases various system resources to reflect the absence of Presentation Manager and saves approximately 135K bytes of memory.

Specify **Yes** to set the following configuration parameters:

:cSemaphoreRecords: 8
:MaxInstanceSeg: 0
:sNameHeap: 0
:sExitProcStack: 512
:sMemMgrHeap: 512
:sLoaderHeap: 4096
:sEnvironment: 0

(Each of the parameters listed above is also described individually, in alphabetical order, throughout this section.)

Parameters in *Config.sys* are read from top to bottom, and when a parameter occurs more than once, the last occurrence in the file is the one that remains set. Therefore, to override a parameter that is set by *:SuppressPmDefaults:*, the keyword for the parameter you want to override must be placed after the *:SuppressPmDefaults:* keyword in *Config.sys*.

In the following example, the system is configured for 100 semaphore records, because the value specified for *:cSemaphoreRecords:* overrides the value of 8 that was set by *:SuppressPmDefaults:*.

:SuppressPmDefaults: Yes
:cSemaphoreRecords: 100

In the following example, however, the system is configured for 8 semaphore records, because the value of 8 that is set by *:SuppressPmDefaults:* overrides the value that was specified for *:cSemaphoreRecords:*.

:cSemaphoreRecords: 100
:SuppressPmDefaults: Yes

:SwapFile:

Protected mode and virtual memory only

Default: *[Sys]<Sys>CrashDump.sys*

This parameter specifies the file to which the contents of a memory partition are swapped. If *[Sys]<Sys>CrashDump.sys* is specified, that file serves a dual purpose; it is used as the swap file while the system is running and as the crash dump file when the system crashes. This conserves disk space by minimizing the number of large files needed on a workstation. You can, however, specify some other file.

If the specified file is not available, as in the case of a diskless workstation, for example, the alternate swap file is used. See the *:SwapFileAlternate:* parameter below. The alternate swap file is also used if a size is specified for *CrashDump.sys* that is larger than the current size of that file; that is because the size of a crash dump file cannot be expanded. See the *:SwapFileSize:* parameter, below.

If the swap file you specify requires a password, a valid password for it must be built (SysGen'd) into a customized version of the operating system. See Section 18, "Building a Customized Operating System."

:SwapFileAlternate:

Protected mode and virtual memory only

Default: *[Sys]<Sys>SwapArea*NNN.*sys*

This parameter sets the swap file specification to be used when the primary swap file, as specified in the *:SwapFile:* parameter, cannot be accessed. The default is *[Sys]<Sys>SwapArea*NNN.*sys*, where *NNN* is a number from 0 to 127.

When specifying a file other than the default, its file specification must contain the string 00, which is incremented by one as subsequent swap files are opened. That allows multiple diskless workstations to swap simultaneously, each using a different swap file.

Periodically, alternate swap files should be deleted from the disk. Those that are in use when you perform the delete operation will not be deleted.

:SwapFileSize:

Protected mode only

Default: Current size of the swap file
Minimum: 512
Maximum: All available disk space

This parameter specifies the starting size of a swap file. Specify a number of sectors. On virtual memory operating systems, if 0 is specified as the swap file size, a swap file is neither created (if it does not already exist) nor used by the operating system.

:SwapFileSizeMax:

Protected mode and virtual memory only

Default: 0 (see below)
Minimum: 0
Maximum: 65535

This parameter controls the maximum size of a disk swap file. Specify a number of sectors, for example, 5000. The swap file size starts at the number of sectors specified for the *:SwapFileSize:* parameter and grows to the size specified for *:SwapFileSizeMax:*. By limiting the size of the swap file, disk space is retained for other uses. Swap files created in memory are not expanded.

:SwapRemoteRunFilesLocally:

Virtual memory only

Default: No

This parameter controls whether pages from run files located on the server are copied to the local swap file. Specify **Yes** if you want the paging service to copy pages read from the server to the local swap file and subsequently read them from there. This may improve swapping performance but also may increase the size of the swap file. The default is to read pages from the run file located on the server.

:UsedFromVirtualRealMode:

See *:ModuleType:*.

:VDMFile:

Protected mode only

Default: *[Sys]<Sys>InstallVDM.run*

This parameter determines which video service is installed on the workstation. The default file installs the appropriate video manager for the workstation hardware, as listed below:

Character	*[Sys]<Sys>VDM_Ch.run*
Bit-mapped	*[Sys]<Sys>VDM_Bm.run*
VGA (X-Bus and X-Bus+)	*[Sys]<Sys>VDM_Vga.run*
VGA (EISA/ISA)	*[Sys]<Sys>VDM_StdVga.run*

To install a video manager with windowing capabilities, specify the following value for the *:VDMFile:* keyword:

[Sys]<Sys>InstallVDM_w.run

For workstation servers without video, specify the following value for the *:VDMFile:* parameter:

[Sys]<Sys>VDM_dmy.run

:WakeUpInterval:

Protected mode without virtual memory only

Default: 0 (see below)
Minimum: 0
Maximum: 6553

This parameter controls how often the operating system searches for contexts to swap back into memory. Specify a time interval in tenths of seconds. The default, 0, indicates that swapping occurs only through user action, for example, with Context Manager.

:XBlocks: (Number = *number*, Size = *bytes*)

Protected mode only; servers only

Default: See subparameters, below

This parameter defines the number and size of large X-blocks to be allocated on the processor. See "Adjusting Memory Blocks" in Section 15, "Optimizing System Performance," for information about how X-blocks are used by the operating system. Subparameters are described below.

Caution

Do not change the value of the *Size* subparameter, described below, unless instructed to do so by a Technical Support engineer. An incorrect value can cause total system failure.

Subparameter	Value
Number	Default: See below Minimum: 1 Maximum: 64K bytes divided by X-block size Specify the number of large X-blocks to be allocated on the processor. The default number of X-blocks differs among operating systems; for example, *pSrvrL* allocates more X-blocks than *pSrvrM*. Use the Cluster Status command to determine the number of X-blocks that are currently allocated.
Size	Default: 4160 bytes Minimum: 2656 bytes Maximum: 64K bytes Specify the number of bytes to be allocated for each large X-block.

:XBlocksSmall: (Number = number, Size = bytes)

Protected mode only; servers only

Default: See subparameters below

This parameter defines the number and size of small X-blocks to be allocated on the processor. See "Adjusting Memory Blocks" in Section 15, "Optimizing System Performance," for information about how X-blocks are used by the operating system. Subparameters are described below.

Caution

Do not change the value of the *Size* subparameter, described below, unless instructed to do so by a Technical Support engineer. An incorrect value can cause total system failure.

Subparameter	Value
Number	Default: See below Minimum: 0 Maximum: 64K bytes divided by small X-block size Specify the number of small X-blocks to be allocated on the processor. The default number of small X-blocks differs among operating systems. Use the Cluster Status command to determine the number of small X-blocks that are currently allocated.
Size	Default: 96 bytes Minimum: 0 Maximum: 64K bytes Specify the number of bytes to be allocated for each small X-block.

:XBusWindowSize:

See *:ModuleType:*.

Section 17
Configuring Shared Resource Processor Operating Systems

The SRP Operating System Configuration File

The CTOS/XE operating systems are designed to work well in a variety of settings. In some situations, however, you might need to configure an operating system to function optimally in your environment. Communications or database applications, software development environments, and unusual hardware configurations most often benefit from modifications to the SRP operating systems.

In addition, you may need to configure the operating system to recognize all disks on the SRP. By default, only disks in the primary cabinet are preconfigured. Therefore, you must configure the disks in the other cabinets for them to be recognized by the operating system. See the *:MassStorage:* parameter, in this section, for specific information.

The operating system configuration file for SRPs is named *[Sys]<Sys>SrpConfig.sys*. It consists of a *boot section*, which identifies the processors and operating systems; and a *processor section*, which defines parameters for specific processors. Each section is described in detail later in this section. The default file boots only a minimal SRP hardware configuration, so in most cases, you will need to modify this file.

This section describes how to configure SRP operating systems. For information about configuring workstation operating systems, see Section 16, "Configuring Workstation Operating Systems."

A sample SRP configuration file is shown in Figure 17-1.

Figure 17-1. SRP Operating System Configuration File

```
:LogUnknownEntries:  Yes
:Boot: (Processor = GP00, Dump = No)
:Boot: (Processor = GP00, Dump = No)
:Boot: (Processor = CP00, OS = [Sys]<Sys>rSrpCp.run, Dump=No)
:Boot: (Processor = CP01, OS = [Sys]<Sys>rSrpCp.run, Dump = No)
:Boot: (Processor = FP00, OS = [Sys]<Sys>rSrpFp.run, Dump = No)
:Boot: (Processor = TP00, OS = [Sys]<Sys>rSrpTp.run, Dump = No)
:Boot: (Processor = SP00, OS = [Sys]<Sys>rSrpSp.run, Dump = No)
:Boot: (Processor = GP01, OS = [Sys]<Sys>pSrpGp.img, Dump = No)
:Boot: (Processor = DP00, OS = [Sys]<Sys>rSrpDp.run, Dump = No)

:Processor: Default
:ClusterLine1:  (Speed = 1.8M)
:ClusterLine2:  (Speed = 1.8M)
:CrashDumpPath:  [Sys]<CrashDump>

:Processor: GP00
:WatchDogStatus: SetFlag
:ClusterLine2: (Speed = 3.7M)
:FileCacheService: (BlockSize = 4096, BlockCount = 512)
:MassStorage: (Class = SCSI, Adaptor = 0, Target = 1,
               LUN = 0, Device = d1, Password = d1)
:MassStorage: (Class = SCSI, Adaptor = 0, Target = 2,
               LUN = 0, Device = d2, Password = d2)
:MassStorage: (Class = SCSI, Adaptor = 0, Target = 3,
               LUN = 0, Device = d3, Password = d3)
:SequentialStorage:  (Class = SCSI, Adaptor = 0,
                      Target = 0, LUN = 0, Device = QIC)
:XBlocks: (Number = 64)
:ZBlocks: (Number = 200)

:Processor: CP01
:ClusterLine1: (Speed = 307K, MaxWs = 4)

:Processor: FP00
:MassStorage: (Class = ST506, Unit = 1, Device = d4, Password = d4)
:MassStorage: (Class = ST506, Unit = 2, Device = d5, Password = d5)

:Processor: GP01
:FileCacheService: (BlockSize = 4096, BlockCount = 3000)
:RemoteCachePool: (Name = Local)
:RemoteCacheClient: (Name = FP00, Pool = Local)
:RemoteCacheClient: (Name = DP00, Pool = Local)

:Processor: DP00
:MassStorage: (Class = SMD, Unit = 0, Device = s0, Password = s0)
:MassStorage: (Class = SMD, Unit = 1, Device = s1, Password = s1)
:SequentialStorage: (Class = HalfInch, Unit = 1, Device = Tape)
```

Using Keyswitch Files

On an SRP, you can create different operating system configuration files to be used in different circumstances. The keyswitch on the front of the SRP controls the file that is read when the SRP boots.

The corresponding keyswitch configuration files are named as follows:

*[Sys]<Sys>SrpConfig.*k*.sys*

where *k* corresponds to the keyswitch position.

In the absence of a configuration file for a particular keyswitch position, the SRP reads *[Sys]<Sys>SrpConfig.sys.*

See "Keyswitch Positions," in Section 5, "Bootstrapping," for a description of the SRP keyswitch.

Editing the SRP Configuration File

To make changes to *SrpConfig.sys*, you use the Editor application, as described in "Editing the Configuration File" in Section 16, "Configuring Workstation Operating Systems."

To efficiently edit this file, it is helpful to be familiar with certain editing techniques, such as copying and moving lines of text. See the *CTOS Editor User's Guide* for detailed information about the Editor application.

Boot Section

The boot section identifies the processors and operating systems to be booted by the master processor. It consists of the following keywords and subparameters:

:Boot: (Processor=*Xpnn*, OS=*FileSpecification*, Dump=*Yes* or *No*)

Subparameters are separated by commas and enclosed in parentheses.

Subparameters are described below.

Subparameter	Value
Processor	Specify the four-character processor identifier, for example, **GP01**.
OS	Specify the file specification for the operating system to be booted on the processor, as shown in the following example: **[Sys]<Sys>pSrpGp.img**
Dump	Specify either **Yes** or **No**. This subparameter defines whether a memory dump takes place automatically when the processor crashes.

The boot section precedes the processor section in *SrpConfig.sys*. Only the *:LogUnknownEntries:* parameter, as shown in Figure 17-1, precedes the boot section. See its description under "Configurable Parameters," later in this section.

Note: *For the master processor, any* OS *value in the* :Boot: *list is ignored, because the master processor bootstraps from* [Sys]<Sys>SysImage.sys. *You can, however, include a* Dump *subparameter for the master processor, for example,* :Boot: (Processor=GP00, Dump=Yes)

Processor Section

The processor section contains default parameter values as well as specific parameters for individual processors. The processor-list format is described below. Configurable parameters are described in detail later in this section.

Processor entries are written in the following format:

:Processor: Xpnn

where *Xpnn* is the four-character processor ID. (See also "Processor Boards," in Section 2, "Understanding Hardware.")

A list of configurable parameters immediately follows each *:Processor:* entry. Some parameters consist of keywords and values only, as in the workstation configuration file. Others consist of keywords, subparameters, and values. The following examples show the formats for processor and parameter entries.

:Processor: Xpnn
:Keyword:Value

or

:Processor: Xpnn
:Keyword: (Subparam = Value, Subparam = Value)

where

Xpnn	Is the four-character processor identifier.
:Keyword:	Is the name of a parameter.
Subparam	Is the name of a subparameter.
Value	Is a configurable value for a parameter or subparameter.

Note the following rules of syntax for parameters and subparameters:

- The colon preceding each keyword must be the first character on the line; no spaces or characters may precede it.
- A space after the colon following a keyword is optional.
- Parentheses enclose the entire set of subparameters.
- Spaces or an equal sign (=) separate subparameters from their values.

If subparameters do not fit on a single line, they may continue on the next line, as shown below:

:Processor: Xpnn
:Keyword: (subparam = value, subparam = value,
subparam = value)

The amount of white space between keywords, parameters, subparameters, and values is not significant. In this manual, white space is used to enhance the readability of *SrpConfig.sys*.

Processor Defaults

The first entry in the processor section defines default parameters. Processor default entries are optional; however, they can reduce the number of parameters you need to specify for individual processors.

To define processor defaults, specify **Default** in place of a processor ID, as shown below:

:Processor: Default

Beneath that entry, you define parameters, as shown in the following example:

```
:Processor:  Default
   :YBlocks:   (Number = 20, Size = 2560)
   :ZBlocks:   (Number = 40, Size = 180)
   :ClusterLine1:   (Speed = 1.8M, MaxWs = 8)
   :ClusterLine2:   (Speed = 1.8M, MaxWs = 8)
   :XBlocks:   (Number = 15, Size = 2624)
```

As each processor is booted, parameters that do not apply to it are ignored. For example, cluster line defaults are implemented on Cluster Processors and General Processors only.

Entries for Specific Processors

After the list of defaults, you define parameters for individual processors. Parameters for specific processors override default parameters.

In the following example, the cluster line parameters specified for GP00 override those that were set as defaults.

```
:Processor:  GP00
   :ClusterLine1:   (Speed = 3.7M, MaxWs = 16)
```

Configurable Parameters

Configurable parameters are described below in alphabetical order. You do not need to configure every parameter; in some cases, for example, you need only to add *:MassStorage:* parameters for your disks.

In addition, all parameters do not necessarily pertain to every type of processor. For example, the *:MassStorage:* parameter pertains to processors with disk controllers only. Relevant processor types are listed for each parameter.

:ClusterLine1: (Speed = *bps*, MaxWs = *number*)
:ClusterLine2: (Speed = *bps*, MaxWs = *number*)

Processor type: Processors with cluster lines only

Default: See subparameters, below

This parameter defines the line speed and number of workstations supported on cluster communications channels. In most cases, maximum cluster line speed is dictated by the workstation hardware connected to the line. Check your workstation hardware specifications. Subparameters are described below.

Subparameter	Value
Speed	Default: 307kbps Specify **307k**, **1.8M**, or **3.7M** ("k" means kilobits per second; "M" means megabits per second).
MaxWs	Default: 16 for General Processors, 8 for Cluster Processors Specify the maximum number of workstations to be connected to the cluster line. To disable a cluster line, specify **0**.

:cParExitRunFile:

Processor type: Any

Default: 3200
Minimum: 0
Maximum: 65535

This parameter defines the minimum memory partition size for chaining or exiting to another run file. Specify a number of paragraphs.

:cParSpecHeap:

Processor type: Any

Default: 128
Minimum: 32
Maximum: 4095

This parameter defines the size of the file-specification expansion memory heap. Specify a number of paragraphs.

:cParSysCommonHeap:

Processor type: Any

Default: 16
Minimum: 10
Maximum: 4095

This parameter defines the size of the system common heap. Specify a number of paragraphs.

:CrashDumpPath:

Processor type: Any

Default: *[Sys]<Sys>*

This parameter defines a volume and directory for crash dumps and extended crash dumps of processors other than the master processor. (The master processor always dumps to *[Sys]<Sys>CrashDump.sys.*) Such crash files are named with the four-digit processor ID, followed by *.crash*, for example, *GP01.crash*.

Specify a volume and directory connected to the master processor.

:CreateDirectoryProtection:

Processor type: Processors with disk controllers only

Default: No

This entry controls whether a volume password is required to create a directory. The default is **No**. If you specify **Yes**, a volume password must be supplied to create a directory on a password protected disk.

:DebugPort:

Processor type: Any

Default: No

This parameter configures an asynchronous port for using the Debugger via Basic ATE. This parameter accepts the following values:

No specifies that a debugger port is not configured.

Yes specifies that the processors's default port is configured as the debugger port using default subparameters. Default ports are as follows:

General Processor	Channel B
Cluster Processor	Channel 3
Terminal Processor	Channel 10

Subparameters, as described below, configure the port with values that you specify.

:DebugPort: (Speed = *baud*,
Parity = *parity*,
Stopbits = *number*,
Charbits = *number*,
Modem = *Yes* or *No*,
Port = *alphanumeric*,
Processor = *x*P*nn*)

Subparameters are described below.

Subparameter	Value
Speed	Default: 9600 baud Specify **50**, **75**, **110**, **134**, **150**, **200**, **300**, **600**, **1200**, **1800**, **2400**, **4800**, **9600**, or **19200** baud.
Parity	Default: None Specify **Odd**, **Even**, **None**, **One**, **Zero**, **1** or **0**.
Stopbits	Default: 1 Specify **1**, or **2**.
Charbits	Default: 8 Specify **5**, **6**, **7**, or **8**.

Modem	Default: No Specify **Yes** or **No**.
Port	Default: See below To configure the port of your choice, rather than the default, specify one of the following: GP — **A** or **B** (B is the default) GP+CI — **A** to **H** (B is the default) CP — **0** to **2** (2 is the default) TP — **0** to **9** (9 is the default)
Processor	Default: None To use the Debugger on a File Processor, Data Processor, or Storage Processor, specify the four-character processor ID of a remote board equipped with an asynchronous port. When you specify a remote processor, you must also identify a port, as shown in the following example: *:DebugPort:* (Processor = GP00, Port = B)

:DiskAllocationLimit:

Processor type: Any

Default: All available disk space
Minimum: 512 bytes
Maximum: 4G bytes

This parameter specifies the maximum size for any one file, to prevent errant programs from consuming all disk space. Specify a number of bytes.

:DiskLogThreshold:

Processor type: Processors with disk controllers only

Default: 0
Minimum: 0
Maximum: 1024

Specify a number between **0** and **1024** to define the number of acceptable retries before I/O errors are logged. Unrecoverable errors are always logged.

:DiskRetryCount:

Processor type: Processors with disk controllers only

Default: 4
Minimum: 0
Maximum: 1024

Specify a number between **0** and **1024** to define how many times a disk operation is retried before it terminates with an I/O error.

:EnterDebuggerOnFault:

Processor type: Protected mode only

Default: No

This parameter defines whether the debugger is automatically entered when a protected-mode fault occurs.

Specify **Yes** to enable, or **No** to disable this option. Exercise caution when using this option. When an application faults and enters the Debugger, other processes running on the board are also suspended.

:FileCacheDefaultEnable:

Processor type: Processors with disk controllers only

Default: Yes

This parameter sets the default condition for file system caching. Specify **Yes** to enable caching on all files that are not specifically disabled. Specify **No** to disable caching on all files that are not specifically enabled.

This parameter alone does not configure or implement file system caching on the processor; see the *:FileCacheService:* parameter, below. See also Section 14, "Using a File System Cache," for detailed information about file cache configuration requirements.

:FileCacheService: (BlockSize=***bytes***, BlockCount=***number***, MinWorkingSetBlockCount=***number***)

Processor type: Protected mode only

Defaults: See subparameters, below

This parameter installs a file system cache on a protected-mode processor. Subparameters are described below.

Subparameter	Value
BlockSize	Default: 4096 on GP+SI, 0 (no cache) on GP or GP+CI Specify the number of bytes to be allocated for each block. It must be a multiple of 4096.
BlockCount	Default: 128 on GP+SI, 0 (no cache) on GP or GP+CI Minimum: 64K bytes divided by block size Maximum: Depends on available memory Specify the number of blocks to be allocated. Block count times block size equals the total amount of memory required for the cache. In most cases, increase this value, rather than *BlockSize*, to increase the size of the cache.
MinWorkSetBlkCnt	Default: See below Specify the number of blocks that cannot contain locked-in files. The default is 64K bytes divided by the block size. For example, if block size is 4096 bytes, the default is 16. If you change the block size, the default value changes accordingly.

:FileStructureVerify:

Processor type: Processors with disk controllers only

Default: No

This entry defines whether file system self-checking is implemented. Specify **Yes** to enable, or **No** to disable this option. Specifying **Yes** can retard disk operations by 50 percent.

:LoadableRequestFile:

Processor type: Any

Default: *[Sys]<Sys>Request.sys*

This parameter defines the file specification of the request file to be loaded when a processor boots. It must be located on a volume connected to the master processor.

:LoadDebugger:

Processor type: Any

Default: No

Specify **Yes** to load the Debugger into memory when the workstation boots. If you specify **No** or leave this field blank, the Debugger is not loaded.

:LogUnknownEntries:

Processor type: Any

Default: Yes

This parameter determines whether unrecognized keywords in *SrpConfig.sys* are written to the system error log. It is placed first in the file, before the *:Boot:* list and *:Processor:* entries.

If you specify **No**, unrecognized entries are not logged. If you specify **Yes**, unrecognized entries are written to the system error file.

:MassStorage: (see below for subparameters)

Processor type: Processors with disk controllers only

Defaults: None

This parameter defines device names and device passwords for the disk drives connected to the processor. It is also used to configure a memory disk.

Subparameters for SCSI Drives:

:MassStorage: (Class = *type*,
Adaptor = *number*,
Target = *number*,
LUN = *number*,
Device = *name*,
Password = *password*)

Subparameters are described below. See Figure 17-1 for examples.

Subparameter	Value
Class	Default: SCSI Specify **SCSI**.
Adaptor	Default: 0 Specify **0** for devices connected to the first channel on the SCSI interface, **1** for the second channel.
Target	Default: 0 Specify a number from **0** to **7** to designate the target setting on the device (see the documentation for your SCSI device).
LUN	Default: 0 Specify **0**. (At a future time, logical unit numbers from 0 to 63 will be supported.)
Device	Default: See below Specify a device name to be assigned to the drive. By convention, SCSI disk devices are named d*n*, where *n* is a number (usually matching the *Target* subparameter setting), for example *d1*. Default device names for SCSI drives connected to the master processor are *d1* to *d15*. If, however, fewer than 15 SCSI disks are connected to the master processor, you can reassign device names to disks controlled by different processors. For example, if the master processor controls *d1* to *d3*, you can continue sequential numbering (*d4* to *d7*) in the next cabinet.

Password	Default: See below Specify a password to be assigned to the drive. By convention, device passwords match device names. For disks connected to the master processor, default device passwords match device names.

Subparameters for Non-SCSI Drives:

:MassStorage:	(Class = *type*, Unit = *number*, Device = *name*, Password = *password*)

Subparameters are described below. See Figure 17-1 for examples.

Subparameter	**Value**
Class	Specify **ST506** or **SMD**.
Unit	Specify a number to denote the position of the drive's connection to the processor (see the appropriate hardware installation manual). For a memory disk, use the number you assign in the device name (see the *Device* subparameter, below).
Device	Default: See below Specify a device name to be assigned to the drive. By convention, ST-506 disk drives are named d*n*, where *n* is a number (for example, *d1*); SMD drives are named s*n* (for example, *s1*). Default device names for the master processor are *d1* to *d3* for ST-506 drives and *s0* to *s5* for SMD drives.
Password	Default: See below Specify a password to be assigned to the drive. For disks connected to the master processor, default device passwords match device names.

Subparameters for Memory Disks

:MassStorage:	(Class = *type*, Unit = *number*, Device = *name*, Password = *password* Volume = *name* MaxSectors = *number* MaxDirectories = *number* MaxSysFiles = *number* MaxTempFiles = *number* MaxFiles = *number*)

Subparameters are described below. See Figure 17-1 for examples.

Subparameter	Value
Class	Specify **CacheMemory** or **Memory**. A cache memory disk can be used as a scratch volume; see "Using a Cache Memory Disk" in Section 15, "Optimizing System Performance." A memory disk can also be used to create a bootable tape for software installation; see the Create Boot Tape command in the *CTOS Executive Reference Manual.*
Unit	Specify a number to uniquely identify each memory disk or cache memory disk on the processor. In most cases, this value will be 0, because it is unusual to configure more than one disk in memory per processor.
Device	Specify a device name. You can assign any name, but by convention, cache memory disks are named CMd*n* (for example, *CMd0*) and memory disks are named m*n* (for example, *m0*), where *n* matches the number you assigned as the Unit subparameter.
Password	Specify a password to assign to the memory disk.

Note: *The remaining subparameters, described below, pertain to cache memory disks only.*

Volume	Specify a volume name for the cache memory disk.
MaxSectors	Specify the maximum amount of memory, in sectors, for the cache memory disk to occupy.
MaxDirectories	Specify the maximum number of directories that can be created on the cache memory disk.
MaxSysFiles	Specify the maximum number of files that can be created in the cache memory disk's *<Sys>* directory.
MaxTempFiles	Specify the maximum number of files for the *<$000>* directory.
MaxFiles	Specify the maximum number of files that can be created on the cache memory disk.

Caution

Do not change the values of the *:MaxXBlocksOut:* and the *:nRepollActive:* parameters, described below, unless instructed to do so by a Technical Support engineer. Incorrect values can cause total system failure.

:MaxXBlocksOut:

Processor type: Processors with cluster lines only

Default: 5
Minimum: 1
Maximum: Number of allocated X-blocks

Specify the maximum number of X-blocks that can be outstanding for each workstation.

:nRepollActive:

Processor type: Processors with cluster lines only

Default: 0
Minimum: 0
Maximum: 5

Specify the number of times to repoll workstations.

:nRkvsUsers:

Processor type: Any

Default: 2
Minimum: 0
Maximum: 14

This parameter defines the number of Remote User Manager sessions that can be started on a protected-mode processor. Specify a number. Although you can specify a large number of sessions, the maximum number of users is actually determined by the Remote User Manager.

:RemoteCacheClient: (Name = *client*, Pool = *pool*)

Processor type: Protected mode only

Defaults: None

This parameter defines a real-mode processor that can access a remote cache. Multiple occurrences of this parameter are permitted, so that more than one remote processor can be defined.

Subparameters are described below.

Subparameter	Value
Name	Specify the four-character ID of the processor that is to access a remote cache.
Pool	Specify the name of the cache pool the processor is to access. (The cache pool must be defined with the *:RemoteCachePool:* parameter.)

:RemoteCachePool: (Name = *pool*, Password = *password*, BlockSize = *bytes*, BlockCount = *number*, MinWorkingSetBlockCount = *number*)

Processor type: Protected mode only

Defaults: See subparameters, below

This parameter defines a remote cache pool that can be accessed by real-mode processors. Multiple occurrences of this parameter are permitted to define more than one cache pool.

Subparameters are described below.

Subparameter	Value
Name	Specify the name of a cache pool to be used by remote processors. Specify **Local** to enable remote access of the cache installed with the *:FileCacheService:* parameter. Or, define an additional cache by specifying a unique name that does not conflict with any device name on the system. Suggested names are *c0*, *c1*, and so on.
Password	Default: No password Specify a password if you want to assign one to the remote cache pool.
BlockSize	Default: 0 (See the *:FileCacheService:* subparameters for minimum and maximum values.) If you specify a unique cache-pool name, specify the size of blocks to be allocated for the remote cache. If you specify "Local" as the cache pool name, leave this field blank.
BlockCount	Default: 0 (See the *:FileCacheService:* subparameters for minimum and maximum values.) If you specify a unique cache pool name, specify the number of blocks to be allocated for the remote cache. If you specify "Local" as the cache pool name, leave this field blank.
MinWorkSetBlkCnt	Default: 0 (See the *:FileCacheService:* subparameters for minimum and maximum values.) If you specify a unique cache-pool name, specify the number of blocks that cannot contain locked-in files. If you specify "Local" as the cache pool name, leave this field blank.

:RemoteCacheService:	(Priority = *number*, StackSize = *bytes*, Descriptors = *number*)

Processor type: Protected mode only

Default: See subparameters, below

This parameter establishes optional internal parameters for the remote cache service. Subparameters are described below.

> **Caution**
>
> Do not change the value of the *Priority*, *StackSize*, or *Descriptors* subparameters, described below, unless instructed to do so by a Technical Support engineer. An incorrect value can cause total system failure.

Subparameter	Value
Priority	Default: 7 Minimum: 0 Maximum: 127 Specify a value from **0** (highest priority) to **127** (lowest priority) for remote cache requests.
StackSize	Default: 512 bytes Minimum: 512 bytes Maximum: No maximum Specify the number of bytes for the remote-cache stack size.
Descriptors	Default: 100 Minimum: 33 Maximum: 1000 (approximately) Specify the number of descriptors for the remote cache. These determine the number of concurrent requests the remote cache service can process.

:RequestTracker:

Processor type: Any

Default: No

This parameter enables request tracking.

Specify **Yes** if you want outstanding requests to be copied to an exchange for tracking.

:RkvsFile:

Processor type: Any

Default: *[Sys]<Sys>Rkvs.run*

This parameter defines the name of the Remote Keyboard Video Service run file to be loaded when the processors boots.

Specify the file specification of the RKVS run file. It must be located on a volume attached to the master processor.

:sBroadcastHeap:

Processor type: Any

Default: 1024
Minimum: 100
Maximum: 65520

This parameter defines the count of bytes to limit the number of outstanding requests being broadcast. Specify a number of bytes.

:ScratchVolumeName:

Processor type: Any

Default: *Sys*

This parameter specifies the disk that is used as the "scratch" volume (*[Scr]*). The scratch volume is used for temporary storage by certain commands and applications.

Enter a volume or device specification, without square brackets. The disk you specify must be connected to the master processor. If you specify a volume other than *[Sys]*, you must create a directory named *<$000>*, with a capacity of 750 files, on that disk.

:SCSIManagerName:

Processor type: Processors with disk controllers only

Default: See below

Enter up to 12 characters to specify a SCSI manager name for the processor. This name is used by applications to route requests to multiple SCSI managers.

Default values begin with SCSI, for the SCSI manager on the first GP, SCSI1, for the second, and so on.

:SequentialStorage: (see below for subparameters)

Processor type: Processors with tape drive controllers only

Defaults: None

This parameter defines the device name for a QIC or half-inch tape drive.

This parameter can also be implemented on the QIC Interface board (which controls non-SCSI QIC tape drives) by making an entry for any one of the processors in the same cabinet. For clarity in the configuration file, such an entry is usually listed under the File Processor or General Processor adjacent to QIC Interface board.

Subparameters for SCSI Drives

:SequentialStorage: (Class = *type*,
Adaptor = *number*,
Target = *number*,
LUN = *number*,
Device = *name*,
Password = *password*)

Subparameters are described below. See Figure 17-1 for an example.

Subparameter	Value
Class	Default: None Specify **SCSI.**
Adaptor	Default: 0 Specify **0** for devices connected to the upper channel on the SCSI interface, **1** for the lower channel.
Target	Default: 0 Specify a number from **0** to **7** to designate the target setting on the device (see the documentation for your SCSI device).
LUN	Default is 0. Specify **0**.
Device	Default: None Specify a device name to assign to the tape drive. This can be any name you choose, for example, *QIC, QIC0,* or *Fred.*
Password	Default: None Specify a password to assign to the drive. By convention, device passwords match device names.

Subparameters for non-SCSI Tape Drives

:SequentialStorage:

(Class = *type*,
Unit = *number*,
Device = *name*,
Password = *password*)

Subparameters are described below. See Figure 17-1 for an example.

Subparameter	Value
Class	Default: None Specify **QIC36** or **HalfInch**.
Unit	Default: None This subparameter applies to half-inch tape drives only. Specify the number between **0** and **7** that corresponds to the ID switch setting on the tape drive unit. (See the documentation for the tape drive.)
Device	Default: None Specify a device name to assign to the tape drive. This can be any name you choose, for example, *QIC, QIC0,* or *Sam.*
Password	Default: None Specify a password to assign to the drive. By convention, device passwords match device names.

:SwapFile:

Processor type: Protected mode only

Default: *[Sys]<Sys>CrashDump.sys*

This parameter defines the file to be used for swapping the contents of a memory partition to disk. Specify a file specification for the swap file. The default is *[Sys]<Sys> CrashDump.sys*, which serves a dual purpose. It is used as the swap file while the system is running, and as the crash dump file, should the system crash.

:SwapFileSize:

Processor type: Protected mode only

Default: 1500
Minimum: 1
Maximum: Depends on available disk space

This parameter defines the minimum starting size of a swap file. Specify a number of sectors for the swap file.

:SwapFileSizeMax:

Processor type: Protected mode only

Default: See below

This parameter controls the maximum size of a disk swap file. If you limit its size, you maintain room on the disk for other uses. Specify a number to designate the maximum number of sectors for the swap file. The default is 0, which indicates that all available disk space will be used.

:VDMFile:

Processor type: Any

Default: *[Sys]<Sys>InstallVDM.run*

This parameter defines the file that installs video services. It must be located on a volume connected to the master processor. Specify the file specification for video installation.

:WakeUpInterval:

Processor type: Protected mode only

Default: See below
Minimum: 0
Maximum: 6553

This parameter specifies whether a time-slicing swap policy is to be used and determines how often the scheduler searches for contexts to swap back into memory. Specify a time interval in seconds. The default, 0, indicates that swapping occurs only on demand.

:WatchDogStatus:

Processor type: Master processor only

Default: SetFlag

The master processor is the first processor in the primary cabinet. This parameter defines the action the master processor takes when another processor crashes. If you specify **SetFlag**, the front panel is set to 40, while unaffected processors continue to run. This is helpful for quickly detecting crashes on processors other than the master processor.

If you specify **None**, other processors continue to run; however, the front panel is not reset. This can cause delays in detecting the crash, and unpredictable errors can occur because interprocessor communications cannot take place with the processor that has crashed. You might, however, want to omit the watchdog when running diagnostics or using the Debugger.

If you specify **Crash**, the master processor forces the entire system to crash. This immediately alerts all users that there is a problem. The front panel status is set to 40 when the watchdog shuts down the system.

:WBlocks: (*Number* = number, *Size* = bytes)

Processor type: Any

Defaults: See subparameters, below

This parameter specifies the number of W-blocks to be allocated on the processor. W-blocks are very large memory buffers used for interprocessor data transfers. Subparameters are described below.

Subparameter	Value
Number	Default: 0 Minimum: 0 Maximum: 64K bytes divided by W-block size Specify the total number of W-blocks to be allocated on the processor.
Size	Specify the number of bytes to be allocated for each W-block.

:XBlocks: (*Number* = number, *Size* = bytes)

Processor type: Processors with cluster lines only

Default: See subparameters, below

This parameter defines the number and size of large X-blocks to be allocated on the processor. Subparameters are described below.

> **Caution**
>
> Do not change the value of the *Size* subparameter, described below, unless instructed to do so by a Technical Support engineer. An incorrect value can cause total system failure.

Subparameter	Value
Number	Default: 5 Minimum: 1 Maximum: 64K bytes divided by X-block size Specify the number of large X-blocks to be allocated on the processor.
Size	Default: 4160 Minimum: 2656 Maximum: 64K Specify the number of bytes to be allocated for each large X-block.

:XBlocksSmall: *(Number = number, Size = bytes)*

Processor type: Processors with cluster lines only

Default: See subparameters, below

This parameter defines the number and size of small X-blocks to be allocated on the processor. Subparameters are described below.

> **Caution**
>
> Do not change the value of the *Size* subparameter, described below, unless instructed to do so by a Technical Support engineer. An incorrect value can cause total system failure.

Subparameter	Value
Number	Default: 28 Minimum: 0 Maximum: 64K bytes divided by small X-block size Specify the number of small X-blocks to be allocated on the processor.
Size	Default: 96 Minimum: 0 Maximum: 64K Specify the number of bytes to be allocated for each small X-block.

:YBlocks: (*Number* = number, *Size* = bytes)

Processor type: Any

Defaults: See subparameters, below

This parameter defines the number of Y-blocks to be allocated on the processor. Y-blocks are large memory buffers used for interprocessor data transfers. Subparameters are described below.

> **Caution**
>
> Do not change the value of the *Size* subparameter, described below, unless instructed to do so by a Technical Support engineer. An incorrect value can cause total system failure.

Subparameter	Value
Number	Default: 4 Minimum: 1 Maximum: 64K bytes divided by Y-block size Specify the total number of Y-blocks to be allocated on the processor.
Size	Default: 2656 bytes Minimum: 2656 bytes Maximum: 64K bytes Specify the number of bytes to be allocated for each Y-block.

:ZBlocks: (Number = *number*, Size = *bytes)*

Processor type: Any

Defaults: See subparameters, below

This parameter defines the number of Z-blocks to be allocated on the processor. Z-blocks are small memory buffers used for interprocessor data transfers. Subparameters are described below.

Caution

Do not change the value of the *Size* subparameter, described below, unless instructed to do so by a Technical Support engineer. An incorrect value can cause total system failure.

Subparameter	Value
Number	Default: 28 Minimum: 0 Maximum: 64K bytes divided by Z-block size Specify the total number of Z-blocks to be allocated on the processor.
Size	Default: 180 bytes Minimum: 0 Maximum: 64K bytes Specify the number of bytes to be allocated for each Z-block.

Section 18
Building a Customized Operating System

When to Build an Operating System

It is possible to customize parameters that are not included in the operating system configuration file. To do so, you *build* or *SysGen* a customized version of the operating system.

You do not need to be a programmer to build a customized operating system; however, as a system administrator, you most likely will be carrying out instructions you receive from someone else, such as a Technical Support engineer. If your questions are not answered in this manual, check with a Technical Support engineer before you build and use a customized operating system.

Building an operating system includes the following steps, which are described in this section.

1. Installing the System Build Utilities and the CTOS Development Utilities
2. Making changes to the source code
3. Assembling and linking the new operating system

To customize an operating system, you must know how to use the Editor application; see the *CTOS Editor User's Guide* for detailed information.

Installing the System Build Utilities

Workstations

The workstation System Build Utilities and are packaged separately from the operating system distribution media. See the CTOS Software Release Announcement for installation instructions.

Shared Resource Processors

The SRP System Build Utilities are supplied with the operating system distribution media. See the CTOS/XE Software Release Announcement for installation instructions.

In addition to the System Build Utilities software, you will need to install the CTOS Development Utilities. See the Software Release Announcement for that product for installation instructions.

Making Changes to the Source Code

After you install the appropriate software, use the Editor application to open and read one of the following files; each file contains instructions for customizing operating system parameters:

CTOS I

<3.4rGen>rmSysGen.asm

CTOS II

<3.4pGen>SysGen.asm

CTOS III

<1.0vGen>SysGen.asm

CTOS/XE

<Gen>SysGen.asm

Customizable source code is contained in *prefix files*, which you modify to customize the operating system. Prefix files are named as follows:

Prefix*.asm*

where *Prefix* is a prefix, as listed in Table 18-1 for CTOS I, Table 18-2 for CTOS II, Table 18-3 for CTOS III, and Table 18-4 for CTOS/XE.

For CTOS I, all customizable parameters are contained in a single prefix file. Therefore, regardless of the parameters you need to change, you edit only that single file.

For CTOS II, CTOS III, and CTOS/XE, the file system parameters are in a separate prefix file. In addition, for CTOS II and CTOS III, cluster agent parameters are in yet another prefix file. Therefore, depending on the changes you need to make, you may need to edit more than one file.

To modify a prefix file, you use the Editor application, as described in the following procedure.

1. Use the **Path** command, as shown below, to set your path to the build directory containing the files you want to modify.

```
Path
  [Volume]                     d0
  [Directory]                  BuildDir
  [Default file prefix]        ____________
  [Password]                   ____________
  [Node]                       ____________
```

   Note: *In the command form shown above, substitute the name of the actual build directory in place of* BuildDir. *For CTOS I, the build directory is named* **3.4rGen**; *for CTOS II,* **3.4pGen**; *for CTOS III,* **1.0vGen**; *and for CTOS/XE,* **Gen**.

2. On the Executive command line, type **Editor**; then press RETURN.
3. Type the file name of the prefix file you want to modify, as shown in the following example:

```
Editor
  File name(s)                 pSrvrM.asm
  [Read only?]                 ____________
  [Alternate user name]        ____________
```

4. Press GO to open the file.
5. Edit the parameter(s) you want to customize.
6. After you have finished making changes, press FINISH, then GO, to exit the Editor and save the file.

Table 18-1. CTOS I Prefix Files

Prefix	Description
t1Srvr	B26 real-mode workstation server
t1Clstr	B26 and CWS real-mode cluster workstations (diskless)
t1ClstrLfs	B26 real-mode cluster workstation (local file system)
t1Stnd	B26 real-mode standalone workstation
v1Clstr	B24 real-mode cluster workstation (diskless)
bawsClstrLfs	B27 real-mode cluster workstation (local file system)
bawsClstr	B27 real-mode cluster workstation (diskless)
bawsStnd	B27 real-mode standalone workstation

Table 18-2. CTOS II Prefix Files

Prefix	Description
FsS_N	File system for server supporting less than 16 workstations or for cluster workstation with local file system
FsM_N	File system for server supporting 16 to 24 cluster workstations
FsL_N	File system for server supporting 24 to 32 cluster workstations
Srvr	Cluster agent for server
Clstr	Cluster agent for cluster workstation
pSrvrS	Protected-mode server supporting less than 16 cluster workstations
pSrvrM	Protected-mode server supporting 16 to 24 cluster workstations
pSrvrL	Protected-mode server supporting 24 to 32 cluster workstations
pClstrLfs	Protected-mode cluster workstation (local file system)
pClstr	Protected-mode cluster workstation (diskless)
pStnd	Protected-mode standalone workstation

Table 18-3. CTOS III Prefix Files

Prefix	Description
FsS_N	File system for server supporting less than 16 workstations or for cluster workstation with local file system
FsM_N	File system for server supporting 16 to 24 cluster workstations
FsL_N	File system for server supporting 24 to 32 cluster workstations
Srvr	Cluster agent for server
Clstr	Cluster agent for cluster workstation
vSrvrS	Virtual memory server supporting less than 16 cluster workstations
vSrvrM	Virtual memory server supporting 16 to 24 cluster workstations
vSrvrL	Virtual memory server supporting 24 to 32 cluster workstations
vClstrLfs	Virtual memory cluster workstation (local file system)
vClstr	Virtual memory cluster workstation (diskless)
vStnd	Virtual memory standalone workstation

Table 18-4. CTOS/XE Prefix Files

Prefix	Description
Fs_Si	File system for General Processors with SCSI Interface
Fs_Fp	File system for File Processors
Fs_Dp	File system for Data Processors
pSrpGp	Protected mode General Processors
rSrpFp	Real mode File Processors
rSrpCp	Real mode Cluster Processors
rSrpTp	Real mode Terminal Processors
rSrpDp	Real mode Data Processors
rSrpSp	Real mode Storage Processors

Assembling and Linking

After you have made changes to the prefix file(s), you assemble and link them to build a new operating system. Procedures vary according to the prefix files you customized, as described in the following sections.

File System Prefix File

Note: *If you are customizing a CTOS I operating system, or if you did not make changes to the file system, skip this section.*

To assemble and link the file system, follow these steps:

1. On the Executive command line, type **Assemble**, then press RETURN.
2. Enter the name of the file system prefix file, as shown in the following example:

```
Assemble
 File name          FsM_N.asm
 [Errors only?]     yes
```

3. Press GO.

   If no error messages occur during the assembly operation, proceed to the next step. If errors occur, see "Troubleshooting SysGen Errors," later in this section.

4. On the Executive command line, type **Link File System**; then press RETURN.
5. Fill in the command form, as shown in the following example.

```
Link File System
 File system type (e.g. FsM_N)  FsM_N
 Version                        3.4-7/30
 [File system name (FileSys)]
```

   Notice that you can include a date or other brief description to differentiate the customized file system from the prebuilt version.

6. Press GO.
7. Assemble and link the cluster agent and/or the operating system prefix file, as described in the sections that follow.

Cluster Agent Prefix File

Note: *If you are customizing a CTOS I or CTOS/XE operating system, or if you did not make changes to the cluster agent, skip this section.*

To assemble and link the cluster agent, follow these steps:

1. On the Executive command line, type **Assemble**; then press RETURN.
2. Enter the name of the cluster agent prefix file, as shown in the following example:

```
Assemble
 File name         Srvr.asm
 [Errors only?]    yes
```

3. Press GO.

   If no error messages occur during the assembly operation, proceed to the next step. If errors occur, see "Troubleshooting SysGen Errors," later in this section.

4. On the Executive command line, type **Link Agent**; then press RETURN.
5. Fill in the command form, as shown in the following example.

```
Link Agent
 Agent type (Srvr or Clstr)  Srvr
 Version                     3.4-7/30
```

   Notice that you can include a date or other brief description to differentiate the customized agent from the prebuilt version.

6. Press GO.
7. Assemble and link the operating system prefix file, as described in the next section.

Operating System Prefix File

To assemble and link the operating system prefix file, follow these steps:

1. On the Executive command line, type **Assemble**; then press RETURN.
2. Enter the name of the prefix file for the operating system you want to build, as shown in the following example.

   ```
   Assemble
    File name         pSrvrM.asm
    [Errors only?]    yes
   ```

   Notice that you can include a date or other brief description to differentiate the customized operating system from the prebuilt version.
3. Press GO.

   If no error messages occur during the assembly operation, proceed to the next step. If errors occur, see "Troubleshooting SysGen Errors," later in this section.
4. On the Executive command line, type the appropriate command name from the list below; then press RETURN.

   Link CTOS I (real-mode workstation)

   Link CTOS II (protected-mode workstation)

   Link CTOS III (virtual-memory workstation)

   Link CTOS (real-mode SRP)

   Link CTOS VM (protected-mode SRP)
5. Enter the operating system prefix and a version number, as shown in the following example:

   ```
   Link CTOS II
    Operating system type (e.g. pClstr)  pSrvrM
    Version                              3.4-7/30
   ```
6. Press GO.

Testing the New Operating System

On a workstation, you use the **Bootstrap** command to test the new operating system before you copy it to *[Sys]<Sys>SysImage.sys*. On an SRP, you use different methods for the master processor and the other processors. All methods are described below.

When the new operating system is running on the workstation or SRP, try to reproduce the problems or inadequacies you were experiencing with the original operating system. Keep track of whether or not they improve.

If the operating system will not boot, see "Troubleshooting SysGen Errors," later in this section.

On a Workstation

To load the new operating system, use the **Bootstrap** command, as described in the following procedure:

1. Make sure your path is still set to the appropriate build directory.
2. On the Executive command line, type **Bootstrap**; then press **RETURN**.
3. Specify the name of the operating system file. After build, operating systems are named with the operating system prefix and one of the following suffixes:
   - *.img* for protected-mode operating systems
   - *.run* for real mode-operating systems

   For example:

   ```
   Bootstrap
    File name                [d0]<3.4pGen>pSrvrM.img
    [Sys volume or wsNNN]
   ```
4. Press **GO**.

On an SRP Master Processor

To test an operating system on the master processor of an SRP, you can copy it to a QIC tape and then bootstrap from the tape.

To copy the customized operating system to a QIC tape, follow these steps:

1. Insert a blank QIC tape into the tape drive on the SRP.
2. On the Executive command line, type **Tape Copy**; then press RETURN.
3. Fill in the command form, as shown in the following example:

```
Tape Copy
 File from          [d2]<Gen>pSrpGp.img
 File to            [QIC]0
 [Overwrite ok?]
```

4. Press GO to copy the file.

To bootstrap from the tape, reset the SRP to the M (manual) keyswitch position.

If you prefer, you can boot an SRP master processor with the **Bootstrap** command via Cluster View. In some cases, though, real mode SRP processors do not contain enough memory to execute the **Bootstrap** command.

After the SRP boots, you can use the **Partition Status** command via Cluster View to make sure that it has booted the customized operating system.

On Other SRP Processors

To test the new operating system on SRP processors other than the master processor, edit the operating system configuration file, as shown in the following example. Note that the value of the *OS* subparameter is the file specification of the customized operating system.

```
:Boot: (Processor = GP01, OS = [d2]<Gen>pSrpGp.img)
```

See Section 17, "Configuring Shared Resource Processor Operating Systems," for more detailed information about the operating system configuration file.

Installing the New Operating System

On workstations and SRP master processors, the new operating system bootstraps on a one-time basis with the methods described above. If you turn off or reset the system, the original operating system bootstraps from *[Sys]<Sys>SysImage.sys*. Once you are satisfied that the new operating system is functioning correctly, you can permanently install it in *[Sys]<Sys>SysImage.sys*.

To install the new operating system into *[Sys]<Sys>SysImage.sys*, use the **Copy** command, as shown in the following example:

```
Copy
  File from           [d0]<3.4pGen>pSrvrM.img
  File to             [Sys]<Sys>SysImage.sys
  [Overwrite ok?]     yes
  [Confirm each?]
```

Troubleshooting SysGen Errors

If you have made inappropriate changes or typographical errors when editing a prefix file, errors will occur during the assembly or link operations.

Assembly Errors

The most common cause of errors during the assembly operation is that you have specified a value that is too large for the parameter field, or that you have accidentally deleted the punctuation marks that enclose comments.

With the Editor or the **Type** command, examine the *.lst* file that is created during the assembly operation. This file contains a list of errors that occurred while the source file was being assembled. Such files are named as follows:

Prefix.*lst*

where *Prefix* matches the prefix of the source file you are attempting to assemble. (See Table 18-1 for CTOS I, Table 18-2 for CTOS II, Table 18-3 for CTOS III, or Table 18-4 for CTOS/XE.)

Reedit the source file to fix the error; then reassemble the source file with the **Assemble** command. Repeat this process until the assembly operation is executed with no errors.

Link Errors

The most common cause of errors during the link operation is not having all the required files in your build directory. This can happen if you did not use the proper command to install the System Build Utilities software, or if some files have been accidentally deleted from the build directory.

If this should happen, examine the *.map* file that is created during the link operations. This file contains information about which files are missing or invalid during the link operation. Such files are named as follows:

Prefix.*map*

where *Prefix* matches the prefix of the file you are attempting to link. (See Table 18-1 for CTOS I, Table 18-2 for CTOS II, Table 18-3 for CTOS III, or Table 18-4 for CTOS/XE.)

To correct link errors, copy the required files from the distribution media into your build directory; then execute the link operation again.

Bootstrap Errors

Some errors that prevent the operating system from booting are not detected during assembly and linking. Bootstrap errors usually occur because you have allocated too much memory for certain parameters, particularly file system parameter values, such as *nFab* and *nVhb*.

If this should happen, make the parameter values smaller by editing the prefix file again; then rebuild the operating system.

Section 19
Customizing Standard Software

What You Can Customize

For Standard Software and many other applications, you can customize messages and prompts that appear on the screen. The following components of Standard Software can be customized:

- *Message files*, which contain prompts and error messages for most Standard Software commands
- *Template files*, which contain menus and function key labels for certain commands only
- The SignOn messages for both the conventional and window-interface screens

Customization is most frequently performed to translate messages into other languages. However, it can also be performed to clarify or change the wording of English messages. *Be aware, though, that significant changes to screen messages can make it difficult for users to follow the published documentation.*

Customizing Message Files

Screen messages are contained in *binary message files*. Binary message files contain machine-readable code, which is interpreted by programs to display messages. Messages for Standard Software are contained in a number of different message files. See the Software Release Announcement for a list of message files and the commands for which they contain messages.

This section contains procedures for customizing screen messages. See also the *CTOS Executive Reference Manual* for more information about the **List Message File** and **Create Message File** commands.

Generating a Message Text File

The binary message file does not contain readable text. You can, however, generate an ASCII output file that translates a binary message file into readable text. Such a file is called a *message text file*.

To generate a message text file, follow these steps:

1. On the Executive command line, type **List Message File**; then press RETURN.
2. Fill in the command form, as shown in the following example:

   ```
   List Message File
    Binary file   [Sys]<Sys>ExecMsg.bin
    [Text file]
   ```

   In the *Binary file* field, enter the name of the message file you want to customize.

   You can leave the *[Text file]* field blank. By default, the text file name is the same as the binary file, except for its suffix. Listed text files end with *.txt*, while binary files end with *.bin*.
3. Press GO.

Editing a Message Text File

To make changes to a message text file, open the message text file with the Editor, as shown in the following example:

```
Editor
 [File name(s)]              [Sys]<Sys>ExecMsg.txt
 [Read only?]
 [Alternate user name]
```

General instructions for using the Editor are provided in Section 4, "Using Administrative Tools." For more detailed information, see the *CTOS Editor User's Guide*.

Observe the following rules and guidelines as you edit a message text file:

- Change only messages and prompts enclosed in quotation marks (").
- Do not change numbers surrounded by colons (:).
- Do not change characters preceded by a percent sign (%).
- Do not change file specifications.
- Keep new messages approximately the same length as the original messages.
- Add comments by preceding text with a semicolon (;).

Creating a Binary Message File

After you have edited the message text file, you create a new binary message file. To do so, use the **Create Message File** command, as described in the following procedure.

Caution

Before you begin the following procedure, make a copy of the original binary message file. By doing so, you can easily restore the original version, if necessary. For example, you could copy *ExecMsg.bin* to a file named *CustomExecMsg.bin*, or a similar descriptive name. Do not, however, rename any of the message files. The exact file names, as listed in the Software Release Announcement, are required for message files to work properly .

1. On the Executive command line, type **Create Message File**; then press **RETURN**.
2. Fill in the command form, as shown in the following example:

```
Create Message File
  Text file         [Sys]<Sys>ExecMsg.txt
  [Message file]
  [Print file]
```

   In the *Text file* field, specify the name of the message text file that you modified with the Editor.

   You can leave the *[Message file]* field blank. By default, the binary file name is the same as the text file, except for its suffix. Binary message files end with *.bin*, while text files end with *.txt*.

3. Press GO.

After the command has been executed, changes in the new binary message file are automatically implemented. You do not need to log out or reboot the workstation.

Merging Message Files

When you update software on your system, you can merge customized message files with those that are newly released. This preserves the customized files you have created but merges new messages into them. See the **Merge Message Files** command in the *CTOS Executive Reference Manual* for more detailed information.

Customizing Templates

The **System Manager** and **User File Editor** commands use template files to display menus and function-key labels. What you see on the screen can be changed by modifying the template file.

Template files for the **System Manager** and **User File Editor** commands are named as follows:

[Sys]<Sys>SystemMgrConfig.sys

[Sys]<Sys>UserFileTemplate.sys

Template files are written in the *:Keyword:Value* format, which is described for other configuration files throughout this manual. (For an example, see Section 7, "Customizing User Environments.") To make changes to a template file, you use the Editor application.

For detailed information about template file formats and keywords, see the **System Manager** and **User File Editor** commands, in the *CTOS Executive Reference Manual.*

Customizing the SignOn Screen

By default, the conventional SignOn screen, as shown in Figure 19-1, is displayed after a workstation bootstraps. Optionally, you can configure a workstation to display the window-interface SignOn screen, as shown in Figure 19-2. In addition, you can customize the SignOn screen to display messages that are appropriate for your workplace. Procedures for customizing the SignOn display are included in this section.

Activating the Window-Interface SignOn Screen

To activate the window-interface SignOn program, the XVT software package must be installed on the workstation. It is not included with Standard Software, but is packaged and installed separately. See the software release announcement for XVT for installation instructions. For information about installing software, see Section 8, "Installing Applications," in this manual.

Note: *Some workstations do not contain enough memory to display the window-interface SignOn screen, even though the XVT software has been properly installed. In such cases, the conventional SignOn screen is displayed, and it is not unusual to notice a delay before it appears.*

Disabling the Window-Interface SignOn Screen

In some cases, you may need to install the XVT software for other applications that require it but prefer to use the conventional SignOn screen. To disable the window interface SignOn screen, delete or rename the file named *[Sys]<Sys>XVTSignOn.run*. If that file is not present, the conventional SignOn screen is displayed.

Figure 19-1. SignOn Screen: Conventional Interface

SignOn 12.2.0 (OS CTOS II 3.4.0)
This message can be customized. Mon Jul 27, 1992 10:25 AM

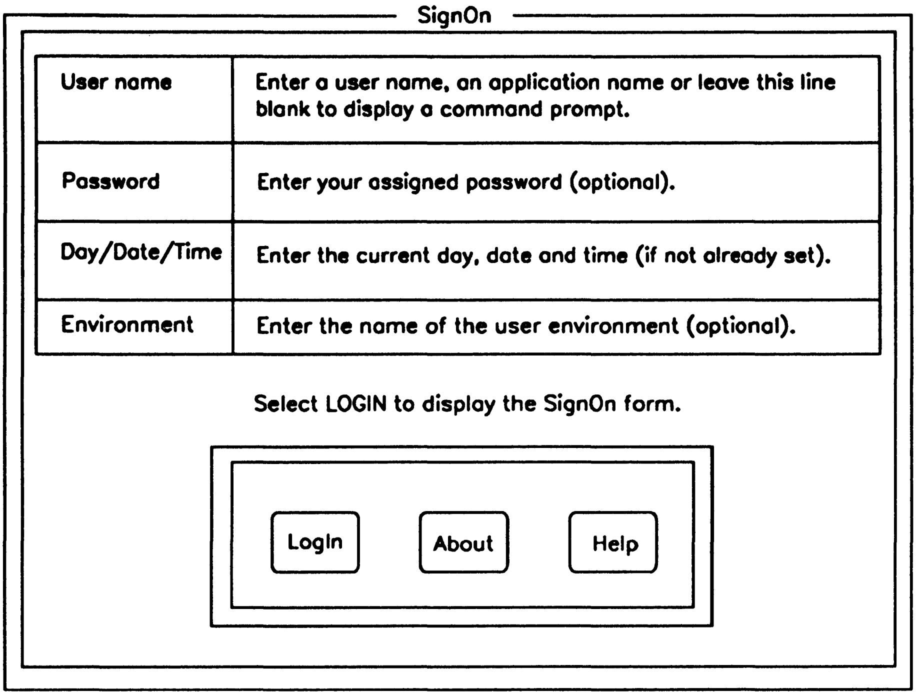
User name	Enter a user name, an application name or leave this line blank to display a command prompt.
Password	Enter your assigned password (optional).
Day/Date/Time	Enter the current day, date and time (if not already set).
Environment	Enter the name of the user environment (optional).

Then press the GO key.

User name (e.g. Pam)
Password
Date/Time (e.g. Fri Dec 13, 1991 10:00am)
User environment

543.19–1

Figure 19-2. SignOn Screen: Window Interface



543.19-2

Editing the SignOn Heading

On the conventional SignOn screen, a heading is displayed above the double bars (see Figure 19-1). It contains the version number of the SignOn program and the system date and time. Below the version number, 50 characters of customized text can be displayed on a single line. For example, this line could be used to display the name of a company or a network node.

To customize the SignOn heading, perform the steps listed below. See "Customizing Message Files," earlier in this section, for detailed procedures for creating and editing message files.

1. Create a message text file from *[Sys]<Sys>SignOnMsg.bin*.
2. Edit line 505 of that message text file to contain the 50-character line of text you want to display.
3. Create a new version of *[Sys]<Sys>SignOnMsg.bin*.

The customized message will be displayed on the conventional SignOn screen the next time you log out.

Editing the SignOn Text File

The text displayed on the SignOn screen is contained in an editable text file, named *[Sys]<Sys>SignOn.txt* for the conventional screen or *[Sys]<Sys>XVTSignOn.txt* for the window-interface screen. You can edit either SignOn text file to display a message that is appropriate for your workplace.

In addition, you can create customized SignOn text files for cluster workstations that boot from server. Such files must be located in *[Sys]<Sys>* on the server and are checked in the following order:

Conventional SignOn

*[Sys]<Sys>Hw*NNN>*SignOn.txt*

*[Sys]<Sys>Ws*NNN>*SignOn.txt*

[Sys]<Sys>Ws>SignOn.txt

Window-Interface SignOn

*[Sys]<Sys>Hw*NNN*>XVTSignOn.txt*

*[Sys]<Sys>Ws*NNN*>XVTSignOn.txt*

[Sys]<Sys>Ws>XVTSignOn.txt

(*NNN* is the three-digit hardware ID or workstation-type number.)

If none of the files listed above exists, *SignOn.txt* or *XVTSignOn.txt* is displayed.

See Section 5, "Bootstrapping," for more information about hardware IDs and workstation-type numbers.

Section 20
Troubleshooting

Diagnosing Problems

As a system administrator, you are responsible for isolating and correcting a wide variety of problems. The following commands, which are described in this section, are useful tools for diagnosing problems:

- **PLog**
- **Cluster Status**
- **Partition Status**

See also the *CTOS Status Codes Reference Manual* and your hardware and software applications manuals for more detailed information about specific problems.

PLog

You use the **PLog** command to view the system log file. It is a good idea to read the log file regularly, even if no specific problems occur. This can help you to identify marginally functioning hardware or ISAM database errors. In addition, many applications record messages and errors in the system error log.

The following events and errors are recorded in the system log file:

- System bootstrap events
- System initialization errors
- System crashes
- Disk errors
- Cluster communication errors
- ISAM errors

To invoke **PLog**, type the command name on the Executive command line; then press **GO**.

With **PLog**, you can optionally view a selected group of errors, based on error type or certain dates. You can also print the error log. See also the **PLog** command in the *CTOS Executive Reference Manual* for more information.

Each entry in the error log contains the date and time of the error, the error type, and additional information about the error. As the log file fills, newer entries replace the older entries. The following samples will help you interpret the information in the system error log.

Example 1: System Bootstrap Event

The following sample shows an entry for a system bootstrap event. It records the type of workstation, amount of memory, date and time of the event, and version of the operating system that was booted.

```
NGENT3, Cluster Workstation, With File System
Memory Size: 3584K,  SignOn User Name:   ...
SYSTEM BOOT -       Tue Jul  7, 1992  2:35 PM
Os Booted: pClstrLfs 3.4.0
```

Example 2: Disk Error

The following sample shows an entry for a disk error. Notice that this error was *recovered*; this means that no damage occurred, but such an error can signal impending problems with a disk. Check the *CTOS Status Codes Reference Manual* for a description of error status codes.

```
NGENT3,  Cluster Workstation,  With File System
Memory Size: 3584K,  SignOn User Name: Tricia
DISK ERROR - Winchester Unit 0 (ERC = 301)
Description: CRC error in data field
Number of Retries: 2 (Recovered), Volume Name: TF1
Cylinder: 418, Head: 0, Sector: 11, Num of Sectors: 5
Command: 2D
Main Status: 51, Error Status: 40
```

Example 3: System Initialization Error

The following sample shows an entry for a system initialization keyboard error.

```
386i,  Server Workstation,  No CommIop
Memory Size: 4096K,  SignOn User Name: ...
SYSTEM INITIALIZATION ERROR -
Description: No keyboard hardware
Initialization Status: 0080H
```

Example 4: Processor Crash

The following examples show entries for workstation and SRP processor crashes. Check the *CTOS Status Codes Reference Manual* for a description of the error, for example, ERC = 89; the executing instruction and crash information may be useful to a Technical Support engineer or system programmer.

Workstation

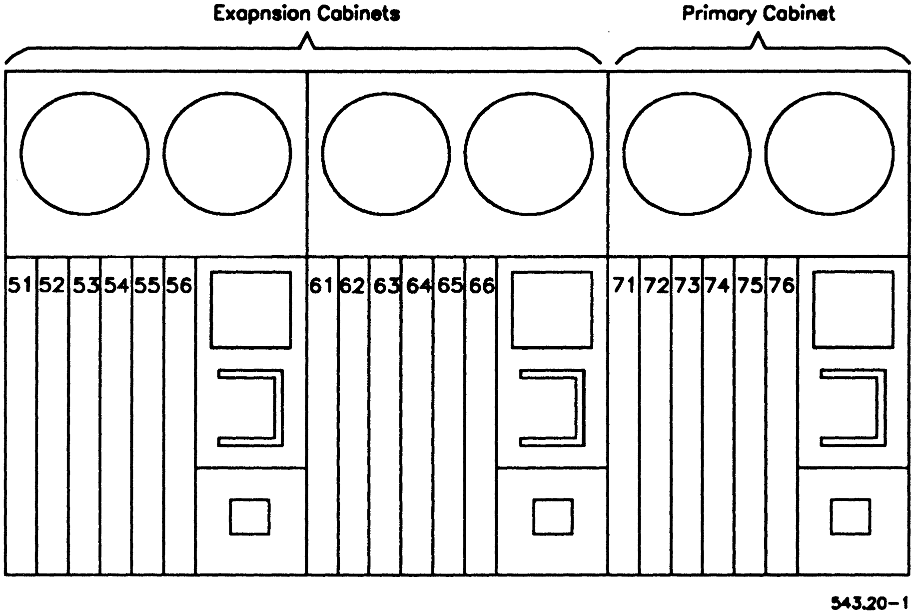
```
NGENT3,  Cluster Workstation,  With File System
Memory Size: 3584K,  SignOn User Name: Tricia
SYSTEM CRASH -  (ERC = 89)
Executing instruction preceding location 02F0:04AC
Crash Information:
0059H 0000H 0000H 0000H 8006H 1900H 02F0H 04ACH
Os Booted: pClstrLfs 3.4.0
```

SRP

```
Cluster Processor in slot 73H
Memory Size:  768K
SYSTEM CRASH - (ERC = 8105)
Executing instruction preceding location
0C84:04B2
Crash Information:
1FA9H 0005H 0000H 0000H 0073H 0000H 0C84H 04B2H
Os Booted:  rSrpCp-3.4.0
```

Note that entries for SRP processors reference a type of processor and a slot number. The slot number identifies the processor where the error occurred. Slots in additional expansion cabinets continue the same numbering scheme. Figure 20-1 shows slot numbers for a three-cabinet SRP.

Figure 20-1. SRP Slot Numbers (In Hexadecimal)



Cluster Status

The **Cluster Status** command provides information about cluster communications. It displays the SignOn user name of all workstations connected to the cluster and lists cluster communication errors.

Although the information displayed by **Cluster Status** is rather technical, certain types of errors can indicate particular problems.

To start **Cluster Status**, type the command name on the Executive command line; then press GO. On an SRP, use F10 (Next), after the command is invoked, to display information about each cluster line on the SRP.

The Errors display is shown in Figure 20-2. Error information is briefly described in Table 20-1. See "Workstation Troubleshooting," later in this section, for more information about correcting these errors. See also the *CTOS Executive Reference Manual* for more information about the **Cluster Status** command.

Figure 20-2. Cluster Status Errors Display

Cluster Status 12.2.0 Line Number 01 Mon Jul 27, 1992 1:30 PM
Server Version: vSrvr 1.0.0 Stats Since: Mon Jul 27, 1992 12:00 M
Line Speed: 1.8Mb Line Use Last sec: 0% Line Use Last 10 sec: 0%
WS Total: 24 WS Active: 16 WS Down/Timeout 1 WS Down/Errors 0

ID	User Name	Timeout	CRC	OvRun	Seq	Proto	Addr	Length
00	SERVER TOTALS	14	0	6	0	0	1	0
11	Linnea	0	0	0	0	0	0	0
12	No Name	0	0	6	0	0	0	0
13	June	0	0	0	0	0	0	0
14	WWS	0	0	0	0	0	0	0
15	Tom	1	0	0	0	0	0	0
16	Doug	0	0	0	0	0	0	0
17	Gregg	0	0	0	0	0	0	0
18	Tricia	0	0	0	0	0	0	0
19	Eric	0	0	0	0	0	0	0
1A	Diane	1	0	0	0	0	0	0
1B	Margaret	0	0	0	0	0	0	0
1C	Gloria	0	0	0	0	0	0	0
1D	Ellen	0	0	0	0	0	0	0
1E	Janet	0	0	0	0	0	1	0
1F	Andrew	1	0	0	0	0	0	0
20	Scott	0	0	0	0	0	0	0

543.20-2

Table 20-1. Cluster Status Errors

Error Field	Description
Timeout	These errors are a normal occurrence when cluster workstations are rebooting. They can also be caused by cabling problems. See "Workstation Troubleshooting," later in this section.
CRC	These errors usually occur when a cluster line is not properly terminated. They can also occur from unusually heavy cluster communications activity or cluster cable lengths that exceed supported limits. (See the *CTOS Cluster and Network Hardware Installation Guide.*)
Overrun	These errors are usually caused by hardware problems.
Sequence	These errors are usually the result of CRC errors (see above).
Protocol	These errors occur when more than one workstation on the same cluster line is running a server operating system.
Address	These errors are caused by improperly terminated cluster lines or other hardware problems.
Length	These errors usually result from cabling problems.

Partition Status

The **Partition Status** command provides information about the random access memory (RAM) on a processor. It is used to find out how memory is being used on a workstation or SRP. System administrators use this command most frequently to obtain the following information:

- Whether system services are installed
- How much memory remains available on a processor
- Whether a portion of memory is not functioning

Instructions for starting **Partition Status** and a description of the screen are contained in "Calculating Memory Requirements" in Section 9, "Installing System Services." See also the **Partition Status** command in the *CTOS Executive Reference Manual.*

Workstation Troubleshooting

This section provides information about common hardware and software problems. It is organized by symptom, as follows:

- Workstation power does not come on
- Workstation does not bootstrap
- Module is not recognized
- Keyboard does not work
- Monitor does not come on
- Workstation does not communicate with the server
- Application cannot be started
- Application is running slowly

Workstation Power Does Not Come On

Use the following list to identify and correct problems that prevent the workstation from turning on.

Possible Cause	Corrective Action
Power supplies might not be plugged in.	Check wall plugs, power cords, and the jumper cords that connect power supplies.
Power supply might be broken.	Replace the power supply with one that you know is working.
Wall outlet might not be supplying power.	Test with an outlet that you know supplies power.

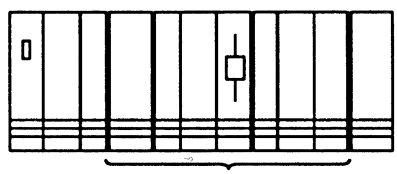
Workstation Does Not Bootstrap

Use the following list to identify problems that prevent the workstation from booting. Several of these problems are described in more detail later in this section (see "What If a System Will Not Bootstrap?," later in this section).

Possible Cause	Corrective Action
System power might not be turned on.	See "Workstation Power Does Not Come On," above.
The disk containing the system volume might not be recognized.	See "Module Is Not Recognized," below.
The system volume might be corrupted.	See "What If a System Will Not Bootstrap?," later in this section.
A customized operating system might not be bootable.	See Section 18, "Building a Customized Operating System" and "What If a System Will Not Bootstrap?," later in this section.
Hardware might be installed incorrectly.	See the appropriate hardware installation manual.
Processor or memory hardware might be failing.	Run diagnostics on the processor and memory; see the manual for diagnostics software.

Module Is Not Recognized

Use the following list to identify and correct problems that prevent a particular module from being recognized on a workstation that is otherwise functioning correctly.

Possible Cause	Corrective Action
Modules might be improperly positioned.	X-Bus and SCSI modules must be properly positioned; otherwise, some of them will not be recognized. In many cases, X-Bus modules must be placed immediately to the right of the processor, followed by SCSI modules. See your hardware installation guides, however, for more detailed information.
Too many modules might be connected to the workstation.	The X-Bus length cannot exceed 24 inches. (X-Bus length is the distance between the first X-Bus connection and the last, as shown in the following illustration. Turn off the workstation and disconnect power. Remove any modules that make the X-Bus too long.  X–Bus Length 543.20–A In addition, the number of SCSI modules is limited, depending on the workstation model. In many cases, that limitation is one SCSI Upgrade module, followed by six SCSI Expansion modules. See your hardware installation guide, however, for more detailed information.

continued

Possible Cause	Corrective Action
Modules might not be properly seated.	Turn off the workstation and disconnect power. Disconnect the offending module, and check for bent or broken pins. If the pins appear to be undamaged, reconnect each module, pressing the X-Bus firmly together before latching. If pins are broken or bent, have the module repaired before replacing it on the workstation.
More power supplies might be needed.	Each power supply can accommodate only a certain number of power units. See the installation guide for your workstation.
A power supply might be broken.	Replace the power supply with one that you know is working.

Keyboard Does Not Work

Use the following list to identify and correct problems that prevent the keyboard from working.

Possible Cause	Corrective Action
The keyboard cable might not be plugged in.	Check the attachment to the keyboard and to the monitor.
The keyboard might be broken.	Replace with a keyboard that you know is working.
The keyboard outlet on the monitor might be broken.	Replace with a monitor that you know is working.
The Remote User Manager might be installed.	Edit the system initialization file to remove the entry that installs the Remote User Manager (*RUM.run*) or that installs the Login Service (*Login.run*); then reboot the workstation. See "Installing the Remote User Manager," in Section 9, "Installing System Services."

Monitor Does Not Come On

Use the following list to identify and correct problems that prevent the monitor from coming on.

Possible Cause	Corrective Action
The brightness control might be turned down.	Adjust the brightness control knob to increase brightness. (See your workstation installation guide.)
The monitor cable might be loose or disconnected.	Check its connection to the workstation or video controller.
The monitor might be plugged into the wrong port.	If the workstation has a Graphics Module, the monitor must be attached to it rather than to the port on the processor module.
The graphics controller might be incompatible with the monitor.	The graphics controller on the workstation must support the type of monitor that is attached to the workstation. See the installation guide for the workstation processor.
The monitor might be broken.	Replace it with a monitor that you know is working.
An On/Off switch might need to be turned on.	Some monitors have an On/Off switch (see the installation guide or owner's manual for the monitor). Turn on the On/Off switch and wait a few moments for the monitor to warm up.

Workstation Does Not Communicate With the Server

Use the following list to identify and correct problems that prevent the workstation from communicating with the server. See the *CTOS Cluster and Network Hardware Installation Guide* for more information about cluster communication problems.

Possible Cause	Corrective Action
The server might be turned off or might have crashed.	Reboot the server.
Cluster cables might be disconnected.	Check connections both at the workstation and at the server.
A workstation in the cluster might not be properly terminated.	Check both ends of the daisy chain and, if required, install a terminator (see the *CTOS Cluster and Network Hardware Installation Guide*).
	With TeleCluster, make sure that each workstation is terminated.
Too many workstations might be connected to the server.	Use the **Cluster Status** command to find out how many total workstations the server supports and how many are already active.
	To support more workstations, you may need a different server or a customized operating system.
A protocol error might have occurred on a cluster workstation.	Reboot the malfunctioning workstation.
A cluster workstation might be running a server operating system.	This may not be the workstation that is receiving the error. Check the Executive status area on each workstation and install the correct operating system on the offender. You may need to reboot the server.
If using TeleCluster, the hub might be malfunctioning.	Call for service from your telephone service company or department.

Application Cannot Be Started

Use the following list to identify and correct problems that prevent applications from starting. If an error code is displayed, write it down in case you need to call Technical Support.

Possible Cause	Corrective Action
The software installation procedure might not have been performed correctly or completely.	Repeat the installation procedure by using the **Installation Manager** command. Never perform a partial installation, such as just copying the run files, of applications products.
A required system service might not be installed.	Check the release documentation for information about the system services that are required to run the application.
The disk might be full.	Check disk space with the **Volume Status** command. Perform a disk cleanup (see the *CTOS Executive User's Guide*) or install the application on a different volume. Consider adding or upgrading to a larger disk.
The installation might have failed because a password is required.	Enter the volume password (use the **Path** command as described in the *CTOS Executive User's Guide*) and repeat the installation procedure.
A password might be required to start the application.	Enter a valid password before starting the application.
System software might require updating.	Check the release documentation to determine the software requirements for the product.

Application Is Running Slowly

Use the following list to identify and correct problems that cause poor performance on your workstation.

Possible Cause	Corrective Action
The application might be running in a small partition.	Use the **Partition Status** command to determine the size of the partition and to determine whether additional memory is available. If using Context Manager, increase the partition size in the Context Manager configuration file. See Section 15, "Optimizing System Performance."
	If the workstation is not running Context Manager, deinstall unnecessary system services.
	Consider expanding memory on the workstation.
The system volume might be more than 90% full.	Use the **Volume Status** command to find out how much disk space has been used (see the *CTOS Executive User's Guide*). Perform a disk cleanup procedure to remove unnecessary files. Then, back up the disk and use **Disk Squash** to compress fragmentation or reinitialize the disk and restore the backup.
	If a printer is attached to the workstation, remove temporary print files from the *<GPS>* directory (see the *CTOS Generic Print System Administration Guide*).
	Consider adding another disk.
A portion of memory might be malfunctioning.	See the hardware diagnostics manual for the workstation.
On a virtual memory system, applications may be contending for oversubscribed memory.	See "Optimizing Virtual Memory," in Section 15, "Optimizing System Performance."

SRP Troubleshooting

Because it contains multiple processors, a number of special considerations pertain to troubleshooting an SRP. This section contains general information about identifying problems and correcting them. See the installation guides and the hardware diagnositcs manual for more detailed information.

Hardware Installation Problems

Many problems associated with the SRP can be traced back to hardware installation. Common hardware installation problems are listed below. See the appropriate installation guide for information about correcting them:

- Seating the boards incorrectly
- Leaving vacant slots between boards within an enclosure
- Installing disk drives improperly
- Placing strapping jumpers incorrectly on memory expansion boards
- Failing to terminate cluster lines (see the *CTOS Cluster and Network Hardware Installation Guide*)

Processor Crashes

The SRP provides the following tools for troubleshooting a processor crash:

- A two-digit front panel display
- An LED display on the back of each processor

This section describes how to read and interpret the status information provided by these displays.

Interpreting the Front Panel Display

The front panel displays a two-digit number to indicate system status, as listed below. See the *CTOS Status Codes Reference Manual* for more detailed information.

00 to 05	Occur during the normal bootstrap sequence when the bootstrap ROM is executing.
06 to 10	Indicate that the bootstrap ROM has completed its job and the operating system has taken over the system initialization sequence.
20	Indicates that the system is running and functioning normally.
21 to 29	Indicate that no bootable operating system was located by the bootstrap ROM.
30 to 39	Indicate a hardware error detected by the bootstrap ROM.
40	Indicates that the watchdog on the master processor has detected a processor crash.
50 to 51	Indicate an error during system service installation.

Interpreting Processor LEDs

When an SRP crashes, the front panel display will usually report the condition. If, however, a watchdog is not set (see the description of that parameter in Section 17, "Configuring Shared Resource Processor Operating Systems"), the front panel may continue to display 20 if a processor, other than the master processor, crashes. Processor crashes can always be detected by examining the processor status lights.

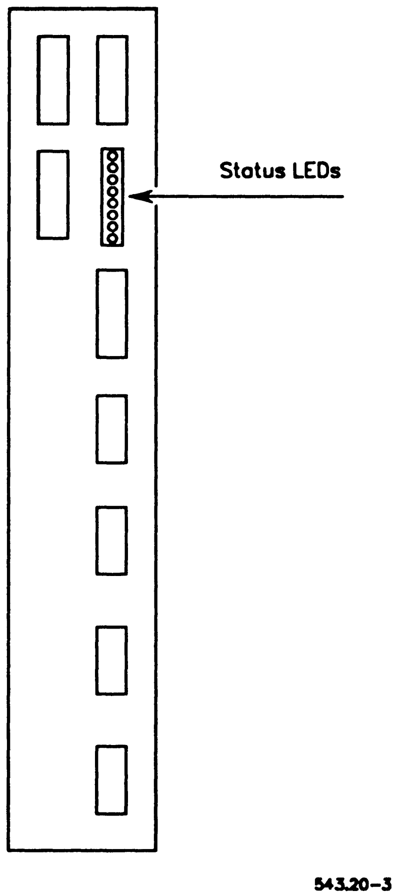
Crash codes are displayed on a processor after it has crashed but before the system is reset. It is important to read and interpret the crash code *before* you reboot the system. Therefore, if you are troubleshooting a recurring processor crash, bootstrap the system from the *M* (manual) keyswitch position so that the processor will not be rebooted automatically after it crashes.

The different status displays for real-mode and protected-mode processors are described below.

Real-Mode Processors

The status display on a real-mode processor consists of eight LEDs, as pictured in Figure 20-3. When the processor is functioning normally, the bottom light flashes on and off in a pulsating "heartbeat-like" pattern.

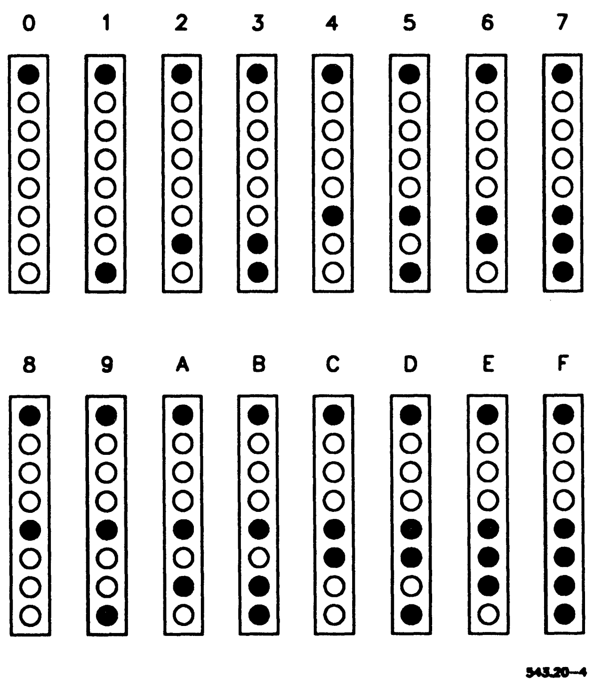
Figure 20-3. SRP Real-Mode Processor Status LEDs



When a processor crashes, the status display enters a loop, during which the error code is displayed by a sequence of LED patterns. The beginning of each loop is heralded by a *walking* pattern. When the LEDs are walking, one light at a time flashes briefly, in sequence from top to bottom (this creates a recycling type of pattern). The next four patterns display the error code. Each pattern is preceded by the top LED blinking off, and then on again. Each pattern corresponds to a hexadecimal digit, which is formed by the LEDs.

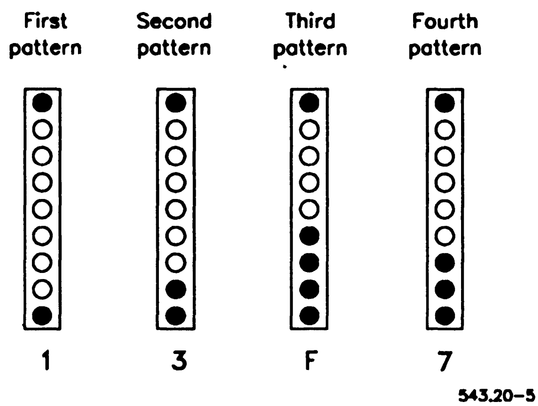
Figure 20-4 illustrates the LED pattern for each hexadecimal digit. It is not necessary to understand the hexadecimal (base 16) number system to read and interpret error codes; simply refer to Figure 20-4 to match the pattern to a digit. For a technical explanation of how these digits are derived, see the *CTOS Status Codes Reference Manual*.

Figure 20-4. LED Patterns for Hexadecimal Digits



A multidigit hexadecimal number is then created from the patterns displayed by the LEDs, with the digit represented by the first pattern occupying the highest place. For example, the LED patterns pictured in Figure 20-5 are displayed sequentially to form the hexadecimal number 13F7. Later you will convert the hexadecimal number to a decimal error code number.

Figure 20-5. LED Sequence Pattern for a Hexadecimal Error Code



To interpret a crash error code, follow these steps:

1. Wait for the walking pattern, which signals the beginning of the display loop.

2. Record the hexadecimal number as displayed by the four subsequent LED patterns. If you need to (which is not unusual), copy the LED patterns themselves and refer to Figure 20-4 later. Observe the loop as many times as is necessary to record accurate information.

3. Convert the hexadecimal number to a decimal number (see "Converting Hexadecimal Error Codes," later in this section). The decimal number is the actual error code.

4. Look up the error code (the decimal number) in the *CTOS Status Codes Reference Manual*. (The *CTOS Status Codes Reference Manual* also contains information about analyzing a system crash.)

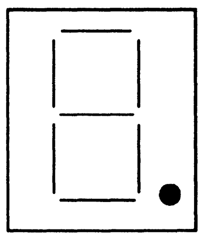
Protected-Mode Processors

The status display on a protected-mode processor consists of a one-digit LED, as shown in Figure 20-6. A pulsating decimal point indicates that the processor is functioning normally.

When a processor crashes, the status display enters a loop, during which the error code is displayed by a sequence of decimal digits. The beginning of each loop is heralded by a circular pattern around the perimeter of the LED display. The next five digits display the error code.

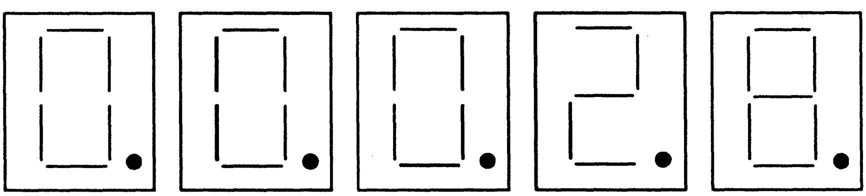
The first digit corresponds to the highest place of the number, for example, error code 28 is displayed as five sequential digits, as shown in Figure 20-7.

Figure 20-6. SRP Protected-Mode Processor Status LED



543.20-6

Figure 20-7. Sequence Pattern for a Decimal Error Code



543.20-7

To interpret a crash error code, follow these steps:

1. Wait for the circular pattern around the perimeter of the LED display.
2. Record each digit that is displayed after the looping pattern. The first digit displayed corresponds to the highest place of the error code number, as described above.
3. Look up the error code in the *CTOS Status Codes Reference Manual*. On protected-mode processors, error codes are already displayed as decimal numbers, so no conversion is necessary.

Note: *If a General Processor crashes while it is booting (before the front panel reaches 20), its crash status is displayed in hexadecimal. See Figure 20-8 for a hexadecimal-to-decimal conversion chart.*

Basic Procedures for SRP Processor Troubleshooting

When a processor board crashes, it usually provides information that can help you determine the cause of the crash. There are several approaches you can take in obtaining and interpreting crash analysis information; the approach you choose in a particular situation will depend on several things, including the kind of work you were doing when the crash occurred, your past experience with this sort of crash, and so forth.

The procedures described below represent the basic approaches to troubleshooting processor crashes and are listed in the recommended sequence. Several procedures are described in more detail later in this section.

1. Read the front panel status display.

   The front panel status display provides only the most rudimentary and general description of system problems and is therefore of limited value in troubleshooting. If, for example, a board crash has set the front panel display, this alerts you to the crash if you do not already know about it.

2. Read the processor LED displays.

   The crash code on the rear processor display provides more specific information about the crash. It should *always* be checked when a processor has crashed. Identify the code *before* you reboot the system.

3. Attempt to reboot the system.

4. After the system reboots, use **PLog** to look for crash information in the system error log (see "Diagnosing Problems," earlier in this section).

   As described earlier, the system attempts to log information relating to the error condition. If the error code is successfully written to the log, you can retrieve this information, along with a brief interpretive message, using **PLog**. Note that the error is not always successfully logged, and you should therefore identify the crash code, as described in step 2, before you reset the system and run **PLog**.

5. Call Technical Support if you are unable to isolate and correct the problem.

6. If required, collect a crash dump and copy it to a QIC tape for analysis by a Technical Support engineer (see "Collecting a Crash Dump," later in this section).

SRP Processor Crashes During Bootstrapping

Some possible causes of a processor crash during the bootstrap sequence are listed below:

- Corrupted system software
- Corrupted system disk
- Malfunctioning processor board

Try booting the system from a QIC tape. If you are successful, copy the operating system for the processor from the tape to the system volume. This would also be a good time to back up the system volume. If you continue to experience problems, reinitialize the system volume, and then restore the backup.

If the SRP will not boot from tape, follow these steps:

1. Turn off power to the expansion cabinets.
2. Remove all processors from the primary cabinet except the master processor and the processor that keeps crashing (move it to a slot adjacent to the master processor).
3. Disconnect all disk drives from the primary cabinet (see the installation guide for your SRP).
4. Attempt to reboot from the QIC tape.

If the processor still crashes while it is booting, you are most likely experiencing a hardware problem with that processor.

Isolating Hardware Problems

To isolate a hardware problem, you reduce the system to the minimum of hardware required to allow a successful bootstrap. Such a technique is recommended when the system is configured in such a way that the symptoms point to a part that may not be removed without causing additional problems.

If the master processor is a real-mode board, a minimum configuration consists of the master processor and one Cluster Processor. If the master processor is a protected-mode board, a minimum configuration consists of the master processor only. The system is then rebuilt using the original boards and drives until the problem is recreated. This suggests that the part most recently introduced to the system is causing the failure.

An alternate technique requires you to remove hardware, one piece at a time, until the problem cannot be reproduced. It is recommended when the system is configured in such a way that the symptoms point to a part that may be removed without causing additional problems. This technique should be used whenever possible to keep the handling of hardware to a minimum and to avoid software reconfiguration.

The system is reduced board by board or drive by drive until the problem no longer exists. The system is then rebuilt using the original boards and drives until the problem is recreated only by the introduction of one remaining part (board or drive), indicating that the part reintroduced to the system is the cause of the original failure.

Isolating Software Problems

This technique requires you to create the following keyswitch files which bootstrap the system with the minimum of required software:

*[Sys]<Sys>SrpConfig.*k*.sys*

*[Sys]<Sys>SysInit.*k*.jcl*

where *k* is the keyswitch position (*M* for manual, *N* for normal, *R* for remote).

Many system administrators use the *R* (remote) keyswitch position for a minimum software configuration. *SrpConfig.R.sys* usually contains only the configuration parameters required to bootstrap a minimum hardware configuration, and *SysInit.R.jcl* is usually empty. In some situations, however, you may want to install QIC tape system services.

Errors During System Service Installation

After all processors are booted, the front panel displays 20. At this point, system services are installed from the system initialization file.

Error 50 or 51 on the front panel indicates an error during system initialization. To troubleshoot these errors, bootstrap the system from your troubleshooting keyswitch position or a QIC tape (the point being to install a minmum of system services). Then, review and correct the system initialization JCL file for the problematic keyswitch position.

In some cases, the front panel continues to display 20, but a processor "hangs" or appears frozen. This indicates that a system service could not be installed and no further processing can take place because the JCL file is, in effect, still running on the processor.

A processor also hangs if the Remote User Manager (RUM) is not installed with the RunNoWait JCL statement, or if RUM is not the last system service installed on the master processor. See "Cluster View" in Section 4, "Using Administrative Tools" and "Installing the Remote User Manager," in Section 9, "Installing System Services."

If you suspect such a condition, use **Partition Status** to determine what is running on the processor. If a system service run file is active, the processor is locked.

If a system hangs during system initialization, follow these steps:

1. Reboot a limited software configuration on the SRP.
2. Attempt to install system services manually one at a time, via Cluster View (see "Installing System Services on an SRP," in Section 9, "Installing System Services").

This allows you to check parameter values and to determine the point at which the installation hangs.

As an alternative to installing system services manually, you can insert FrontPanel statements in the system initialization JCL file. With these statements, you can display a front panel status code at chosen intervals during system initialization. See Table 9-3.

You might need to allocate additional W-, X-, Y-, or Z-blocks if you are installing many system services. See "Adjusting Memory Blocks" in Section 15, "Optimizing System Performance," and the parameters for W-, X-, Y-, and Z-blocks in Section 17, "Configuring Shared Resource Processor Operating Systems."

Intermittent System Crashes

Intermittent system crashes can be caused by anything from inadequate power to faulty hardware to mixed revision levels of hardware and software. Use the troubleshooting techniques described earlier in this section to isolate the source of the problem.

If the system stops functioning and no errors are logged, you might be using a power supply that is not adequate for your hardware. Check its revision level and consider upgrading power supplies to the latest models.

Collecting a Crash Dump

When a processor crashes, a *crash dump* of memory is initiated. During a crash dump, the contents of the processor's memory at the time of the crash are written to a *crash dump file*. That file can then be analyzed by a system engineer to identify and correct the source of the crash.

As a system administrator, you may be responsible for collecting crash dumps. This section describes the procedures for collecting crash dumps on workstation and SRPs. Information about analyzing crash dump files is covered in the *CTOS Debugger User's Guide*.

A file named *<Sys>CrashDump.sys* is created when a disk is initialized, and its size is taken from the specified format template (see Section 11, "Adding Hard Disks"). Crash dumps are written to the first recognized device containing a crash dump file that is larger than 10 sectors.

Note: *Access to a file named* <Sys>CrashDump.sys *that is larger than 10 sectors in size is required for a workstation to be bootable.*

Performing Crash Dumps on a Workstation

On workstations, a crash dump takes places automatically when a processor crashes. While the dump is executing, a *D* followed by a series of dots (as when the workstation bootstraps) appear on the screen.

The following subsections describe the crash dump procedures and requirements on the various models of workstations.

X-Bus Workstations

On real-mode X-Bus workstations, the crash dump is completed in a single stage and is written to the first recognized device containing a crash dump file that is larger than 0 sectors (in most cases, that is *[Sys]<Sys>CrashDump.sys*).

On protected-mode X-Bus workstations, an extended crash dump is required to collect the contents of extended memory. After the initial crash dump takes place (as described above for real-mode workstations) the processor reboots using only the first megabyte of memory. Then, an extended crash dump takes place.

If the crash dump file is large enough, the extended crash dump is written to it. If *CrashDump.sys* is not large enough, the extended crash dump is written to an extended crash dump file. The name of the extended crash dump file (usually *[Sys]<Sys>ExtCrashDump.sys*) is specified in the operating system configuration file. See "Creating an Extended Crash Dump File," later in this section, and the *:ExtCrashDumpFile:* parameter in Section 16, "Configuring Workstation Operating Systems."

By default, video is deactivated during extended crash dumping. On some workstations, however, that option is configurable; see the *:ExtCrashVMDFile*: parameter in Section 16, "Configuring Workstation Operating Systems."

X-Bus+ Workstations

On X-Bus+ workstations, the crash dump is written to the crash dump file, and if that file is large enough, is completed in a single stage. If the crash dump file is not large enough to collect a complete memory dump, an extended crash dump takes place to dump the remaining memory, as described for protected-mode processors, earlier in this section.

EISA/ISA Workstations

On EISA/ISA-bus workstations, the entire memory dump is written to *[Sys]<Sys>CrashDump.sys*. To collect a crash dump on an EISA/ISA-bus workstation, *[Sys]<Sys>CrashDump.sys* must be large enough to contain the entire contents of memory. To determine the correct size for that file, in sectors, multiply the size of system memory (in K bytes) by 2. For example, if the system contains 8192K bytes of memory, the correct size for the crash dump file is 16384 sectors.

A crash dump file is created when a disk is initialized, and its size is specified in the format template that is used to initialize the disk. See the *:CrashFileSize:* parameter in Section 11, "Adding Hard Disks," for more detailed information.

If memory is added to an EISA/ISA-bus workstation, the crash dump file will no longer be large enough to collect a crash dump. In such a case, the disk must be backed up and then reinitialized with a larger crash dump file.

Diskless Workstations

Diskless workstations dump to *[Sys]<Sys>WS>CrashDump.sys* on the server. To perform extended crash dumping on a diskless workstation, that file specification must be specified as the parameter value for *:CrashDumpFile:* in the operating system configuration file for the diskless workstation; see Section 16, "Configuring Workstation Operating Systems."

Creating an Extended Crash Dump File

The extended crash dump file is identified by the *:ExtCrashDumpFile:* parameter in *[Sys]<Sys>Config.sys*, the operating system configuration file. It must be large enough, in sectors, to contain the entire contents of memory. To determine the size, in sectors, for an extended crash dump file, multiply the total amount of processor memory by 2. For example, if a processor has 4096K bytes of memory, the crash dump file must be 8192 sectors long. The extended crash dump file can be accessed from any volume.

To create an extended crash dump file, use the **Create File** command, as described in the following procedure.

1. On the Executive command line, type **Create File**; then press **RETURN**.
2. Fill in the command form, as shown in the following example:

```
Create File
 File name                        [Sys]<Sys>ExtCrashDump.sys
 [Vol or dir password]            ____________________
 [File password]                  ____________________
 [File protection level]          15
 [Size in sectors (0)]            8192
 [Overwrite ok?]                  ____________________
```

3. Press **GO**.

Suppressing Crash Dumping

Automatic extended crash dumping is controlled by the following parameter in the operating system configuration file:

:SuppressAutoDump:

Set this value to **No** (the default) to implement automatic extended crash dumping.

If a system is not set up for automatic extended crash dumping, you can perform it manually; see the **Extended Crash Dump** command in the *CTOS Executive Reference Manual*.

Performing Crash Dumps on an SRP

Crash dumping on SRPs is controlled per processor by the following configuration file entry in *[Sys]<Sys>SrpConfig.sys*:

```
:Boot:        (Processor = CP00,
              OS = [Sys]<Sys>rSrpCp.sys,
              Dump = Yes)
```

After the *Dump* subparameter, specify **Yes** to implement crash dump collection on that processor.

Master Processor

On an SRP, the master processor dumps in the same manner described for X-Bus processors. If it is a protected mode processor, up to 4M bytes of memory are dumped to the crash dump file. The processor then reboots in 4M bytes of memory and an extended crash dump takes place. If *CrashDump.sys* is large enough to accept the entire memory dump (for example, 32768 sectors for a 16M byte processor), the dump of extended memory is appended to that file. If *CrashDump.sys* is not large enough, *[Sys]<Sys>ExtCrashDump.sys* is created and expanded as needed.

After the extended crash dump takes place, the master processor initiates dumping of the other processors and, when that is complete, automatically reboots the system.

Other SRP Processors

Each of the other processors dumps to a unique file specification. In the default or individual processor section of the operating system configuration file, a volume and directory for the crash dump file is designated, as shown in the following example:

```
:Processor: GP00
   :CrashDumpPath: [Sys]<Dump>
```

The specified volume must be controlled by the master processor, and the directory must already exist. Crash dumps files are named Xpnn.*crash*, where *xPnn* is the four-character processor ID, for example, *GP01.crash*.

See the *:CrashDumpPath:* parameter in Section 17, "Configuring Shared Resource Processor Operating Systems," for more information.

What If a System Will Not Bootstrap?

Bootstrapping problems can be caused by the following:

- The system volume is corrupted.
- The operating system in *[Sys]<Sys>SysImage.sys* is not compatible with the workstation or SRP hardware.
- A customized operating system or configuration file contains invalid parameters.

Other causes can be the result of a variety of software and hardware problems. For example, on EISA/ISA workstations, hard disk geometry entries may become corrupted due to loss of battery power. Always check the appropriate hardware manual for additional troubleshooting possiblities.

To troubleshoot a bootstrapping problem, you can attempt to bootstrap from the server or from removable media. If the system still does not boot, you are most likely dealing with a hardware problem.

If the system bootstraps successfully from the server or from removable media, most likely the system volume has been corrupted or an incompatible operating system has been installed.

Bootstrapping From the Server

If you are troubleshooting a cluster workstation connected to a server, you can bootstrap it from the server. This is a convenient technique for troubleshooting, because most of the software you need is already installed on the server.

For example, if you need to restore the system volume on a cluster workstation, simply bootstrap from the server and restore the diskettes or tape to the cluster workstation's disk. When you do this, however, remember to use the volume or device name of the disk to which you are restoring (for example, *[d0]* or *[d1]*); do not use *[Sys]*. See "Bootstrapping a Workstation From a Server," in Section 5, "Bootstrapping," for information about bootstrapping from the server.

Caution

Remember that when a workstation is booted from the server, *[Sys]* is the server's own system volume. Therefore, when you restore to a disk on the cluster workstation, designate the volume or device name (such as *[d0]*), rather than *[Sys]*. Restoring to *[Sys]*, while booted from the server, overwrites the server's system volume.

Bootstrapping From a Floppy Diskette

Standard Software includes a utility that allows you to create a set of diskettes from which you can bootstrap a workstation. You would most likely use this method on a workstation that is not connected to a server or on a workstation that is itself a server. In addition to booting, the diskette set allows you to perform the following tasks:

- Sign on to the workstation
- Back up disks to tape or floppy diskettes
- Restore a backup
- Initialize a disk
- Change the name of a volume

This section contains information and procedures for creating and using a bootable floppy diskette set.

Regular and High-Capacity Drives

You can create a bootable diskette set on either a 5.25-inch or a 3.5-inch floppy drive. 3.5-inch drives are high-capacity drives. In addition, 5.25-inch floppy drives in B39 workstations and in MF1, MS5, and MS6 disk modules are high-capacity drives. 5.25-inch floppy drives in other workstations are regular capacity drives.

If you are not sure whether your drive is high capacity, a 5.25-inch drive will default to regular capacity. Similarly, if you want to create a set of regular capacity bootable diskettes (so that you can use it on any workstation), you can do so on a high capacity drive by using default parameters.

On 3.5-inch drives, however, you must specify high capacity. If you use default parameters, the diskettes will not be formatted and the procedure will fail.

Both default and high capacity parameters are described in the procedure for creating the bootable diskette set, later in this section.

Creating a Message File

Before you create the bootable diskette set, you will need to create a small Executive message file, as described in the following procedure:

1. With the Editor application, open the following file:

   [Sys]<Sys>ExecMsg.txt

2. Select (highlight) messages 32907 through 32957.
3. While highlighted text is still displayed, open a new file named *[Sys]<Sys>CBF>ExecMsg.txt.*
4. Press COPY to copy the highlighted text into the new file.
5. Press FINISH, then GO, to exit the Editor and save the file.
6. On the Executive command line, type **Create Message File**; then press RETURN.

7. Fill in the command form as shown below:

```
Create Message File
  Text file          [Sys]<Sys>CBF>ExecMsg.txt
  [Message file]     ______________________
  [Print file]       ______________________
```

8. Press **GO** to create *[Sys]<Sys>CBF>ExecMsg.bin.*

Creating a Bootable Diskette Set

Before you begin, you will need the following number of unformatted blank diskettes:

High capacity	3 diskettes
Regular capacity	5 diskettes

To create the bootable diskette set, follow these steps:

1. On the Executive command line, type **Create Backup Floppy**; then press **RETURN**.
2. If required, fill in the parameter fields, as described below:

```
Create Backup Floppy
  [Include tape utilities?]    ______________
  [High capacity floppies?]    ______________
```

Note: *Because* **Create Backup Floppy** *is executed from a submit file, any response in a field activates the parameter. Therefore, to respond* **No** *to either field, leave it blank.*

[Include tape utilities?]. Specify **Yes** if you want the Sequential Access Service utilities to be included on the diskette set. If you do not want to include the tape utilities, leave this field blank.

[High capacity floppies?]. The following possibilities exist for this field:

- If the workstation is equipped with a 3.5-inch drive, specify **Yes**. (If you leave this field blank, the command will fail.)
- If the workstation is equipped with a regular-capacity 5.25-inch drive, leave this field blank. (If you specify anything in this field, the command will fail.)

- If the workstation is equipped with a high-capacity 5.25-inch drive, and you want to create a high-capacity diskette set, specify **Yes**.
- If the workstation is equipped with a high-capacity 5.25-inch drive and you do not want a high-capacity diskette set, leave this field blank.

See also "Regular and High-Capacity Drives," earlier in this section.

3. Press GO.

After some introductory messages are displayed, you are prompted to insert the first diskette. Instructions will appear on the screen; continue responding to messages and changing diskettes as prompted.

Labeling the Bootable Diskette Set

As the diskettes are created, label them sequentially, as follows:

Regular Capacity Set

Boot Diskette
Continuation Diskette 1: Optional Tape Services (You will need to use this diskette even if you did not include tape services.)
Continuation Diskette 2: Volume Archive, Change Volume Name
Continuation Diskette 3: Format Disk
Continuation Diskette 4: Restore Archive

High Capacity Set

Character or Bitmap Boot Diskette
VGA Boot Diskette
Continuation Diskette

Using the Bootable Diskette Set

To bootstrap a workstation from the diskette set, follow these steps:

1. Turn on the workstation.
2. Insert the Boot Diskette into the floppy disk drive.

3. Press the Reset button.

   When the workstation has successfully booted, the following message appears:

   ```
   Please insert Continuation 1 Diskette
         Press any key to continue
   ```

4. Remove the Boot Diskette.
5. Insert the Continuation 1 Diskette.
6. Press GO.
7. Insert the appropriate continuation diskette, as described on the screen.
8. Press GO.

With the high capacity diskette set, you do not need to change diskettes between backup, format, and restore procedures. With the regular capacity diskette set, however, you will need to change diskettes. Be sure to wait for commands to finish executing before changing to a different diskette.

In addition to commands listed on the continuation diskettes, the following commands are available:

Batch Foreground

Copy

Create Directory

Create File

Delete

List

Logout

Path

Rename

Run

Type

Bootstrapping From QIC Tape

To troubleshoot an SRP, you can bootstrap from the CTOS/XE Boot Tape, which is supplied with the distribution media. The boot tape creates a limited, but valid, system volume in memory on the SRP.

Using the Bootable Tape

To boot an SRP from tape, the SRP must contain one of the following minimum hardware configurations:

- One protected-mode processor
- Three real-mode processors

(This is the same configuration that is required for you to install the CTOS/XE system software.)

In addition to the SRP hardware requirement, one cluster workstation that boots locally must be connected to the SRP as follows:

- If the master processor is a General Processor with SCSI Interface, connect the cluster workstation to a cluster port on the master processor.
- If the master processor is a File Processor or Data Processor, connect the cluster workstation to channel A on the first Cluster Processor.

You will use this workstation to communicate with the SRP after it boots from tape; therefore, make sure that the Cluster View commands are installed on the workstation.

Because only a minimum number of processors boot from the tape, it is essential to connect the cluster workstation to the correct cluster channel. To simplify troubleshooting, many system administrators connect a workstation, in close proximity to the SRP, to use specifically as a troubleshooting station.

To bootstrap from the CTOS/XE Boot Tape, follow these steps:

1. Insert the tape into the QIC tape drive on the SRP.
2. Turn off the keyswitch, and then turn it back on to its usual position.

When the front panel reads 20, the SRP has bootstrapped a minimum hardware and software configuration. You have access, via Cluster View, to internal Executive commands, as well as the following commands:

Format Disk

Volume Archive

Restore Archive

Install Sequential Access Service

Submit

You can execute other commands, such as **Volume Status, Cluster Status, Files**, and the Editor, on the cluster workstation.

Creating the Bootable Tape

If you use a customized operating system on your SRP, you may want to create your own bootable tape for troubleshooting. In addition, if your SRP exceeds the minimum hardware requirements, you can add more files to those contained in the system volume created in memory. This expands the troubleshooting tools available to you when you boot from tape.

See the **Create Boot Tape** command in the *CTOS Executive Reference Manual* for information about creating a bootable tape.

As a safeguard for your master copy of the CTOS/XE Boot Tape, you can make a copy of it to use for troubleshooting. To do so, use the **Tape Copy** command to copy tape files 0 through 7. See the *CTOS Executive Reference Manual* and the release documentation for the CTOS/XE operating systems.

Error 21 on the Front Panel

Error 21 on the front panel means that the bootstrap ROM cannot locate a bootable file name *<Sys>SysImage.sys*. If the front panel displays 21 when you attempt to boot from QIC tape, suspect a faulty disk-drive cable or a hardware problem on the master processor.

Converting Hexadecimal Error Codes

In most cases, you will need to convert hexadecimal error codes into decimal numbers before you can look them up in the *CTOS Status Codes Reference Manual*. The chart shown in Figure 20-8 will help you perform that task.

Each hexadecimal error code consists of four digits, and as with decimal numbers, the leftmost digit holds the highest place value. For example, in the hexadecimal number 13F7, the digit 1 holds the highest place, 7, the lowest.

Figure 20-8 is arranged in four columns with each column corresponding to one digit of the hexadecimal error code. It is arranged from highest to lowest digit, starting from the left, in the same order as the hexadecimal number. Along the left of each column is the hexadecimal digit itself; along the right is a corresponding decimal value.

To use Figure 20-8 to obtain the decimal error code number, follow these steps:

1. Locate the hexadecimal digit for each place value and note the corresponding decimal equivalent for each.
2. Add all the decimal equivalents together to obtain the decimal error code number.

For example, the decimal equivalents for the hexadecimal number 13F7 are as follows:

Hexadecimal Digit	Decimal Equivalent
1	4096
3	768
F	240
7	7

The total, 5111, is the decimal error code number.

Figure 20-8. Hexadecimal-to-Decimal Conversion Chart

Highest Digit →							→ Lowest Digit
HEX	**DEC**	**HEX**	**DEC**	**HEX**	**DEC**	**HEX**	**DEC**
0	0	0	0	0	0	0	0
1	4,096	1	256	1	16	1	1
2	8,192	2	512	2	32	2	2
3	12,288	3	768	3	48	3	3
4	16,384	4	1,024	4	64	4	4
5	20,480	5	1,280	5	80	5	5
6	24,576	6	1,536	6	96	6	6
7	28,672	7	1,792	7	112	7	7
8	32,768	8	2,048	8	128	8	8
9	36,864	9	2,304	9	144	9	9
A	40,960	A	2,560	A	160	A	10
B	45,056	B	2,816	B	176	B	11
C	49,152	C	3,072	C	192	C	12
D	53,248	D	3,328	D	208	D	13
E	57,344	E	3,584	E	224	E	14
F	61,440	F	3,840	F	240	F	15

543.20–8

Glossary

^

See circumflex.

$ directory

See dollar-sign directory.

<Sys>

A directory on every disk that contains unique information about the disk. This information is used by the operating system when you issue commands and use applications.

[Sys]

A system-assigned pseudonym for the volume from which the workstation or shared resource processor bootstraps.

[Sys]<Sys>

The volume and directory containing the bootable operating system.

A

agent cache

Enables caching of files opened on the server to the cluster workstation's local cache.

application

A program or software package that accomplishes a specific task or set of tasks, such as office publishing, spreadsheet preparation, or database retrieval.

archive dataset

A file or set of files that contains the contents of other files in a compressed form. An archive dataset is created by the **Volume Archive** or **Selective Archive** command.

archive media

Floppy diskettes, hard disks, or tapes that are used to store backups.

active command file

The command file specified in the user configuration file with which the user signed on. If none is specified, *[Sys]<Sys>Sys.cmds* is the default.

active password

The password that is in effect when a command is issued.

ASCII text file

A file containing the alphanumeric characters comprising the American Standard Code for Information Interchange. This term is commonly used to denote a file containing unformatted text, as opposed to a file containing imbedded formatting characters, which may not be visible to the user.

asynchronous terminal emulator (ATE)

A workstation functioning as an asynchronous terminal. Also used to describe a software package that provides this capability. *See also* Basic ATE.

B

bad block table

An area on a disk where the addresses of bad spots are stored. Data is not written to the bad spots identified in the bad block table.

Basic ATE

A software product that allows a workstation to function as an asynchronous terminal. *See also* asynchronous terminal emulator.

baud rate

The speed at which data is transmitted.

binary file

A file containing a set of computer instructions reduced to a choice of two alternative conditions.

bit

The smallest unit of electronic data.

BNet

A communications application used to connect servers together. *See also* network node.

bootstrap (*also* boot)

The hardware-initiated process of loading an operating system into memory.

buffer

An area of memory used as a temporary holding place for data.

byte

A unit of data containing a specific number of bits.

C

cache

A high-speed buffer that improves data access speed.

cache memory disk

An area of memory that is used as a disk, usually as the scratch volume. *See also* scratch volume.

case sensitive

Upper case and lower case letters are interpreted differently. The Executive is not case sensitive.

channel

A connector to which a device can be connected.

circumflex (^)

Precedes a password in a file specification.

click

To press and then immediately release a mouse button.

cluster

A group of workstations connected to a common server. *See also* server.

Cluster File Access (CFA)

A method by which workstations can gain access to disks on other workstations in the cluster.

Cluster View

A software product that connects the keyboard and monitor on a cluster workstation to a processor on the server.

cluster workstation

A workstation connected to a server. *See also* server.

command case

An arbitrary value assigned within a run file that invokes a particular function of the program.

command file

A file the Executive reads to display command names, command forms, and help descriptions, and to associate a run file and command case with each command.

command form

Displays parameter fields for the specified command.

command line

The highlighted line on the Executive screen, where you enter the name of the command you want to issue.

commented text

Text that is ignored when a program is compiled or executed.

configuration

An arrangement of parts, such as computer hardware, or of elements, such as software programs.

configuration file

Contains parameter values for a software product or hardware device.

Context Manager

A program that divides memory into multiple partitions so that more than one program can be started, and in some cases simultaneously executed, on a workstation. *See also* memory partition, static partition, and variable partition.

corrupted volume

An initialized disk that contains damaged or unreconciled data.

CP

Acronym for an SRP Cluster Processor.

CPU

See processor.

crash

See system crash.

crash dump

The process of writing data from memory into a file, so that it can be analyzed and debugged by an operating system engineer.

CRC

See cyclic redundancy check.

CTOS

A comprehensive term encompassing all varieties of the CTOS I, CTOS II, CTOS III, and CTOS/XE operating systems.

CTOS I

A set of real-mode operating systems for workstations.

CTOS II

A set of protected-mode operating systems for workstations.

CTOS III

A set of virtual-memory operating systems for workstations.

CTOS/XE

A set of real-mode and protected-mode operating systems for shared resource processors.

cursor

A movable marker that indicates where on the screen the next typed character will appear.

customized operating system

A an operating system that has been customized with the System Build Utilities. *See also* prebuilt operating system.

cyclic redundancy check (CRC)

An error-checking procedure that takes place during cluster communications.

D

Debugger

A software debugging product used by programmers.

default directory

The directory name that appears in angle brackets (< >) in the path setting on the screen.

default path

The volume and directory that appear in the path setting on the screen. *See also* path.

default value

A predetermined value with which a command is executed when an optional field is left blank.

default volume

The volume or device name that appears in square brackets ([ ]) in the path setting on the screen.

demand paging

The ability of an operating system to swap portions of a program (pages) in and out of memory (frames) as they are needed. *See also* virtual memory.

Development Utilities

A software package containing commands and applications that are used primarily by programmers and software development engineers.

device

A disk drive, printer, tape drive, modem, or other physical device that receives or transmits data.

device name

See device specification.

device password

A password assigned to a piece of hardware. Device passwords are assigned by the operating system or in the operating system configuration file.

device specification

The identifier for a piece of hardware. Device specifications are assigned by the operating system or in the operating system configuration file.

device template

A set of configuration file entries that define the physical characteristics of a disk device. Such parameters are required to format a disk.

digital data storage (DDS) drive

A tape drive on a workstation that uses small, cassette-like tape cartridges.

direct memory access (DMA)

A direct high-speed data transfer between an input/output device and memory.

directory

A subdivision of disk storage space.

disk

A mass-storage device for data.

disk drive

The mechanism that holds the disk.

disk drive heads

The mechanisms that read data from and write data to the disk.

diskless workstation

Consists of a base unit containing the CPU and video controller. It uses disks on the server for data storage; disks are not connected directly to it.

distribution media

The diskettes or tapes on which software is supplied.

DMA

See direct memory access.

dollar-sign directory

A directory that stores temporary files.

DP

Acronym for an SRP Data Processor.

E

ECC

See error checking and correction.

Editor

An ASCII text editing application.

EISA/ISA-bus workstation

Consists of a base unit containing a CPU and other hardware components such as a hard disk, a floppy disk drive, and a video controller. It can be upgraded with "industry standard" hardware components. The SuperGen Series 3000 is an EISA/ISA-bus workstation.

error checking and correction (ECC)

Detection and correction of single-bit errors in the processing unit.

error code

A decimal or hexadecimal number denoting an error condition on a workstation or shared resource processor.

Executive

The CTOS command interpreter.

F

field

The highlighted line in a command form where a parameter value is entered.

file

A set of data that is stored and retrieved as a unit.

file specification

A unique identifier that contains the name of a file, as well as its volume and directory location.

file system

The data and control structures stored on accessible disks.

file system cache

An area of memory where disk sectors are stored dynamically, as they are used.

floppy disk drive

A slot-like opening on a workstation that holds a floppy diskette.

floppy diskette

A small, removable data storage disk.

format

A particular arrangement of data.

format template

A set of configuration file entries that define the characteristics of a volume. Such parameters are required to initialize a disk.

FP

Acronym for an SRP File Processor.

fragmentation

Noncontiguous data storage. A fragmented file is stored in multiple file extents. A fragmented disk contains many small, noncontiguous areas of storage space.

frame

A 4K-byte region of physical memory into which the paging service loads a page of program code or data.

function keys

The keys labeled F1 though F10. Their functions change from program to program.

G

G byte (*also* gigabyte)

1,073,741,824 bytes.

Generic Print System (GPS)

A set of software programs that provide printing services for CTOS applications.

global thrashing

A condition that occurs when the number of frames required by concurrently running programs exceeds the amount of physical memory.

GP

Acronym for an SRP General Processor.

H

half-inch tape drive

A reel-to-reel tape drive for use on shared resource processors only.

hexadecimal number

A number in the base sixteen number system, which is primarily used by programmers. Hexadecimal digits are represented by numerals 0 to 9 and letters A, B, C, D, E, and F.

I

IDE drive

The type of disk drive contained within a Series 3000 EISA/ISA workstation.

input/output (I/O)

Data transfers between subsystem boundaries, such as from disk to memory, then back to disk.

integrated X-Bus workstation

Contains the processor, a hard disk, and a floppy disk drive in the base unit, and can be expanded with X-Bus modules. The B39 style is an integrated X-Bus workstation. *See also* X-Bus workstation.

J

Job Control Language (JCL)

A programming language processed by the Batch facility.

K

K byte (*also* kilobyte)

1,024 bytes.

keyword

A predefined word or string of characters that identifies a parameter. Keywords are used in many configuration files and are written in the form of *:Keyword:* colon. The parameter value follows the closing colon.

L

LFS

See local file system.

loadable request file

A binary file containing request definitions for a system service.

local file system (LFS)

A workstation with its own disks, as opposed to a diskless workstation, which always uses disks on the server.

logging out

The opposite of signing on. Logging out exits the Executive.

M

M byte (*also* megabyte)

1,048,576 bytes.

master processor

The first processor in the primary cabinet of an SRP. It bootstraps itself first and then controls booting of the other processors.

memory

High-speed volatile data storage, the contents of which can be altered at any time. *See also* random access memory.

memory block

Small portions of memory that are allocated by the operating system for use during interprocessor data transfers.

memory disk

A portion of memory on a shared resource processor that functions as a system volume.

message file

A binary file containing the screen prompts and messages displayed by an application.

message text file

The text source file for a binary message file.

memory partition

A discrete area of memory. *See also* Context Manager, static partition, and variable partition.

modify access

An access level that allows changes to be made to a file.

module

A workstation component, such as a disk drive, housed in its own casing and connected as an individual unit.

mouse

An electronic pointing device, used for drawing or selecting items on the screen.

N

network node

A server connected via BNet to other servers in a network. Cluster workstations connected to a node can communicate with other network nodes.

node

See network node.

null device

A valid, but nonexistent, device specification *[Nul]*. It is used to test command execution without generating output.

O

operating system

A program that controls execution of other programs on the computer.

operating system configuration file

A file containing configurable operating system parameters.

output

Data delivered from a program to a file or device.

overwrite

To replace the contents of an existing file with the contents of another file. Overwriting destroys the original file. This is an option with many Executive commands.

oversubscribed memory

When the total number of pages allocated to applications and system services is greater than the number of frames available. *See also* virtual memory.

P

page

A 4K byte section of a program in the linear address space.

page fault

The process of bringing pages (of data) into physical memory.

paging service

The virtual memory operating system service that controls demand paging.

parameter

A definable element of information affecting the way a program is executed.

parameter field

See field.

parameter template

A format or device template read by the **Format Disk** command. *See also* device template and format template.

parameter value

An element of information supplied in a command form or a configuration file.

partition

See memory partition.

partitioned memory management

The memory management system whereby the operating system loads programs into individual memory partitions. *See also* demand paging and virtual memory.

password

An access code that restricts the use of a system. Workstations or servers can have several passwords that allow varying levels of access to different users. *See also* active password.

path

The default volume and directory. This volume and directory are used automatically when you execute a command unless you override the path with a file specification. The path setting appears in the status area of the screen.

physical address space

The actual amount of memory that can be addressed by a program. In real mode, the physical address space is 1M byte. In protected mode, the physical address space is determined by the system processor and its hardware limitations.

prebuilt operating system

An unmodified version of a CTOS operating system distributed by Unisys Corporation.

prefix files

Operating system source code files that are modified, assembled, and linked to build a customized operating system.

primary file headers

The file headers used by the operating system to perform disk read and write operations. *See also* secondary file headers.

primary partition

The memory partition containing the program that is currently active on the workstation or shared resource processor.

private installation

Software that is installed for use on the local workstation only. *Compare to* public installation.

processor (*also* processing unit or CPU)

The unit that interprets and executes instructions.

protected mode

A program or operating system that can use memory above the first megabyte.

protection level

A number assigned to a file that designates read or modify access and whether a volume, directory, or file level password is required to open it.

public installation

Software that is installed on the server and can be accessed by other workstations in the cluster. *Compare to* private installation.

Q

queue

A portion of memory used for storing a list of files or jobs awaiting processing.

Queue Manager

A system service that controls spooled printing and other queue-oriented jobs, such as batch processing.

quarter-inch cartridge (QIC) tape drive

A tape drive on a workstation or shared resource processor that uses cassette-like cartridge tapes.

R

random access memory (RAM)

A high-speed storage area where data is loaded prior to processing. The contents of memory are volatile and can be altered at any time. After processing, data is written back to disk for permanent storage.

RDAT

See digital data storage drive.

real mode

A program or operating system that runs in the first megabyte of memory.

read access

An access level that allows a file to be opened or processed only, but not to be changed.

record

To store a group of commands that can be reexecuted later.

remote caching

The caching of disk sectors to a file system cache located on a processor different from the one to which the disk is physically connected.

Remote Keyboard Video Service (RKVS)

The Cluster View system service.

Remote User Manager (RUM)

A memory manager that allows workstations to execute applications via Cluster View in discrete partitions on a shared resource processor.

request file

See loadable request file.

run file

An executable program.

S

scratch volume

A disk used for storage of the temporary files that some applications create.

SCSI

The acronym for Small Computer Standard Interface. It provides a design standard for hardware device interfaces.

secondary file headers

Duplicate copies of primary file headers. They are used to retrieve data when primary file headers are damaged. *See also* primary file headers.

sector

512 bytes of data.

server

A workstation or shared resource processor (SRP) to which cluster workstations are connected. The server controls many system resources, such as printing and communications. Co-workers can share the files and applications that are stored on disks located on the server.

shared resource processor (SRP)

A multiprocessor computer that is always used as a server. Also called an XE system.

sign on

The procedure that starts a user session. The user signs on with a predefined name, which determines the applications and commands that are available.

SMD disk drive

Acronym for Storage Module Device. It refers to an eight-inch hard disk drive for use on a shared resource processor.

Software Release Announcement (SRA)

A version-specific document containing information about a new release of a software product. Formerly called Release Notes, Release Notice, or Release Information File.

source code

The text of a programming language, before it is compiled and linked to form an executable program.

SP

Acronym for an SRP Storage Processor.

squash

To reduce file and disk fragmentation.

SRP

See shared resource processor.

ST-506 disk

A non-SCSI SRP disk.

Standard Software

A set of programs, configuration files, and commands packaged with the operating system, which are needed to configure the system and perform basic operations.

static partition

An area of memory that is allocated by Context Manager to be a specific predetermined size. *See also* Context Manager, memory partition, and variable partition.

status area

The top two lines of the Executive screen where the default path, user name, and date/time information are displayed.

status code

A number designating a certain condition on the system. In many cases, status codes represent errors. In other cases, they represent a normal operating condition.

SuperGen Series 2000 workstation

Consists of a base unit containing the CPU and video controller. It can be upgraded with expansion cards for graphics or extra memory. The SuperGen Series 2000 is a diskless workstation.

SuperGen Series 3000 workstation

Consists of a base unit containing a CPU and other hardware components such as a hard disk, a floppy disk drive, and a video controller. It can be upgraded with "industry standard" hardware components. The SuperGen Series 3000 is also called an EISA/ISA-bus workstation.

SuperGen Series 5000 workstation

Consists of a base unit containing the CPU and removable cartridges, which house disk drives, graphics controllers, tape drives, and other optional equipment. The SuperGen Series 5000 is also called an X-Bus+ workstation.

Sys

An abbreviation for "system." It is used in file names to denote files that are necessary for the workstation or SRP to boot and function correctly. It is also used as the name for the volume and directory that contain the operating system. *See also <Sys>, [Sys],* and *[Sys]<Sys>.*

System Build Utilities

The set of commands and prefix files used to build a customized operating system.

system crash

An abnormal condition from which the system cannot recover. After a crash, the system usually freezes or reboots automatically.

system error log

A file containing information about many types of hardware and software errors. It can be displayed or printed with the **PLog** command.

System Image

The operating system.

system service

A program than expands the capabilities of the operating system.

system software

The operating system, system services, utilities, and configurations files that are required for a workstation or shared resource processor to function as a computer.

T

tape file mark

A software mark that separates sequential tape files.

TeleCluster

The method of connecting workstations to a server via telephone lines.

template file

A text file containing menu displays and function key definitions for a particular command.

U

user configuration file (*also* user file)

A configuration file that defines one or more software environments for the user.

user name

The name with which a user signs on to the system.

utility

A program that carries out a specific task, such as copying or deleting files.

V

valid volume

A disk that has been formatted and initialized for use on a workstation or shared resource processor.

value

An element of information supplied in a command form or a configuration file.

variable

A predefined character or group of characters that is replaced with an actual value during program execution.

variable partition

An area of memory that is allocated by Context Manager of a size equal to or smaller than a specified value. *See also* Context Manager, memory partition, and static partition.

version number

A number designating the revision level of a software product.

virtual memory

The apparent size of memory (to an application), which is greater than actual physical memory size, that makes it possible to address more memory than physically exists on the processor. *See also* demand paging.

volume

An initialized disk. *See also* valid volume.

volume control structures

The framework within which the file system allocates disk space.

volume name

The name assigned to a disk when it is initialized.

volume password

Allows unrestricted access to a volume.

W

W-block

See memory block.

workstation

A desktop computer that can function as a standalone system or be connected into a workgroup called a cluster. *See also* cluster.

workstation number (Ws*NNN*)

A three-digit number assigned to the operating system for each type of processor.

write access

The capability of opening a file and making changes to it.

X

X-block

Memory that is allocated by the operating system for use during communications between the server and cluster workstations.

X-Bus workstation

A collection of separately housed modules, each containing an individual hardware component, such as a processor, a disk, or a graphics controller. X-Bus workstations include the B26, B28, B38, and B39 styles. *See also* integrated X-Bus workstation.

X-Bus+ workstation

Consists of a base unit containing the CPU and removable cartridges, which house disk drives, graphics controllers, tape drives, and other optional equipment. The SuperGen Series 5000 is an X-Bus+ workstation.

XE

See shared resource processor.

Y

Y-block

See memory block.

Z

Z-block

See memory block.

Index

A

B

C

D

E

F

G

H

I

R

:RqTracker:, 16-25
RUM, 4-10, 9-3, 9-26
run files for system services, 9-15

S

T

W

X

USER'S COMMENT SHEET

We welcome your comments and suggestions. They help us improve our manuals. Please give specific page and paragraph references whenever possible.

Does this manual provide the information you need? Is it at the right level? What other types of manuals are needed?

Is this manual written clearly? What is unclear?

Is the format of this manual convenient in arrangement, in size?

Is this manual accurate? What is inaccurate?

Name ______________________________ Date ____________

Title _________________________ Phone ________________

Company Name/Department ________________________________

Address ____

City ____________________ State _____ Zip Code _________

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 1807 SAN JOSE, CA

POSTAGE WILL BE PAID BY ADDRESSEE

UNISYS
Multimedia Product Information 9–007
2700 N 1st St
San Jose CA 95134–2028

Fold Here