TOPS–20
System Manager's Guide

AA–4169F–TM, AD–4169F–T1, and AD–4169F–T2

April 1982

This document is intended for the person who is responsible for making final decisions for setting up and maintaining the efficient operation of a TOPS–20 installation.

This version of the guide updates the document having these order numbers, AA–4169F–TM and AD–4169F–T1.

OPERATING SYSTEM: TOPS–20 (KS/KL Model A), V4
TOPS–20 (KL Model B), V5

Software and manuals should be ordered by title and order number. In the United States, send orders to the nearest distribution center. Outside the United States, orders should be directed to the nearest DIGITAL Field Sales Office or representative.

Northeast/Mid–Atlantic Region

Digital Equipment Corporation
PO Box CS2008
Nashua, New Hampshire 03061
Telephone:(603)884–6660

Central Region

Digital Equipment Corporation
Accessories and Supplies Center
1050 East Remington Road
Schaumburg, Illinois 60195
Telephone:(312)640–5612

Western Region

Digital Equipment Corporation
Accessories and Supplies Center
632 Caribbean Drive
Sunnyvale, California 94086
Telephone:(408)734–4915

digital equipment corporation • marlboro, massachusetts

First Printing, October 1976
Revised, May 1977
Revised, January 1978
Revised, October 1978
Revised, January 1980
Updated, December 1980
Updated, April 1982

The postage-prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

Contents

Chapter 3 After Software Installation

Chapter 4 Creating Structures

Chapter 8 Tape Storage

Chapter 9 System Problems/Crashes

Chapter 10 System Performance

Chapter 11 Controlling Access To System Resources

Index

FIGURES

TABLES

Preface

The *TOPS–20 System Manager's Guide* is written for the person who is responsible for establishing policies and procedures for a timesharing and/or batch processing installation, using the TOPS–20 operating system. Usually, this person is responsible for setting up and maintaining both the system hardware and software. The *Site Management Guide* and the *TOPS–20 Operator's Guide* provide you and your operations people with the necessary information to maintain your system hardware. These two manuals are referenced throughout this guide.

This guide deals primarily with your system software. It contains general suggestions for planning the installation of your software and for setting up your computer room to begin operations. The guide contains hints and suggestions for your system's operation, including when, and many times why, particular functions or procedures should be considered. It assumes that your system operator is responsible for implementing many of the decisions you make. In most cases, where lengthy implementation procedures are required, the appropriate reference is noted.

Chapters 1 and 2	describe the documentation, system logs, and special forms that you should have available to you, and in some cases, to system users. Chapter 2 also includes preliminary planning functions that you can do before the software is installed.
Chapter 3	describes the system directories and files that your system contains immediately after you install the software. It also describes the mechanisms you can use to change the installed TOPS–20 batch system and to test the integrity of your newly installed or updated system.

Chapter 4	describes using your disk-pack and -drive resources to set up disk structures in a way that best suits your installation's needs. It also includes guidelines for determining the available disk space that you have to create user directories.
Chapter 5	describes creating and maintaining directories. It includes a detailed description of the three methods of administration you can choose from to control the creation and maintenance of directories. It describes how to use directory and file protection codes to expand or limit the type of access users can have to directories and files, and how to place users and directories in groups so that users can share files.
Chapter 6	describes the TOPS–20 accounting facility. This description includes how to choose an accounting scheme, how to create accounting files, and how to set the system to begin validating accounts.
Chapter 7	describes backing up your disk structures onto magnetic tape soon after software installation. It recommends the supplies needed and procedures that you should follow to save all your directories and files on a daily basis, and how to create a system crash tape in the event of a major problem with the file system.
Chapter 8	describes how you can use magnetic tapes to store important files (file archiving) and to save valuable disk space by copying infrequently accessed files to tape (file migration). It also describes how to give control of tape drive usage to the system and the operator (tape drive allocation), and how to set up your system to use labeled tapes (tape labeling).
Chapter 9	describes the procedures you must follow in the event you have a problem with the file system, or a user has lost the files in a directory. It describes using your system crash tape and your daily backup tapes to resolve these problems.
Chapter 10	describes the tuning mechanisms that allow you to change the behavior of your system. Each description includes why you may want to use a particular mechanism, how to use it, and the affects it may have on your system.
Chapter 11	describes the access control mechanism that you can use to alter system policy decisions. It includes the type of policy changes you may want to make at your installation.

The software package that you purchased for your system determines which system model references you should follow when using this guide. If you purchased the QT010 package, follow the descriptions referring to the DECSYSTEM–2040, 2050. If you purchased the QT023 package, follow the descriptions referring to the DECSYSTEM–2060. Note that if you have a DECSYSTEM–2040S, you have purchased the QT023 software package and should also follow the descriptions for the 2060. If you have a 2020, follow the descriptions for the 2020.

Throughout this guide the following conventions and symbols are used:

Convention/Symbol	Description
n–CONFIG.CMD	n refers to the Release 5 version of this file, that is, 5–CONFIG.CMD.
UPPERCASE	In user input representations, indicates information that must be entered exactly as shown.
lowercase	In user input representations, indicates variable information that is determined by you.
red print	Indicates the information that you must type at your terminal.
( )	In user input representations, encloses guide word information. Pressing the ESCAPE or ALTMODE key on your terminal causes guidewords to be printed by the computer.
(RET)	Indicates you should press the RET or RETURN key on your terminal. Unless otherwise noted, pressing RETURN terminates all command or input strings.
CTRL/	Indicates you should press the CTRL key on your terminal. The CTRL key is always used in conjunction with another key, for example, CTRL/Z.

Chapter 1
Documentation

Section 1.1 describes the documentation provided by DIGITAL and recommends the manuals with which you should be familiar to manage your system. Section 1.2 describes adding your own documentation, for example, special forms, to the documentation you receive from DIGITAL. Be sure you have all available documentation convenient to your system users.

1.1 Documents Available From Digital

All documentation for the TOPS–20 Operating System is contained in the *TOPS–20 Software Notebook Set.* This notebook set contains information pertaining to the most recent version of TOPS–20. It is organized functionally to facilitate referencing manuals. Each manual contains cross references to other manuals within the set that further explain a subject.

This manual assumes you are familiar with some of the manuals in the notebook set. In particular, you should be familiar with the information in the *TOPS–20 Operator's Guide,* the *TOPS–20 User's Guide* and either the *TOPS–20 Software Installation Guide* or the *TOPS–20 KL10 Model B Installation Guide.* Any additional documents that you need depend on the configuration of your system. For example, if your system has IBM emulation and termination (DNxx), you should be familiar with the *IBM Emulation–Termination, DN64 DN65 Manual.* It includes installation procedures and descriptions of the operator and user interfaces. If your system is connected to the ARPA network, you should have access to the *TOPS–20AN Monitor Calls User's Guide* and the *TOPS–20AN User's Guide.* Finally, if your system has DECnet, you should be familiar with the various DECnet–20 manuals. The *TOPS–20 Software Notebook Set* includes all of these manuals.

In addition to the *TOPS–20 Software Notebook Set*, you receive the TOPS–20 Beware File Listing. It is distributed with the software installation and distribution magnetic tapes. Before installing a new version of the software on your system, read the Beware File. It contains last–minute changes to the software that have not been documented, and hints or suggestions for installing or using the new software.

With each new system, you should also receive two stand–alone documents, which are documents not included in the notebook set. These manuals assist you in 1) preparing your site for the hardware installation, the *Site Preparation Guide*, and 2) maintaining and reporting problems about your system's software and hardware, the *Site Management Guide*.

NOTE

Your Sales Representative delivers the *Site Preparation Guide*, and your Field Service Representative delivers the *Site Management Guide*.

This manual (the *TOPS–20 System Manager's Guide*) deals primarily with installing and maintaining the software on your system. Therefore, it is assumed that you have already used the *Site Preparation Guide* to install your system hardware.

The *Site Management Guide* is designed for use by both you (along with your operations people) and your Field Service Representative. You should begin using this manual immediately after you install your hardware. It contains schedules, procedures, and logs for recording and evaluating all information pertinent to the operation and care of the system. The manual belongs to DIGITAL, but it is kept and maintained at your computer site. For added convenience and organization, many system managers keep all their important system information in the same binder as the *Site Management Guide*. For example, they keep System Logs and Operator Shift Change information in the same binder, along with other special forms. Section 1.2 describes several forms that you may include in a system log book or, as suggested here, in the *Site Management Guide*.

DIGITAL places a major emphasis on the documentation provided to its customers. The Software Publications Department continues to solicit suggestions for improvement and corrections from the users of its documentation. Encourage users to comment on the manuals you receive with your system. For convenience, a Reader Comment Form is located at the back of each manual.

1.2 Documents Prepared At Your Installation

Sections 1.2.1 through 1.2.5 describe some forms that may be useful at your installation. A sample form is provided in each section.

1.2.1 System Log

Every system must have a system log for recording problems and procedures relating to both hardware and software. All operators and system programmers should record the following types of activities in the log, along with the date, time, and their names.

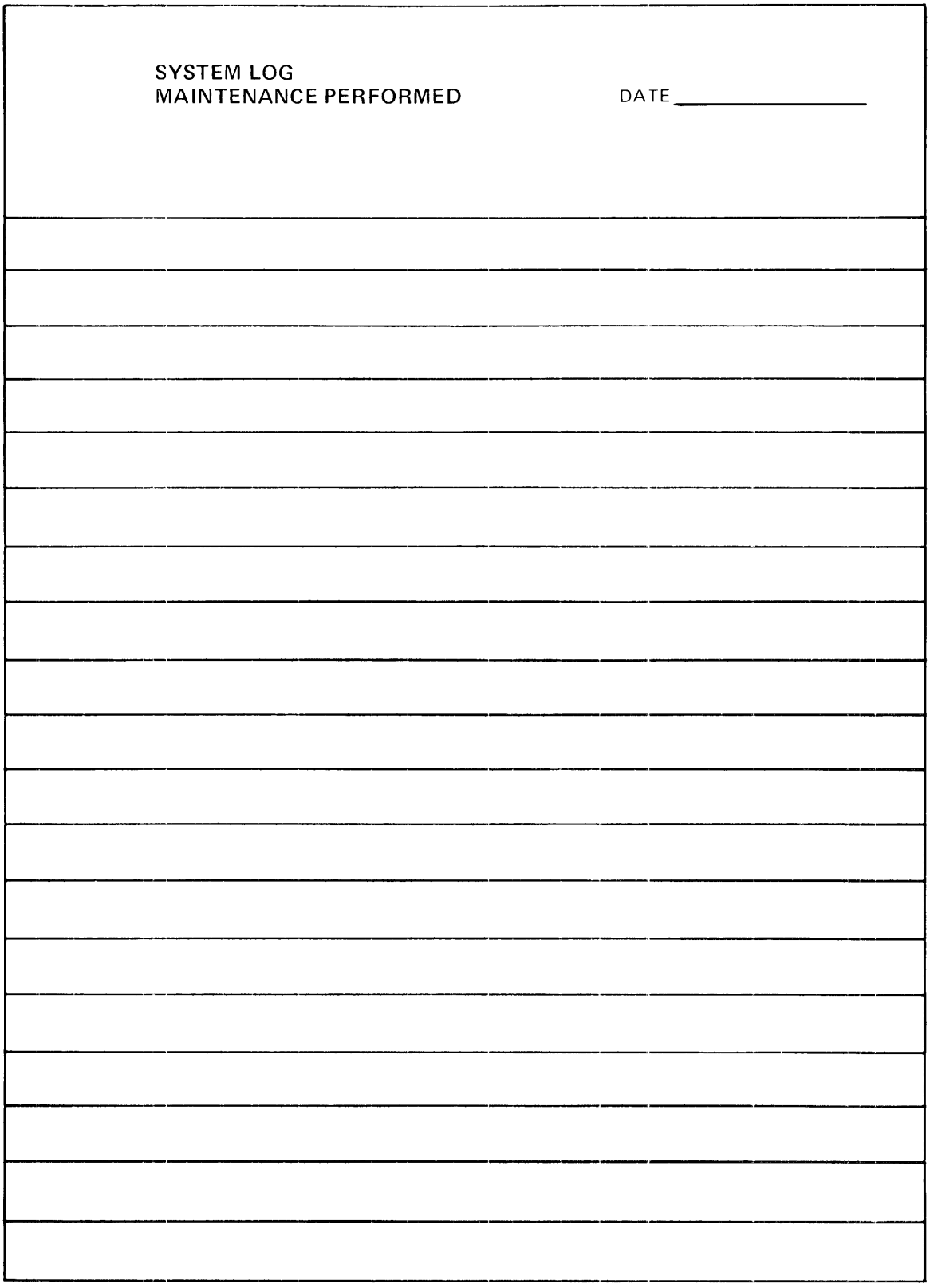
- System backup procedures
- Beginning and ending of timesharing (for example, the times the system was started and stopped for preventive maintenance or repair)
- Problems in hardware or software AND the actions taken to correct the problems (always save the CTY (operator terminal) output or copy of the typescript)
- New or revised software installed
- New users or changes to existing user data or directories
- New structures or changes to existing structures

Most system problems are easier to solve (and, hence, less costly) if you keep an accurate record of all activities. The *Site Management Guide* has a section set aside for system log information. This section contains preprinted forms that you can use to record system log information, or you can design your own forms. You can store these forms in the *Site Management Guide* or in a separate binder.

You should design your log so that it is easy to use and read. Remember, you are likely to have the most problems when the system is new, so NOW is the time to start using the log. The following two pages contain sample left and right-hand pages of a log book. The left-hand page (Figure 1-1) contains information concerning hardware maintenance; the right-hand page (Figure 1-2) is a problem report containing:

- The time of the entry
- A "Y" or "N" answer to whether the system had to be reloaded
- The name of the person making the entry
- A few words describing the nature of the activity
- A record of calls to Digital Field Service (F/S)
- A description of the device or program causing the problem
- Remarks about the entry

Figure 1–1: Sample System Log (Hardware Maintenance)



MR-S-526-80

Figure 1–2: Sample System Log (Problem Report)

PAGE_____

SYSTEM LOG

DATE_____

TIME	RELOAD	NAME	MONITOR OR HARDWARE MAINTENANCE ACTIVITY	F/S ATTN	DEVICE OR PROGRAM	ENTRY

MR-S-527-80

1.2.2 Mountable Structure Sign-Up Log

In addition to keeping the system log, you should also record requests from users to mount structures. (Chapter 4 describes how to set up and use structures.) Without a formal scheduling procedure, some users may monopolize the use of a structure and frustrate other users who do not have the opportunity to mount and use their structures, usually because there are no disk drives available. To avoid this situation, set up a procedure whereby users inform the operator when they need to use a structure. The operator can then schedule the length of time specified on the request log. On a busy day when many users are issuing MOUNT requests for structures, the operator checks the log before granting or denying the mount requests. This scheduling allows you to service many requests for mounting structures in a fair and orderly manner. The sample Mountable Structure Sign-Up Log shown in Figure 1–3 contains:

- The scheduled mounting time
- The scheduled time needed to use the structure
- The actual time the structure was mounted
- The actual time the structure was removed
- The name of the user who initiated the request
- The structure name (or pack ID)
- A column for any special instructions or notes

Remember that this log is only a sample; you should design a form that best suits your own requirements.

1.2.3 System Access Request Form

Some installations have many users requesting access to the system for the first time. You need standard information from these users before you can process their requests and create directories for them. For example, you must know which system they need to access (if you have more than one system), their names, selected passwords, departments, accounts, etc. You can organize these requests by providing a System Access Request Form that is kept in an easy-to-access area, perhaps outside the computer room. You can require signatures of department managers on the access form to ensure that prospective users have approval to charge computer usage to accounts. Figure 1–4 is a sample of a system access request form.

Figure 1–3: Sample Mountable Structure Sign-Up Log

MOUNTABLE STRUCTURE
SIGN - LOG

PAGE_____

DATE_____

SCHEDULED		ACTUAL				
MOUNTING TIME	TIME NEEDED	MOUNTING TIME	TIME REMOVED	USER NAME	PACK ID(s)	NOTES

MR-S-528-80

Figure 1–4: System Access Request

SYSTEM ACCESS REQUEST

SYSTEM NAME: _____ DEPARTMENT: _____
YOUR NAME: _____ ACCOUNT: _____
PROJECT: _____
PERMANENT ACCESS? ☐ YES ☐ NO (FROM: _____ TO: _____)

SUPERVISOR: _____ MANAGER: _____
Signature Signature

DIRECTORY NAME (1-39 CHARACTERS): _____
PASSWORD: _____ DIRECTORY PROTECTION (DEF.777700) ☐ OTHER: _____
*DO YOU REQUIRE PRIVILEGES ON THE SYSTEM ☐ N ☐ Y (TYPE: _____)
DO YOU WANT TO CREATE SUBDIRECTORIES? ☐ N ☐ Y (HOW MANY? _____ MAX.=8)
DO YOU WANT TO BE IN A GROUP WITH OTHER USERS OR DIRECTORIES? ☐ N
☐ Y (NAME OF USER(S) OR DIRECTORY(S): _____ _____ _____)
BRIEFLY DESCRIBE THE TYPE OF WORK YOU WILL PERFORM. FOR EXAMPLE, CREATING AND EDITING FILES, APPLICATIONS PROGRAMMING, COMPILER PROGRAMMING, ETC.

_____

_____

OPERATIONS USE ONLY

DIRECTORY	PASSWORD	STRUCTURE	WORKING QUOTA	PERMANENT QUOTA	USER GROUP	DIRECTORY GROUP	ACCOUNT

SUBDIRECTORIES	SCHED.CLASS	PRIVILEGES	DATA CREATED	COMMENTS:

*MUST BE APPROVED BY OPERATIONS MANAGEMENT

MR-S-529-80

1.2.4 Operator Work Request Form

You may want a form that allows users to request work from the operator. Examples of requests made to the operator are initializing tapes, transferring files between systems, and making changes to directories. You should set up a procedure for handling these requests. Figure 1–5 is a sample of an operator work request form.

1.2.5 Operator Shift Change Log

You may want to set up a binder to contain operator shift change information. Each operator records new procedures, or special instructions that the incoming operator needs to know. The incoming operator reads the operator shift log before starting the new shift. For example, the first shift operator changes the procedure for storing tapes, and records the new procedure in the shift change log. The information in the shift change log should not concern problems with the system, but should contain important information about the system or the computer room. The incoming operator still reads the system log book for the status of the system and any problems that have occurred during the previous shift. Figure 1–6 is a sample of an operator shift change log.

Figure 1–5: Operator Work Request

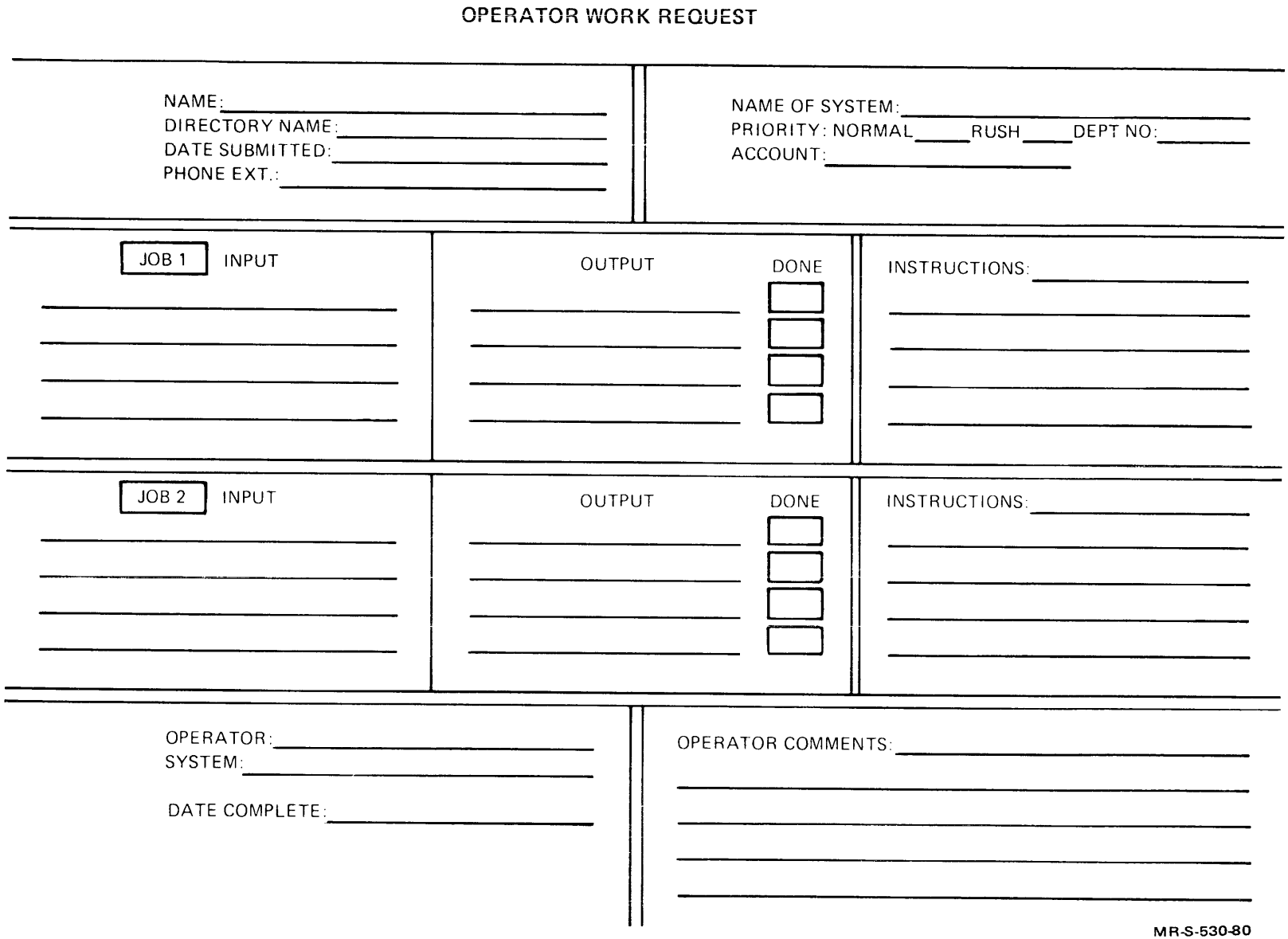


Figure 1–6: Operator Shift Change Log

OPERATOR SHIFT CHANGE LOG			
DATE	OPERATOR	SHIFT	COMMENTS

MR-S-531-80

Chapter 2
Preparing For Software Installation

You can establish many of the policies and procedures for your computer site before you install the software. It may help you later if some of the preliminary decisions and preparations are done before you begin setting up the system and handling requests from users. The following suggestions for preparing your installation are not all inclusive. Some TOPS–20 installations have specific requirements or restrictions that are not considered here. You can use this list as a guideline for the types of decisions you can make in the early stages of setting up your computer site.

2.1 Securing The Computer Room

Select the type of computer room security you need and a method for enforcing it. Many system managers do not allow non–operations people to enter the computer room. Establish an open– or closed–door policy and notify users of your policy. In the case of a closed–door policy, notify users of the procedure they should use to contact you (or the operator) and to submit their job requests.

2.2 Handling User Requests

Determine how user requests will be handled. You can handle jobs on a first–come basis, or on a priority basis. You can set up request boxes outside the computer room that the operator checks occasionally. You can also establish a location where users can leave disks and tapes for the operator to mount. Post a sign–up sheet so users can specify the time they need the tape or disk mounted. Chapter 1 describes sample forms that can be completed by users for requesting initial access to the system and requesting work to be done by the operator.

2.3 Ordering Supplies

Assign someone the responsibility for ordering paper supplies, ribbons, cards, and magnetic tapes. Chapter 7 provides an estimate of the number of tapes you should have to begin a backup procedure immediately after you install the software. Be sure you have enough CTY (operator terminal) and line printer paper to begin operations.

2.4 Scheduling Operator Tasks

The operator performs tasks either on a regular basis or on an as–needed basis. Decide which operator tasks will be performed on a regular schedule. Be sure to include hardware, software, and documentation related tasks. These regularly scheduled tasks can be performed daily, weekly, or monthly.

The following lists are samples of hardware and software related tasks that your operator may perform.

Hardware Related Tasks

Regular Schedule	As Needed Schedule
Clean tops of disk drives.	Replenish paper in the line printer.
Clean magnetic tape drives.	Remove reports from the line printer and distribute (perhaps to mail boxes).
Vacuum line printer to remove paper chad.	Replenish paper in operator's console.
Load mountable structures according to a schedule.	Physically load and unload magnetic tape and disk drives.

Software Related Tasks

Regular Schedule	As Needed Schedule
Bring up system after weekly maintenance.	Bring up system after a crash.
Run scheduled batch production jobs.	Maintain the batch system for users.
Save the contents of disk on magnetic tape.	Save special disk areas on magnetic tape.
Create a system "crash" tape for backup.	Restore selected user disk areas as needed.
Run the SPEAR program for daily error analysis.	Interact with users.
Submit a daily control file for accounting.	Create and update user directories.
Create the Message–of–the Day with the MAIL program.	Monitor disk space.

Establish a location for keeping the hard–copy output from the CTY. Your Field Service Representative needs this information if you have problems with your system. Have the operator tear off the copy and store it daily.

Documentation related tasks include:

- keeping a hand-written log of system activities (System Log)
- recording operator shift change information (Operator Shift Change Log)
- coordinating the mounting and dismounting of structures (Mountable Structure Sign-up Log)

Chapter 1 describes creating a system log, an operator shift change log, and a mountable structure sign-up log.

2.5 Selecting System Features

Determine the system features you want to enable during software installation. When you install the software, you create a file called n–CONFIG.CMD. This file is read by a start-up program (SETSPD) when the system is started for the first time and each subsequent time that you reload and start the system. The n–CONFIG.CMD file defines the line speeds for your terminals and many system parameters. Most of the decisions you must make concerning the parameters in this file are described throughout this manual. As you read each chapter, you can list the parameters that you want to place in the n–CONFIG.CMD file. Many system managers choose to slowly introduce new pieces of software. Therefore, you may want to disable some of the parameters until you have run the new software for awhile. You can edit the n–CONFIG.CMD file to add new software features to the system. You should edit the file at a convenient time before you reload the system. Then, when the system restarts, the new software features are enabled.

Chapters 3 through 11 describe setting up and maintaining your system. Read these chapters thoroughly. They contain important information to help you make decisions both before and after you install the software.

Chapter 3
After Software Installation

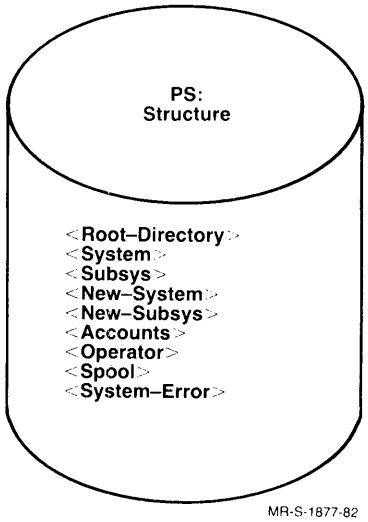
3.1 Overview

After you install the TOPS–20 software, your system contains all the directories and files necessary for you to start preparing for timesharing and batch processing. This chapter describes the directories, files, and system logical names created during software installation. Also included are suggestions for creating additional directories and logical names to assist you and system users.

3.2 Special System Directories

You initialize the file system during software installation. At this time, the system automatically creates nine directories on the disk that you defined as the system structure. These directories are shown in Figure 3–1:

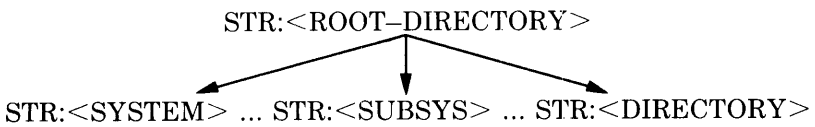
Figure 3–1: Special System Directories



Sections 3.2.1 through 3.2.7 describe these directories and their use. Section 3.2.8 describes additional directories you can create, and how they are useful.

3.2.1 <ROOT–DIRECTORY>

The <ROOT–DIRECTORY> contains a separate file for each first level directory on the system structure as follows



(where STR: is the name of the structure)

The <ROOT–DIRECTORY> is the most important directory created. Without it, directories and files cannot be accessed. You must *NEVER* modify this directory. The system maintains a backup copy of <ROOT–DIRECTORY> that can be accessed if the original copy is destroyed. (Refer to Section 9.3, Restoring <ROOT–DIRECTORY>.)

Chapter 5 describes creating directories and includes diagrams showing the structure of directories.

Each structure you create in addition to the system structure has a <ROOT–DIRECTORY>. The <ROOT–DIRECTORY> on any structure points to all the first level directories created under the <ROOT–DIRECTORY>.

After you install the software, give the DIRECTORY command for the directory <ROOT–DIRECTORY>. The output on your terminal appears similar to the example below. Note that each directory is a file in the <ROOT–DIRECTORY>. The differences between this list and the one on your terminal depend on the model system you have and the type of unbundled software you have purchased; for example, a 2020 may have different files than a 2050.

```
$DIRECTORY (OF FILES) STR:<ROOT-DIRECTORY>RET

   STR:<ROOT-DIRECTORY>
ACCOUNTS.DIRECTORY.1
BACKUP-COPY-OF-ROOT-DIRECTORY.IMAGE.1
BOOTSTRAP.BIN.1
DSKBTTBL..1
FRONT-END-FILE-SYSTEM.BIN.1
NEW-SUBSYS.DIRECTORY.1
NEW-SYSTEM.DIRECTORY.1
OPERATOR.DIRECTORY.1
ROOT-DIRECTORY.DIRECTORY.1
SPOOL.DIRECTORY.1
SUBSYS.DIRECTORY.1
SYSTEM.DIRECTORY.1
SYSTEM-ERROR.DIRECTORY.1
UETP.DIRECTORY.1

Total of 14 Files
```

3.2.2 <SYSTEM>

The directory <SYSTEM> contains data and program files that the system uses during normal operation. Table 3–1 lists many of the files that appear in this directory. The names and types of files differ occasionally, depending on the type of system you have installed. For example, if you are installing TOPS–20 on a DECSYSTEM–2020, <SYSTEM> contains all the microprocessor files for that system. However, the directory <SYSTEM> on a DECSYSTEM–2050 may not have these same files. Files that are specific to a particular system are noted accordingly.

Table 3–1: <SYSTEM> Files

File Name	Explanation
0DUMP11.BIN	Contains a dump of front–end memory after the front end crashes. (2040, 2050, 2060 only)
2020–MONMED.EXE	The largest runnable monitor on the 2020.
2020–MONSML.EXE	The smallest runnable monitor on the 2020.
2060–MONBIG.EXE	The smallest runnable monitor on the 2060.
2060–MONMAX.EXE	The largest runnable monitor on the 2060.
n–CONFIG.CMD	Contains definitions of line speeds, system logical names, printer VFU files, magnetic tape logical unit numbers, and additional system–dependent parameters. These system parameters are set every time the system starts. The value n equals the latest release of TOPS–20.
n–SETSPD.EXE	Program that reads the n–CONFIG.CMD file and sets up the parameters that it contains. The value n equals the latest release of TOPS–20.
ACCOUNTS–TABLE.BIN	Contains the information necessary to validate accounts.
AN–MONBIG.EXE	A large ARPANET timesharing monitor.
AN–MONLGE.EXE	The largest ARPANET timesharing monitor.
AN–MONMED.EXE	A medium ARPANET timesharing monitor.
AN–MONSML.EXE	A small ARPANET timehsaring monitor.
BUGS.MAC	Contains a complete description of all TOPS–20 BUGCHK, BUGHLT, and BUGINF messages.
BUGSTRINGS.TXT	Contains a list of all BUGHLT, BUGINF, and BUGCHK messages.
CHECKD.EXE	Program that creates structures and checks file system consistency.
DEVICE–STATUS.BIN	Contains status information for tape drives and disk drives. It is maintained by MOUNTR.

(continued on next page)

Table 3–1: <SYSTEM> Files (Cont.)

File Name	Explanation
DUMP.CPY	Contains a copy of main memory at the time of the last system crash. It is copied from DUMP.EXE to maintain a history of crashes. This file is written by the n–SETSPD program when the system is rebooted.
DUMP.EXE	Contains a copy of main memory at the time of the last system crash. You must have this file to get a system dump after a crash.
ERRMES.BIN	Contains binary system error messages.
EXEC.EXE	The TOPS–20 Command Processor.
FEDDT.EXE	A DDT program used for debugging the front–end.
KS10.RAM	Microcode. (2020 only)
KS10.ULD	Microcode used by the SMFILE program to build the microprocessor file system. (2020 only)
MONBCH.EXE	The runnable Batch monitor on the 2040 and 2050.
MONBIG.EXE	The largest runnable monitor on the 2040 and 2050.
MONITR.EXE	The current monitor.
MONMED.EXE	The medium–sized runnable monitor on the 2040 and 2050.
MONNAM.TXT	Contains the monitor name printed at the beginning of the system greeting line.
MONSML.EXE	The smallest runnable monitor on the 2040 and 2050.
MOUNTR.CMD	Contains a list of structure characteristics.
MTBOOT.RDI	Magnetic tape boot program. (2020 only)
PROGRAM–NAME–CACHE.TXT	Contains a list of the programs that should be loaded into the program–name cache. Read by the MAPPER program.
PTYCON.ATO	Contains the commands that are given automatically at the operator's console every time the system starts. You may modify this file to suit your own installation.
REAPER.CMD	Contains a list of default commands to REAPER. The REAPER program reads this file each time it is run.
RSX20F.MAP	Contains symbol locations for the front–end. It is used by the FEDDT program.
SMBOOT.EXE	Disk boot program. (2020 only)

(continued on next page)

Table 3–1: <SYSTEM> Files (Cont.)

File Name	Explanation
SMFILE.EXE	Program that is used to build the microprocessor file system. (2020 only)
SMFILE.HLP	Contains information about the SMFILE program. (2020 only)
SMMTBT.EXE	Input file to the SMFILE program to create MTBOOT.RDI. (2020 only)
SYSJOB.EXE	Program that runs in a process created by the monitor and takes commands from the file SYSJOB.RUN.
SYSJOB.HLP	Contains information about the SYSJOB program.
SYSJOB.RUN	Contains commands that SYSJOB processes.
SYSTEM.CMD	Contains OPR commands and is read by the OPR program at system startup.
TAPNAM.TXT	Text file that contains the installation identifier that is written in VOL1 labels on labeled tapes.
TGHA.EXE	Program that analyzes and corrects MOS memory problems. (2040, 2050, 2060)
TGHA.HLP	Contains information about the TGHA program.
UETP.EXE	Program that runs the User Environment Test package.

3.2.3 Restoring the Directory <SYSTEM>

If the contents of <SYSTEM> are accidentally lost or destroyed, you can restore the directory from the TOPS–20 Installation Tape or your latest system backup tape. (Refer to Chapter 7 for information about creating system backup tapes.) Use the procedure below to restore <SYSTEM> directory. If you have enabled tape drive allocation, use the MOUNT command instead of the ASSIGN command. (Refer to Section 8.3 for information about using tape drive allocation.)

1. Mount the appropriate tape (in this example, it is on drive 0).
2. Give the following commands at your terminal. If your system is a DECSYSTEM–2020, replace the command SKIP (DEVICE) MTA0: 4 FILES with SKIP (DEVICE) MTA0: 6 FILES.

```
@ENABLE (CAPABILITIES)(RET)
$ASSIGN (DEVICE) MTA0:(RET)
$SKIP (DEVICE) MTA0: 4 FILES(RET)
$RUN (PROGRAM) MTA0:(RET)

DUMPER>TAPE (DEVICE) MTA0:(RET)
DUMPER>RESTORE (TAPE FILES) DSK*:<*>*.*.* (TO) <SYSTEM>(RET)

DUMPER TAPE #1, , WEDNESDAY 1-NOV-78 330
LOADING FILES INTO <SYSTEM>

END OF SAVESET
DUMPER>EXIT(RET)
$
```

3.2.4 <SUBSYS>

The directory <SUBSYS> contains system programs (and their help files) that the user may want to run. The directory protection code set for <SUBSYS> prevents users from changing the files in this directory. Many of the file protections require users to enable WHEEL or OPERATOR capabilities to use the files. (Refer to Chapter 5 for information about directory and file protections and special capabilities.) Table 3–2 lists the programs and files commonly placed in <SUBSYS>. An asterisk precedes all unbundled software.

Table 3–2: STR:<SUBSYS> Files

Programs	Explanation
ACTGEN.EXE	Program that takes information from accounting files and creates the account validation data base.
ACTGEN.HLP	Contains information about the ACTGEN program.
*BASIC.EXE	The BASIC compiler.
BATCON.EXE	Program that controls batch jobs.
*BLIS10.EXE	BLISS compiler for building FORTRAN.
BLIS10.HLP	Contains information about the BLIS10 program.
CDRIVE.EXE	Program that controls card readers.
CHECKD.EXE	Program that creates structures and checks file system consistency (same as in <SYSTEM>).
CHECKD.HLP	Contains information about the CHECKD program.
CHKPNT.EXE	Program that makes accounting entries in the file <ACCOUNTS>CHECKPOINT.BIN.
CHKPNT.HLP	Contains information about the CHKPNT program.
CNVDSK.EXE	Program that extends FDBs to Release 4 format.
*COBDDT.HLP	Contains information about COBDDT.
*COBDDT.REL	The COBOL debugging program.
*COBOL.EXE	The COBOL compiler.
*COBOL.HLP	Contains information about the COBOL compiler.
CREF.EXE	Program that produces a cross–reference listing.
CREF.HLP	Contains information about the CREF program.
DLUSER.EXE	Program that saves and restores the directory parameters.
DLUSER.HLP	Contains information about the DLUSER program.
DUMPER.EXE	Program that saves and restores files to and from magnetic tape.
DUMPER.HLP	Contains information about the DUMPER program.

(continued on next page)

Table 3–2: STR:<SUBSYS> Files (Cont.)

Programs	Explanation
DX20LD.EXE	Loads (TU70 and TU72) DX20 microcode, if these devices are available.
DXMCA.ADX	Microcode for DX20. (Used only with TU70 tape drives.)
EDIT.EXE	A line–oriented text editor.
EDIT.HLP	Contains information about the EDIT program.
*FAL.EXE	Program that 'listens' for DECnet file transfers.
FE.EXE	Program that is used when copying files from the front–end file system to the TOPS–20 file system and vice versa. (2040, 2050, 2060 only)
FE.HLP	Contains information about the FE program. (2040, 2050, 2060 only)
FILCOM.EXE	Program that compares the contents of two files.
FILCOM.HLP	Contains information about the FILCOM program.
FILDDT.EXE	A DDT program used for examining the contents of system dumps (DUMP.CPY).
*FORDDT.HLP	Contains information about the FORDDT program.
*FORDDT.REL	The FORTRAN debugging program.
FORMAT.EXE	Program used to format RP04/RP06 disk packs while the system is in timesharing mode. (2040, 2050, 2060 only)
FORMAT.HLP	Contains information about the FORMAT program. (2040, 2050, 2060 only)
*FOROTS.EXE	The FORTRAN object time system (operating system interface).
*FORTRA.EXE	The FORTRAN compiler.
GALGEN.EXE	Program that creates the parameter file for building the batch system.
GLXLIB.EXE	Object time system used by the GALAXY programs.
HELP.HLP	Contains information about the HELP command.
*IBMSPL.EXE	Spooling program that sends IBM–batch–job files to remote IBM host and retrieves the output.
INFO.EXE	Program that gives information to programs using IPCF.
*ISAM.EXE	Program that maintains all COBOL indexed sequential files.
*ISAM.HLP	Contains information about the ISAM program.
*LIBARY.EXE	Program that creates, maintains, and lists the contents of COBOL library files.
*LIBARY.HLP	Contains information about the LIBARY program.
*LIBO12.EXE	The COBOL object time system (operating system interface).

(continued on next page)

Table 3–2: STR:<SUBSYS> Files (Cont.)

Programs	Explanation
*LIBOL.REL	Contains the COBOL library subroutines.
LINK.EXE	Program that loads relocatable binary programs.
LINK.HLP	Contains information about the LINK program.
LP64.RAM	Translation RAM file for a 64–character line printer. Read by SETSPD.
LP96.RAM	Translation RAM file for a 96–character line printer. Read by SETSPD.
LPTSPL.EXE	Program that controls output to the line printer.
MACREL.REL	Run–time file for macros in MACSYM.
MACRO.EXE	The MACRO assembler.
MACRO.HLP	Contains information about the MACRO assembler.
MACSYM.UNV	Contains system macros.
MAIL.EXE	Program that sends messages to users.
MAIL.HLP	Contains information about the MAIL program.
MAILER.EXE	Program that receives mail from the MAIL program and places it in the appropriate mailbox.
MAKDMP.EXE	Program that produces a standard DUMP.EXE file in <SYSTEM>.
MAKLIB.EXE	Program that creates relocatable subroutine libraries.
MAKLIB.HLP	Contains information about the MAKLIB program.
MAKRAM.EXE	Program that creates a translation RAM file for line printers.
MAKRAM.HLP	Contains information about the MAKRAM program.
MAKVFU.EXE	Program that creates a vertical formatting unit (VFU) file.
MAKVFU.HLP	Contains information about the MAKVFU program.
MAPPER.EXE	Program that loads the program–name cache. (Refer to Section 10.4, Improving Program Startup Time.)
MONSYM.REL	Object file that contains monitor call symbol definitions.
MONSYM.UNV	Contains symbol definitions for monitor calls.
MOUNTR.EXE	Program that mounts tapes and structures.
*NETCON.EXE	DECnet program that performs the network control program (NCP) and network control utility (NCU).
*NFT.EXE	DECnet file transfer program.
*NFT.HLP	Contains information about the NFT.EXE program.

(continued on next page)

Table 3–2: STR:<SUBSYS> Files (Cont.)

Programs	Explanation
NORMAL.VFU	Vertical formatting unit file for line printers.
OPR.EXE	Program that the operator uses to interface with all jobs and devices on the system.
OPR.HLP	Contains information about the OPR program.
ORION.EXE	Program that processes messages sent by the OPR, MOUNTR, LPTSPL, QUASAR, EXEC, etc. programs.
OVRLAY.REL	Overlay manager for the LINK program.
PA1050.EXE	The TOPS–10 Compatibility Package.
PAT.EXE	Program that creates the PA1050 program.
PLEASE.EXE	Program that establishes a dialog with the operator.
PLEASE.HLP	Contains information about the PLEASE program.
PTYCON.EXE	Program that controls many jobs from a single terminal.
PTYCON.HLP	Contains information about the PTYCON program.
QMANGR.EXE	Program that manages the batch and print queues.
QUASAR.EXE	Program that does the central queuing and scheduling for the batch system.
RDMAIL.EXE	Program that allows a user to read mail sent with the MAIL program.
RDMAIL.HLP	Contains information about the RDMAIL program.
REAPER.EXE	Program that marks files for migration to magnetic tape.
*RERUN.EXE	Restarts COBOL programs.
*RERUN.HLP	Contains information about the RERUN program.
RSXFMT.EXE	Utility program used for converting TOPS–20 files to a format used by the front end and vice versa. (2040, 2050, 2060 only)
RSXFMT.HLP	Contains information about the RSXFMT program.
RUNINP.HLP	Contains a list of RUNOFF text preparation commands.
RUNOFF.EXE	Program that helps with text preparation.
RUNOFF.HLP	Contains information about the RUNOFF program.
S20TAP.CTL	Control file that creates a system backup tape. (2020 only)
SDDT.EXE	DDT debugger for programs not containing a symbol table.
*SELOTS.EXE	Program that interfaces between the COBOL language and the COBOL object–time system (LIBOL). (Used with earlier versions of COBOL, up to and including version 11.)
*SORT.EXE	Program that sorts files record–by–record.
*SORT.HLP	Contains information about the SORT program.
SPRINT.EXE	Program that creates batch jobs from card input.

(continued on next page)

Table 3–2: STR:<SUBSYS> Files (Cont.)

Programs	Explanation
SPROUT.EXE	Output spooler for card punch, paper tape punch, and plotter.
SPEAR.EXE	Segment of SPEAR program.
SPRRET.EXE	Segment of SPEAR program.
SPRRET.TXT	Segment of SPEAR program.
SPRSUM.EXE	Segment of SPEAR program.
SPRSUM.TXT	Segment of SPEAR program.
SPEAR.HLP	Contains information about the SPEAR program.
SYSTAP.CTL	Control file that creates a system backup tape. (2040, 2050, 2060 only)
TV.EXE	A character–oriented text editor.
UDDT.EXE	DDT debugger for programs with a symbol table.
ULIST.EXE	Program for printing information about directories and users.
ULIST.HLP	Contains information about the ULIST program.
USAG20.EXE	Program (COBOL) that summarizes and reports data contained in the USAGE file.
USAG20.HLP	Contains information about the USAG20 program.
USAH20.EXE	Program (FORTRAN) that summarizes and reports data contained in the USAGE file.
USAH20.HLP	Contains information about the USAH20 program.
VERIFY.EXE	Program that is used during software installation to determine the integrity of files. It verifies checksums and version numbers of the .EXE files.
WATCH.EXE	Program for observing system performance.
WATCH.HLP	Contains information about the WATCH program.

NOTE

All the .HLP files can be printed using the HELP command; for example, the command @HELP WATCH prints the WATCH.HLP file.

3.2.5 Restoring the Directory <SUBSYS>

If the contents of <SUBSYS> are accidentally lost or destroyed, you can restore the directory from the TOPS–20 Installation Tape or your latest system backup tape. (Refer to Chapter 7 for information about creating system backup tapes.) Use the procedure below to restore the <SUBSYS> directory. If you have enabled tape drive allocation, use the MOUNT command instead of the ASSIGN command. (Refer to Section 8.3 for information about using tape drive allocation.)

1. Mount the appropriate tape (in this example, it is on drive 0)
2. Give the following commands at your terminal. If your system is a DECSYSTEM–2020, replace the command SKIP (DEVICE) MTA0: 4 FILES with SKIP (DEVICE) MTA0: 6 FILES.

```
@ENABLE (CAPABILITIES)(RET)
$ASSIGN (DEVICE) MTA0:(RET)
$SKIP (DEVICE) MTA0: 4 FILES(RET)
$RUN (PROGRAM) MTA0:(RET)

DUMPER>TAPE (DEVICE) MTA0:(RET)
DUMPER>SKIP (NUMBER OF SAVSETS) 1(RET)
DUMPER>RESTORE (TAPE FILES) DSK*:<*>*.*.* (TO)      <SUBSYS>(RET)

DUMPER TAPE # 1, , FRIDAY, 3-NOV-78 330
LOADING FILES INTO STR:<SUBSYS>

END OF SAVESET

DUMPER>EXIT(RET)
$
```

3.2.6 <NEW–SYSTEM> and <NEW–SUBSYS>

The first time you install the TOPS–20 software, the DLUSER program creates the directories <NEW–SYSTEM> and <NEW–SUBSYS>. They do not contain files. You use these directories when a new release becomes available and you are updating the existing system. When DIGITAL distributes an updated monitor on the TOPS–20 Installation Tape, you restore the first two savesets from this tape to the directories <NEW–SYSTEM> and <NEW–SUBSYS> respectively. You use these directories until you feel comfortable with the new software. Should you have any problems with the new software, you can easily revert to using the old software. Appendixes A and B of the *TOPS–20 Software Installation Guide* detail the procedures to update one software release to another.

If you have no problems and you are comfortable with the new monitor, copy all the files in the directory <NEW–SYSTEM> into the directory <SYSTEM> and all the files in the directory <NEW–SUBSYS> into the directory <SUBSYS>. You can now delete all the files in <NEW–SYSTEM> and <NEW–SUBSYS>. The directories <NEW– SYSTEM> and <NEW–SUBSYS> remain empty until a new version of the TOPS–20 software is distributed.

NOTE

After you copy the new files into the directories <SYSTEM> and <SUBSYS>, you cannot revert to the old system software unless you reinstall the system using the old monitor tapes.

3.2.7 <ACCOUNTS>, <OPERATOR>, <SPOOL>, and <SYSTEM–ERROR>

<ACCOUNTS> — After installation, the directory <ACCOUNTS> contains one file, SYSTEM–DATA.BIN. This file contains all the accounting system entries for each user. If the directory <ACCOUNTS> is destroyed, the accounting system creates a new SYSTEM–DATA.BIN file.

After the first LOGIN on the system, the system creates the <ACCOUNTS>CHECKPOINT.BIN file. This file stores accounting entries for each user during the time the user is logged in. After a user logs out, the accounting data stored in CHECKPOINT.BIN is copied to the SYSTEM–DATA.BIN file. When the system comes up after a crash, the monitor examines <ACCOUNTS>CHECKPOINT.BIN to determine which users were logged in at the time of the crash and stores the data in CHECKPOINT.BIN in SYSTEM–DATA.BIN. Therefore, users who did not log out as a result of a crash are still charged for their log–in time.

<OPERATOR> — The directory <OPERATOR> normally contains the file, PTYCON.LOG. This file contains a record of all the activities that occur under the operator jobs. The directory <OPERATOR> also contains any files the operator needs to run the system.

<SPOOL> — The directory <SPOOL> contains files that the spooling system needs before performing any input or output. The file PRIMARY–MASTER–QUEUE–FILE.QUASAR is created in this directory. It contains a copy of the input queues so that they are not destroyed if the system crashes. You must delete this file or process all entries in the queues before installing a new version of the batch system that has a different queue format. The GALAXY.DOC file describes the new software components and tells you if the queue format has changed.

<SYSTEM–ERROR> — The directory <SYSTEM–ERROR> contains the file ERROR.SYS. The ERROR.SYS file contains entries about system errors and is read by the system error recovery program, SPEAR.

3.2.8 Other Useful Directories

You may want to create additional directories for storing different versions of programs or text. Some useful directories are listed below. You should give these directories the proper protection number and make them files–only directories.

Directories and files that are executed or read by the entire user community should not be given the default protection 777700, which allows no access. They should be given the directory protection 777740 and the file protection 777752. (Section 5.7 describes directory and file protections.)

<NEW> The directory <NEW> can contain versions of your software that are not completely tested or that are drastically different from the current versions. If you create a directory <NEW>, users will find it more convenient if you also create the system logical name NEW: defined as <NEW>, SYS:. This logical name allows them to run all new software by merely typing NEW: and the program name. If there is no file with the given name in <NEW>, the system uses the version currently on <SUBSYS>. (Refer to Section 3.3 for a description of logical names.)

<OLD> The directory <OLD> can contain the old version of software as newer versions appear on <SUBSYS>. If programs or data do not work with new software, the user has a chance to correct the problems before the older software is no longer available. Users will find it convenient if you also define the system logical name OLD: as <OLD>,SYS:.

By creating the directories <NEW> and <OLD>, you gradually introduce new software to your users. When a new version becomes available, place it in the directory <NEW>. When the software appears to work correctly, move the version in <NEW> to <SUBSYS> and the version in <SUBSYS> to <OLD>. Store the version in <OLD> on a system backup tape. Each time you change a version of the software, you should send a systemwide message to all users.

<HELP> The directory <HELP> contains documents and help files that describe the system software. As different versions of software appear on <SYSTEM>, <NEW>, and <OLD>, you should make a list of changes incorporated in the new versions and place it in the directory <HELP>. You can move all files with the file type .HLP from <SUBSYS> to the directory <HELP>. The HELP command still works correctly if you define the system logical name HLP: to be <HELP>,SYS:.

<REMARKS>

The directory <REMARKS> contains messages from users to the operator. These messages are usually general system comments or complaints. When a user wants to send the operator a message that does not require an immediate response, he can send a message to the directory <REMARKS> using the MAIL program. (Refer to the description of the MAIL program in the *TOPS–20 User Utilities Guide.*) A typical message may be a request for supplies, for example, LA36 paper or ribbon. Creating the directory <REMARKS> avoids constant interruptions to the operator from users issuing PLEASE requests. The operator can read the messages in <REMARKS> at a specified time each day, or simply when he has time.

3.3 System Logical Names

A logical name is a descriptive word used to establish a search route to locate files in directories. It can be up to 39 alphanumeric characters; however, it is usually three to six alphanumeric characters. Because logical names are used in place of device names, they are always followed by a colon. Logical names tell the system where and in what order to search for files. When a user types a logical name, the system searches the directories in the order they were defined by the logical name. Although users can define logical names for their own use (refer to the *TOPS–20 User's Guide*), the logical names described here can be used by all users of the system. You can define system logical names in the n–CONFIG.CMD file.

After the system is installed, there is one systemwide logical name SYS: that is defined as <NEW–SUBSYS>, <SUBSYS> and another systemwide logical name SYSTEM: that is defined as <NEW–SYSTEM>, <SYSTEM>. Both are defined in the n–CONFIG.CMD file during installation. You may decide to add other logical names to aid your users in accessing files. If you want the logical names to be permanent, place the definitions (using an editor) in <SYSTEM>n–CONFIG.CMD file. SYSTEM:, SYS:, and some other frequently used logical names are explained below.

3.3.1 SYSTEM:

The logical name SYSTEM: defines a search list that contains all the system programs and files that the system needs to operate. SYSTEM: should always contain the directory <SYSTEM>. If you are updating the system with a new monitor, the definition of SYSTEM: in the n–CONFIG.CMD file also contains the directory <NEW–SYSTEM>, for example,

```
DEFINE SYSTEM: STR:<NEW-SYSTEM>,STR:<SYSTEM>
```

3.3.2 SYS:

The logical name SYS: defines a search list that contains all the system programs a user may want to run. SYS: should always contain the directory <SUBSYS> and any other library directories that contain commonly used programs. If you are updating the system with a new monitor, the definition of SYS: in the n–CONFIG.CMD file also contains the directory <NEW–SUBSYS>, for example,

```
DEFINE SYS: STR:<NEW-SUBSYS>,STR:<SUBSYS>
```

Be sure to set the protection on the library files in <SUBSYS> (or <NEW–SUBSYS>) to 777752. This protection allows access by all users.

3.3.3 NEW:

The logical name NEW: defines a search list containing a directory that has new software and then the system logical name SYS:. The definition you would put in n–CONFIG.CMD is:

```
DEFINE NEW: STR:<NEW>,SYS:
```

With this systemwide logical name, the user can give the command:

```
@DEFINE (LOGICAL NAME) SYS: (AS) NEW:(RET)
```

Now, when the user runs a program, the system looks first in the directory <NEW>, and then in the normal system search list SYS:. Therefore, the user always gets the most recent version of any program.

3.3.4 OLD:

If you have old versions of programs, defining the system logical name OLD: may be helpful to users. The usual definition of the logical name OLD: is:

```
DEFINE OLD: STR:<OLD>,SYS:
```

This definition has the same type of effect as defining NEW:. If the user gives the command:

```
@DEFINE (LOGICAL NAME) SYS: (AS) OLD:(RET)
```

whenever he runs a program, he gets the oldest available version.

3.3.5 HLP:

If you want to keep programs and documentation in separate directories, you should store the documentation in <HELP>. The HELP command looks in the directories identified by the logical name HLP:, so you must define the logical name HLP: to include the directory <HELP>. If the HELP command cannot locate a file using the logical name HLP:, it looks for a file using the logical name SYS:.

The definition of HLP: in n–CONFIG.CMD should be:

```
DEFINE HLP: STR:<HELP>,SYS:
```

3.3.6 SERR:

The logical name SERR: defines a search list that contains the system error file ERROR.SYS. The SERR: logical name tells the system to search the directory <SYSTEM–ERROR> for the ERROR.SYS file. This file is later used to produce reports.

The definition of SERR: in the n–CONFIG.CMD should be:

```
DEFINE SERR: STR:<SYSTEM-ERROR>
```

3.4 Console Front–End Files (2040, 2050, 2060 only)

The console front–end computer on a DECSYSTEM–2040, 2050, 2060 consists of a PDP–11 with 28K of memory. When the system is brought up for timesharing, the front–end monitor, RSX20F, is loaded in the PDP–11 memory and started. The TOPS–20 monitor is loaded in main memory and started. Thus, you have two computers working together. Both computers have their own monitor and related software.

The front–end file system consists of the RSX20F monitor and related programs (tasks) and files. During software installation, these front–end files are transferred from floppy disks to a special area on PS: unless an RP07 is being used as the system structure. If an RP07 is being used as the system structure, only the files on the TOPS–20 Installation Tape will be placed on the RP07. The front–end files, on the floppy disks, must be placed on either a dual–ported RP04 or RP06 disk drive. (Refer to the TOPS–20 KL10 Model B Installation Guide for the procedure for creating the front–end file system when using an RP07 disk drive as the system structure.)

The area the front–end files are placed on is called the FRONT–END FILES area, or FILES–11 area. Once this area has been set–up, there is normally no need to get these files again from floppy disks. The floppy disks used to install the system become backup devices in case the public structure is destroyed, or in the case where an RP07 is being used as the system structure, they can be used to recreate your front–end file structure. It is a good idea to make an extra copy of your installation floppies in the event one of your original floppies is destroyed and you need to restore the FRONT–END FILES area. Refer to Chapter 7, System Backup Procedures, for a description of the COP program that is used to copy floppy disks.

As previously stated, the front–end files must always be placed on a dual–ported RP04 or RP06. This allows the front–end processor to access these files while the main processor accesses TOPS–20 files on the same or different disk packs.

The RSX20F monitor and its related tasks do the following:

- Control input/output and communications devices.
- Interface with the main computer.
- Load the TOPS–20 monitor at system startup, and reload TOPS–20 if a crash occurs.
- Report system errors and write these in a file.
- Perform system diagnostic functions.

Table 3–3 lists the programs and files located in the FRONT–END FILES area with a brief description of each. Those files with the file type TSK are programs that can be run under the front–end monitor (RSX20F). Files with the type MCB contain the microcode for the main memory, KL20. Files with the type EXB are bootstrap programs used to load the TOPS–20 monitor.

Table 3–3: Console Front–End Files

File	Contents or Function
F11ACP.TSK	File handler for front–end disk files.
PARSER.TSK	The front–end command parser (prompts PAR>). The primary means of access to front–end programs.
TKTN.TSK	Terminates tasks, reports errors and requests reloads.
MOU.TSK	Mounts a device for use with the front end.
BF16N1.A11	MOS memory timing RAM file. It is a nonexecutable file containing MOS memory data.
SETSPD.TSK	Sets system parameters, such as line speeds.
KLXFER.TSK	Transfers KLERRO.SNP to TOPS–20 error file.
KLE.TSK	KLERR program for error processing.
KLI.TSK	KLINIT program for initializing the central processor (KL20). Loads microcode, configures cache and main memory, loads bootstrap.
KLA.MCB	Microcode file.
KLX.MCB	Microcode file.
BOOT.EXB	The central processor disk bootstrap program that boots TOPS–20 from disk.
RED.TSK	Tells the system where to look for the front–end files device, SY0:.
SAV.TSK	Saves the front–end monitor and bootstrap on disk.
DMO.TSK	Dismounts a front–end device and allows a reboot.
T20ACP.TSK	Interfaces between front–end and TOPS–20 file systems.
UFD.TSK	Sets up user–file–directories in the front–end files area.
INI.TSK	Initializes a front–end files area.
PIP.TSK	Front–end program for file transfer.
COP.TSK	Copies the contents of floppy disks.
KLRING.TSK	Provides the KLINIK line ring service.
MTBOOT.EXB	Boots a TOPS–20 monitor from magnetic tape.
KLDISC.TSK	Provides the KLINIK line disconnect service.
MIDNIT.TSK	Updates the time of day through midnight.
ZAP.TSK	Makes binary modifications to task images.
RSX20F.SYS	Virgin image of front–end monitor (RSX20F).
BOO.TSK	Used to boot RSX20F.

3.5 Microprocessor Files (2020 only)

The microprocessor file system is created during software installation, or when the entire file system is rebuilt. The microprocessor is in the same cabinet that contains the central processor, internal MOS memory, and mass storage controllers. The microprocessor serves as the 2020 console and handles the following functions and has the following responsibilities:

- Boots the system.
- Controls the console and lights.
- Monitors the keep–alive count.
- Monitors parity errors and memory refresh errors.
- Controls the diagnostic console.
- Controls the CTY driver.

The microprocessor needs a special set of files to operate properly. These files are programs that the microprocessor reads from disk. Table 3–4 lists these programs and their basic functions. The microprocessor files are contained in the directory PS:<SYSTEM>.

Table 3–4: Microprocessor Files

File	Contents or Functions
KS10.RAM	Microcode for the main processor.
KS10.ULD	Microcode used with SMFILE.EXE to build the microprocessor file system. (Refer to the *TOPS–20 Software Installation Guide* for a description of running SMFILE.)
MTBOOT.RDI	Magnetic tape boot program.
SMBOOT.EXE	Disk boot program.
SMFILE.EXE	Program used in conjunction with the BOOTSTRAP routines to create the microprocessor file system.
SMMTBT.EXE	Input file to SMFILE.EXE to create MTBOOT.RDI.

The *TOPS–20 Operator's Guide* contains a detailed description of communicating with the microprocessor via the microprocessor console program. The console program is used for boot procedures, diagnostics, error recovery, and maintenance. The *TOPS–20 Operator's Guide* also describes the KLINIK facility, which allows a DIGITAL Field Service Representative or Software Specialist to diagnose a problem in your system from a remote location.

3.6 Tailoring The Batch System

Most installations use the parameters and defaults in the distributed version of the batch system. However, you can modify some of these parameters if required by the batch processing procedures at your installation.

DIGITAL distributes a program with the TOPS–20 software that allows you to tailor the standard batch system to the requirements of your installation. This program, called GALGEN.EXE, is located in the directory PS:<SUBSYS>. You can run GALGEN at the time the system software is installed or at a later date. In either case, you must have a working batch system before you can generate a new one using GALGEN. This means if you are installing the system, you must first install the batch system that is distributed with every new version of the TOPS–20 software (on the software installation tape). You can then run the GALGEN program and tailor the batch system before it becomes available for general use.

If you tailor the batch system at a later date, you can run most of the GALGEN program with users logged in. However, for safety reasons, the system should be stand–alone during the critical phase of stopping the old batch system and starting the new one. The batch queues, however, need not be empty. That is, batch jobs can be waiting to be processed at the time you bring the system down.

The *TOPS–20 Software Installation Guide* contains the procedures for running the GALGEN program.

3.7 Checking The Software (UETP)

After the system software is installed, you or the Software Specialist can run the User Environment Test Package (UETP). UETP is a collection of programs, data files, and batch control files designed to allow you to test the integrity of various system elements. In addition to testing that the hardware has been properly installed, UETP ensures that the TOPS–20 Operating System is running and that the languages you have selected for your operation are available.

UETP creates a moderate load on the system, consisting of various defined procedures that closely resemble the load in an actual operation. Later, you may want to tailor UETP to test a software load that more closely resembles your particular system's use.

The *TOPS–10/TOPS–20 User Environment Test Package Reference Manual* describes UETP, the individual component tests, typical message information, and the procedures for adding new tests.

Chapter 4
Creating Structures

4.1 Overview

One of the first decisions you must make about your new (or upgraded) installation is what type of disk storage environment best suits your needs. Some of the considerations that determine your decision are:

- How large is the data base?
- How many users will be using the system?
- How experienced are these users?
- Will there be a full–time operator?
- How often will you run diagnostics and how critical is it that the system remain available during this maintenance?
- Must all files be available to all users at all times during system operation?

The mountable structure facility of TOPS–20 provides several options for making this decision. The option you choose depends on the answers to the previous questions and the number of disk packs and drives that are available. For example, if your installation has a number of disk packs and two or more drives, you can store data and program files on different structures.

A structure is a collection of data and program files contained on one or more disk packs and referenced under one name.

When you install your software, you create a structure known as the system structure. All packs in this structure remain on–line at all times during system operation. If your system structure does not encompass all of your available drives you can create and mount other structures. Throughout the remainder of this manual the system structure will be referred to as PS: or public structure.

Sections 4.2 through 4.9 describe the public structure and how you can best utilize your disk resources and create and use other structures.

4.2 The Public Structure

Sections 4.2.1 and 4.2.2 provide an overview of what the public structure is, including its relationship to the system and its contents.

4.2.1 What Is The Public Structure?

The public structure, commonly named PS:, is the most important structure on your system. It is created and brought on–line at system installation when you answer the appropriate questions in the installation dialog. (Refer to the *TOPS–20 Software Installation Guide.*) The name of the public structure can be up to six characters.

PS: can be one or more disk packs, depending on the configuration of your system and your disk drive resources. You may NOT use an RP20 as PS:.

While installing the software on a DECSYSTEM–2040, 2050, or 2060, you copy the console front–end files to the PS: pack that is mounted on a dual–ported drive (usually drive 0). The dual port allows the front–end processor and the central processor to access the data on PS:.

Only one structure with the name PS: can be on–line at any one time. This means you can have another structure on–line that is capable of being used as the public structure, but it cannot have the name PS: while it is mounted. If you want to mount another structure named PS:, you must first rename that structure. Section 4.5.2 provides more information about mounting structures having the same name. Section 4.5.5 describes why you would have another structure on–line that is capable of being the public structure.

All disk packs in PS: must be on--line at all times, because PS: contains all the programs, files, and swapping area that the system needs to operate. PS: also contains all user directories necessary to support users logging into the system. If the file system is destroyed on PS:, or if a drive that contains all or part of PS: malfunctions, the system halts. Refer to Chapter 9 in this manual and to the *TOPS–20 Operator's Guide* for the steps that you and the operator must follow if you have problems with the file system or if a drive goes down.

4.2.2 The Contents of PS:

The following list provides an overview of the contents of PS:.

PS: contains:

1. The TOPS–20 command processor.
2. A <ROOT–DIRECTORY> (Section 3.2.1) that points to the location on disk of all first level directories on PS:.
3. All the files in the directories <SYSTEM> and <SUBSYS> (Sections 3.2.2 and 3.2.4).
4. The directories <NEW–SYSTEM>, <NEW–SUBSYS>, <ACCOUNTS>, <OPERATOR>, <SYSTEM–ERROR> and <SPOOL> (Sections 3.2.6 and 3.2.7).
5. The front–end monitor (RSX20F) and the console front–end files for the DECSYSTEM–2040, 2050, and 2060 (Section 3.4). If you are using the RP07 as the public structure, the front–end file system must reside on either an RP04 or RP06 dual–ported disk drive.
6. The microprocessor files for the DECSYSTEM–2020 (Section 3.5).
7. The required swapping area. The size of this area depends on the TOPS–20 monitor you are using. For example, MONBIG uses 10,000 pages of disk space for swapping. (Refer to Section 4.8 for a description of the swapping area.)
8. The spooled files area. Files are kept in this area until they can be input from or output to a slow speed device such as a line printer.
9. A HOME block that contains the following parameters.
   - the structure name, in this case, it is PS:
   - the number of disk packs in the structure
   - the number of pages used for the front–end file system (usually 950)
   - the number of pages set aside for swapping
10. A directory for every user who requires access to the system. Users must log into a directory on PS: to use the system. Afterwards, they can mount and connect to a different structure and directory.

4.3 One–Structure Systems

A one–structure system consists of a single structure, PS:, which is always on–line. All packs in the structure must be on–line for the system to operate.

Usually, a one–structure system has only one or two disk drives. Smaller TOPS–20 installations choose to keep all their directories and files on one structure (PS:) for some of the following reasons.

- It is the simplest system.
- It is the easiest system to maintain.
- The installation has no major security concerns (for example, there is no requirement to physically remove packs from the drives).
- The majority of users are inexperienced.
- All files are available at all times, and thus are easy to access.
- A full–time operator may be unnecessary.
- There is only one disk drive (only one structure supported).

Chapter 5 describes the methods you can use to create and maintain directories on your one–structure system.

4.4 Mountable Structures

If PS: does not encompass all available disk drives, you can create and mount other structures on the unused drives. Non–PS: or mountable structures are created using the CHECKD program. The *TOPS–20 Operator's Guide* describes creating structures with CHECKD.

NOTE

PS: is the only structure created at installation time. All other structures are created (using the CHECKD program) and brought on–line during system operation.

4.4.1 Differences Between Mountable Structures and PS:

Unlike PS:, a mountable structure can be mounted and dismounted during timesharing. Also, it need not contain a front–end file system. Therefore, on the 2040, 2050, and 2060, a mountable structure does not have to reside on a dual–ported disk drive. Although a mountable structure has its own <ROOT–DIRECTORY> and directory system, a user cannot log into a mountable structure, but must log in as a user on PS:. A user can then mount a different structure and connect to directories. Table 4–1 summarizes the differences between a mountable structure and PS:.

4.4.2 Similarities Between Mountable Structures and PS:

There are, however, many similarities between the public structure, PS:, and mountable structures. Both contain user directories and files. A mountable structure can have a front–end file system, and can be used in place of PS: to load the system for timesharing. A mountable structure is created with the eight special directories (mentioned in Chapter 3) as for a public structure. Like PS:, a mountable structure has a HOME block that contains information such as the name of the structure and the number of disk units in the structure. These and other similarities between PS: and mountable structures are summarized in Table 4–2.

Table 4–1: Differences Between Mountable Structures and PS:

PS:	Mountable Structure
Always up during timsharing	Can be mounted and dismounted
Has a front-end file system	Need not have a front-end file system
Resides on a drive that is dual ported with the front-end computer	Need not reside on a dual-ported disk drive
Known to the system as PS:	Cannot have the name PS:, or else must have an ALIAS when mounted during timesharing (Section 4.5.2)
Used for logging into the system	Cannot be used for logging into the system
Belongs to the system	Can belong to a private user
Has the <SYSTEM> and <SUBSYS> directories	Need not have the <SYSTEM> and <SUBSYS> directories unless the structure will be used as PS:
Must contain a swapping area	Need not contain a swapping area unless the structure is to be mounted as PS:

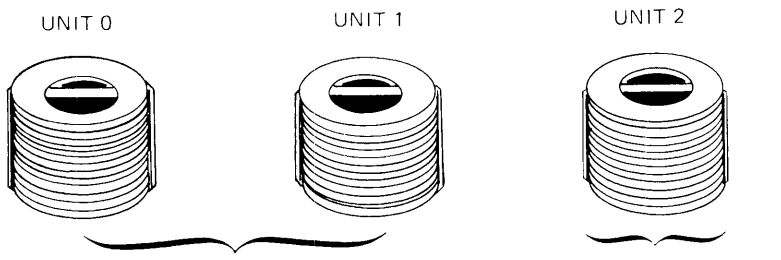
Table 4–2: Similarities Between PS: and Mountable Structures

PS:	Mountable Structures
Has a HOME block	Has a HOME block
Has a front-end files area	Can have a front-end files area
Is used to load the system	Can be used to load the system if the proper file areas and files have been established
Contains user files	Contains user files
All packs must be on-line for the system to operate	All packs in the structure must be on-line to use the structure

4.5 Multiple Structure Systems

A multiple structure system consists of a PS: and one or more additional structures. Figure 4–1 illustrates a system with three disk drives and two structures. The two-pack PS: structure must be on-line during timesharing. The one-pack mountable structure ADMIN: can be removed during timesharing. Another one-pack structure can be mounted in its place.

Figure 4–1: System With 3 Disk Drives and 2 Structures

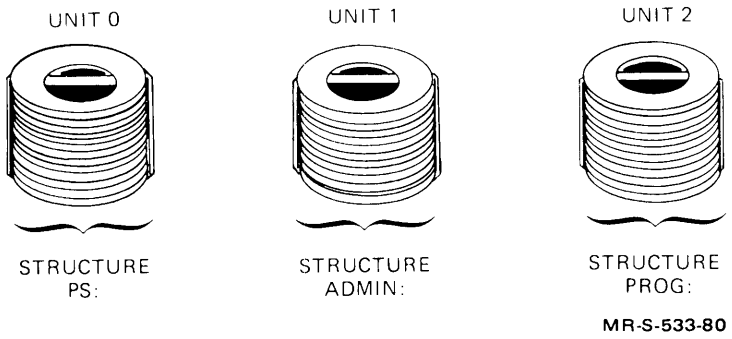


Using Figure 4–1, suppose you want structure ADMIN: to remain on-line at all times during system operation. The structure is automatically mounted if you turn on the drive that contains the structure before the system is brought up. The *TOPS–20 Operator's Guide* describes mounting structures automatically.

In addition to PS: and perhaps another permanent on-line structure, you may choose to keep one or more disk drives available for users to mount and dismount "private" packs during timesharing.

Figure 4–2 illustrates a system with three disk drives and three one-pack structures, PS:, ADMIN:, and PROG:.

Figure 4–2: Three Structure System



In this example, PS: contains all the directories necessary to support logins. ADMIN: contains the same or a subset of the same directories as those on PS: and remains on-line at all times during system operation. The drive that contains PROG: is used for short-term mounting of different one-pack structures. PROG: remains on-line only for the time it is needed.

The DECSYSTEM–2040, 2050, and 2060 can have as many as 16 structures on-line at one time. The DECSYSTEM–2020 can have 8 structures on-line at one time. Section 4.5.3 explains why using the maximum number of structures may be useful for your installation.

Several of the advantages in a mountable structure environment are:

- Some users or groups of users may require a structure exclusively for their use. They can 'own' or possibly pay for the use of certain structures.
- Service engineers can mount their own pack on the short-term drive and perform some diagnostics without disturbing normal system operation.
- Creating structures on mountable packs provides additional security to that already within the system. For example, you can create a structure that contains highly confidential data, remove it from the drive(s) when you are done with it, and lock it in a security cabinet or safe. At the end of the day, the operator locks up any confidential structures.
- In this type of environment, you are not limited in the size of your system's data base. You can create as many structures as you have disk packs to contain them, and you can mount as many at one time as your system can support.

After the system is operating and structures have been created, the operator responds to requests from users to mount and dismount structures. Section 4.5.5 describes how to place user directories on your mountable structures to obtain maximum availability to priority jobs.

4.5.1 Choosing Mountable Structure Names

Each device on the system has a name, called the physical device name, which is used when giving commands to the software. Unlike the generic device name that applies to a class of devices, for example: TTY:, DSK:, LPT:, the physical device name applies to a particular device on the system; for example, TTY6: LPT0:, PS:. The physical device names for disks are structure names. A structure name can be from one to six alphanumeric characters of your choice, and, like other device names, must be followed by a colon. The colon indicates to the software that a device is being used and not, for example, a file.

It is important to carefully assign a unique name to each structure that you create. Section 4.5.2, Mounting Structures Having The Same Name, explains why this precaution avoids confusion for users if an operator is unavailable during timesharing. Because structure names are used in the device field (dev:) of a file specification, you should not create any structures with the same name as a defined (or valid) device name. Table 4–3 lists sample device names that may be defined in your system.

For the same reason, avoid naming a structure with a defined logical name, for example, SYS:, SYSTEM:, NEW:, OLD:, HELP:, etc. because the system searches the list of defined systemwide logical names before device names. Refer to Section 3.3 for a description of logical names.

Table 4–3: Sample Device Names

DSK:	CDP:
PS:	FEn:
MTAn:	TTY:
MTn:	TTYn:
LPT:	PTYn:
LPTn:	NUL:
PLPTn:	PLT:
CDR:	PLTn:
PCDRn:	DCN:
PCDPn:	SRV:
NET:	
Where n is the unit number of the device.	

Because the system is aware only of structures that are on-line and cannot list the structure names already used, you should keep an accurate list of all the existing names. The log book may be an appropriate place to keep this list.

4.5.2 Mounting Structures Having the Same Name

A situation may arise requiring you to mount a structure that has the same name as a structure that is already on-line. Perhaps another installation has requested you to mount its public structure (named PS:) for testing purposes, but you already have a structure named PS: on-line. Because the system notices ambiguous structure names, you must mount the structure under a different name.

Each structure that is mounted is identified with two names: the physical identification and the alias. Usually, these names are the same. The physical identification is the actual structure name written in the HOME block of that structure. The alias is the name that you use to reference the structure while it is mounted. After a structure is on-line, it is known only by its alias. The MOUNT command is used to mount a structure and give it an alias different from the physical identification. This allows two or more structures with the same physical name to be on-line simultaneously. The system distinguishes them by their different aliases. (The *TOPS–20 Operator's Guide* describes this procedure.)

4.5.3 Maximum Size of Structures

The maximum size of any structure you create (including PS:) depends on the type of system you have and your disk drive resources. The maximum size of a structure on the DECSYSTEM–2020, 2040, and 2050 is approximately 152,000 pages. A structure of this size requires 4 RM03/RP04 disk drives, or 2 RP06 disk drives. The RM03 drive is used with the 2020 and the RP04 is used with the 2040, 2050, or 2060. The RP06 can be used with any DECSYSTEM–20. The RP07 can be used only with a DECSYSTEM–2060. The maximum size of a structure on a DECSYSTEM–2060 is approximately 604,260 pages. A structure of this size requires 2 RP20 disk drives (3 spindles).

Structures of the maximum size, however, may not be practical for your installation. Smaller structures enhance the reliability and availability of the system. Remember that you can have as many as 16 structures on–line at one time on a DECSYSTEM–2040, 2050, or 2060, and 8 structures on–line at one time on a DECSYSTEM–2020.

Also, if a structure is contained on more than one disk pack, the packs and drive units for that structure must all be the same type, that is, either all RM03s, RP04s, RP06s, RP07s, or all RP20s. For example,

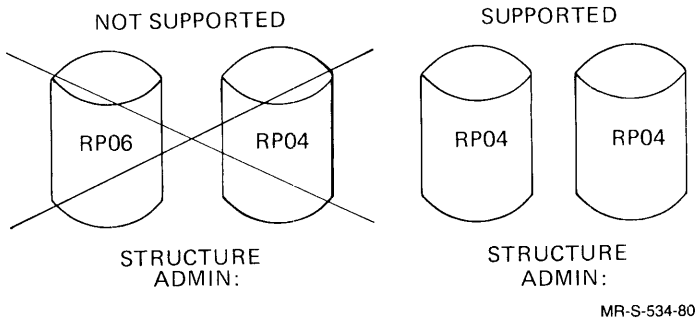


Table 4–4 lists the individual configurations for the maximum size structure on a DECSYSTEM–2020, 2040, 2050, and 2060, using RM03, RP04, RP06, RP07, and RP20 disk drives. Note that when a disk drive is mentioned with only one type of system, you can use that drive only with that system.

NOTE

The number of directories per structure and files per directory that can be created is approximate. This is because the disk space needed to create a directory or file varies. *Also, if you plan to mount 2060 structures on other DECSYSTEM–20's and vice versa, refer to Section 4.7 for the size limitations that you should consider before doing this interchange.*

Table 4–4: Maximum Size Structures

Type of Disk Drive	Max.No.Packs Per Structure	No. Pages Per Pack	No. Directories Per Structure	No. Files Per Dir.
DECSYSTEM–2020				
RM03	4	30340	4,000	365
RP06	2	76000	4,000	365
DECSYSTEM–2040,2050				
RP04	4	38000	4,000	365
RP06	2	76000	4,000	365
RP20	1	201420	4,000	365
DECSYSTEM–2060				
RP07	1	216376	12,000	5,000
RP04	6	38000	12,000	5,000
RP06	3	76000	12,000	5,000
RP20	3	201420	12,000	5,000

4.5.4 Increasing the Size of Structures

You can add more disk packs to increase the size of a mountable structure (not PS:) during timesharing. To do this, you must:

- Dump the entire file structure onto magnetic tape using the DUMPER program
- Run the DLUSER program
- Run the CHECKD program, specifying the new configuration, to re–create the structure
- Restore all the directories and files from magnetic tape using the DUMPER program

IMPORTANT

If possible, re–create the structure and restore the files to a different set of packs from the structure that you dumped. This precaution ensures that you do not lose any valuable data should you have problems reading the tape back to disk; that is, you still have the original structure intact and can rerun DUMPER and copy the structure to another tape.

To increase the size of PS:, you must shutdown the system and follow the installation procedure for bringing up PS: with more disk packs. Refer to Chapter 9, System Problems/Crashes, and to the *TOPS–20 Software Installation Guide*.

4.5.5 Setting Up Structures for Maximum Availability

Before you create structures and place user directories on them, you should determine which users must be on the system at all times. Place these users' directories and files on PS:. Divide the remaining users of the system by priority and place their directories and files on the other structures. Although these users have log-in directories on PS:, their large working area where they create and store files is on the other on-line structures. Dividing users into categories and placing them on structures accordingly ensures that the failure of one disk drive does not prevent the most important users from using the system. For example, if the drive that contains ADMIN: goes down, you can remove the ADMIN: pack from the broken drive and mount it on another drive that contains a less critical structure.

Also, on-line disk diagnostics can be performed during timesharing. Sometimes, the service engineer can dismount a non-critical structure, mount the maintenance pack, and perform preventive maintenance or troubleshooting with only a portion of the user community off-line.

To increase system availability, you can create another PS: for backup using the CHECKD program. After you create this structure, you should follow the procedures in the *TOPS–20 Software Installation Guide* for creating the front-end files area. The *TOPS–20 Operator's Guide* describes using the CHECKD program to create a backup PS:. If you have problems with the primary PS:, having a second PS: available allows you to resume timesharing without reinstalling the system.

The backup PS: can be mounted and on-line at all times under another name, or it can be kept in storage and mounted as a backup if the regular PS: is destroyed. If the backup PS: is kept in storage, the operator must update the structure periodically with PS: System Backup Tape and the latest incremental dumper tapes. (Chapter 7, System Backup, describes creating and using your PS: System Backup Tape and incremental tapes.) Occasionally updating your PS: backup (in storage) keeps it reasonably up to date.

If you keep your backup PS: on-line at all times, and you have important files that are constantly accessed by the user community, you can improve your system performance by placing these files on the backup PS:. Now your swapping area and the files that you access frequently are not on the same disk. This procedure is useful with any structure that you keep on-line at all times.

4.5.6 Taking Structures Off-line

When a structure must be taken off-line, the operator should notify users that it will be dismounted at a certain time. Users should give the DISMOUNT command for the structure before the specified time. If the users

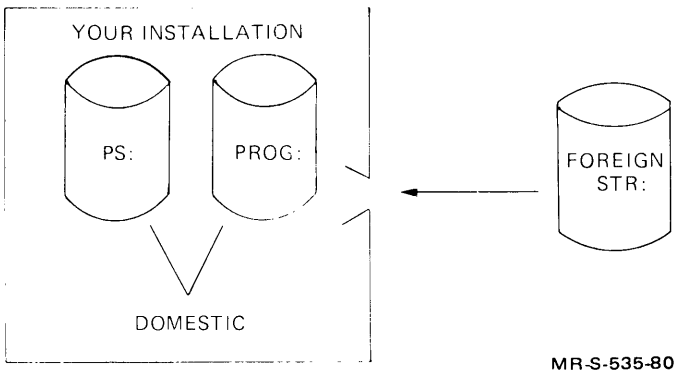
do not cooperate, the operator can dismount the structure (via the DISMOUNT command to OPR) without leaving files in an unknown state. Files that are open simply become inaccessible and the user who had the files open receives an error.

If a structure is taken off-line, all the packs in that structure must be taken off-line, not just one.

4.5.7 Mounting Structures from Another Installation

If you mount a structure from another installation or perhaps a structure that contains confidential data, some of the user names on this structure may match the user names on your public structure. You must mount this structure in what is called a FOREIGN state to avoid the mishap of your users accessing directories that do not belong to them. The same is true if you bring one of your structures to another installation. You should have the operator at the installation mount your structure as a FOREIGN structure. Figure 4–3 illustrates this concept.

Figure 4–3: Domestic and Foreign Structures



Structures are brought on-line in one of two states, DOMESTIC or FOREIGN. If you do not specify the DOMESTIC state in PS:<SYSTEM> MOUNTR.CMD file, the system uses the FOREIGN state as the default. The structure remains in the FOREIGN state for as long as it is mounted or until the operator changes the state with the SET STRUCTURE command to OPR.

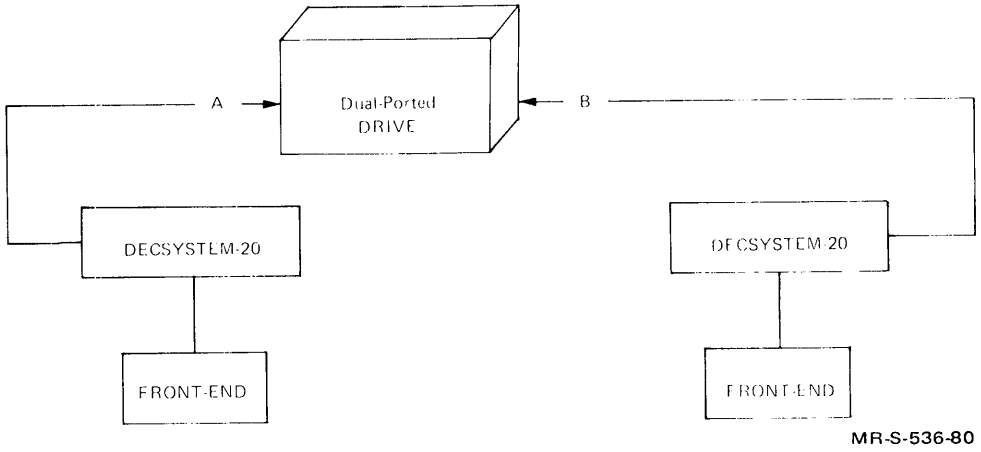
You should bring a structure on-line as DOMESTIC only if the directories on that structure were created for the same people as those on PS:. One can be a subset of the other, but a given directory name should represent the same person on both. Conversely, you should bring a structure on-line as FOREIGN if the directories on that structure were not necessarily created for the same people as those on PS:. This is because a user can log into PS: and give the CONNECT or ACCESS command to a directory with the same user name on a DOMESTIC structure, without giving a password (providing the directory protection allows this type of access). However, a user who logs into PS: and gives the CONNECT or ACCESS command to a directory with an identical user name on a FOREIGN structure must give the associated password.

4.6 Sharing Structures (Disk Drives) Between Two Systems

If you have two DECSYSTEM–20's and one or more structures that contain data common to both these systems, you may want to set up your system to share disk drives concurrently. For example, you could allow System A to use the drive that contains structure ADMIN: in the morning and allow System B to use this structure on the same drive in the afternoon. *THE SYSTEMS CANNOT, HOWEVER, ACCESS THE DRIVE AT THE SAME TIME.* Also, if one of your systems goes down, you can still use the drive that is connected to both systems.

A drive that is to be shared by two systems must be supported by both systems. For example, you cannot connect an RM03 disk drive to a DECSYSTEM–2020 and a DECSYSTEM–2040 because the 2040 system does not support RM03s. Also, the shared drive must be dual ported. Your field service representative must make the appropriate connections from each DECSYSTEM–20 to a port on the disk drive. Be sure to have the field service representative tell you which system is connected to which port on the drive. Figure 4–4 illustrates this connection.

Figure 4–4: Shared Disk Drive



The port switch on this drive must be in either the A or B position. *NEVER have the port switch in the A/B position.* If users on both systems attempt to access the structure at the same time and the port switch is in the A/B position, the contents of the disk pack can be destroyed.

To use the drive, place the port switch in the position that corresponds to the first system that is using the drive (A or B). The operator mounts a structure using the normal procedure. After the first system is no longer allowed to use the drive, the operator gives the DISMOUNT command for the structure.

To use the drive on the second system, the operator leaves the pack on the drive (if you are using the same structure), turns the drive off-line, changes the port switch to the corresponding system, and turns the drive back on.

Then, the operator (or a user) gives the MOUNT command to mount the structure on this system. The system automatically recognizes that another drive is on-line and mounts the structure.

4.7 Interchanging Structures Between Different Systems

Sections 4.7.1 and 4.7.2 contain information to assist you if you have more than one DECSYSTEM–20 and plan to interchange structures among them.

4.7.1 Mounting 2060–Structures on Other DECSYSTEM–20's

If you create a structure on a DECSYSTEM–2060, determine if the structure will ever be mounted on a DECSYSTEM–2020, 2040, or 2050. (In the case of the 2020, the 2060-structure must be on RP06 disk packs to move it to the 2020. The 2020 cannot have RP04 disk drives.) When interchanging 2060-structures with other DECSYSTEM–20's, you should limit the number of directories and files created on the 2060 structures to the maximum allowed for other systems. The 2060 can have approximately 5,000 files per directory and approximately 12,000 directories per structure. The limit for all other DECSYSTEM–20's is approximately 365 files per directory and approximately 4,000 directories per structure.

If you do not limit the number of directories and files created on a 2060-structure, you may have problems mounting that structure on another system. When you subsequently mount a 2060-structure on another DECSYSTEM–20, all the files in directories that have grown beyond the maximum limit for these other systems cannot be accessed. Files cannot be accessed because the symbol table that keeps track of and locates (maps) each file in a directory, including <ROOT–DIRECTORY>, is located in the last page(s) of the directory. When a file is created in a directory, the symbol table moves to the last page. Therefore, a directory on a 2060-structure that exceeds the file limit on other DECSYSTEM–20's cannot map any of its files because the system cannot find the symbol table. In the case of <ROOT–DIRECTORY>, the structure cannot be mounted because the system cannot find the symbol table and cannot map <ROOT–DIRECTORY>. Note that you cannot mount a structure unless <ROOT–DIRECTORY> is found.

To limit the number of directories and files that are created on a 2060 structure, mount the structure, enable capabilities, and run the CHECKD program with the LIMIT command. For example,

```
CHECKD>LIMIT (Number of Files Per Directory On Structure)str:
```

limits the number of directories and files to the maximum allowed for other systems. Unless the LIMIT command is given, the 2060 structure is unlimited and can have the maximum number of files and directories allowed on that structure.

When a 2060-structure is no longer used on other 20's, you can run CHECKD again and give the UNLIMIT command:

```
CHECKD>UNLIMIT (Number of Files Per Directory On Structure)str:
```

The 2060-structure can now grow to its maximum limits.

4.7.2 Mounting 2020, 2040, 2050 Structures on Other DECSYSTEM–20's

If you create a structure on a DECSYSTEM–2020 and plan to mount that structure on a non-2020 system, be sure to create the structure on an RP06 disk pack. The RM03 that is supported on the DECSYSTEM–2020 is not supported on the other system models.

Also, if you plan to interchange a 2020, 2040, or 2050 structure with a DECSYSTEM–2060, you MUST run CHECKD and limit the potential size of the structure while it is mounted on the 2060 system, so it can return to the smaller system. It is imperative that you do this; otherwise you risk not being able to return your structure to its original non-2060 system.

4.8 Determining Swapping Space On PS:

Sections 4.8.1 and 4.8.2 describe what swapping space is and how to determine the amount of swapping space that you should allocate for your system.

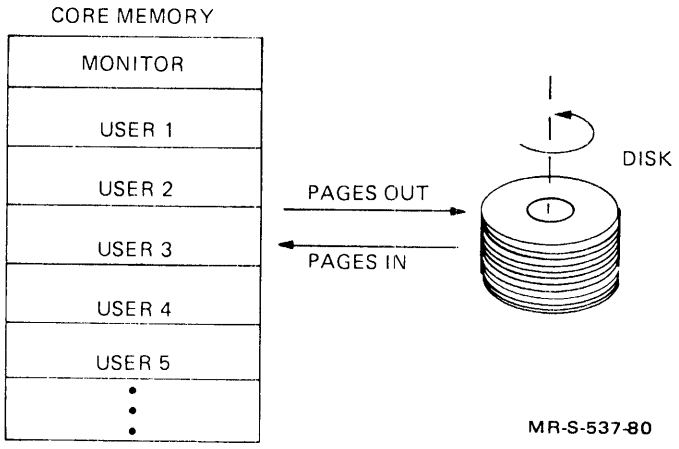
4.8.1 What is Swapping?

The number of user jobs that can fit into main memory simultaneously depends on the size of the individual jobs, the size of the memory-resident portion of the monitor, and the size of memory physically available. (Only a portion of the TOPS–20 monitor resides in main memory at one time.) If a user wishes to run a job that is not currently in memory, space must be provided. This may necessitate moving some other job out of memory. The user's program or data that is transferred out of memory is placed on disk in the swapping area. The system sets aside a portion of the disk storage space on PS: specifically for this purpose.

On some timesharing systems, a program must be entirely in main memory to execute. Swapping then consists of moving entire programs between disk and memory. Under TOPS–20, only portions of a program (those containing the instructions and data currently being referenced) need be in memory. Other portions of the program are brought into memory from disk as they are needed. In this case, swapping consists of moving portions of a program or data between disk and memory. The monitor decides which portions of which programs to swap, and when.

The size of a program is measured in a unit called a page. When swapping occurs, some of these pages are copied between memory and disk. Figure 4–5 illustrates the concept of swapping.

Figure 4–5: Swapping Concept



4.8.2 When to Increase Swapping Space

For the most part, the size of your swapping space depends on the number of jobs you estimate will be on the system at any one time. Table 4–5 contains guidelines for estimating the amount of swapping space required for an approximate number of user jobs. This amount is given in response to the question "HOW MANY PAGES FOR SWAPPING" in the software installation procedures.

The actual disk space used for swapping depends on the number of pages you give. The system rounds the number of pages given upward to an integral number of cylinders. The algorithm that follows Table 4–5 can be used to determine if the number of pages you choose for swapping space is the optimum number for the actual disk space used. For example, the number 5500 may require an additional cylinder on the disk, whereas 5000 may give you the number of pages you require in the least amount of disk space. As you can see in the algorithm, the number you should choose also depends on the number of disk packs in the structure. The swapping space is divided equally among the disk packs in the public structure.

You can allow for swapping space on structures other than PS:. However, it is necessary only if you plan to mount the structure as PS: in the future. Allocating swapping space avoids re-creating the structure should you decide to mount it as PS:.

All the monitors are designed to default to an appropriate number of pages for swapping. In most cases, you can take this default. The algorithm following Table 4–5 simply provides you with a means of checking that the default is sufficient for your particular system.

The guidelines in Table 4–5 apply to systems whose users perform many editing jobs and an average or small amount of debugging programs and production jobs. If your users perform a great number of debugging and production jobs and only a small amount of editing, you should double the

size of your swapping space. However, if you double the size of your swapping space, check the maximum swapping space allowed for the monitor you are running. (The *TOPS–20 Software Installation Guide* lists the maximum number of swapping pages you can use with each monitor.) You cannot exceed this maximum. If you enter a number that is larger than the maximum, the monitor uses the maximum allowed. If you must exceed the maximum, you can bring up a larger existing monitor, or you can tailor your monitor by following the instructions in the BUILD.MEM file. This file is located in the saveset documentation files on the TOPS–20 Software Distribution tape.

Table 4–5: Determining Swapping Space

Estimated Number of Jobs	Recommended Number of Pages for Swapping*
20 or less	2500
21 to 30	3500
31 to 40	4500
41 to 50	5500

* For each additional 10 jobs, increase the number of pages for swapping by approximately 1,000.

Your estimate of the required number of pages for swapping is inserted as the value of X in the following algorithm.

X = the number of swapping pages you chose from Table 4–5

Y = the number of pages per cylinder (Y = 95 pages for an RP04 or RP06 or 37 pages for an RM03).

Z = the number of packs in a structure * Y

$$\text{ACTUAL DISK SPACE FOR SWAPPING} = \frac{X + Z - 1}{Z} * Z$$

(Note that this is integer arithmetic; do not cancel out any factors)

The following example uses the system default number 5000 as the value of X (the number of swapping pages) and shows how much disk space will actually be allocated for a structure that is composed of two disk packs.

Example:

X = default swapping space for monitor MONSML. The default for MONMED, MONBCH, and MONBIG is 7,000 pages. The default for monitor MONMAX is 10,000 pages.

Y = 95

Z = 2 (packs in structure) * 95 = 190

```
1)   5000 + 190      = 5190
2)   5190 - 1        = 5189
3)   5189 / 190      =   27
4)    190 *  27      = 5060
```

5060 = The actual swapping space used for swapping in a 2–pack structure.

4.9 Determining The Available Disk Space

4.9.1 Determining Disk Space Before Installation

To determine the available disk space that you will have to divide among your users before installing the system, first calculate the swapping space required by your system (Section 4.8.2). Second, insert the number you calculated for swapping space into the formula shown in Table 4–6 and perform the appropriate steps.

Table 4–6 outlines how to approximate the available disk space on PS:. If you are calculating the available disk space on structures other than PS:, follow this same procedure but eliminate reserving space for any directories or areas that are not on that structure. If any possibility exists that a structure may be used as a PS:, reserve the swapping space.

NOTE

Remember that as your system expands, the number of pages in the <SYSTEM> and <SUBSYS> directories increases. Also, the number of pages reserved for directory <SPOOL> should be increased if you predict a continual backlog on the line printer.

When figuring disk space for a DECSYSTEM–2020, reserve the console front–end file system only if you plan to use your PS: RP06 packs on a 2040, 2050, or 2060. RM03 disk packs cannot be used on a 2040, 2050, or 2060.

Table 4–6: Calculating Available Disk Space

TOTAL DISK SPACE:

Number of RM03 disk drives	*	30340 pages per drive	= _____ TOTAL DISK SPACE
OR:			
Number of RP04 disk drives	*	38000 pages per drive	= _____ TOTAL DISK SPACE
OR:			
Number of RP06 disk drives	*	76000 pages per drive	= _____ TOTAL DISK SPACE
OR:			
Number of RP07 disk drives	*	216376 pages per drive	= _____ TOTAL DISK SPACE
OR:			
Number of RP20 disk drives	*	201420 pages per spindle	= _____ TOTAL DISK SPACE

RESERVED DISK SPACE:

BOOTSTRAP.BIN file	=	64 (2020 only)		
Front–end file system	=	950		
Swapping Space (Enter number of pages selected and allocated for swapping)	=			
		2060	*2040,2050*	*2020*
<SYSTEM>	=	1876	2523	1904
<SUBSYS>	=	1780	1780	1780
<NEW–SYSTEM>	=	2	2	2
<NEW–SUBSYS>	=	2	2	2
<OPERATOR>	=	3	3	3
<UETP.*>	=	1701	1701	1701
<ROOT–DIRECTORY>	=	9	9	11
<SPOOL (you should reserve)>	=	1000	1000	1000

_____ TOTAL RESERVED SPACE

SUBTRACT TOTAL RESERVED SPACE FROM TOTAL DISK SPACE _____

AVAILABLE DISK SPACE

4.9.2 Determining Disk Space After Installation

Shortly after you have installed the system, you can log in as OPERATOR and give the command INFORMATION (ABOUT) DISK–USAGE. One line of output tells you the actual number of system pages that are available on PS:. You can divide this disk storage among your users. (Refer to Chapter 5, *Creating Directories*.)

Chapter 5
Creating Directories

A prospective user who requires access to the system must be assigned a user name (normally the user's surname), a password, an account, and disk storage quotas, and must have a directory created for him on PS:. Optionally you can assign certain capabilities and/or make the user or the user's directory a member of one or more groups. (Refer to Section 5.8 for a description of how to establish group relationships among users and Section 5.9 for a description of the capabilities you can assign to users.) You can also create additional directories for users on non-PS: structures.

This chapter describes three methods you can use to create and maintain directories. Using one method, the operator creates and maintains all the directories on the system. A second method allows you to delegate the responsibility for creating and maintaining directories to project administrators. The third method combines the first two methods thus providing additional flexibility. Sections 5.1 through 5.3 explain these methods and the determining factors for choosing one of them. These sections also include some of the decisions necessary to assign user names and capabilities, and how to allocate disk storage according to the method you use to create and maintain directories. (Refer to Chapter 4 to determine the amount of disk space available to divide among the directories you create.)

Chapter 6 describes how to set up an accounting scheme and assign and validate accounts. You should read Chapter 6 if you want to allow users to log into the system immediately after you have created their directories and charge their computer usage to valid accounts.

5.1 Having the Operator Create and Maintain All Directories (Central Control)

In this type of installation, the operator creates a directory on the public structure, PS: for each new user, and specifies the appropriate parameters. The name of the directory is the same as the assigned user name. Each user informs the operator when a change to his directory parameters is required. This type of operation allows you as system manager to have *central* administrative *control* over all directories and parameters.

Therefore, central control means that you or the operator create and maintain all the directories for all your system users. Central control has two types of directory schemes. One scheme allows you to create up to approximately 365 directories per structure on a DECSYSTEM–2020, 2040, or 2050, and up to approximately 5,000 directories per structure on a DECSYSTEM–2060. The other scheme allows you to use subdirectories and create up to approximately 4,000 directories per structure on a DECSYSTEM–2020, 2040, 2050, and up to approximately 12,000 directories on a DECSYSTEM–2060.

NOTE

The number of directories allowed per structure is approximate because the disk space needed to create a directory or file varies. Throughout this chapter, the notations 365(5,000) and 4,000(12,000) mean that the number of directories you can create per structure depends on the model system you have.

5.2 Delegating the Creation and Maintenance of Directories To Project Administrators (Project Control)

An alternative type of installtion involves *project* administrative *control.* Under this type of control, the operator creates directories only for the users who have been designated as project administrators, (e.g., the representatives of major departments). The project administrators, in turn, create subdirectories for users within their departments or projects and control the assignment of those users' directory parameters. This type of control allows you to delegate the responsibility for creating and maintaining directories for other users and still maintain ultimate control over your system and its resources.

Therefore, project control means that most of the directories created by you or the operator are project directories (e.g., MATH might be the assigned name of a project). The system's resources, such as disk space, are divided among these project directories either equally or according to the expected size of the project. Subdirectories are then created under project directories for users within the project. The people who have been appointed project leaders or administrators are responsible for creating, assigning parameters to, and maintaining the subdirectories within their project. The resources that you allocate to the project directory are divided among its subdirectories by the project administrators. Under project control, you are allowed to create up to approximately 4,000 directories (including subdirectories) per structure on a DECSYSTEM–2020, 2040, or 2050 and approximately 12,000 directories (including subdirectories) per structure on a DECSYSTEM–2060.

5.3 Combining Central and Project Control

A combination of central and project control can be used if you want to keep the majority of the user directories at the management level of control and separate only a portion of your system into projects and administrative control.

Therefore, combining central and project control means that the operator creates and maintains directories for most of the system users and creates project directories for special projects. The project administrators create directories under the project directories and are responsible for maintaining them. Combining the two types of control still allows up to approximately 4,000(12,000) directories per structure.

5.4 Central and Project Control Descriptions

Section 5.4.1 through 5.4.4 describe the two types of central control, project control, and the combination of central and project control. Each description includes:

- The determining factors for choosing a particular control
- The format, including a diagram, of each type of control
- The procedure for assigning user names
- The procedure for creating user directories
- The procedure for creating files-only directories
- The restrictions, if any, that apply to using a particular control

Also, any additional considerations that apply to headings within each description are included.

Read each description thoroughly. The first central control description contains very general information and suggestions that apply to all the directory schemes.

5.4.1 Central Control

DETERMINING FACTORS:

- Your business installation is relatively uncomplicated; therefore, there is no need to separate projects and assign the creation and control of directories to various administrators. All the directories on the system are created and maintained by you or the system operator.
- You are sure that the number of directories you need is less than approximately 365(5,000).

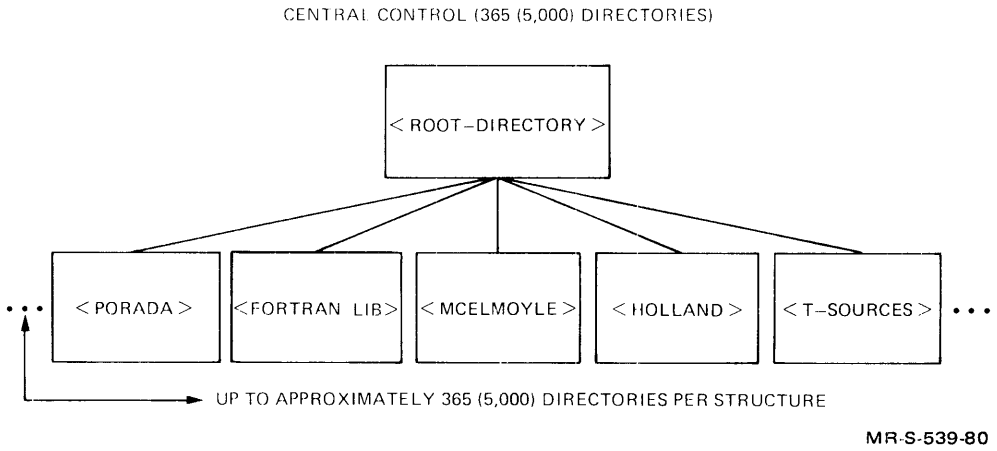
5.4.1 Central Control (Continued)

FORMAT:

<ROOT–DIRECTORY> can point to approximately 365(5,000) directories per structure.

All directories under <ROOT–DIRECTORY> are on one level.

DIAGRAM:



ASSIGNING USER NAMES:

The user name that you assign should be as close as possible to the user's last name. This convention is true for any type of directory scheme that you use. The system uses this name when recording the authors of files, sending mail to users, and displaying system status. If you follow this convention, you can easily identify who is using the system when you give a SYSTAT command.

CREATING USER DIRECTORIES:

All directories are created using the ^ECREATE command. (Only users who have WHEEL or OPERATOR capability enabled can use this command.) In the next example, the operator is connected to PS: and uses the ^ECREATE command to create a new directory named <BECKER> for the user who has been assigned the user name BECKER. Also, the operator assigns the password MARTIN.

```
@ENABLE (CAPABILITIES)(RET)
$^ECREATE (NAME) PS:<BECKER>(RET)
[NEW]
$$PASSWORD MARTIN(RET)
$$(RET)
$DISABLE (CAPABILITIES)(RET)
@
```

This directory is called the user's logged-in directory and is always on PS:. Whenever the user logs into the system, he is connected to this directory. He can remain in this directory or connect to and use files in another directory.

5.4.1 Central Control (continued)

Refer to the *TOPS–20 Operator Command Language Reference Manual* for a complete description of the ^ECREATE command that the operator uses to create new directories, and the ULIST program that prints information about all the directories on the system.

After creating a new directory (either files-only or user), remember to update the tape containing the monitor, TOPS–20 Command Processor, DLUSER, DLUSER data, DUMPER, <SYSTEM>, and <SUBSYS>. (Refer to Chapter 7, System Backup Procedures.)

CONSIDERATIONS:

If two users have mistakenly been assigned the same user name and you try to create the second directory with this duplication, the system prints [OLD] instead of [NEW]. Give the ABORT subcommand, assign the user a slightly different user name, and reissue the ^ECREATE command with the new directory name. A common practice is to precede such names with the user's first initial. This allows recognition on the user or directory name without typing the entire name and the distinguishing character. For instance, if you have the two users Stephanie Sheldon and Andrew Sheldon, you should assign them the user names S–SHELDON and A–SHELDON or SSHELDON and ASHELDON rather than use the names SHELDON–S and SHELDON–A.

CREATING FILES–ONLY DIRECTORIES:

If a user wants to have a library area in addition to his logged-in directory, you can create a files-only directory on PS: or on another structure. The user can gain owner privileges to this directory by giving the CONNECT command and the password associated with the directory. If the directory is located on a structure different from PS:, the user must also give the MOUNT command to use the structure before he gives the CONNECT command. The user cannot give the ACCESS command for or log into a files-only directory.

For example, if you have a user named BECKER who processes payroll on a regular basis, he may want to develop the payroll programs in his directory and keep the payroll data in a more restricted directory. To accomplish this, you can create the logged-in directory PS:<BECKER> and the files-only directory PS:<PAYROLL>. The directory <PAYROLL> can be on some other structure, (e.g., ADMIN:<PAYROLL>). BECKER now has normal protection on his directory, more restrictive protection on the directory <PAYROLL> and can still CONNECT to <PAYROLL> by giving its password. BECKER cannot, however, give the ACCESS command for or log into <PAYROLL>.

5.4.1 Central Control (continued)

The next example shows how to create the directory <PAYROLL> on PS:.

```
@ENABLE (CAPABILITIES)(RET)
$ CREATE (NAME) PS:<PAYROLL>(RET)
[NEW]
$$PASSWORD MONIES(RET)
$$FILES (ONLY)(RET)
$$PROTECTION (OF DIRECTORY) 774000(RET)
$$DEFAULT (FILE) PROTECTION 770200(RET)
$$(RET)
$DISABLE (CAPABILITIES)(RET)
@
```

Now if user BECKER logs in and wants to use the files in <PAYROLL>, he can give the following CONNECT command.

```
@CONNECT (TO DIRECTORY) <PAYROLL>(RET)
Password: monies(RET)
```

The *TOPS–20 Operator Command Language Reference Manual* describes all the parameters you can give to directories and describes how to create directories on structures other than PS:.

CONSIDERATIONS:

When you create additional directories on structures other than PS:, consider if files–only directories are suitable. Some users will not want to give the CONNECT command to a directory each time they require owner access to the files in that directory. Also, files–only directories are members of groups only as directory group members and not user group members. Therefore, if you create ALL the directories on a structure as files–only, you cannot establish any valid user group relationships among those directories. (Refer to Section 5.8 for a description of setting up groups.)

Conversely, if you create user directories, users can give the ACCESS command to their additional directory and gain owner and group privileges without connecting to the directory, and they can use other directories on the structure as group members. Also, if the name of the directory you create is the same as the user's logged–in directory, and the structure is mounted as DOMESTIC, the user does not have to specify a password when giving the CONNECT or ACCESS command to the directory. (Refer to Section 4.5.7.) This is valuable when the user is submitting a batch job. No password is required on the batch input, therefore, security is preserved. In addition to creating user directories on the structure, you can create files–only directories to be used as library areas.

It is also possible to create only one user directory on a structure and create all other directories as files-only. In this case, all users required to use a files-only directory on the structure could give the ACCESS command to the user directory (gaining owner and group privileges) and use the files in a files-only directory according to the group protection codes set. This would

5.4.1 Central Control (continued)

be useful if you have a private structure that contains several library areas that are common to the owners of the private disk pack(s). Each owner could give the ACCESS command for the one user directory and gain group privileges to all the library directories. Therefore, these users would need only one password to gain access to all the information on the pack.

Refer to the *TOPS–20 User's Guide* or the *TOPS–20 Commands Reference Manual* for a complete description of the CONNECT and ACCESS commands.

RESTRICTIONS:

The number of directories you create per structure cannot exceed approximately 365(5,000). This number is an approximation because the disk space that it takes to create a directory or file varies.

5.4.2 Central Control Using Subdirectories

DETERMINING FACTORS:

- As stated in the previous directory scheme, your installation does not warrant segregation of projects and control. However, this directory scheme allows more directories per structure than the previous central control scheme. You or the operator can create up to approximately 4,000(12,000) directories per structure and assign and maintain all the directory parameters.
- You can easily expand into a form of project control by adding project directories, and still maintain control at the management level over the majority of the user directories. (Refer to Section 5.4.4, Combined Central and Project Control.)

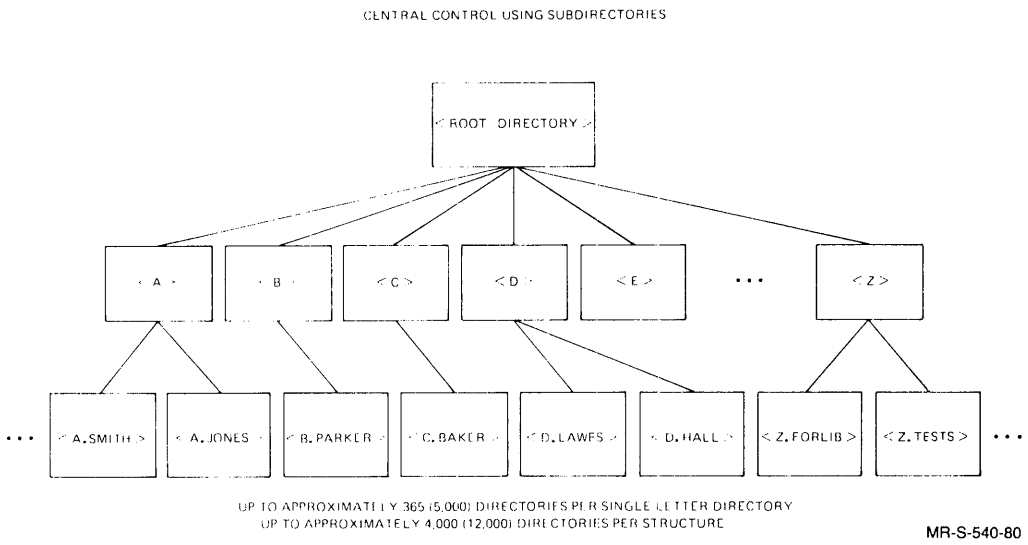
FORMAT:

<ROOT--DIRECTORY> points to 26 directories. The name of each directory is a letter of the alphabet, <A>, <B>, <C>, ..., <Z>. The directories point to all user and files-only directories. The single letter directories are on one level below <ROOT–DIRECTORY>. The user and files-only directories are on a second level below <ROOT–DIRECTORY> and are pointed to by the alphabetic directories.

The diagram below illustrates this flow. Also, although it is not shown in the diagram, the directories pointed to by the single letter directories, (e.g., <A.JONES>), can be allowed to create directories under them. Perhaps user A. JONES wants to create one or two subdirectories to store special files, such as memos. The user is responsible for maintaining the directory he created and is allowed to use only the disk quota you originally allocated to his logged-in directory. Refer to Section 5.4.3, Project Control, if you would like to allow some users to create directories of their own.

5.4.2 Central Control Using Subdirectories (Continued)

DIAGRAM:



ASSIGNING USER NAMES:

Each user name that you assign should be as close as possible to the user's last name prefixed by a first initial and a period, for example, Charles Baker would be assigned the user name C.BAKER. Under this type of directory scheme, you must follow the principle of prefixing the name with the first initial and a period.

CREATING USER DIRECTORIES:

Have the operator create 26 directories using the ^ECREATE command. The name of each directory is a letter of the alphabet, that is, <A> through <Z>.

The theory behind creating these alphabetic directories is the same as described in Section 5.4.3. That is, you must create directories that are allowed to have subdirectories. The directories <A>, <B>, ..., <Z> can have approximately 365(5,000) user and files-only directories under them. Therefore, you must include some of the same parameters in these directories, as you would in project directories.

Refer to the *TOPS–20 Operator Command Language Reference Manual* for a complete description of the parameters that are defined when the operator uses the ^ECREATE command to create directories, and the ULIST program that prints information about all directories on the system.

The procedures for creating the alphabetic directories and the user and files-only directories under them are described below.

5.4.2 Central Control Using Subdirectories (continued)

NOTE

The general considerations described in Section 5.4.1 for creating directories are also applicable to this directory description.

Even though the alphabetic directories are not associated with users, they must be created as log-in directories. However, do not assign passwords. This prevents users from gaining access to these directories.

```
@ENABLE (CAPABILITIES)RET
$ ECREATE (NAME) PS:<A>RET
[NEW]
$$
```

Assign each directory a large number for creating subdirectories, that is, at least 400.

```
$$MAXIMUM SUBDIRECTORIES (ALLOWED) 400RET
```

Because many user directories are created under each of these alphabetic directories and the page quota (disk space) from these alphabetic directories is divided among (or passed on to) the user directories, you must assign the alphabetic directories a very large permanent and working page quota. Assigning them a sufficiently large page quota prevents any of these alphabetic directories from exceeding their page quota, thus requiring you to make a change to the quota at a later time. Therefore, assign each directory at least 500,000 pages of permanent and working disk page quota.

```
$$ PERMANENT (DISK STORAGE PAGE LIMIT) 500000RET
$$ WORKING (DISK STORAGE PAGE LIMIT) 500000RET
```

Assign a list of SUBDIRECTORY–USER–GROUP numbers to each directory. The list should be the same for each directory. The range of numbers you use depends on how many groups you plan to establish, (e.g., 5 groups, 10 groups, 50 groups). The numbers used in the following examples are for illustration; you can choose any sequence:

```
$$ SUBDIRECTORY-USER-GROUP (ALLOWED) 200RET
$$ SUBDIRECTORY-USER-GROUP (ALLOWED) 201RET
$$ SUBDIRECTORY-USER-GROUP (ALLOWED) 202RET
$$ SUBDIRECTORY-USER-GROUP (ALLOWED) 203RET
$$ SUBDIRECTORY-USER-GROUP (ALLOWED) 204RET
```

Later, when you create a user directory and place that user in a user group, you must enter one of the numbers in this list. No other numbers will be valid. (Refer to Section 5.4.3, for a more detailed description of why you are using this list of numbers, and Section 5.8 for establishing valid group relationships.)

5.4.2 Central Control Using Subdirectories (continued)

In the following example, the operator is connected to PS: and creates the first two directories, <A> and <B>:

```
@ENABLE(CAPABILITIES) RET
$ ^ECREATE(NAME) PS:<A> RET
[NEW]
$$ MAXIMUM-SUBDIRECTORIES(ALLOWED) 400 RET
$$ WORKING 500000 RET
$$ PERMANENT 500000 RET
$$ SUBDIRECTORY-USER-GROUP(ALLOWED) 200 RET
$$ SUBDIRECTORY-USER-GROUP(ALLOWED) 201 RET
$$ SUBDIRECTORY-USER-GROUP(ALLOWED) 202 RET
$$ SUBDIRECTORY-USER-GROUP(ALLOWED) 203 RET
$$ SUBDIRECTORY-USER-GROUP(ALLOWED) 204 RET
$$ RET
$
$ ^ECREATE(NAME) PS:<B> RET
[NEW]
$$ MAX 400      !You can use abbreviations RET
$$ WORKING 500000 RET
$$ PERMANENT 500000 RET
$$ SUB 200 RET
$$ SUB 201 RET
$$ SUB 202 RET
$$ SUB 203 RET
$$ SUB 204 RET
$$ RET
$ !Continue creating directories <C>
$ !through <Z> in exactly the same manner.
```

After all 26 directories are created, you can give the DIRECTORY command to see all the directory files that have been created under <ROOT–DIRECTORY>.

```
$DIR PS:<ROOT-DIRECTORY>

 PS:<ROOT-DIRECTORY>

A.DIRECTORY.1
B.DIRECTORY.1
C.DIRECTORY.1
      .
      .
      .
Z.DIRECTORY.1
```

Next, after all 26 alphabetic directories have been successfully created, the operator again uses the ^ECREATE command and creates all the user directories.

In the example below, the operator is connected to PS: and creates a directory named <A.JONES> for the user who has been assigned the user name A.JONES. This directory is A. Jones' log-in directory on PS:. Each time A. Jones logs into the system, he is connected to directory <A.JONES>. In this example, the operator also assigns the password 2BY4 . The operator gives

the directory the system default of 250 pages for both working and permanent disk quota. Because he is using the default, he does not have to make any entries for these two parameters. The 250 pages given to this directory are taken from the superior directory's quota (directory <A>). The operator also places user A.JONES in user group 202 and places directory <A.JONES> in directory group 202.

```
@ ENABLE(CAPABILITIES) RET
$ ^ECREATE(NAME) PS:<A.JONES>RET
[NEW]
$$ PASSWORD 2BY4RET
$$ USER-GROUP(NUMBER) 202RET
$$ DIRECTORY-GROUP(NUMBER) 202RET
$$ RET
$ DISABLE(CAPABILITIES) RET
@
```

After creating any new directories (either files-only or user), you should update the backup tape that contains the Monitor, TOPS–20 Command Processor, DLUSER, DLUSER data, DUMPER, <SYSTEM>, and <SUBSYS>. (Refer to Chapter 7, System Backup Procedures.)

CONSIDERATIONS:

If users have duplicate first initials and last names, you can use middle initials. For example, if two users have the name C.BAKER, you can assign either one of them a user name in the form CL.BAKER or C.LBAKER. If you use the form CL.BAKER you must create a directory <CL> in addition to directory <C>. If you do not create this additional directory with the user's first and middle initial, you will receive error messages and will not be able to create the directory <CL.BAKER>. If, instead, you assign user Baker the user name C.LBAKER, (the preferred method) you can create the directory <C.LBAKER> as described above using the standard procedure. You do not need to create the additional directory <CL>.

If a user requires special capabilities to perform privileged functions, the operator can include the parameter for the capability in the user's directory accordingly. (Refer to Section 5.9 for a description of the capabilities you can assign to certain users who require them.)

CREATING FILES–ONLY DIRECTORIES:

If a user wants a library area in addition to the logged-in directory, you can create a files-only directory.

5.4.2 Central Control Using Subdirectories (continued)

Files-only directories can also be prefixed by a letter and a period. Because the first name initials of users do not always encompass every letter in the alphabet, you may want to use those infrequently used letters as the prefix to the files-only directories, e.g., <X.FORLIB> or <Z.TESTS>. This method allows you to distribute your disk storage resources equally. The example below shows how to create the directory <X.FORLIB> on structure PS:. The operator assigns the password SQUASH, makes the directory files-only, takes the 250-page default for working and permanent disk storage quotas, and places the directory in directory group number 202. (Users who are in user group 202 can now access this directory as group members.)

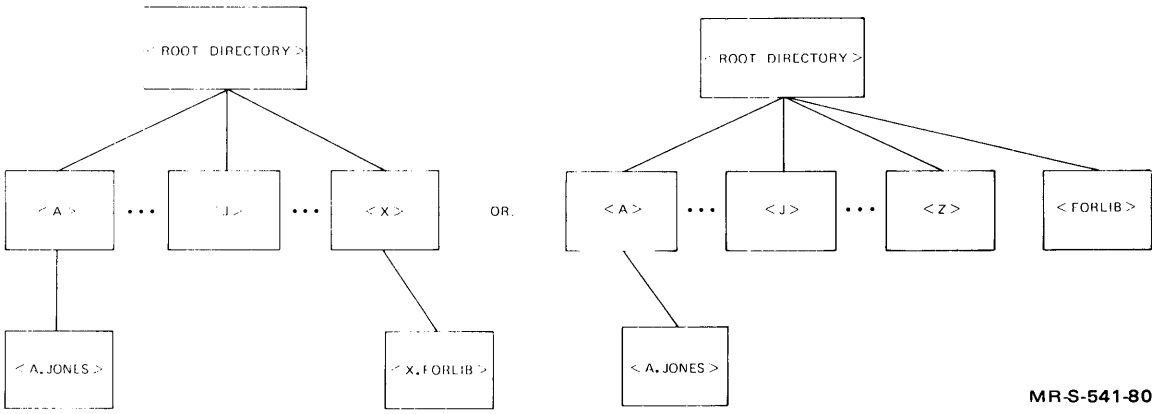
```
@ENABLE (CAPABILITIES) (RET)
$ CREATE (NAME) PS:<X.FORLIB>(RET)
[NEW]
$$ PASSWORD SQUASH(RET)
$$ FILES-ONLY(RET)
$$ DIRECTORY-GROUP (NUMBER) 202(RET)
$$ (RET)
$ DISABLE (CAPABILITIES) (RET)
@
```

Follow the procedures in the *TOPS–20 Operator's Guide* for creating directories on structures other than PS:.

CONSIDERATIONS:

The CONSIDERATIONS described in the previous central control description (Section 5.4.1) for files-only directories also apply to this description.

If the number of files-only directories you want to create is small, you can create them on the same level as the alphabetic directories. That is, <X.FORLIB> can be created as <FORLIB>. Your directory scheme may look like:



5.4.2 Central Control Using Subdirectories (continued)

RESTRICTIONS:

The number of directories you can create per structure cannot exceed approximately 4,000(12,000).

The number of subdirectories under a single-letter directory, e.g., <A>, cannot exceed approximately 365(5,000).

If you reach the maximum number of directories allowed per structure, the system prints the message:

```
MAXIMUM DIRECTORY NUMBER EXCEEDED; INDEX TABLE NEEDS EXPANDING
```

If you reach the maximum number of subdirectories that a single letter directory can point to, the system prints the message:

```
SUPERIOR DIRECTORY FULL
```

5.4.3 Project Control

DETERMINING FACTORS:

- The complexity and perhaps geography of your organization warrants separating small or large groups of users into projects. The responsibility for creating and maintaining the directories within a project can be given to an administrator. This is especially helpful if a large number of users have directories on the system. This method frees the operator from spending an excessive amount of time creating directories and changing directory parameters.
- Even though you delegate the task of creating and managing groups of directories to project administrators, you still maintain ultimate control of the overall system and its resources. This means that you still determine and allocate the disk space that each project uses. The administrator distributes the disk space you allocate to directories within the project. Also, administrators can create and maintain directories for their projects without having WHEEL or OPERATOR capabilities by using the TOPS–20 BUILD command. Therefore, you do not weaken the security of your system. Unless you give WHEEL or OPERATOR capabilities to an administrator, he cannot assign those capabilities to other users.
- In addition to allowing administrators to create directories for a project, you can allow other users of the system to create subdirectories. These users can separate and store files in a subdirectory. According to the protection they place on their subdirectories, they can share their files with other users without losing the security of their superior directory. The users are responsible for maintaining the directories they create.

5.4.3 **Project Control** (Continued)

- Up to 4,000(12,000) directories (including subdirectories) can be created per structure.

FORMAT:

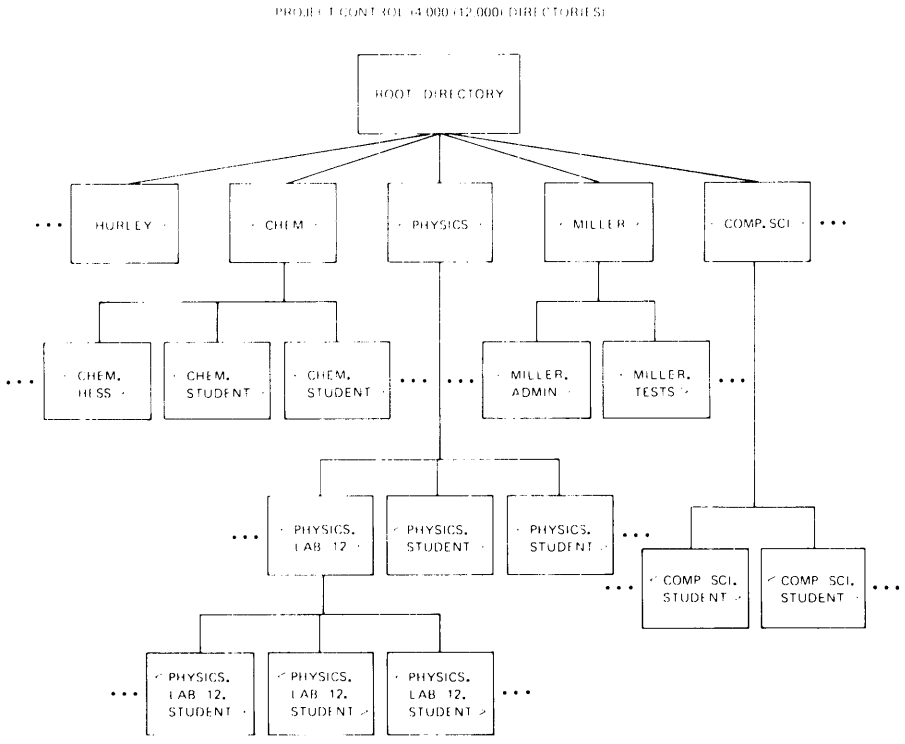
<ROOT–DIRECTORY> can point to approximately 365(5,000) directories per structure.

Each directory under <ROOT–DIRECTORY> can point to approximately 365(5,000) subdirectories.

Each subdirectory can also point to approximately 365 or 5,000 subdirectories under it, depending on the type of system you have.

The number of subdirectory levels is determined by a maximum length of 39 alphanumeric characters, because each subdirectory name contains the name or names of any superior directories above it. Using the diagram below, the user who owns directory <PHYSICS> under <ROOT–DIRECTORY> creates the subdirectory <PHYSICS.LAB–12>. The new subdirectory name (LAB–12) has its superior directory's name (PHYSICS) as its prefix. The period separates the different levels of the directory name and is counted as one of the characters in the directory name.

DIAGRAM:



5.4.3 Project Control (continued)

ASSIGNING USER NAMES:

The names that you assign to users should be as close as possible to the user's last name. In addition, the project names that you assign and that will be used for project directory names should be closely related to the project, e.g., PHYS might be used for Physics and PHYED for Physical Education.

When you give a SYSTAT command, the user surnames and obvious project names make it easier to identify who is using the system, and under which project.

CREATING PROJECT AND USER DIRECTORIES:

The user and project directories that <ROOT–DIRECTORY> points to (first-level directories) are created by you or the operator using the ^ECREATE command.

The procedures you should use and the parameters that you must include in these directories are described below.

Create all project directories as log-in (user) directories. You would not create a project directory as files-only because files-only directories cannot have log-in directories created under them. However, log-in project (or user) directories can have both log-in and files-only subdirectories.

Assign a disk storage quota to each project directory. This quota must be large enough to accommodate both the files that are contained in the directory and the directories that are created under it. Each time a directory is created under a project directory, that directory's disk quota is taken from the project directory's disk quota. The total disk quota for directories created under a project directory cannot exceed the quota originally given to the project directory.

In the example below, the operator begins to create the project directory <CHEM>. He creates the directory as a log-in directory and assigns the password H20. This procedure allows an administrator to log into the directory, giving its password, and create the required subdirectories. He may also want to store his files in this directory. The operator gives the directory a 10,000-page working and a 10,000-page permanent disk storage quota.

```
@ENABLE (CAPABILITIES)RET
$ ^ECREATE (NAME) PS:<CHEM>RET
[NEW]
$$ PASSWORD H20RET
$$ WORKING 10000RET
$$ PERMANENT 10000RET
$$
```

5.4.3 Project Control (continued)

Next, the operator enters the parameter that allows the owner of the project directory to create subdirectories. This parameter, called MAXIMUM–SUBDIRECTORIES (ALLOWED), specifies how many directories can be created under the directory. Unless you enter this parameter (the default is 0), the owner of the directory cannot create subdirectories. For example, all users of the system can type the BUILD command to the TOPS–20 Command Processor; but, only those users who have the MAXIMUM SUBDIRECTORIES (ALLOWED) parameter in their directory with a number greater than zero can actually use the BUILD command to create subdirectories.

The following entry in the sample project directory <CHEM> allows the administrator to create 100 subdirectories.

```
$$MAXIMUM-SUBDIRECTORIES (ALLOWED) 100<RET>
```

The administrator who is responsible for this sample project might create 60 directories under the project directory and give each subdirectory approximately 50 pages of working and permanent disk quota. He keeps enough pages in the project directory to allow that directory's files to grow and to create additional subdirectories. (Refer to the *TOPS–20 Commands Reference Manual* for the description of the BUILD command, including distributing working and permanent storage quotas and maximum subdirectory quotas.)

Also, some of the MAXIMUM–SUBDIRECTORIES (ALLOWED) quota given to the project directory can be given to a subdirectory so that directories under it can be created. The quota for the project directory is decremented by the amount of quota given to the subdirectory.

For example, directory <CHEM> is given a subdirectory quota of 100. The administrator creates the directory <CHEM.STUDENT> under <CHEM> and gives the directory a subdirectory quota of 10. The number of subdirectories that can now be created under <CHEM> is 89. If the administrator creates another subdirectory under <CHEM> called <CHEM.STUDENT2> and gives that directory a subdirectory quota of 6, the number of subdirectories that can now be created under <CHEM> is 82.

5.4.3 Project Control (continued)

If the administrator gives an INFORMATION (ABOUT) DIRECTORY <CHEM> command, the output line for maximum subdirectory quota is:

```
MAXIMUM NUMBER OF SUBDIRECTORIES ALLOWED 84
```

The two directories <CHEM.STUDENT> and <CHEM.STUDENT2> that were created under <CHEM> account for the two subdirectories not shown in the subtraction.

Next, the operator enters the parameter that allows the administrator for this project to place users in groups. The administrator can use the group facility as described in Section 5.8 to set up library directories and allow file sharing among members of the project.

The SUBDIRECTORY–USER–GROUP parameter accepts a number between 1 and 262143 as its argument. You can list a range of numbers that the administrator can use to establish groups within the project; however, you must enter each number separately. Be careful to assign a range of numbers that are unique to that project. For example, project directory <CHEM> may be given the range:

```
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 2600RET
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 2601RET
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 2602RET
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 2603RET
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 2604RET
```

Project directory <PHYSICS> may be given the following range of numbers different from project CHEM.

```
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 3001RET
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 3002RET
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 3003RET
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 3004RET
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 3005RET
```

If you assign the same range of numbers to different projects you can cause a security break among projects. For example, a user in group 2602 in project CHEM should not be able to access, as a group member, the directories and files in project PHYSICS.

The range of numbers placed in a project (or user) directory's parameter list does not imply that the directory or any of its subdirectories have access to those groups. It means only that the administrator (or owner of the directory) can use those group numbers to establish group relationships among that directory and its subdirectories.

5.4.3 Project Control (continued)

The following example shows the completed parameter list for the sample project directory <CHEM>:

```
@ENABLE (CAPABILITIES)(RET)
$^ECREATE (NAME) PS:<CHEM>(RET)
[NEW]
$$PASSWORD H2O(RET)
$$WORKING 10000(RET)
$$PERMANENT 10000(RET)
$$MAXIMUM SUBDIRECTORIES (ALLOWED) 100(RET)
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 2600(RET)
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 2601(RET)
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 2602(RET)
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 2603(RET)
$$SUBDIRECTORY-USER-GROUP (ALLOWED) 2604(RET)
$$(RET)
$DISABLE (CAPABILITIES)(RET)
@
```

Refer to the *TOPS–20 Operator's Guide* for a complete description of the ^ECREATE command that the operator uses to create new directories, and the ULIST program that prints information about all the directories on the system.

After creating a new directory (either user or files-only), remember to update the backup tape that contains the monitor, TOPS–20 Command Processor, DLUSER, DLUSER data, DUMPER, <SYSTEM>, and <SUBSYS>. (Refer to Chapter 7, System Backup Procedures.)

CONSIDERATIONS:

If two projects or users have mistakenly been assigned the same name and you try to create the second directory with this duplication, the system prints [OLD] instead of [NEW]. Give the ABORT subcommand, assign the user or project a slightly different name, and reissue the ^ECREATE command with the new directory name.

A subdirectory is just like any other directory. It can be logged into (if it is not specified as files only), it can be a member of user and directory groups, and it obeys the usual protection mechanisms. Therefore, there are no implied rights between a directory and its subdirectories or between two subdirectories of the same directory. Files have three protection fields; owner, group, and world, and each directory has the same three protection fields. Refer to Section 5.7 for a description of directory and file protections.

The only additional rights that the owner of a directory has over that directory's subdirectory is the power to change its parameters (e.g., directory protection, password, or group memberships), or to use the KILL subcommand and delete the subdirectory.

If you or another user choose to delete a directory, you must first delete any subdirectories under the directory. You cannot delete a directory or sub-

directory that has existing subdirectories. This protection insures that someone (possibly an administrator of a project) does not accidentally delete a directory that points to a large portion of the data base. The operator or administrator must connect to the directory immediately above the lowest level subdirectory to begin deleting any directories. For example, using the diagram in the FORMAT description, if the owner of the directory <PHYSICS> wants to delete the directory <PHYSICS.LAB–12>, he must first connect to directory <PHYSICS.LAB–12> and delete the three <PHYSICS.LAB–12.STUDENT> directories. Then, he connects to directory <PHYSICS> and gives the KILL subcommand to delete directory <PHYSICS.LAB–12>. Note that the operator is the only person who can delete the directory <PHYSICS>.

If you or the administrator choose to grant special capabilities to a user, you can include the parameter for the capability in the user's directory. (Refer to Section 5.9 for a description of the capabilities you can assign to certain users.) You should instruct the administrator to inform you when special capabilities are given to a system user. You are protected against users randomly giving other users special capabilities, because the operator or the administrator who assigns special capabilities to a user must have (as a user) those same capabilities. A person with WHEEL or OPERATOR capabilities can assign any capability to another user. Also, the user or operator who is assigning the capabilities must have those capabilities enabled at the time the privileged parameter is entered into the user's directory.

CREATING FILES–ONLY DIRECTORIES:

Administrators or users can have library areas in addition to their logged-in directories. They can use the BUILD command and create files-only directories under their logged-in directories, providing you have given them the capability to do so by adding the MAXIMUM SUBDIRECTORIES (ALLOWED) parameter to the directory that will contain the subdirectories.

CONSIDERATIONS:

Refer to the CONSIDERATIONS under the first description of Central Control, Section 5.4.1. These considerations also apply to Project Administrative Control.

RESTRICTIONS:

- You cannot exceed approximately 4,000(12,000) directories per structure.

  The number of directories that a superior directory points to cannot exceed approximately 365(5,000).

5.4.3 Project Control (continued)

If you reach the maximum number of directories that you can create on a structure, the system prints the message:

```
MAXIMUM DIRECTORY NUMBER EXCEEDED; INDEX TABLE NEEDS EXPANDING
```

If either you or an administrator reach the maximum number of directories that can be created under a superior directory, the system prints the message:

```
SUPERIOR DIRECTORY FULL
```

- Files-only directories cannot have log-in subdirectories. If you want to allow a user to create user (log-in) subdirectories under his directory, you must make his directory a log-in directory.

5.4.4 Combined Central and Project Control

DETERMINING FACTORS:

- Only a portion of your organization warrants being separated into projects. The directories for the majority of the user community are created and maintained at the central management level. But, where project administration is appropriate, the task of creating and managing directories within a project is given to administrators.

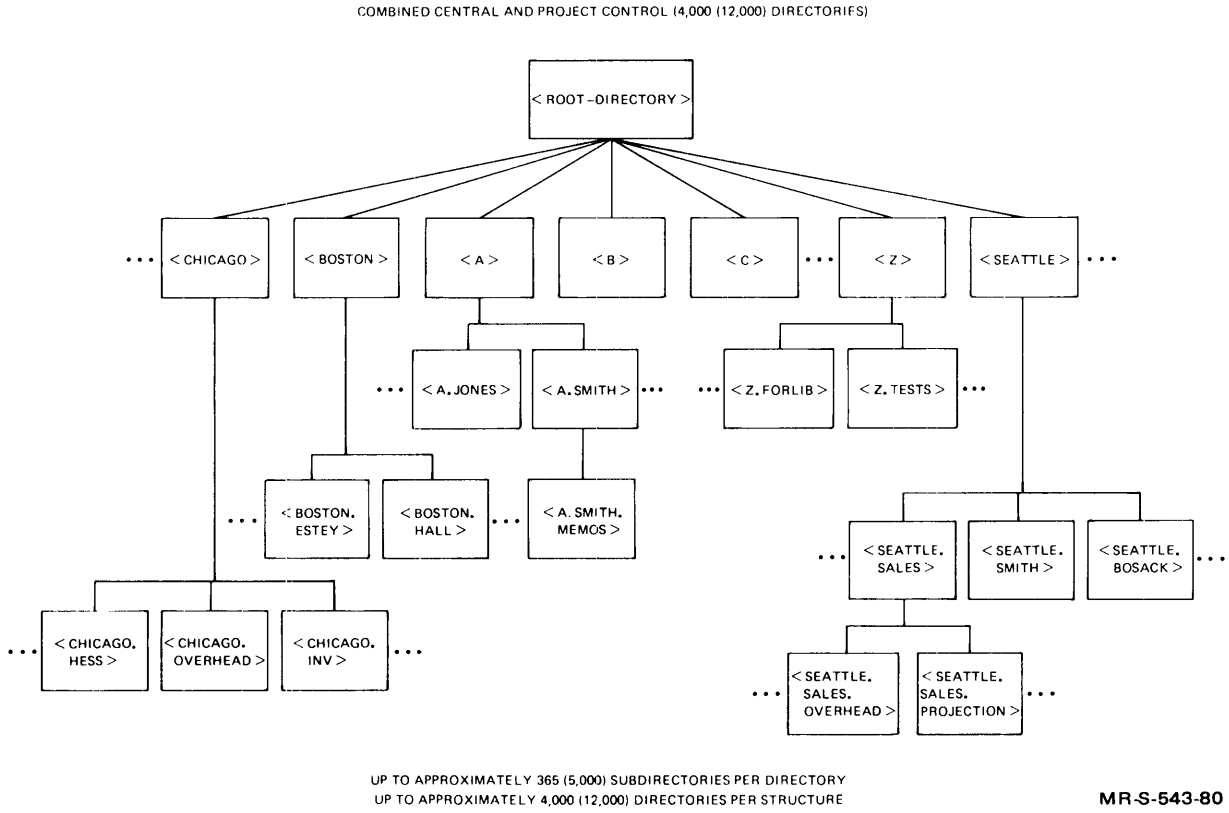
- For example, if your company has groups of users with terminals in several distant locations, you may want to have the administrator at the remote location create and maintain all the directories for that site. You can create a project directory for the remote location, perhaps using the name of the site as the project directory name (e.g., <CHICAGO> or <CHIC>, <SEATTLE>, etc.). The remaining user directories at the central location are created by the system operator.

FORMAT:

<ROOT–DIRECTORY> points to all the project directories and 26 alphabetically named directories, <A> through <Z>. The project directories point to the user and files-only directories that an administrator creates for a given project. The directories <A> through <Z> point to user and files-only directories created and maintained by the operator. These directories can also be allowed to have subdirectories.

5.4.4 Combined Central and Project Control (Continued)

DIAGRAM:



ASSIGNING USER NAMES:

If you create any user directories that are pointed to by <ROOT-DIRECTORY>, assign project names and user names in the same manner as described under Project Control, Section 5.4.3. Again, assign the 26 directories that will point to the majority of the user directories, the names <A>, <B>, <C>....<Z>. The user names that will be the directory names under the alphabetic directories should again be the user's surname prefixed by a first initial and a period. (Refer to Section 5.4.2, Central Control Using Subdirectories.)

CREATING USER AND FILES–ONLY DIRECTORIES:

Create the directories <A> through <Z>, following the instructions in Section 5.4.2. Create the user and files-only directories, following the instructions in the same description.

Create the project directories according to the instructions in Section 5.4.3, and distribute the description of the BUILD command (Appendix A) to the administrators who are responsible for creating the user directories within their project.

5.4.4 Combined Central and Project Control (continued)

CONSIDERATIONS:

All the considerations that apply to both Central and Project Control also apply to combining the two types of control.

You may want to allow users whose directories are created by the operator to create several directories under their logged-in directories. The diagram under FORMAT illustrates this facility. User A.SMITH has created the subdirectory <A.SMITH.MEMOS> to store files that he wants to keep separate from his programming files. This user uses the BUILD command to create the number of subdirectories that he is allowed to create and divides the quota for his logged-in directory among the directories he creates.

In general, users can store files in these directories or, if they set the appropriate protection, share the files in these directories with other users.

RESTRICTIONS:

Combined Central and Project Control allows up to approximately 4,000(12,000) directories per structure.

The number of directories that a superior directory can point to cannot exceed approximately 365(5,000).

If you reach the maximum number of directories per structure, the system prints the message:

```
MAXIMUM DIRECTORY NUMBER EXCEEDED; INDEX TABLE NEEDS EXPANDING
```

If you reach the maximum number of directories that a superior directory can point to, the system prints the message:

```
SUPERIOR DIRECTORY FULL
```

5.5 Allocating Disk Storage Quotas

In Chapter 4 you determined the amount of disk space that is available on PS: after installation. Once you know the available disk space, you can decide how to allocate it among the directories you create. Each directory is given a number of pages for both working-storage and permanent-storage allocations. Working storage refers to the disk space that a user can have during the time he is logged-in. Permanent storage refers to the total disk space that a user can have to store files after he has logged-out.

The number of pages that you should assign to directories depends on whether you (or the operator) are creating all the directories on the system (central control) or you are delegating the task of creating and maintaining directories to project administrators (project control). When using central control, you may divide the disk space equally among directories giving

regard to special requirements of certain users. In the case where the operator creates project directories, you should allocate a disk quota large enough to accommodate the expected size of each project. Remember that project directories must distribute their disk space to the directories created under them.

Several important points about working and permanent allocations are discussed below.

Assign a large (2000–3000 page) working-storage allocation to users who perform considerable sorting because the temporary files required for this operation can occupy substantial disk space.

As the number of users on the system increases and your disk space on PS: becomes low, you can decrease the working-storage and permanent allocations on PS: to add new log-in directories. If you have additional disk drives not used by PS:, you can accommodate the directories with many or large files by creating other structures and directories. Users will log into their directories on PS:, request the operator to mount the proper structure using the MOUNT command, and access their additional directories with the ACCESS and/or CONNECT command.

5.6 Enforcing Disk Storage Quotas

Working-storage allocations are strictly enforced. Users cannot exceed their working-storage allocations unless they enable WHEEL or OPERATOR capabilities. (Refer to Section 5.9 for a description of the special capabilities that can be given to users who require them). If users request additional space, you can increase their allocations as required.

If a user exceeds his working-storage allocation and attempts to create or change a file, the system prints the following error message.

```
?QUOTA EXCEEDED OR DISK FULL
```

The user must decrease his disk usage to less than the working-storage allocation for the directory (in which the file is being changed or created) before he can create or change any more files.

The system informs a user if he is over this permanent storage allocation when he logs off the system or connects to a different directory. The system prints the following message after the CONNECT or LOGOUT commands.

```
<directory>OVER PERMANENT STORAGE ALLOCATION BY nn PAGES
```

This message reminds users that although they may not be over their working-storage allocation, they have exceeded their expected total disk usage. Users should delete any files that are unnecessary for their job. Also, because permanent quotas are not enforced, it is wise to instruct the operator or administrator to police each directory's disk usage. The operator should run the CHKPNT program daily to keep a record of each directory's

disk usage. The *TOPS–20 Operator's Guide* contains the description of running the CHKPNT program. If you are using the file migration facility (refer to Chapter 8), you may want to run the REAPER program with the TRIM command to force users to stay below their permanent quotas.

Every time the available disk space on a structure is less than 500 pages, the system prints the following warning message.

```
[CAUTION-DISK SPACE LOW ON structure name] [DELETED FILES WILL BE
EXPUNGED IN 30 SECONDS]
```

After 30 seconds, the system prints the following message and starts expunging any deleted files in all directories on the structure mentioned in the warning message.

```
[EXPUNGING DELETED FILES]
```

The system prints this message when the expunging is complete.

```
[SYSTEM EXPUNGE COMPLETED]
```

If anyone tries to create or change a file when there is no more disk space available, the system prints an error message similar to the one below.

```
?FILE OR SWAPPING SPACE EXCEEDED
```

Again, the operator or administrator should check to see how many users are over permanent allocations. Also, if you are using the file migration facility, you may want to migrate files on the system more frequently if you are constantly running low on systemwide disk space. (Refer to Chapter 8 for a description of file migration.)

5.7 Protecting Directories and Files

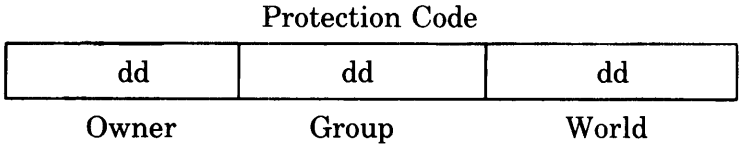
Every directory and file has a protection number associated with it. The system uses a default protection number for each directory and file when the directory or file is created.

Whenever a user accesses a file, the system first checks the directory protection. If that protection allows the user the appropriate access to the directory, the system then checks the protection of the individual file.

5.7.1 Directory and File Protection Digits

The directory and file protection numbers have three 2-digit fields. The first field applies to the owner of the directory or file, the second field to members of the same group as this directory, and the third field to all other users (or world).

5.7.1 Protecting Directories and Files (continued)



The default protection for directories and files is 777700. A directory or file protection of 77 in any given field allows full access. For example, the default protection allows the owner and members of his group full access but all other users no access.

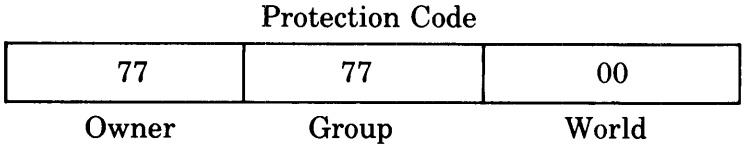


Table 5–1 contains a list of the directory protection digits.

Table 5–1: Directory Protection Digits

Digits	Privilege
04	Permits creating files in the directory.
10	Permits connecting to the directory without giving a password and changing the accounts and protection numbers of the files therein. Thus it gives many of the privileges the directory owner has. (Refer to the *TOPS–20 Monitor Calls* manual.)
40	Permits, subject to the protection on the individual file, listing the names of the files with the DIRECTORY command and reading the file, e.g., via the TYPE, PRINT, or LIST commands.

These protection codes are actually bits in a protection word. To get more than one protection, add the digits (octal) corresponding to the protection you want. Thus, 44 allows listing the files and creating new files. There are unused bits in the protection number; therefore, to provide complete access to files, use 77. Useful digit pairs are:

00 Permits no access
40 Permits the files to be listed and read
77 Permits full (owner) access

A file protection number has the same format as a directory protection number, but the meanings of the digits are different. Table 5–2 contains a list of file protection digits.

Table 5–2: File Protection Digits

Digits	Privilege
02	Permits listing the file with the DIRECTORY command.
04	Permits appending to the file.
10	Permits executing the file.
20	Permits writing and deleting the file.
40	Permits reading the file.

Obtain a protection number by adding the file protection digits of the different protections you need. For example, protection number 775200 allows the owner full privileges; the members of the same group reading, executing, and directory listing privileges; and all other users no privileges. Useful digit pairs are:

00 Permits no access

12 Permits executing and using the DIRECTORY command to list the file only.

This protection is useful when, for example, you purchase a program and agree in your contract not to allow any of your system users to read, write into, or copy the file. Set the protection on an execute-only file to 771212. The TOPS–20 Beware file provides additional considerations for setting up execute-only files.

52 Permits reading, executing, and using the DIRECTORY command to list the file

77 Permits full access

The system checks protection numbers starting with the two rightmost digits. Therefore, users do not restrict members of a group by assigning the file protection 770052, because the group gets at least the execute, read, and directory list access (52) granted to all users.

Also, because the system checks the directory protection before the file protection, files that have been given a low file protection are still secure in a directory with the default directory protection. For example, suppose the user KOHN tries to type the file EDIT.MAC in the directory <HESS>. The protection on the directory <HESS> is 777700 and the protection on the file EDIT.MAC is 777752. User KOHN and directory <HESS> are not in the same group, so the world protection applies. First, the system checks the directory protection, 777700. The last two digits (00) apply and permit no access to the directory. User KOHN is not allowed to type the file, even though the corresponding protection on the file (52) allows the file to be read, executed, and listed with the DIRECTORY command.

5.7.2 Changing Directory and File Protection

Users can change file protection numbers via the SET FILE PROTECTION command or the RENAME command.

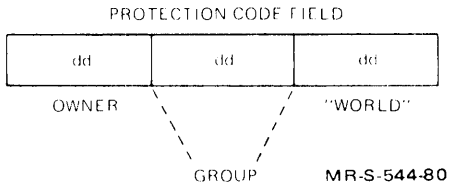
Users can change directory protection numbers via the SET DIRECTORY PROTECTION or BUILD command. You can, however, prevent users from making changes to their directory protection numbers by including the DISABLE DIRECTORY–PARAMETER–SETTING command in the system file called PS:<SYSTEM>n-CONFIG.CMD. If you make this entry in n-CONFIG.CMD, only users with WHEEL or OPERATOR capabilities can change directory parameters (via the ENABLE and SET DIRECTORY PROTECTION commands).

NOTE

Make an entry in n-CONFIG.CMD only if you DO NOT want to allow users to change their directory protections; otherwise, the system assumes that you want to use the system default command of ENABLE DIRECTORY–PARAMETER–SETTING. (Refer to the *TOPS–20 Software Installation Guide* for a description of the parameters that are placed in the n-CONFIG.CMD file.)

5.8 Establishing Groups

You can let users share files by placing users and directories in groups. Members of a group can access directories and files in that group according to the middle digits of the directory and file protection code fields.



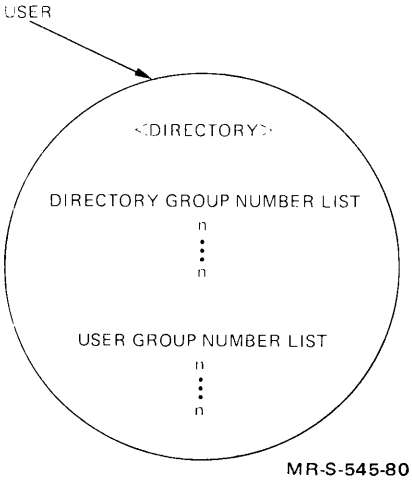
Each group that you establish has two types of members: USERS and <DIRECTORIES>. Each group is identified by a number. This number is included as one of the directory parameters in each directory belonging to the group. Any directory (including subdirectories) or user can belong to as many as 19 groups. You can set up group relationships in the individual directories by using the DIRECTORY–GROUP and USER–GROUP subcommands to the ^ECREATE and BUILD commands. The following example shows that you have placed user Smith in USER–GROUP 268 and DIRECTORY–GROUPS 268 and 418:

```
@ENABLE (CAPABILITIES)RET
$^ECREATE (NAME) PS:<SMITH>RET
$$PASSWORD SOARRET
$$WORKING 500RET
$$PERMANENT 500RET
$$USER-GROUP 268RET
$$DIRECTORY-GROUP 268,418RET
$$
```

The DIRECTORY–GROUP or USER–GROUP parameter that you place in the user's directory determines: 1) if this user can access another directory's files as a group member 2) if the files in this user's directory can be accessed by another user as a group member, or 3) both. The diagrams on the following pages illustrate the difference between being a member of a group as a directory and/or as a user.

When a user accesses a file in a directory that is a member of the same group, the system first checks to see if this user is the owner of the directory. When, in this example, it finds that the user is not the owner, the system then checks to see if the user is in the same group as this directory. In this case, the user and directory are in the same group; that is, the group numbers match. The system now checks the group protection code field of the directory being accessed. If the group protection allows the type of access that the user requested, the system proceeds to check the group protection on the individual file.

If you are setting up groups on different structures, there is no correlation between a group number on one structure and the same group number on another structure. For example, group 202 on PS: does not necessarily have the same user and directory members as group 202 on another structure.

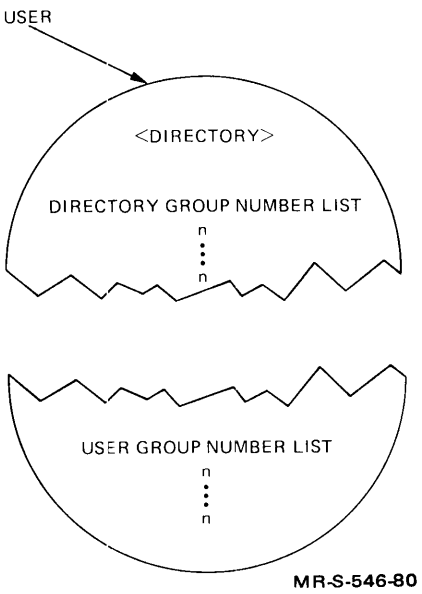


MR-S-545-80

Each directory has two lists of group numbers: Directory Group Numbers and User Group Numbers.

Directory Group Numbers identify the various groups of which this <DIRECTORY> is a member.

User Group Numbers are associated with users and identify the various groups of which each user is a member.



The Directory Group Numbers are important to users who require access to this directory. Those users who have a matching group number in the User Group Number List can access this <DIRECTORY> according to its group protection code.

The User Group Numbers are important to the owner of this directory. This owner can access any directory that has a matching group number in its Directory Group Number List. Note: Because files-only directories are not associated with a user, they do not contain User Group Numbers.

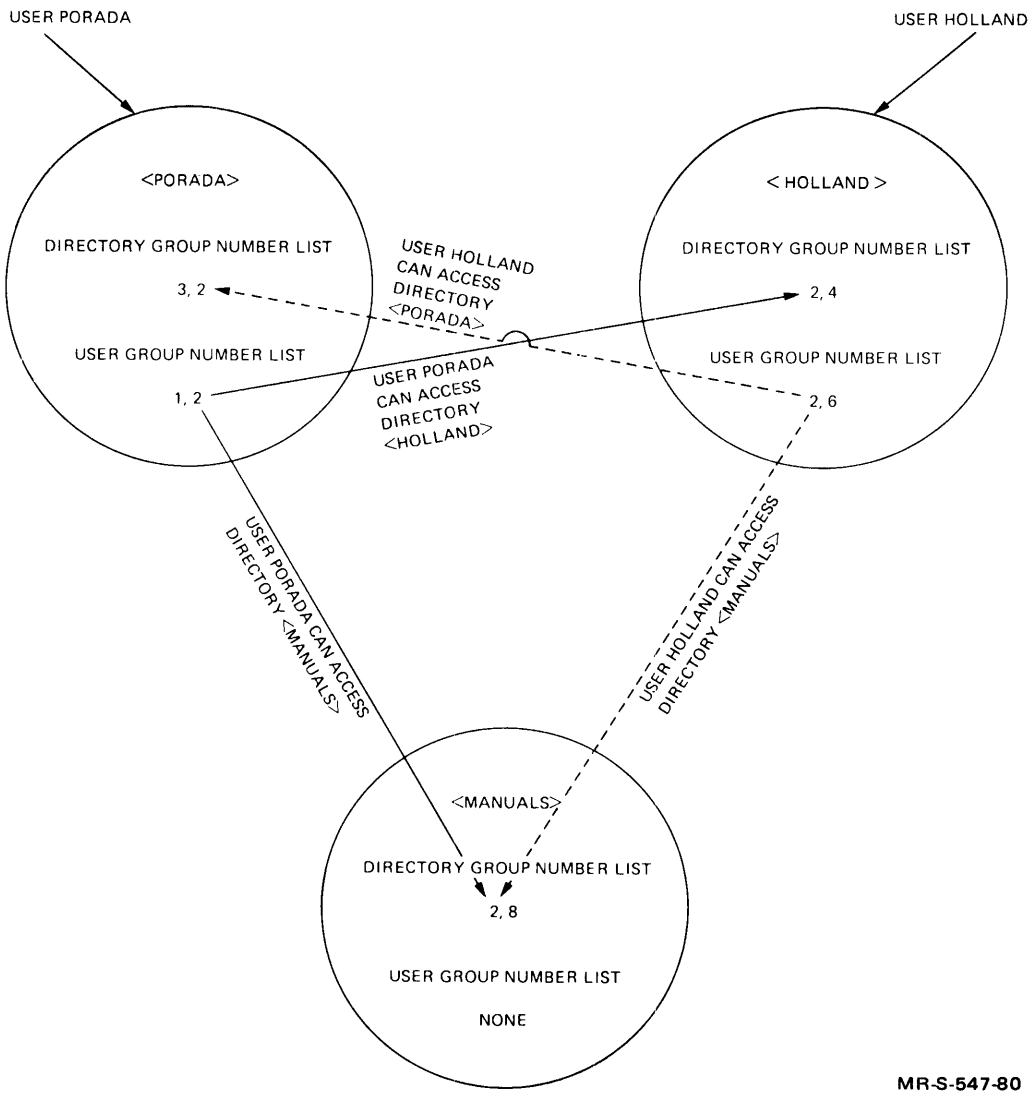
There are three common types of groups:

1. A file-sharing group, whose users can access a set of library directories and each other's logged-in directories.
2. A library group, whose users can access a set of library directories and their own logged-in directories, but not each other's logged-in directories.
3. A teacher-student group, in which the teacher can access the students' directories and the students can access their own logged-in directories, but not their classmates' directories or the teacher's directory.

Figures 5–1 through 5–3 illustrate these three common groups and the association between USER and <DIRECTORY> members of a group.

In a file-sharing group (Figure 5–1), users share all their files according to the group protection field, both in the library directories (here it is <MANUALS>) and in their logged-in directories.

Figure 5–1: File-Sharing Group



In Figure 5–1, the two users, PORADA and HOLLAND, are members of the same group (group 2). The directories <PORADA>, <HOLLAND> and <MANUALS> are also members of group 2. Users PORADA and HOLLAND can access their own directory and files according to the owner protection code fields. PORADA can access directories <HOLLAND> and <MANUALS> according to the group protection code fields, and conversely, HOLLAND can access directories <PORADA> and <MANUALS> according to their group protection code fields. The other numbers shown in the figure indicate that a user or directory can be a member of more than one group.

In a library group (Figure 5–2), USER members can access all the <DIRECTORY> members but not each other's logged-in directories. The library directories are usually files-only directories. This figure illustrates a library group that consists of the files-only directories: <SUBROUTINES>,

<TAPE–TESTS>, <MACROS> and users: ALUSIC, BROPHY and KOHN. This library group illustrates that just because you are a member of a group as a user, your logged-in directory need not belong to the same group.

Figure 5–2: Library Group

In Figure 5–2, users KOHN, ALUSIC and BROPHY are not directory group members of the same group; however, they are all user group members in the same group (group 2). User KOHN can access directory <KOHN> according to the owner protection field and can access directories <SUBROUTINES>, <TAPE–TESTS>, and <MACROS> according to the group protection field. KOHN can access <BROPHY> and <ALUSIC> according to the "world" protection field. Although the arrows have not been drawn from users BROPHY and ALUSIC, their access privileges are the same as KOHN's.

Figure 5–3: Teacher-Student Group

In a teacher-student group (Figure 5–3), the teacher, WILEY, is a member of the group as a USER, while the directories <HURLEY>, <HALL>, <MILLER> and <RUSSELL> are <DIRECTORY> members. The teacher, WILEY, can access the files in the directories <HURLEY>, <HALL>, <MILLER> and <RUSSELL> according to the group protection. The students whose logged-in directories are in this group as <DIRECTORY> members can access the files in <WILEY> according to the protection set for all users, because only their directories are members of the group; they are not members of the group as users.

5.9 Giving Users Special Capabilities

You can give special capabilities to certain users; they are WHEEL, OPERATOR, CONFIDENTIAL, MAINTENANCE, IPCF, ENQ–DEQ, ARPANET–WIZARD*, and ABSOLUTE–ARPANET–SOCKETS*. Each capability that you give to a user is placed in the user's directory parameter list when you create or change the directory. The person who enters the capability in a user's directory must have that capability himself, and have it enabled at the time the capability is entered into the directory parameter list. You should grant these capabilities only to users who absolutely need them. Table 5–3 lists all the available capabilities and a brief description of their function.

Table 5–3: Special Capabilities

Capability	Description
WHEEL	Allows the user to modify any system parameters or data. In particular, the WHEEL capability is needed if the user wants to give the ^EEDDT or ^EQUIT commands.
OPERATOR	Allows the user all capabilities required to control the system. The user cannot, however, give the ^EEDDT or ^EQUIT commands.
CONFIDENTIAL	Allows the user to obtain confidential information about the system or other users.
MAINTENANCE	Allows the user (usually the field service representative) to perform certain maintenance functions, but he cannot give the ^E commands.
IPCF	Allows the user to perform the privileged functions of IPCF. (Refer to the *TOPS–20 Monitor Calls Reference Manual.*)
ENQ–DEQ	Allows the user to perform global ENQUEUE/DEQUEUE functions. (Refer to the *TOPS–20 Monitor Calls Reference Manual.*)
ARPANET–WIZARD	Allows the user to perform certain ARPANET privileged functions. (Refer to the *TOPS–20AN Monitor Calls User's Guide.*)
ABSOLUTE–ARPANET–SOCKETS	Allows the user to place absolute socket numbers in his programs.
ARPANET–ACCESS	Allows the directory owner to establish ARPANET network connections.
DECNET–ACCESS	Allows the directory owner to establish DECNET network connections.

* Not applicable to the DECSYSTEM–2020.

With the exception of WHEEL and OPERATOR, these capabilities are not listed in a format where having one capability means you also have the capabilities listed below it. The user who has WHEEL capabilities can perform the OPERATOR, CONFIDENTIAL, MAINTENANCE, IPCF, and ENQ–DEQ functions. The user who has OPERATOR capabilities can also perform these privileged functions with the exception of the ^EEDDT and ^EQUIT commands. But the user who has CONFIDENTIAL capabilities can only perform functions that are allowed by this capability; that is, he cannot perform MAINTENANCE, IPCF, ENQ–DEQ, ARPANET WIZARD, and ABSOLUTE ARPANET SOCKETS functions, unless he has been given the individual capability. The same principle is true for the remaining capabilities.

Also, you are giving capabilities to a user, not the user's directory. Therefore, if user HALL has WHEEL capabilities, other users who connect to the directory <HALL> do not obtain WHEEL capabilities. However, if they log in as user HALL, they will obtain HALL's capabilities.

5.10 Printing Directory Information

The ULIST program prints information about directories on the system and is described in the *TOPS–20 Operator's Guide*. In addition to listing information about each directory, the ULIST program can list information about groups, capabilities, and related information.

The LIST subcommand of the ^ECREATE command prints information about the directory or user name you are currently creating, and the ^EPRINT command also prints the information on an individual basis.

Chapter 6
Creating Accounts

The TOPS–20 accounting facility allows you to assign and charge computer usage to valid accounts. It provides you with a means for 1) adding security to your system, 2) determining charges for computer usage and billing users by account, and 3) associating classes with accounts for use by the class scheduler. You can use account validation for one or all of these reasons.

One or more accounts can be assigned to a user for specific tasks and validated each time they are used. All accounting data, including records of CPU time, structures used, and peripherals used under a valid account, are stored in a usage file and can be used later for reports and billing.

This chapter describes how to set up the system to use accounts and establish an accounting data base. The *TOPS–20 USAGE File Specification* describes how to create accounting reports from the Usage file and establish billing procedures. The following sections include:

- How to set up your system to use accounts
- How to select an accounting scheme
- How to use your accounting scheme and create the necessary base account and subaccount files
- How to run the account generator program (ACTGEN), which takes these account data files and creates the accounting data base
- What the operator can do if the accounting data base does not work properly
- How to initialize your system to start validating accounts

6.1 Setting Up The System To Use Accounts

6.1.1 Enabling or Disabling Account Validation

During software installation, you can specify whether you wish to create the account data base and validate accounts. You can make an entry in the n-CONFIG.CMD file that specifies either DISABLE ACCOUNT–VALIDATION or ENABLE ACCOUNT–VALIDATION. If you do not make an entry in the n-CONFIG.CMD file for accounting, the system assumes ENABLE ACCOUNT–VALIDATION.

If you enter DISABLE ACCOUNT–VALIDATION, meaning you do not wish to use the account validation facility, the system checks each account only for length. The purpose of the check is to ensure that the maximum number of alphanumeric characters has not been exceeded in each account. No other checking is performed. If a user attempts to use or create an account greater than 39 characters, the system simply truncates the entry to the 39-character maximum.

If you have instructed the system to ENABLE ACCOUNT–VALIDATION but have not yet created an account validation data base, you receive a warning on the console terminal (CTY) when the system starts operation. The message is:

```
<SYSTEM>ACCOUNTS-TABLE.BIN NOT FOUND - ACCOUNT VALIDATION IS DISABLED
```

The system continues its normal operation; however, no accounts are validated (except for length checking) until you create the necessary account data files and run the account generator program (ACTGEN) to create your account data base.

You should not receive the above warning message if you have created your account data base prior to bringing the system up for operation. Users can log into the system using their valid accounts.

6.1.2 Setting up Account Validation with Existing Files

If you are using account validation on a system that already has files, the accounts for these existing files should be updated before account validation is enabled in the n-CONFIG.CMD file. Notify the users who created these files to change the existing account on every file to their new account(s). This procedure ensures accurate billing immediately after the system is brought up and that daily DUMPER tapes contain files with valid accounts. This means that if you must restore files from a backup tape, the correct account for each file is properly restored; therefore, the disk file storage continues to be accurately charged. (Refer to the *TOPS–20 Operator's Guide* for the procedure to follow if all files do not get updated.)

6.1.3 Setting up the System for Accounting Shift Changes

The accounting facility also allows you to change your billing rates for system usage at selected times during the day. This action is called an accounting shift change. Accounting shift changes are selected by day-of-week and time-of-day.

You must enter the appropriate commands in the n-CONFIG.CMD file to initiate accounting shift changes. The SETSPD program reads these commands each time the system is reloaded. The format of the command placed in the n-CONFIG.CMD file is:

CHANGE time days-of-week

You can use any format of the time and day, that is, 1500, 15:00, 3:00pm, MONDAY, MON. Or, you can use the keywords ALL, WEEKENDS, and WEEKDAYS. The default for days-of-week is ALL. The following is a typical set of commands that may appear in the n-CONFIG.CMD file:

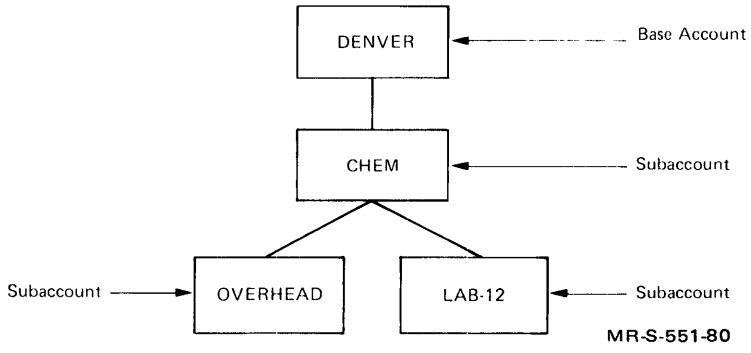
```
CHANGE 9:00 WEEKDAYS
CHANGE 10:00 WEEKENDS,MONDAY
CHANGE 12:00 TUESDAY,THURSDAY,SAT
CHANGE 17:00
```

The CHANGE (ACCOUNTING SHIFT NOW) command to the CHKPNT program provides you with a means of changing shifts during system operation. This command causes an accounting session to end and a new accounting session to begin for all active jobs on the system. Refer to the *TOPS–20 Operator's Guide* for a description of all the commands that can be given to the CHKPNT program.

6.2 Selecting An Accounting Scheme

The first thing you must do before you create account data files is set up an accounting scheme. This procedure includes deciding which accounts you wish to create, their expiration dates if you are going to open and close accounts, the names of the users who can use (or charge to) those accounts and, if you are using the class scheduler, the scheduling class associated with each account.

The TOPS–20 account validation facility allows several levels of project administration in a group of accounts having the same base account. For example:



The accounts you would create using the above example are:

DENVER
DENVER.CHEM
DENVER.CHEM.OVERHEAD
and
DENVER.CHEM.LAB–12

In this example, users at Denver University taking a particular lab course in chemistry (e.g., 12) would log in and charge to their assigned account, DENVER.CHEM.LAB–12.

All accounts that you assign to users can have a maximum length of 39 alphanumeric characters. The system allows you to use a hyphen (-) within the accounts you create (e.g., LAB–12), but no other punctuation (including spaces) can be used. Note that the system uses the period (.) as a delimiter to separate each part of multi-level accounts and the period is counted as one of the 39 characters. Therefore, DENVER.CHEM.OVERHEAD is a user account with 20 characters.

The type of accounting scheme you use depends on the form of project administration you have at your installation. Multi-level accounts are usually created through a form of project administration similar to that used when allowing certain users (perhaps heads of departments) to create subdirectories. (Refer to Chapter 5, Creating Directories.) Remember that subdirectories are just like any other directory. Therefore, users must have accounts to log into their directory.

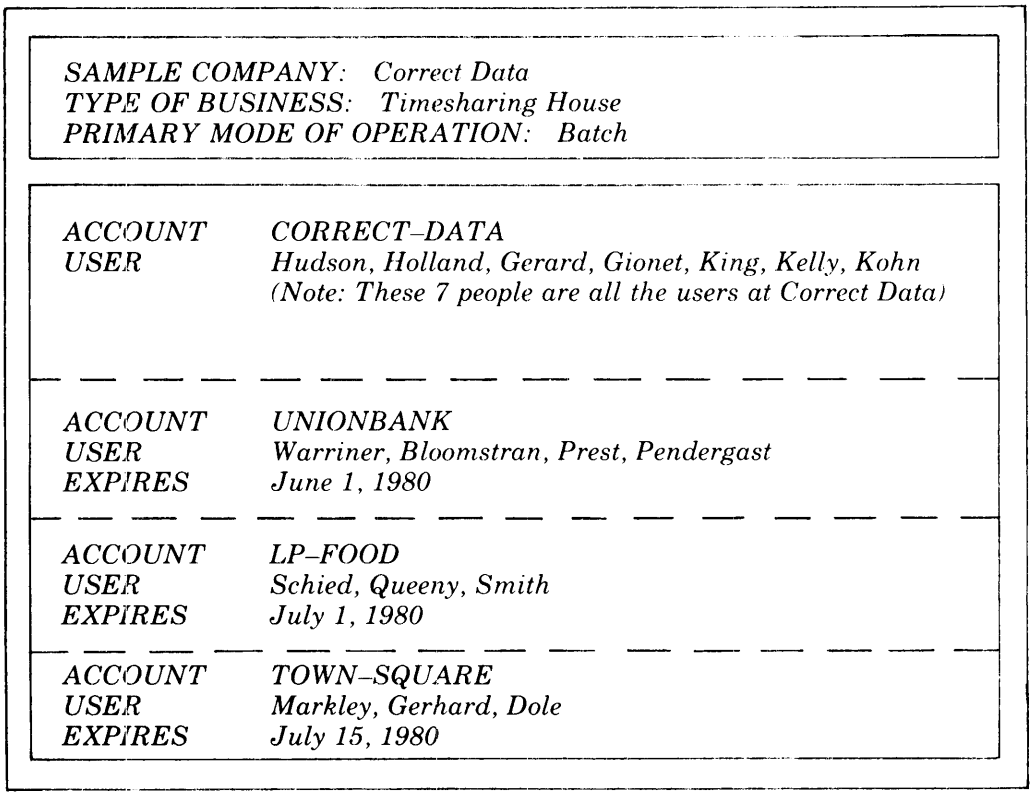
Generally, all files that contain data pertaining to base accounts are created by you or the operator, and all files that contain subaccount data are created by one, or perhaps more project administrators. A project administrator, for example, might be the head of the Chemistry Department. (This could be the same person who handles the subdirectory creation for a group or groups of users.) Allocating the subaccount file creation to a project administrator allows you to collect or budget for one base account (e.g., DENVER.CHEM) and not be directly concerned with the subaccounts. In the example, the head of the Chemistry Department is responsible for creating the subaccount files under DENVER.CHEM., that is, LAB–12 and OVERHEAD. Section 6.3 describes how to create these account data files using a sample accounting scheme.

Figures 6–1 and 6–2 illustrate several ways that you can set up your accounting scheme. Figure 6–1 is a simple scheme that a small organization might use. It also allows you, as system manager, to have complete control over all accounts because you are aware of every account assigned.

Figure 6–1 shows that the manager at Correct Data Company has decided to set up one base account for Correct Data and one base account for each customer using his system. He used the customer name for the accounts. All the people who use the system at Correct Data can charge their computer usage to the Correct Data account, CORRECT–DATA. Unionbank, L & P Food and Town Square Magazine submitted the names of those

people who will be using the system from their respective sites. These are the only people who will be able to log in from their site and charge to their assigned account. The manager at Correct Data also planned expiration dates for each customer account.

Figure 6–1: Accounting Scheme 1

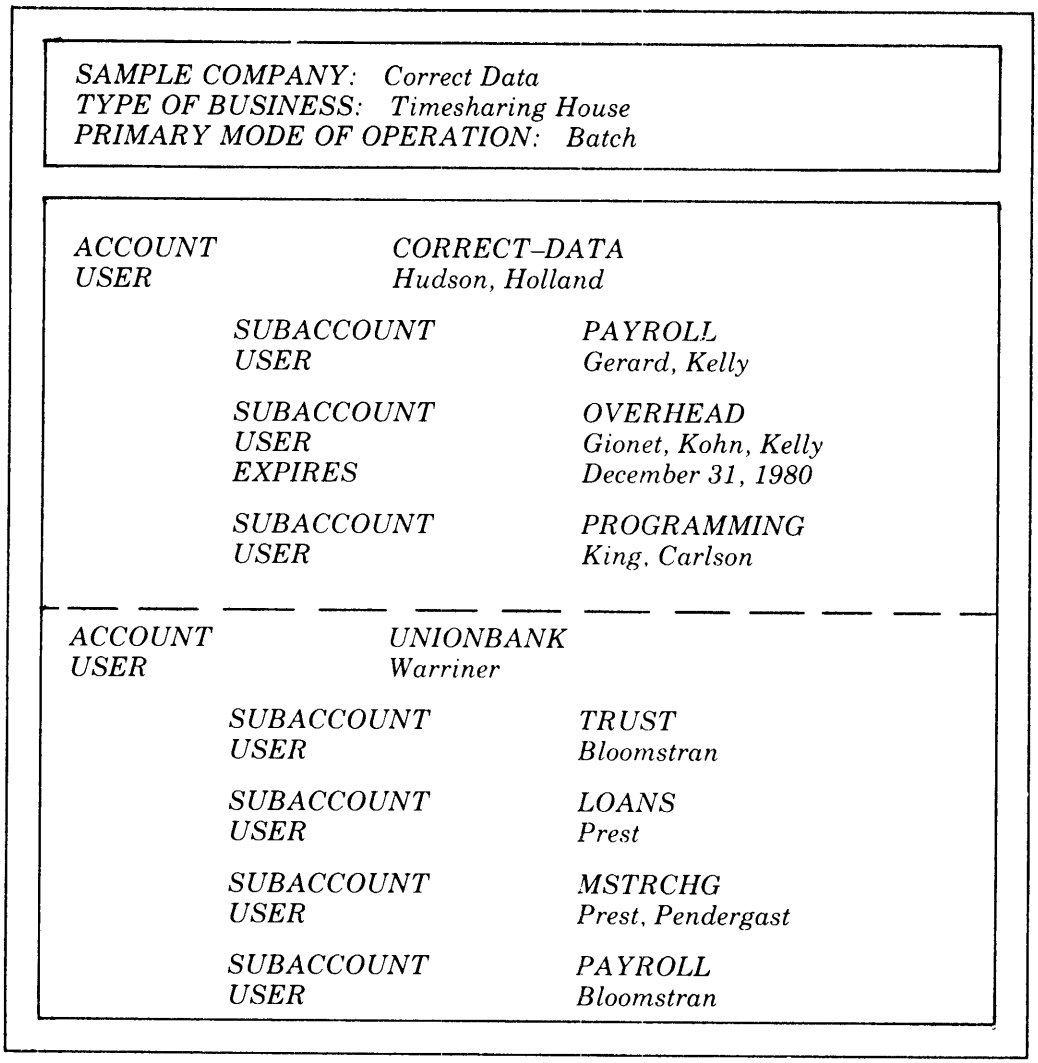


MR-S-552-80

Using the same sample company, Figure 6–2 shows how a simple accounting scheme of this type can be expanded into a form of project administration. Here, Correct Data and one of its customers, Unionbank, broke down the base accounts into subaccounts.

Because Correct Data bills Unionbank for all its computer usage as one account, the manager at Correct Data is not concerned with how Unionbank subdivides its account, and is probably not aware of the subaccounts at Unionbank. The manager at Unionbank, however, is concerned with the computer usage costs incurred by each department within his company. He supplies Correct Data with the name of the file that contains his subaccount information.

Figure 6–2: Accounting Scheme 2



MR-S-553-80

Section 6.3 describes how to enter the information for Figures 6–1 and 6–2 into files that are used to create an account data base.

6.3 Creating An Account Data Base

Sections 6.3.1 through 6.3.3 describe how to use your selected accounting scheme and create the necessary files for your data base. Specifically, these sections include how to enter accounting data into files, how to run the account generator program (ACTGEN) to create the data base, and what to do if an error occurs.

6.3.1 Entering Accounting Data Into Files

Base and subaccount files are created using a text editor. The format below shows the combination of entries you can make in accounting files using

the CREATE command. Each file you or a project administrator creates can contain one or more accounts. Each account can point to one subaccount file, where additional account information is stored pertaining to that account. Following the format is a summary of the valid commands in an account file.

ACCOUNT DATA FILE FORMAT

```
@CREATE (FILE) <directory>filename.type
Input:  filename.type.1

00100  ACCOUNT account/SUBACCOUNT:dev:<dir>filename.type-
00200  /CLASS:n/ALLOW:n,n
00300  USER name,name,name,...
00400  DIRECTORY dev:<directory>
00500  GROUP (ON STRUCTURE) dev:/USER:user group number
00600  GROUP (ON STRUCTURE) dev:/DIRECTORY:directory group number
00700  $
*EU

<filename.type.1>
```

In addition to the above entries, each entry in the file can have an expiration date in the form:

```
/EXPIRES:dd-mm-yy hh:mm
```

This switch indicates when the account will no longer be valid for that entry in the file. For example:

```
USER name1,name2/EXPIRES:10-JAN-80,name3,name4
```

In the above example, name2 can no longer use this account after 10 January 1980. Name1, name3, and name4, however, can continue to use the account beyond that date. You could also place the switch immediately following the USER entry. For example:

```
USER/EXPIRES:10-JAN-80, name1, name2, name3, name4
```

This format specifies that none of the users in the list can use the account after a certain date. The account, however, remains open and you can place another list of users in the file who can use the account.

Table 6–1 summarizes the account data file commands. You can type the entire command, or just the characters necessary to distinguish one command from another. For example, ACCOUNT can be typed as AC.

Because the ACCOUNT command has several modifiers, you may have to continue typing the modifiers on the next line. To do this, use a hyphen at the end of the line and continue typing the ACCOUNT modifiers on the next line. For example,

```
0100 ACCOUNT TEST/SUBACCOUNT:SYSA:<MARK> ACCOUNT.TXT-
0200 /CLASS:2/ALLOW:1,3
```

Table 6–1: Summary of Account Data File Commands

Command	Description
ACCOUNT	Specifies the name of the account that you or a project administrator wish to assign. Note: The ACCOUNT command must be the first entry in an account data file because all subsequent entries up to the next ACCOUNT entry are modifiers.

/SUBACCOUNT:	Modifies the ACCOUNT command. It includes the specification of the file where additional data for the account can be found.
	
	Note: The ACCOUNT command accepts only one /SUBACCOUNT: modifier.
	
	Example: One of your accounting files contains the following commands.
	
	`ACCOUNT CORRECT-DATA/SUBACCOUNT:`
	`<GERHARD>ACCT.TXT`
	`ACCOUNT UNIONBANK/SUBACCOUNT:`
	`<WARRINER>ACCTG.TXT`
	
	ACTGEN looks in <GERHARD>ACCT.TXT for more account data for the account CORRECT-DATA, and it looks in <WARRINER>ACCTG.TXT for more data for account UNIONBANK.

/CLASS:n	Modifies the ACCOUNT command and is used in conjunction with the class scheduler. It specifies the scheduling class that is valid for this account. For example, `ACCOUNT CHEM-207/CLASS:3` means that class 3 is valid when using the account CHEM–207. ACTGEN places this information in the system's accounting data base for use by the class scheduling routines. Use the /CLASS:n switch only if you have selected to specify class scheduling by account. (Refer to Section 10.1 for a complete description of using the class scheduler by account.) When a user gives an account that does not have a valid class associated with it, the system uses the default class, class 0. If the account has a class associated with it, that class is used. The percentage of CPU time that classes can receive is defined in the n–CONFIG.CMD file. To use class scheduling by account, you must have: • made the appropriate entries in the n–CONFIG.CMD file. • updated your ACCOUNTS.CMD file (and subaccount files) to specify the classes that are associated with each account. • run ACTGEN with the INSTALL command to update the ACCOUNTS–TABLE.BIN file. • given the ENABLE CLASS–SCHEDULER command to OPR or brought the system down and back up again to start class scheduling.

(continued on next page)

Table 6–1: Summary of Account Data File Commands (Cont.)

Command	Description

/ALLOW:n,n	Modifies the ACCOUNT command and is used in conjunction with the class scheduler. It allows you to delegate the assigning of classes to subaccounts by project administrators. It specifies the class or classes that can be used by subaccounts of this account. For example,
	
	`ACCOUNT CHEM/SUBACCOUNT:<ABC>MORE.TXT-`
	`/CLASS:2/ALLOW:1,3`
	
	means the CHEM account is in class 2, and that subaccounts created under CHEM can be in either class 1 or class 3. If no /ALLOW switch is given, the administrator is not restricted to using certain classes and; therefore, can give the subaccounts any class. For example, the administrator can give them the same class as the superior directory. The /ALLOW switch is useful when you want the superior account to be in perhaps a higher percentage class than its inferior accounts. Remember that if the administrator does not give a /CLASS switch to the subaccount, users who log into or change to this subaccount will be in class 0.

USER	Specifies one user or the list of users who are allowed to use this account.

*	Specifies that an account is valid for all users on a system. The * is a special argument to the USER command.
	
	Example: One instance when you might use * is if you have not established an accounting scheme but would like to allow users to log into the system. You could set up one account and use the * to indicate that all users can use that account.
	
	You could also use the * as follows:
	
	`ACCOUNT MATH-101`
	`USER: MATH,*`
	
	This means that all users with a user name beginning with MATH can use the MATH–101 account. For example, the users assigned the user names MATH.SMITH, MATH.JONES, and MATH.BROWN can all use this account.

DIRECTORY	Specifies a directory name. It indicates that the account is valid for anyone with write access to the directory. This feature allows users to create or store files in systemwide or groupwide directories and to charge them to an "overhead" account different from their own account. The usage charged to this directory could be absorbed by system or project administration. This command also prevents users from being charged for file storage in files-only directories. Note: The DIRECTORY command also accepts a form of the wildcard entry. The valid forms are: *:<*>, dev:<*>, or *:<dir>. The asterisk indicates that users with write access to any of the directories matching the wildcard entry may charge their file creation to a certain account.

(continued on next page)

Table 6–1: Summary of Account Data File Commands (Cont.)

Command	Description
DIRECTORY (continued)	Example: The file that contains account data for account CORRECT–DATA.UNIONBANK.FUND has the entry: `DIRECTORY PS:<FORLIB>` This entry means that anyone with write access to directory <FORLIB> can use the account CORRECT–DATA.UNIONBANK.FUND when storing files there.
GROUP	Specifies that an account is valid for use by certain user and directory groups on a structure. (Refer to Section 5.8 for a description of groups.)
/USER:nnn /DIRECTORY:nnn	Modifies the GROUP command and can be used in any combination. (nnn is a decimal user or directory group number.) /EXPIRES: can be placed after either modifier to indicate when an account becomes invalid for use by the group. Note: Using the GROUP command is helpful if many people are eligible to use an account. Specifying their group number (if they are in a group) eliminates typing a long list of names incorrectly.

6.3.2 Sample Data Files

The examples below show how you could enter the information in Figures 6–1 and 6–2 into account data files. The first example shows the data file for Figure 6–1. Tabs and comment lines beginning with an exclamation point (!) can be used within the file for ease in reading or formatting the file. This file in particular contains all the base accounts and should be stored in your directory. You may find it easier when you run ACTGEN if you name the file ACCOUNTS.CMD. ACCOUNTS.CMD is the default file that the TAKE command under ACTGEN looks for. (Refer to Section 6.3.3 for running ACTGEN and giving the TAKE command.)

```
@CREATE (FILE) <HUDSON>ACCOUNTS.CMD(RET)
Input: ACCOUNTS.CMD.1

00100  !This file contains definitions of top-level accounts(RET)
00200  ACCOUNT CORRECT-DATA(RET)
00300    USER Hudson,Holland,Gerard,Gionet(RET)
00400    USER King,Kelly,Kohn(RET)
00500  ACCOUNT UNIONBANK /EXPIRES:1-JUN-80(RET)
00600    USER Warriner,Bloomstran,Prest,Pendergast(RET)
00700  ACCOUNT LP-FOOD /EXPIRES:1-JUL-80(RET)
00800    USER Schied,Queeny,Smith(RET)
00900  ACCOUNT TOWN-SQUARE /EXPIRES:15-JUL-80(RET)
01000    USER Markley,Gerhard,Dole(RET)
01100  $
*EU(RET)

<ACCOUNTS.CMD.1>
```

NOTE

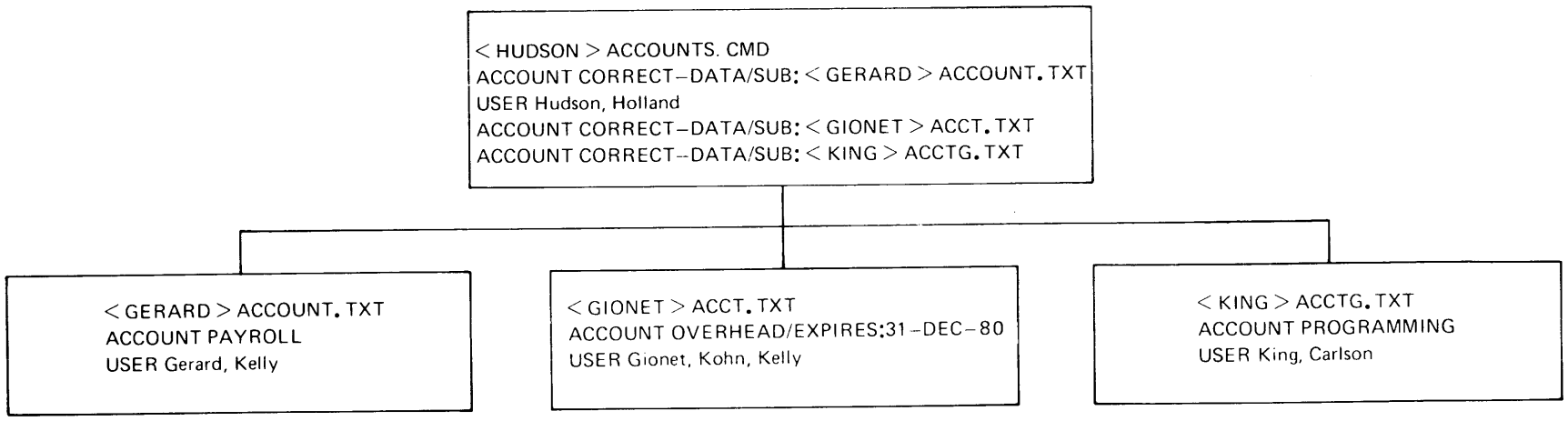
Lines 300 and 400 contain all the users at Correct Data. However, you would not use the asterisk (*) because it would enable all the users of the system, including the users at Unionbank, L & P Food, and Town Square to charge to the CORRECT–DATA account.

Figures 6–3 and 6–4 show what information to enter into base and subaccount files for Figure 6–2. Each block in Figures 6–3 and 6–4 contains information to be entered into a separate file. Some of the blocks contain subaccount (/SUB:) entries that point to other files containing more information about that particular account.

Figure 6–3 shows which entries to make for account CORRECT–DATA and its subaccounts (the top half of Figure 6–2.)

Figure 6–4 shows which entries to make for account UNIONBANK and its subaccounts (the bottom half of Figure 6–2.) Note that all the base account entries for both Correct Data and Unionbank are contained in the file <HUDSON>ACCOUNTS.CMD.

Figure 6–3: Correct-Data Accounting Files



NOTES

The accounts that will be created are:

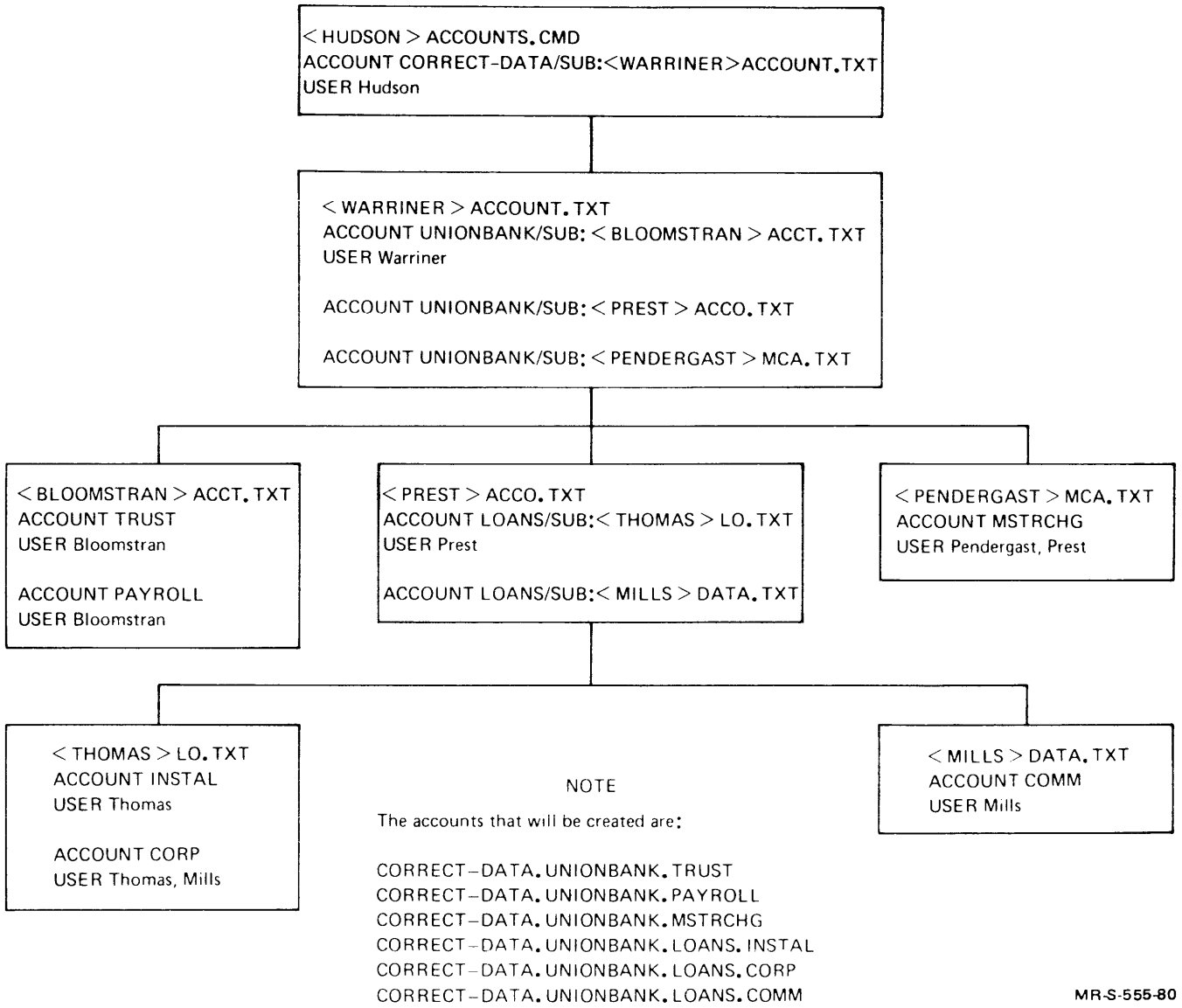
CORRECT–DATA. PAYROLL
CORRECT–DATA. OVERHEAD
CORRECT–DATA. PROGRAMMING

Note that users Hudson and Holland can use any account number that begins with CORRECT–DATA, but user Gerard can use only the account CORRECT–DATA. PAYROLL. User Kelly can use the accounts CORRECT–DATA. PAYROLL and CORRECT–DATA. OVERHEAD.

The file type for account data files is optional. Your project administrators can use any file type, e. g. , .TXT, .CMD or .ABC.

MR-S-554-80

Figure 6–4: Unionbank Accounting Files



6.3.3 Running the ACTGEN Program

After you create the base account files and the project administrator notifies you that all his subaccount files are complete, you can tell the operator to run the ACTGEN program. ACTGEN takes the accounting information in these files and creates an account validation data base. It is through this data base that the monitor validates all accounts entered by the users of your system. ACTGEN is a privileged program, so you must enable WHEEL or OPERATOR capabilities before giving the ACTGEN command. The command appears as follows:

```
@ENABLE (CAPABILITIES) RET
$ACTGEN RET
ACTGEN>
```

Valid commands that can be given to ACTGEN are HELP, EXIT, TAKE, CTRL/A, and INSTALL.

The HELP command lists information to assist you when running the ACTGEN program.

The EXIT command exits the program and returns you to the TOPS–20 command level ($).

The TAKE command accepts as its argument either a file specification or a carriage return. A carriage return defaults to your connected directory and the filename ACCOUNTS.CMD. If you do not name your base account file ACCOUNTS.CMD, the file you specify should be the one that contains your base account information and points to all the existing subaccount files. The TAKE command tells ACTGEN to look in the specified (or default) file for account information. It also tells ACTGEN to look at any subaccount file specifications for additional information pertaining to the account(s) in the base account file. Using Figures 6–1 and 6–2, the manager, Hudson, would specify the TAKE command as follows:

```
@ENABLE (CAPABILITIES) RET
$ACTGEN RET
ACTGEN> TAKE (COMMANDS FROM FILE) RET
```

If the manager in these examples had named his base account file MACCT.TXT, he would specify the TAKE command as follows:

```
ACTGEN> TAKE (COMMANDS FROM FILE) <HUDSON>MACCT.TXT RET
```

CAUTION

If the data files that are pointed to by your base account file are located on structures other than PS:, be sure the required structures are mounted. Otherwise, the ACTGEN program will fail on those accounts that have subaccount files on unmounted structures.

ACTGEN takes all the information specified in the account files, forms the valid accounts, and creates a new version of the file ACCOUNTS–TABLE.BIN in the directory where ACTGEN is running. Each time the ACTGEN program is run successfully, a new version of this file is created.

While ACTGEN is creating the data base file, it checks for duplicate entries, e.g., two accounts of the same name, and the length of the accounts. If an error occurs, an appropriate message is printed on the terminal where ACTGEN is running but the program continues to build the data base, using only the accurate data. You should make a note of the error on an error log sheet that you have prepared and later correct the appropriate files using an editor.

ACTGEN also checks expiration dates. If two or more expiration dates are given for the same entry in a file, the system uses the earliest date. For example: if you specify May 15, 1980 as the expiration date for account MATH and your project administrator specifies August 30, 1980 for the account MATH.LAB–201, the system will stop validating all accounts beginning with MATH as of May 15, 1980.

You can press CTRL/A while ACTGEN is running to stop the program and return to ACTGEN command level. The data files are closed and no new version of the data base file ACCOUNTS–TABLE.BIN is created from this session.

The INSTALL command starts account validation. When you enter this command, ACTGEN copies the ACCOUNTS–TABLE.BIN file in your connected directory (or the directory where ACTGEN created the file) to PS:<SYSTEM>ACCOUNTS–TABLE.BIN and enables account validation using this new data base.

Because the new version of ACCOUNTS–TABLE.BIN is kept in the directory where ACTGEN was run and not in the directory <SYSTEM>, you have a means of correcting any errors that might occur without disturbing the version currently running in the PS:<SYSTEM>ACCOUNTS–TABLE.BIN file. You can give the INSTALL command after you have corrected any problems.

If you do not receive any errors while ACTGEN is creating the data base file (ACTGEN has successfully completed the accounting file and you receive the ACTGEN prompt), you can give the INSTALL command immediately.

You should keep track of which version of the <SYSTEM>ACCOUNTS–TABLE.BIN file you are using. A log book that contains the date that ACTGEN was run and the version number of the data base file is helpful should a system problem occur and you are not sure of which data base file you were using. To find out which version you are using, enable capabilities and give the DIRECTORY command for PS:<SYSTEM>ACCOUNTS–TABLE.BIN. The generation number indicates the version that is presently running. The system looks in the current data base file each time it validates a given account.

6.3.4 Data Base Failures/Recovery

If your accounting files were set up inaccurately, or you entered random incorrect data into the data base file, account validation will not work properly. You are aware of this because users cannot log in and/or use accounts that are normally valid for them. Therefore, the account OPERATOR is set up for the user OPERATOR and is always valid. The operator can log into <OPERATOR> with the OPERATOR account, fix the files that are in error, and run ACTGEN to get account validation working again.

6.4 Validating Accounts

An account is validated when a user gives any one of the following TOPS–20 commands.

- LOGIN — A user must have a valid account to successfully log into the system
- SET ACCOUNT (TO) argument
- Any queue commands, for example, PRINT, SUBMIT, if the account is different from the currently validated account
- SET FILE ACCOUNT (OF FILES) arguments (TO) argument
- File creation with an explicit account

The system records the computer time used for valid accounts. This includes CPU time, time used per structure[1], and peripherals used per job, that is, the number of pages printed on the line printer, tape mounts, tape records read/written, card reader usage, and disk storage. The computer usage incurred by each account is stored in the accounting USAGE.OUT file. This file is used for reports and billing. (Refer to the *TOPS–20 Usage File Specification* for information about reading and using this file).

How often you run ACTGEN and create a new data base file depends on how frequently you change your accounts. If you expect to have frequent changes (e.g., opening and closing accounts), you may want to establish a standard time each week to run ACTGEN. Your administrators should inform the operator when changes are made to their subaccount files.

NOTE

Once ACTGEN is run and the data base file has been created, you can dump all the account files to magnetic tape and conserve some of your disk space. You must copy the files to disk the next time you need to run the ACTGEN program.

---

[1] To account for the time used on a structure, you must set the structure as REGULATED. Refer to the *TOPS–20 Operator's Guide* for a description of REGULATED and NONREGULATED structures.

Chapter 7
System Backup Procedures

All the disk packs on your system must be backed up on magnetic tape. This procedure provides both a permanent record of the contents of the disk and a precautionary measure in the event a disk pack and/or its contents are destroyed. On the first day of operation, start a system backup procedure that includes:

1. Saving all the files in all the directories on all structures
2. Saving the directory parameters and critical system programs
3. Saving the front-end file system (one time only)

These procedures should become a part of the operator's scheduled duties.

It is important to start backing up the system immediately after installation. If you follow the backup procedures as they are outlined here and in the *TOPS–20 Operator's Guide*, you can restore the file system quickly and easily should a mishap occur.

7.1 Saving All Files In All Directories

Have the operator run the DUMPER program (with the /FULL–INCREMENTAL switch to the SAVE command) to save all files in all directories. This procedure includes saving all the directories on the public structure, PS:, and all the directories on any additional structures you have created. You can save all the files (a full dump) or just the files that have changed since the last time the operator ran DUMPER (an incremental dump).

Start a library of the magnetic tapes from the DUMPER operations. Each structure should be copied to a separate tape(s). Each tape should be identified with the system model number or name, for example, 2020, 2050, or System-A, the date, the type of save (full or incremental), the name of the structure, and the tape number. (A tape set name may replace the tape number, if labeled tapes are used.) A typical identification may look like:

SYSTEM–A (2050)		SYSTEM–A (2050)
30–JANUARY–1980		30–JANUARY–1980
Incremental	OR:	Full
ADMIN:		ADMIN:
Tape #1 of 2		Tape #3 of 3

In addition to keeping the tapes, keep the listing of their contents. (The operator includes a command to DUMPER to list the contents of the magnetic tapes on the line printer.) These DUMPER log files can be conveniently kept in a binder with the most recent listing on top. Identify each binder with the system model or name, for example, 2040 or System-A. (Chapter 9, System Problems/Crashes describes how to use these log files to restore directories and files.)

Tell users that backup files do exist and post the times when the operator normally creates the backup tapes. Many system managers do not allow users to enter the computer room to mount and use the tapes themselves.

NOTE

The DUMPER program *DOES NOT* save the files in the console front-end file system on the DECSYSTEM–2040, 2050, and 2060. If you lose these files, you must restore them from the floppy disks. (Refer to Section 7.6.) Also, the DUMPER program does not save the <ROOT–DIRECTORY>BOOTSTRAP.BIN file on the DECSYSTEM–2020. You must run the SMFILE program to restore this file. (Refer to Chapter 5 in the *TOPS–20 Software Installation Guide* for a description of running SMFILE.)

7.1.1 Full Dumps

Full dumps contain all the information on the system, with the exception of the BOOTSTRAP.BIN file on the 2020 and the console front-end files on the 2040, 2050, and 2060, and can be used to restore many of the files that were on the system to their previous state. Therefore, full dumps contain a copy of every file in every directory on every structure.

NOTE

Full dumps are known as FULL–INCREMENTAL dumps. This name corresponds to the /FULL–INCREMENTAL switch that you give with the SAVE command to DUMPER.

7.1.2 Incremental Dumps

Incremental dumps (using the /INCREMENTAL switch with the SAVE command) cause DUMPER to save the files that have never been saved (new files) and the files that have been updated since the last time an incremental DUMPER operation was performed. Many managers request an incremental dump Monday through Thursday and a full dump on Friday.

The File Descriptor Block (FDB) of each file contains the information necessary to determine if the file has been updated since it was last saved during an incremental DUMPER operation. A file that has changed since the last time it was saved is automatically saved again on tape; otherwise, it is passed over.

The operator should give the INCREMENTAL switch to DUMPER and specify each structure one at a time. By running DUMPER for each structure individually, the operator can copy each structure onto a separate tape. After running DUMPER and copying the structures that are presently on–line, the operator should mount any additional structures that have been used that day and run DUMPER for them also.

An incremental dump is faster than a full dump and requires less magnetic tape.

7.2 A Common Backup Policy

A common backup policy is outlined below. You can set up your own backup policy.

1. Each day, take an incremental save of the files that have changed from the previous day's backup tape. Keep the incremental saves until a full save is made, at which time you can recycle the incremental tapes.
2. At the end of the week, take a full save of all the files on the system. Keep the full saves for six months, at which time you can recycle the tapes into the backup system.
3. Every six months, take a full save and keep it for a number of years, or if you choose, indefinitely.

7.3 Magnetic Tape Requirements

You need a supply of magnetic tapes to start a system backup procedure. This section provides a guideline for the number of tapes you should have on hand for your installation.

It takes approximately seven 2400 foot reels of magnetic tape, using a 1600 bit–per–inch (bit/in) drive, to back up a full RP06 disk. It takes approximately four tapes (2400 ft., 1600 bit/in) to back up a full RP04 disk and two tapes (2400 ft., 1600 bit/in) to back up a full RM03 disk. It takes approximately 19 tapes (2400 ft., 1600 bit/in) to back up one spindle of an RP20 disk or approximately 37 tapes (2400 ft., 1600 bit/in) to back up one RP20

drive (both spindles). It takes approximately 19 tapes (2400 ft., 1600 bit/in) to back up a full RP07 disk pack. It takes approximately 6 tapes (2400 ft., 6250 bits/in) to back up one spindle of an RP20 disk or approximately 12 tapes (2400 ft., 6250 bits/in) to back one RP20 drive (both spindles). It takes approximately 7 tapes (2400 ft., 6250 bits/in) or 19 tapes (2400 ft., 1600 bits/in) to back up an RP07 disk drive.

Therefore, the number of tapes you stock depends on the type of disks at your installation. It also depends on the backup procedure you use. For example, if you save your daily incremental tape dumps for a longer time than usual, it takes longer to recycle these tapes into the backup system, and thus you need more tapes.

Generally, during the first month after installation, you may need approximately 36 (2400 ft.) tapes for each RP06, 24 tapes for each RP04, 16 tapes for each RM03 disk, 45 tapes (6250 bit/in) or 180 tapes (1600 bit/in) for each RP20 disk (2 spindles), and 72 tapes for each RP07 (1600 bit/in).

NOTE

These estimates assume a magnetic tape blocking factor of 1. You can specify a higher blocking factor and save some space on your tapes. Before doing this, however, there are cautions that you must consider. The description of DUMPER in the *TOPS–20 User Utilities Guide* explains how and when you can increase blocking factors.

7.4 Making A System Crash Tape

As the name implies, the system crash tape is used to re–create PS: should it become unusable. This tape is created in addition to the tapes that contain FULL–INCREMENTAL and INCREMENTAL saves of all files and directories. You should make a new system crash tape whenever you add a new user, change any directory parameters, or make a change (patch) to:

- The monitor you are running
- The TOPS–20 Command Processor
- The DLUSER program
- The DUMPER program

Therefore, the crash tape contains only the files necessary to recover user directory parameters and important system programs. User files themselves are restored from the FULL–INCREMENTAL and INCREMENTAL saves of PS:.

Label this tape PS: SYSTEM BACKUP TAPE and include the DECSYSTEM–20 model number or name, for example, 2050 or System–B, and the date and time the tape was created. You should follow this procedure once a

day if users are allowed to change their own directory parameters. (Refer to Section 5.7.2 for information about allowing users to change directory parameters.)

NOTE

Do not use a labelled tape when creating a System Crash Tape. The reason for this is MOUNTR must be running to read the label.

The order of files on the crash tape for a DECSYSTEM–2040, 2050, and 2060 is:

1. PS:<SYSTEM>MONITR.EXE
2. PS:<SYSTEM>EXEC.EXE
3. PS:<SYSTEM>DLUSER.EXE
4. DLUSER data files
5. PS:<SUBSYS>DUMPER.EXE
6. DUMPER save sets containing the directories:

   PS:<SYSTEM>
   PS:<SUBSYS>
   PS:<NEW–SYSTEM>
   PS:<NEW–SUBSYS>
   PS:<UETP>
   PS:<GALAXY–SUBSYS>

The order of files on the crash tape for a DECSYSTEM–2020 is:

1. KS10 microcode file
2. Bootstrap routines
3. PS:<SYSTEM>MONITR.EXE
4. PS:<SYSTEM>EXEC.EXE
5. PS:<SUBSYS>DLUSER.EXE
6. DLUSER data file
7. PS:<SUBSYS>DUMPER.EXE
8. DUMPER save sets containing the directories:

   PS:<SYSTEM>
   PS:<SUBSYS>
   PS:<NEW–SYSTEM>
   PS:<NEW–SUBSYS>
   PS:<UETP>

Notice that the format of this tape is the same as the *TOPS–20 Installation Tape* that you used to install the TOPS–20 software.

7.5 Making A Crash Tape Using Batch

You can create a batch job to make your crash tape or type the commands at the operator's terminal. Example 1 shows the DECSYSTEM–2040, 2050, and 2060 standard control file that you can submit as a batch job to create this tape. Example 2 shows the DECSYSTEM–2020 standard control file that you can submit as a batch job for that system's tape.

EXAMPLE 1

DECSYSTEM–2040, 2050, 2060 SYSTAP Control File

```
@TYPE (FILE) SYS:SYSTAP.CTL(RET)

! Obtain a tape drive
@MOUNT TAPE TAPE:/WRITE/LABEL:UNLABELED

!Systems not using Tape Drive Allocation must replace the
!MOUNT TAPE command with @ASSIGN MTA0: and @DEFINE TAPE:
!(AS) MTA0: commands.

@ENABLE (CAPABILITIES)
@REWIND (DEVICE) TAPE:

!Save the monitor
@GET (PROGRAM) PS:<SYSTEM>MONITR.EXE
@SAVE (ON FILE) TAPE:

! Save the TOPS-20 Command Language Interpreter
@GET (PROGRAM) SYSTEM:EXEC.EXE
@SAVE (ON FILE) TAPE:

!Save the DLUSER program
@GET (PROGRAM) SYS:DLUSER.EXE
@SAVE (ON FILE) TAPE:

!Run the same DLUSER program, saving the directory structure
!on tape
@START
*DUMP (TO FILE) TAPE:
*EXIT

!Save DUMPER
@GET (PROGRAM) SYS:DUMPER.EXE
@SAVE (ON FILE) TAPE:

!Run the same DUMPER, saving SYSTEM: and SYS:
@START
*TAPE (DEVICE) TAPE:
*LIST (LOG INFORMATION TO FILE) SYSTAP.LPT
*SSNAME SYSTEM-FILES
*SAVE (DISK FILES) PS:<NEW-SYSTEM>,PS:<SYSTEM>
*SSNAME SUBSYS-FILES
*SAVE (DISK FILES) PS:<NEW-SUBSYS>,PS:<SUBSYS>
*EXIT

!Print the DUMPER log file
@PRINT SYSTAP.LPT/NOTE:BACKUP TAPE

@DISMOUNT TAPE:
@
```

```
!Systems not using Tape Drive Allocation must replace the
!DISMOUNT TAPE: command with @UNLOAD (DEVICE) TAPE: and
!@DEASSIGN TAPE: commands.
```

To run SYSTAP, submit the batch control file using the TOPS–20 SUBMIT command.

EXAMPLE 2

DECSYSTEM–2020 S20TAP Control File

```
@ TYPE(FILE)  SYS:S20TAP.CTL(RET)

!Obtain a tape drive
@MOUNT TAPE TAPE:/WRITE/LABEL:UNLABELED

!Systems not using Tape Drive Allocation must replace the
!MOUNT TAPE command with @ASSIGN MTA0: and @DEFINE TAPE:
!(AS) MTA0: commands.

@ENABLE (CAPABILITIES)
@REWIND (DEVICE) TAPE:

!Write microcode
@COPY (FROM) <SYSTEM>KS10.RAM (TO) TAPE:

!Write bootstrap
@COPY (FROM) SYSTEM:MTBOOT.RDI (TO) TAPE:

!Save the monitor
@GET (PROGRAM) PS:<SYSTEM>MONITR.EXE
@SAVE (ON FILE) TAPE:

!Save the TOPS-20 Command Language Interpreter
@GET (PROGRAM) SYSTEM:EXEC.EXE
@SAVE (ON FILE) TAPE:

!Save the DLUSER program
@GET (PROGRAM) SYS:DLUSER.EXE
@SAVE (ON FILE) TAPE:

!Run the same DLUSER program, saving the directory structure
!on tape
@START
*DUMP (TO FILE) TAPE:
*EXIT

!Save DUMPER
@GET (PROGRAM) SYS:DUMPER.EXE
@SAVE (ON FILE) TAPE:

!Run the same DUMPER, saving SYSTEM: and SYS:
@START
*TAPE (DEVICE) TAPE:
*LIST (LOG INFORMATION ON FILE) S20TAP.LPT
*SSNAME SYSTEM-FILES
*SAVE (DISK FILES) PS:<NEW-SYSTEM>,PS:<SYSTEM>
*SSNAME SUBSYS-FILES
*SAVE (DISK FILES) PS:<NEW-SUBSYS>,PS:<SUBSYS>
*EXIT

!Print the DUMPER log file
@PRINT S20TAP.LPT/NOTE:BACKUP TAPE

@DISMOUNT TAPE:
@
```

```
!Systems not using Tape Drive Allocation must replace the
!DISMOUNT TAPE: command with @UNLOAD (DEVICE) TAPE: and
!@DEASSIGN TAPE: commands.
```

To run S20TAP, submit the batch control file using the TOPS–20 SUBMIT command. For example:

```
@ SUBMIT SYS:S20TAP.CTL(RET)
[INP:S20TAP=/SEQ:2346/TIME:00:05:00]
```

In the event PS: becomes unusable, the crash tape can now be used by following the instructions in the *TOPS–20 Software Installation Guide.*

HINT:

Before you store the crash tape, verify that you have made a usable tape. That is, mount the new crash tape, follow the instructions in the *TOPS–20 Software Installation Guide* to load the Monitor, and use DUMPER to get a listing of the tape's contents.

7.6 Saving The Console Front-End File System (2040, 2050, 2060)

The DUMPER program does not save the contents of the console front-end file system. You can, however, make a backup copy of the file system by copying your floppy disks using the front-end program COP (for copy). You should make at least one backup copy of your console front-end file system (refer to Section 3.4).

To run the COP program, follow the steps outlined below. You need not stop timesharing when following these steps.

1. At the operator's console, type CTRL/backslash; the system prints PAR>.

   CTRL/backslash

   ```
   PAR>
   ```

2. Type MCR COP and press the RETURN key; the system prints the COP prompt.

   ```
   PAR> MCR COP(RET)
   COP>
   ```

3. Place the floppy disk to be copied in drive 0 and the floppy to contain the new files in drive 1. Be sure to mount the floppy disks correctly; this includes checking that the paper containing the floppy directory is not accidentally attached to the back of the floppy disk.

4. Type DX1:=DX0: and press the RETURN key; the system starts the copying, which takes a few minutes. After the copying is complete, the system verifies the copy and prints a message. Type CTRL/Z to exit COP.

```
COP>DX1:=DX0:<RET>
COP - STARTING VERIFY

COP>^Z
```

5. To return to TOPS–20 Command Level, type a CTRL/backslash; the system prints PAR>; type another CTRL/Z, or type QUIT and press the RETURN key.

   CTRL/backslash

```
PAR>^Z

@
```

Chapter 8
Tape Storage

Chapters 1 through 6 deal primarily with setting up and using your disk resources. Chapter 7, System Backup, describes how to save all the data from your disk structures onto magnetic tape. These tapes are the backup tapes that you use to restore directories and perhaps entire disk structures if something happens to the disks (refer to Chapter 9). In addition to using magnetic tapes for system backup tapes, you can use magnetic tapes to store other types of data and save valuable disk space.

This chapter describes two other uses for magnetic tapes, File Archiving and File Migration. It also describes how you can allow the system and the operator to control all tape drive assignments, Tape Drive Allocation, and how you can set up some or all of your tapes to contain labels, Tape Labeling. These uses are described in the following order.

- FILE ARCHIVING
- FILE MIGRATION
- TAPE DRIVE ALLOCATION
- TAPE LABELING

File archiving provides you and users of the system with a voluntary way to move important files from the disk to magnetic tape for long-term storage. These tapes are stored separately from your system backup tapes. Users can access these tape files as easily as they access files on the disk. When users want to restore archived files to disk, they give a command to the TOPS–20 command processor. The system then tells the operator which tapes to mount and proceeds to restore the files. Section 8.1 describes why you would use the file archiving facility, and how to set up your system to archive files to magnetic tape.

File migration provides you with a means of controlling the use of disk space. File migration is especially useful if your disk space is very low on a particular structure, for example, PS:. This type of disk space control is, for the most part, involuntary on the part of the user. Old unused disk files are periodically moved (migrated) to magnetic tape by the system operator. Again, you should store these tapes separately from your system backup tapes. Users still maintain easy access to these files and retrieve them the same way as they retrieve archived files. Section 8.2 describes why and when you would use the file migration facility and how to set up your system to migrate files to magnetic tape.

If you use the file archiving or file migration facilities, or both, remember that these tapes are in addition to your system backup tapes. They are not replacements. You must continue to run the DUMPER program and create full and incremental system backup tapes.

Tape drive allocation provides the system and computer operator with complete control over tape drive usage. This means that it prevents users from issuing the ASSIGN command and reserving tape drives for their jobs. When users issue the MOUNT command, the TOPS–20 Tape Drive Allocation system and the operator control the allocation of tape drives. You must use the tape drive allocation facility if you use tape labeling; however, using tape drive allocation does not restrict you to using labeled tapes.

Tape labeling provides a means of storing label information on the tape itself that identifies the tape and describes the data on the tape. This label information is in an industry standard format so you can read and write tapes to be used with different computers. Tape labels can also add more security and reliability to your tape system. Section 8.3 describes why you would use tape labels, and also how to set up your system to begin labeling tapes.

There are no dependencies among file archiving, file migration, and tape labeling. For example, you can use the file archiving facility without using file migration or tape labeling. Each tape facility can be used separately. There is, however, the dependency that tape drive allocation must be turned on to use tape labels.

8.1 File Archiving

File archiving provides a means of storing data on magnetic tape and freeing valuable disk space. This type of off-line storage allows users to store (archive) important files on tape, keeping their disk space below their permanent allocation, and still have easy access to those files.

If your installation has more than one computer system, the archive tapes can be common to all systems. You can put files on tape from one system, move a directory and its files to another system, and still retrieve files from the tape in the ordinary manner.

Unlike general system backup tapes, the tapes that contain archived files are usually kept for a much longer time, for example, seven to ten years.

8.1.1 Setting Up the System to Use File Archiving

If you did not use the file archiving feature in Release 4 but plan to do so in Release 5, you must run the <SYSTEM>CNVDSK.EXE program. This program extends each file's File Descriptor Block (FDB) to allow for tape archival information. That is, each file now has room to contain a pointer to the location of the file on magnetic tape. The CNVDSK program reformats all file FDBs on the system, with the exception of those files that are opened at the time this program is run. Therefore, you should schedule running this program at installation time, or when the system is stand–alone (no timesharing users are logged in).

If you are installing TOPS–20 Version 5 for the first time, you do not have to run the CNVDSK program.

NOTE

When a user restores a file(s) from an old backup medium, where FDBs are not extended, you, or someone with WHEEL or OPERATOR capability, can run the CNVDSK program for the individual directory. This ensures that all files in the directory have extended FDBs and allows the user to later archive or migrate these files. You may want to run CNVDSK and convert all files on your backup medium to the extended FDB format.

When you receive the TOPS–20 Installation Tape and have brought up the TOPS–20 monitor, your system contains a built–in default of 3650 days for recycling archived tapes. To change the 3650 day (10 year) default, you can enter a command in the n–CONFIG.CMD file. The command you use is:

ARCHIVE–TAPE–RECYCLE–PERIOD days

Select a length of time that is appropriate for your installation. Place the ARCHIVE–TAPE–RECYCLE–PERIOD command in the n–CONFIG.CMD file during software installation, or edit the file at a later date when you are planning to reload the system.

Each time the DUMPER program copies an archived file to tape, it places the expiration date argument in the FDB of the file. The MAIL program notifies users when a file on tape has reached its expiration date. If the file is no longer needed, the user can discard (using the DISCARD command) the information in the file's FDB that points to the file on tape. After all the files on a tape have passed their expiration dates and no users have FDBs in their directories that point to tape, the tape can be recycled. Refer to Section 8.2.5 for additional information on how to recycle tapes.

8.1.2 What Happens When Users Archive Files

Users archive files voluntarily by giving the ARCHIVE command. After a specific generation of a file has been archived (e.g., MYFILE.CBL.6), it cannot change. Users can obtain copies of archived files by using the RETRIEVE command, but cannot alter those files. The *TOPS–20 User's Guide* describes the ARCHIVE and RETRIEVE commands.

When you establish your installation's policy for file archiving and notify users of its availability, you may want to instruct users to archive source files only. For example, files with a file type

.CBL, .MAC, .TXT, .RNO, or .FOR

should be archived; but, files with the file type

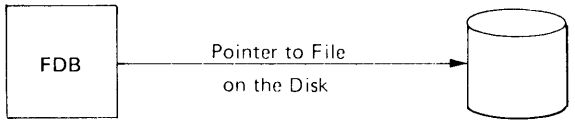
.REL, .EXE, or .MEM

should not be. This restriction saves space on your magnetic tapes.

To completely archive a file, two copies must exist in the archives. This means that each archived file is stored on two tapes. Having the archived file on two tapes provides you with a backup tape if later you cannot retrieve a file off one of the tapes. The DUMPER program, which is used to archive files, records both tape identifying numbers in the FDBs of the files being archived.

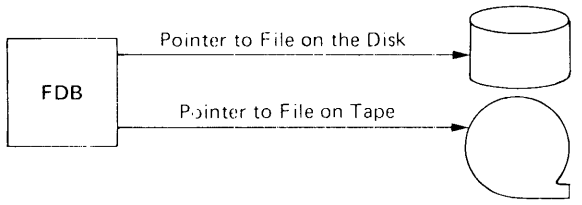
The diagrams below illustrate what happens when a user archives a file.

First, the user creates a file. The File Descriptor Block (FDB), among other file information, contains a pointer to the location of the file on the disk.



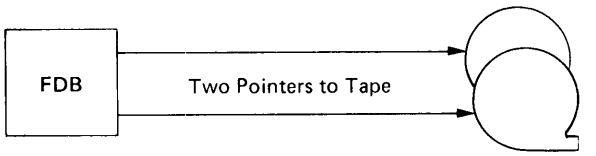
MR-S-557-80

The user gives the ARCHIVE command for this file. The first or next time the operator runs DUMPER with the SAVE /ARCHIVE command, the system locates all files that have been marked for archival by the ARCHIVE command and copies these files to tape. The FDB now contains pointers to the file on the disk and to the file on the tape.



MR-S-558-80

The second or subsequent time the operator runs DUMPER with the SAVE /ARCHIVE command (remember that archived files are contained on two tapes), DUMPER copies the file again to the second tape. DUMPER then deletes the pointer to the location of the file on the disk. DUMPER places another pointer in the file's FDB to the second tape that contains the file and deletes the contents of the file on disk. After a user archives a file, the file name no longer appears in the user's directory list. The user must give the DIRECTORY command with the ARCHIVE or INVISIBLE subcommand to see the file name.



MR-S-556-80

8.1.3 What Happens When Users Retrieve Files

Users request files be returned to disk (retrieved) by using the RETRIEVE command. When the operator runs DUMPER to process retrieval requests, DUMPER notifies the operator of the second tape that contains the file. If the file cannot be copied from the tape (e.g., the tape is bad), DUMPER notifies the operator of the first tape that contains the file. When DUMPER returns a file to disk, the FDB of that file now contains two pointers to tape and one to the disk. The pointer to the file on the disk remains in the FDB until the file is deleted from disk.

8.1.4 When To Create Archive Tapes

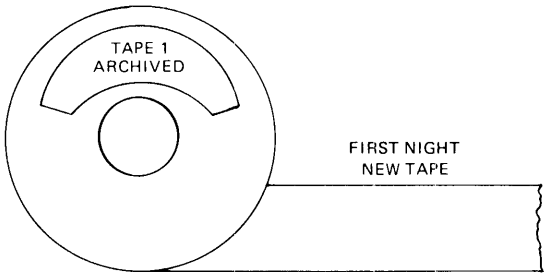
You can select how often the operator runs DUMPER to archive files. However, running DUMPER (with SAVE /ARCHIVE) every day before or after your general system backup procedure is probably closest to your present schedule.

The steps below provide an example of a typical procedure. These steps assume that this is the first time you are archiving files. The diagrams that accompany the steps show you what information is on the archive tape after each DUMPER operation. Although you do not have to use this procedure, it is one that best utilizes your tape resources. (The *TOPS–20 Operator's Guide* describes the procedure for running DUMPER with the SAVE /ARCHIVE command.)

Step	Procedure

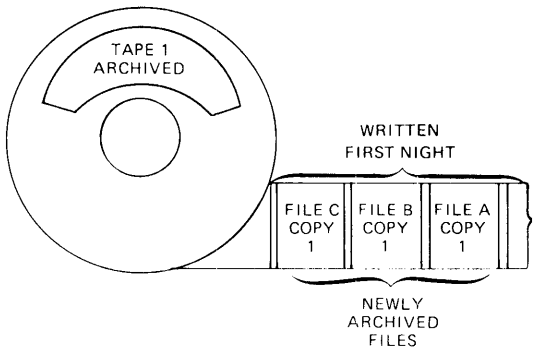
1. The operator runs DUMPER and performs the normal (incremental or full) backup procedures for the entire system. (Refer to Chapter 7, System Backup Procedures.)

2. The operator mounts a brand new tape (a tape that has been initialized if you are using tape labels; refer to Section 8.4.3) to contain the archived files, for example, TAPE 1.



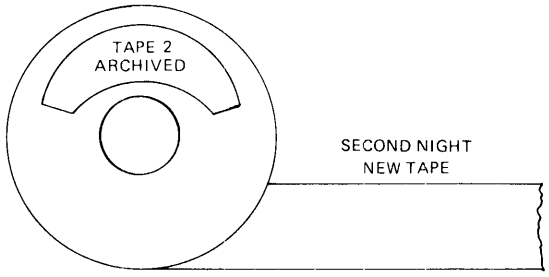
MR-S-559-80

3. The operator runs DUMPER with the SAVE /ARCHIVE command. DUMPER locates all files marked for archival and copies them to tape.



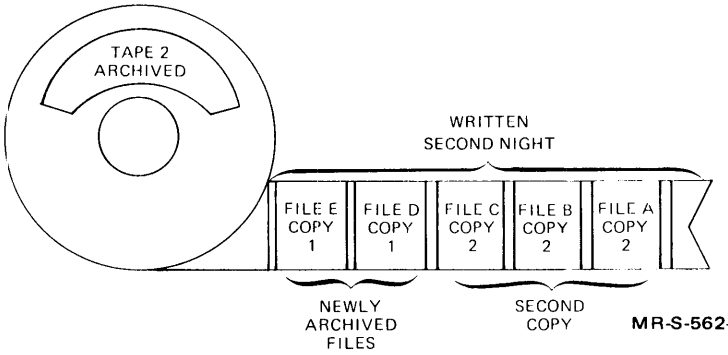
MR-S-560-80

4. The next evening (or the next time system backup is performed), the operator mounts a brand new tape, e.g., TAPE 2. He does NOT mount the tape used the first time.



MR-S-561-80

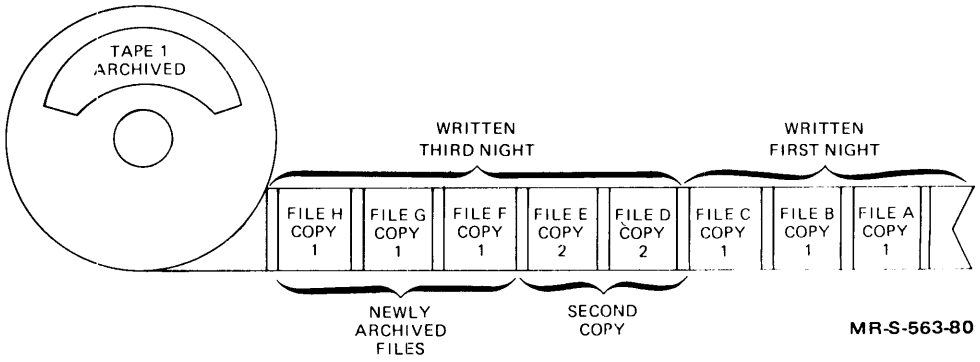
5. The operator runs DUMPER with the SAVE /ARCHIVE command. This time DUMPER finishes the archival run of the previous night by making a second copy of those files. In addition, DUMPER locates all the files newly marked for archival and copies them to tape for the first time.



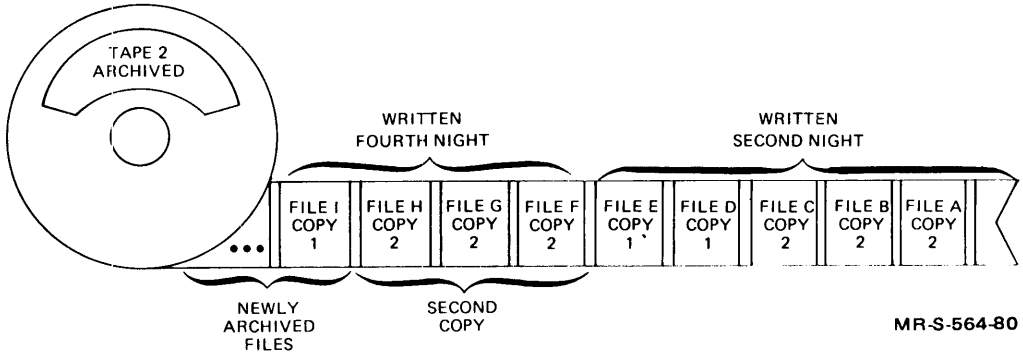
6. The operator repeats this process every day until the tapes are full.

7. For example, the third night, the operator mounts the first tape, TAPE 1.

8. The operator runs DUMPER. DUMPER finishes the archival run of the previous night by making a second copy of the previous nights files. It also locates all the files newly marked for archival and copies them to tape.



9. The fourth night, the operator mounts the second tape, TAPE 2. Again, DUMPER finishes the archival run of the previous night by making a second copy of those files. It also locates all the files newly marked for archival and copies them to tape.



NOTE

DUMPER checks tapes for duplicate files. It does not write both copies of the same file on the same tape. If the wrong tape is mounted, DUMPER outputs an error message.

8.1.5 Processing Retrieval Requests

When a user gives the RETRIEVE command to request an archived file, the request is stored in a queue. You must establish a policy for how often the operator should process the retrieval requests contained in the queue. (The *TOPS–20 Operator's Guide* describes how to process retrieval requests.) If you have encouraged users to archive their files, you should instruct the operator to process the request queue frequently.

8.2 File Migration

Some installations must control the use of disk space by periodically migrating files to magnetic tape. This forced file migration allows the management of an installation to move old unused disk files to a less expensive storage medium. Similar to archived files, migrated files are still easily accessible to the user. File migration also allows you to keep users' directories below their permanent allocation. (Refer to Section 5.5 for a description of permanent and working storage allocations.)

Whether you use the file migration facility depends almost entirely on your disk space resources. If users are archiving files regularly, if their directories are usually below their allowed permanent disk allocation, and your system is not continuously interrupted with "disk space low" messages, you may choose not to migrate files. Otherwise, if you are constantly receiving the [CAUTION — DISK SPACE LOW ON structure name] message, you may want to forcibly migrate files from the disk.

Sections 8.2.1 through 8.2.3 describe using file migration. They include:

- the program you must run before migrating files to tape
- the command that can be placed in the n–CONFIG.CMD file to change the default tape recycle period
- when to run the REAPER program that marks files for migration and marks for deletion the contents of archived and/or migrated files
- a sample of the REAPER.CMD file that you can use as a default file to be read by the REAPER program
- when to run the DUMPER program that locates files marked for migration and copies them to tape
- when to process retrieval requests for migrated files
- when to recycle migrated (and archived) tapes

8.2.1 Setting Up the System to Use File Migration

If you did not use the file migration feature in Release 4 but plan to do so in Release 5, you must run the <SYSTEM>CNVDSK.EXE program. This program performs the same function as it does for file archiving (refer to Section 8.1.1). That is, it extends the File Descriptor Block (FDB) for each file on the system to allow for additional file information. If you are using both file archiving and file migration, you need run the CNVDSK program only once.

If you are installing TOPS–20 Version 5 for the first time, you do not have to run the CNVDSK program.

When you receive the TOPS–20 Installation Tape and have brought up the new TOPS–20 monitor, your system contains a built–in default of 180 days for recycling migrated tapes. This default is placed in the FDB of each file as it is migrated to tape.

To change the 180–day default, you can enter a command in the n–CONFIG.CMD file to inform the system when you plan to recycle your migrated tapes. This command is:

TAPE–RECYCLE–PERIOD days

Select a length of time that is appropriate for your installation. The default of 180 days, however, is a standard time period. You can place the TAPE–RECYCLE–PERIOD command in the n–CONFIG.CMD file at the time you install the system (refer to the *TOPS–20 Installation Guide*). Or, you can edit the n–CONFIG.CMD file at a later date. Remember that if you edit the file at a later date, you must reload the system to process the commands in the n–CONFIG.CMD file.

CAUTION

If you decide to change the 180 day default, place the TAPE–RECYCLE–PERIOD command with the new argument in the n–CONFIG.CMD file and reload the system, BEFORE you run the <SYSTEM>CNVDSK.EXE program. Otherwise, the default recycling period does not change until the next system reload.

8.2.2 Using the REAPER Program

The REAPER program is the tool used to free disk space. It performs the following functions:

- Marks for migration the files that have not been referenced for a specified period of time
- Marks for deletion the disk contents of archived or migrated files, either after they have been successfully copied to tape, or after they have been returned to disk with the RETRIEVE command, and have not been referenced for a specified period of time
- Trims directories that are over permanent disk allocation by marking files in those directories for migration
- Deletes (purges) the tape pointers in FDBs on the disk that have reached their tape storage expiration date. That is, the file's FDB will no longer contain a pointer to the contents of the file on tape.

You can instruct the operator to run the REAPER program and perform one, several, or all of these functions. The operator can give a list of commands to REAPER or use the TAKE command with the default argument SYSTEM:REAPER.CMD. After the system is installed, the directory <SYSTEM> contains a default REAPER.CMD file. You can use this file as it is or use an editor and change the default parameters. The default SYSTEM:REAPER.CMD file appears similar to the following.

```
$TYPE (FILENAME) SYSTEM:REAPER.CMD(RET)

! Sample REAPER policy file

! Directories not to be considered (specify the structure and directory)

SKIP PS:<NEW-SUBSYS>,PS:<NEW-SYSTEM>,PS:<SYSTEM>,PS:<SUBSYS>

PERIOD 60                   !Specifies the age limit on
                            !files

MIGRATE                     !Files older than PERIOD days

DELETE-CONTENTS             !Of unreferenced files older than
                            !PERIOD with tape backup

TRIM                        !Directories over perm allocation
                            !back to perm allocation

ORDER *.TMP,*.LST,*.REL     !The order to take files during TRIM
```

Note that the SKIP command includes a list of directories that are not to be touched by the REAPER program. You can add other system or user directories to this list. The list can contain approximately 75 directories. Be sure that the operator always includes this command when running the REAPER program; otherwise, you may accidently migrate some very important files from the disk.

The REAPER program accepts the following commands.

BEGIN (Processing files)
DELETE-CONTENTS (Of old offline files)
EXIT (To monitor)
LIST (Output to file)
MIGRATE (Old files to offline storage)
ORDER (For trimming)
PERIOD (For migration)
PURGE (Expired FDBs from disk)
SCAN (Only)
SKIP (Directories)
TAKE (Commands from file)
TAPE (Check of tapes in use)
TRIM (Directories over allocation)

The *TOPS-20 Operator's Guide* provides a complete description of all the commands that can be given to the REAPER program or placed in the REAPER.CMD file. Typically, you give a number of commands to REAPER, one for each operation you want performed.

The availability of disk space determines how often you run the REAPER program. If your disk space is low, you may want to run the REAPER program daily to free up as much disk space as possible. Other installations may run it once a month or less.

8.2.3 Using the DUMPER Program

After the REAPER program marks files for migration, the operator runs the DUMPER program to copy the files to tape. Similar to an archived file, a migrated file is not completely migrated until two copies of the file exist on magnetic tape. Section 8.1.4 describes a procedure for copying archived files to tape. You can use this same procedure for migrated tapes, by using the SAVE /MIGRATE command instead of SAVE /ARCHIVE.

If you use both the file archiving facility and the file migration facility, do not merge archived and migrated files on the same tapes. The expiration dates for migrated files differ greatly from the expiration dates on archived files. If you put them on the same tape, you will end up saving migrated files for approximately ten years and use up all your tape resources very quickly.

8.2.4 Processing Retrieval Requests for Migrated Files

When a user gives a DIRECTORY command, the files that have been migrated to tape still appear in the directory list; however, each file has a notation (;OFFLINE) beside the filename to indicate that the file is contained on tape and not on the disk. The versions of migrated files that have been copied to tape can be returned to disk, and unlike archived files, they can be altered and/or renamed in the ordinary manner. The user requests that a migrated file be returned to disk with the RETRIEVE command. These requests are stored in the same queue as archive requests until the operator processes the queue. The *TOPS–20 Operator's Guide* describes how to process retrieval requests. Retrieval requests for migrated or archived files should be processed frequently.

8.2.5 Recycling Migration (and Archive) Tapes

When all the files on a migrated or archived tape have passed their expiration dates and all pointers to these files on the disk have been deleted, you can recycle the tape.

The PURGE command to REAPER checks tape expiration dates and notifies users by the MAIL program when migrated or archived files on tape are about to expire. Users can retrieve the file to disk again or discard the tape pointer on disk if they no longer need the file.

The operator can determine if a tape can be recycled by giving the TAPE command to REAPER. If a tape is not mentioned in the TAPE output list, this means that none of the disk structures that are on-line at this time and specified to REAPER contain FDB pointers to that tape. However, be sure that you check all possible places for on-line (disk) pointers to this tape. That is, run REAPER with the TAPE command on all the disk structures on all systems that may contain pointers to this tape. If files have passed their expiration date and pointers to them still exist on the disk, the operator can run REAPER with the PURGE command to delete these pointers. The operator should be certain that files are no longer needed before using the PURGE command.

HINT

When a migration tape is full, have the operator use the PRINT command to DUMPER to obtain a hard-copy listing of the tape contents.

8.3 Tape Drive Allocation

Tape drive allocation provides the system, and not the user, with complete control over tape drive usage. When accessing a magnetic tape, the user must give a MOUNT command to request that the operator mount the tape on a drive. Once the operator responds to the user's request, the user can access the tape. When the user is finished with the tape, the user gives the DISMOUNT command to release the tape drive back to the system. From the user's point of view, the MOUNT and DISMOUNT commands replace the TMOUNT, ASSIGN, and DEASSIGN commands. The operator selects the drive for the user, and the system informs the user how to access the tape. Using tape labeling requires that you use tape drive allocation; however, this does not restrict you to the use of labeled tapes only.

8.3.1 When to Use Tape Drive Allocation

Table 8–1 lists the differences between using and not using tape drive allocation.

Table 8–1: Tape Drive Allocation

Tape Drive Allocation	No Tape Drive Allocation
You must make an entry in the n–CONFIG.CMD file to use tape drive allocation.	No entry required in the n–CONFIG.CMD file.
Users can use labeled and unlabeled tapes.	Users can use only unlabeled tapes. This means that all tapes, whether they contain labels or not, are treated as unlabeled.
Users cannot give the ASSIGN and DEASSIGN commands for tape drives, but must give the MOUNT and DISMOUNT commands.	Users can give the ASSIGN, TMOUNT, and DEASSIGN commands for tape drives and cannot use the MOUNT and DISMOUNT commands.
The operator should not use the UNLOAD button on tape drives, but should use the DISMOUNT command to OPR.	The operator may unload tapes using the UNLOAD button on the tape drive.

8.3.2 How to Enable/Disable Tape Drive Allocation

To use tape drive allocation enter the command

```
ENABLE TAPE-DRIVE ALLOCATION
```

in the n–CONFIG.CMD file.

You can disable tape drive allocation on a tape drive by using the SET TAPE–DRIVE MTn: UNAVAILABLE command.

8.3.3 Tape Mounting Policy

Occasionally, you may mount a tape that the system cannot read. For example, the operator mounts a tape that has a density of 800 bits per inch (bits/in) on a drive that does not support this density. The system checks tapes only for labels; therefore, even though this tape may contain labels, the incorrect density prevents the system from recognizing them.

The system can be set up to immediately unload the tape and protect it from being accidentally erased, or it can treat the tape as unlabeled and continue processing.

If you do not want the system to classify these tapes as unlabeled, you can place the TAPE–RECOGNITION–ERRORS command in the n–CONFIG.CMD file with the appropriate argument. The format of this command follows:

```
                                REGARD-AS-UNLABELED
TAPE-RECOGNITION-ERRORS
                                UNLOAD
```

The system uses REGARD–AS–UNLABELED if no entry is made in the n–CONFIG.CMD file.

8.4 Tape Labeling

This section describes what tape labels are and how, as system manager, you can initiate using them.

Magnetic tape labels are records that are interspersed with user–defined data on a tape. They are informational records that describe the user data in a standard fashion that is recognized by many computer systems.

The TOPS–20 tape labeling system allows you to read and write tape labels that conform to ANSI (American National Standards Institute) and DEC standards. The tape labeling system also allows you to read tapes that are labeled according to IBM labeling standards.

Tape labeling is an option. You can start or continue to run your system using unlabeled tapes. If you have hundreds of unlabeled tapes at your installation, you may decide not to convert entirely to a labeled shop. Instead, you may have a combination of labeled and unlabeled tapes.

Sections 8.4.1 through 8.4.3 describe the advantages of using tape labels and how to set up your system to begin labeling tapes. The *TOPS–20 Tape Processing* manual provides a complete description of the ANSI, DEC, and

IBM standard label formats and how to use them. It also describes the interface between the operator and magnetic tapes and the user and magnetic tapes.

8.4.1 Why Tape Labels?

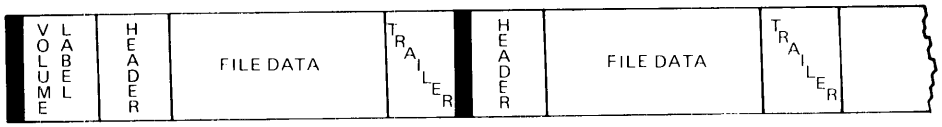
An unlabeled tape has a gummed label on the outside of the reel that identifies the contents of the tape. When the operator selects, mounts, and types the identity of an unlabeled tape, the system assumes that the mounted tape is the one that the user (or job) requested. No checking is performed by the system.

A labeled tape, however, contains standardized information on the tape itself that identifies and describes the data on the tape. This internal label information is in addition to the gummed label on the outside of the reel. With labeled tapes, the operator selects a tape (by looking at the outside gummed label), mounts the tape on any available drive, but does not type in any identifying information at the terminal. When a user issues a MOUNT request for a labeled tape that is already mounted, the system automatically locates the drive containing the tape requested, and checks to ensure that the correct tape has been mounted. This facility for automatically locating and checking tapes is described further below.

A labeled tape consists of a volume label group, followed by one or more files. (A volume is a reel of magnetic tape.) The volume label group is a set of one or more records at the beginning of the tape. It contains a volume identifier, commonly referred to as a VOLID, and other identifying information. (See Figure 8–1.) You, as system manager, select the VOLIDs to be used at your installation. The VOLID is a name containing from one to six alphanumeric characters. A user requests access to a specific volume by specifying its VOLID to the system.

Each file on a labeled tape contains a file header label group, the file data (written by the user program), and a file trailer label group. (See Figure 8–1.) Optionally, the file can contain user labels, whose contents are specified and examined only by user programs. Volume labels, header labels, trailer labels, and user labels are described in the *TOPS–20 Tape Processing* manual.

Figure 8–1: Organization of Labeled Tapes



MR-S-565-80

Because every labeled tape contains a unique identifier, or VOLID, the system can read this VOLID and ensure that the correct tape has been mounted. This automatic checking improves the reliability of your tape system. It eliminates the possibility of an operator mounting the wrong tape.

Also, some or all of your tape drives can be set to automatically recognize tape volumes as they are mounted. This process is called automatic volume recognition (AVR). Setting AVR means that after the operator mounts a tape, the system automatically reads the first record and inspects it for label information. If the tape contains no labels, the system classifies it as unlabeled and the operator must key in a volume identifier for the tape. If a request for the tape is pending, the system readies the tape for use by the requesting job. If a request for the tape is not pending, the system stores the VOLID in a table and waits for a request. Therefore, automatic volume recognition provides the following benefits.

- The operator does not have to type tape-identifying information to the system when mounting a labeled tape.
- It provides a faster connection between a user's job and the tape requested.
- The operator can mount a tape long before it is needed. When a job requests the tape, the system locates the drive that contains the requested tape and readies it for use.

Tape labels also improve the security of your tape system. DEC-standard labels identify the owner, as well as the volume, in the volume label and each file label. These labels protect a tape from being inadvertently written on and valuable data destroyed by a user who does not have the appropriate access rights to the tape or its files.

In addition to the added reliability, security, and volume recognition, labeled tapes provide you with a means of interchanging tapes between DECSYSTEM–20's and other computers. This interchange capability extends the mobility of data between different systems. You can write ANSI- and DEC-standard labeled tapes and mount these tapes on other systems using ANSI- or DEC-standard labels, and vice versa. You can mount a labeled tape that was written on an IBM system (in EBCDIC) and read it on a DECSYSTEM–20.

Finally, if you are using the TOPS–20 tape drive allocation facility, you can charge users for their tape usage. Note that you must use tape drive allocation with labeled tapes, but you can use it with unlabeled tapes also. The acounting usage file contains entries for all tape-mount requests. The *TOPS–20 USAGE File Specification* describes the formats of these entries and how they are used in reports and billing.

8.4.2 Setting Up the System to Use Tape Labels

To use tape labels, you must have at least one tape drive that is 9-track and has the capability of using tapes at a density of 800, 1600, or 6250 bits per inch (bits/in). There is no restriction on the number of these drives you use. The TOPS–20 Tape Labeling system can be used with as many drives as are allowed for your system.

Also, you must enter the ENABLE TAPE–DRIVE–ALLOCATION command in the n–CONFIG.CMD file. The *TOPS–20 Software Installation Guide* describes the format of this command and how to enter it into the n–CONFIG.CMD file at the time you install the system. If you do not enter this command during software installation, you can edit the n–CONFIG.CMD file at a later date. However, if you edit the file at a later date, you must shut the system down and bring it back up again to process the commands in the n–CONFIG.CMD file.

8.4.3 Initializing Tapes and Drives to Use Labels

Tapes must be initialized before they can contain labels. An initialized tape contains a volume label set followed by an empty file. The operator issues commands to OPR to initialize tapes for use by the TOPS–20 Tape Labeling system. All the necessary volume labels are then created on a tape. (Refer to the *TOPS–20 Operator's Guide* for a description of using OPR to initialize tapes.)

You should initialize as many scratch tapes as you will need to store your system's data before users start issuing MOUNT requests for tapes. Then, when a user issues a MOUNT command without specifying a VOLID, the operator mounts an initialized scratch tape of the appropriate label type. TOPS–20 then readies (loads) the tape for write access by the user program.

In addition to initializing tapes, you can set some or all of your tape drives to use the automatic volume recognition facility (AVR). As described earlier, AVR sets your tape drives to automatically recognize tape volumes as they are mounted. To turn on automatic volume recognition, have the operator enter the following command in the <SYSTEM>SYSTEM.CMD file:

```
ENABLE AUTOMATIC-VOLUME-RECOGNITION (FOR) object
```

Where object is either TAPE–DRIVE MTAn: or TAPE–DRIVES.

You can turn off AVR by entering the following command in the <SYSTEM>SYSTEM.CMD file:

```
DISABLE AUTOMATIC-VOLUME-RECOGNITION (FOR) object
```

These commands can be given to OPR at any time to enable or disable AVR for any or all drives on the system.

It is generally a good practice to enable AVR for all drives.

Chapter 9
System Problems/Crashes

Errors that require you to correct a problem in the file system seldom occur. However, if a problem of this nature arises, you can perform four classes of file system corrections. From the least to most severe, they are:

- Restore a single file in a directory
- Restore a single directory (other than <ROOT–DIRECTORY>)
- Restore <ROOT–DIRECTORY>
- Restore the entire file system

The *TOPS–20 Operator's Guide* provides all the necessary information for the operator to correct these types of problems. Sections 9.1 through 9.4 provide you with an overview of how these problems are solved.

9.1 Restoring a Single File

If you receive a request to restore a file for a user, you can use the following procedure.

1. Look in the binder that contains the listing of the DUMPER log files. (Chapter 7 describes creating DUMPER log files.)
2. Write down the file specification (including the structure and directory) to be restored, and the date and number of the tape containing the file. Be sure to indicate the destination structure and directory if one or both are different from the structure and directory from which the file was saved.
3. Submit a request to the operator to restore the file.

9.2 Restoring a Single Directory

If you receive a request to restore a directory, you can use the following procedure.

1. Determine the structure that contains the directory.
2. Make sure you have a copy of the files in this directory on a DUMPER tape. (Check the log files.)
3. Give the ^ECREATE command with the LIST subcommand. Write down the list of parameters, for example, the directory number, password, allocation, etc. You may need this information when you re-create the directory.
4. Give the ^ECREATE command with the KILL subcommand for the directory. (The *TOPS–20 Operator's Guide* provides additional procedures for deleting a single directory if the ^ECREATE command with the KILL subcommand is unsuccessful.)
5. Using your DUMPER backup tapes, first restore the files from the last FULL–INCREMENTAL DUMPER operation. Then, restore files from each INCREMENTAL tape until the time when the files were lost. Be sure the operator gives the CREATE command to DUMPER to restore the directory parameters.

If the directory contains unreproducible information and it is not backed up on tape, call Digital Software Services for assistance. It *may* be possible to reconstruct the directory without losing the valid information in it.

9.3 Restoring <ROOT–DIRECTORY>

Each structure has its own <ROOT–DIRECTORY> and a backup <ROOT–DIRECTORY> that is used by the system if the primary <ROOT–DIRECTORY> is bad. The directory <ROOT–DIRECTORY> contains a pointer to each first-level directory on a structure. (Chapter 5 illustrates how <ROOT–DIRECTORY> points to directories.) If <ROOT–DIRECTORY> is lost on the public structure, PS:, users cannot access any files in the system. If <ROOT–DIRECTORY> is lost on a non-PS: structure, users cannot access files on that structure.

You can tell that the <ROOT–DIRECTORY> on PS: is bad if the operator's console prints any one of the BUGHLTs listed in Table 9–1. When a BUGHLT appears on the console, the system stops. A BUGHLT appears in the form:

```
**********
* BUGHLT name AT dd-mm-yy hh:mm:ss
* JOB:n, USER: user-name
* ADDITIONAL DATA: data,data,data
**********
```

The lines beginning with JOB: or ADDITIONAL DATA: may not appear.

Table 9–1: <ROOT–DIRECTORY> BUGHLTS

BUGHLT	Meaning
BADROT	<ROOT–DIRECTORY> is invalid
FILIRD	The system could not initialize <ROOT–DIRECTORY>
FILMAP	The system could not map <ROOT–DIRECTORY> into memory
BADXT1	The index table is missing and cannot be created

<ROOT–DIRECTORY> may also be bad if you get the BOOT error ?FIL NOT FND. If this error appears, be sure you have mounted all devices correctly.

To recover from a bad <ROOT–DIRECTORY>, first determine which structure contains the bad directory. If the bad <ROOT–DIRECTORY> is on a structure other than PS:, you can run CHECKD with RECONSTRUCT ROOT–DIRECTORY and specify the proper structure. The *TOPS–20 Operator's Guide* details the procedure for determining the structure and using CHECKD for reconstructing <ROOT–DIRECTORY> on non-PS: structures.

If the bad <ROOT–DIRECTORY> is on PS:, you can instruct the system to use the backup PS:<ROOT–DIRECTORY> and rebuild this directory. Section 9.3.1 describes this procedure for the DECSYSTEM–2040, 2050, and 2060. Section 9.3.2 describes this procedure for the DECSYSTEM–2020.

NOTE

If your first attempt to rebuild <ROOT–DIRECTORY> fails, call your DIGITAL Field Service Representative. *NEVER* try to rebuild this directory twice on any structure.

9.3.1 Rebuilding PS:<ROOT–DIRECTORY> (2040, 2050, and 2060)

To rebuild <ROOT–DIRECTORY> on PS:, halt the central processor and perform Steps 7 through 18 and Steps 36 and 37 in Chapter 2 of the *TOPS–20 Software Installation Guide*. The steps are shown below for reference.

1. Type CTRL/backslash on the operator's console; the system prints PAR>.
2. Type SHUTDOWN and press the RETURN key; the system prints a few message lines.

```
PAR> SHUTDOWN(RET)
**HALTED**

%DECSYSTEM-20 NOT RUNNING
```

3. Mount System Floppy A in drive 0 (Step 7).
4. Mount System Floppy B in drive 1 (Step 8).
5. Mount the TOPS–20 Installation Tape on MTA0: (Step 9).

   If the TOPS–20 Installation Tape is not your most recent system backup tape, mount your PS: SYSTEM BACKUP TAPE that contains the monitor, TOPS–20 Command Processor, DLUSER, DLUSER data, DUMPER, and save sets containing <SYSTEM> and <SUBSYS>. (Refer to Section 3.2 for a description of special system directories.)
6. Place the front–end HALT switch in the ENABLE position (Step 10).
7. Set the front–end switch register to 000007 (octal) (Step 11).
8. Press the ENABLE and SWITCH REGISTER switches simultaneously (Step 12).

```
RSX-20F YB14-45 8:55 1-JAN-81

[SY0: REDIRECTED TO DX0:]
[DX0: MOUNTED]
[DX1: MOUNTED]
KLI -- VERSION V812-26 RUNNING
KLI -- ENTER DIALOG [NO,YES,EXIT,BOOT]?
KLI>
```

NOTE

The version and edit numbers in this manual may differ from the numbers printed on your console. The numbers on your console must be equal to or greater than the numbers in this manual.

9. Type YES and press the RETURN key (Step 13).

```
KLI>YES(RET)
KLI -- KL10 S/N: 2136, , MODEL B, 60 HERTZ
KLI -- KL10 HARDWARE ENVIRONMENT
       MOS MASTER OSCILLATOR
       EXTENDED ADDRESSING
       INTERNAL CHANNELS
       CACHE
KLI -- RELOAD MICROCODE [YES,VERIFY,FIX,NO]?
KLI>
```

10. Type YES and press the RETURN key (Step 14).

```
KLI>YES(RET)
KLI -- MICROCODE VERSION 275 LOADED
```

NOTE

If your system has cache memory, the following question appears previous to the question in Step 11. (Step 15.)

```
KLI -- RECONFIGURE CACHE (FILE,ALL,YES,NO)?
```

Type ALL and press the RETURN key (Step 16).

```
KLI>ALL(RET)
KLI -- ALL CACHES ENABLED
```

11. Type ALL and press the RETURN key (Step 17).

```
   KLI -- CONFIGURE KL MEMORY [FILE,ALL,REVERSE,FORCE,YES,NO]?
KLI>ALL(RET)
LOGICAL MEMORY CONFIGURATION
                          CONTROLLER
ADDRESS        SIZE   RQ0   RQ1   RQ2   RQ3   CONTYPE    INT
00000000       256K   00    01    00    01    MB20       4
KLI -- LOAD KL BOOTSTRAP [YES,NO,FILENAME]?
KLI>
```

If you have a DECSYSTEM–2040 or 2050, the output is slightly different. (Refer to the *TOPS–20 Software Installation Guide* for examples of the memory configuration output for DECSYSTEM–2040, 2050, and 2060.)

12. Type MTBOOT and press the RETURN key (Step 18).

```
KLI>MTBOOT(RET)
KLI -- CONFIGURATION FILE ALTERED
KLI -- WRITE CONFIGURATION FILE [YES,NO]?
KLI -- NO
BOOTSTRAP LOADED AND STARTED

BOOT V10.0(151)

MTBOOT>
```

13. Type /L and press the RETURN key (Step 36).

```
MTBOOT>/L(RET)
[BOOT: STARTING CHN:n DX20x:0 MICROCODE Vn(n)][OK]
[BOOT: LOADING RESIDENT MONITOR][OK]
MTBOOT>
```

NOTE

The message CHN:2DX20:0 MICROCODE VERSION 1(0) LOADED, VERIFIED, AND STARTED appears only if you have a DX20 Tape Controller.

14. Type /G143 and press the RETURN key (Step 37).

```
MTBOOT>/G143(RET)

[FOR ADDITIONAL INFORMATION TYPE "?" TO ANY OF THE
FOLLOWING QUESTIONS.]
DO YOU WANT TO REPLACE THE FILE SYSTEM ON THE SYSTEM
STRUCTURE?
```

NOTE

Read Step 15. carefully before answering this question.

15. Type N and press the RETURN key. You DO NOT want to clear all the information on the disk packs. If you want to retain all the information in the file system, always type N.

```
DO YOU WANT TO REPLACE THE FILE SYSTEM ON
THE SYSTEM STRUCTURE? N(RET)

[PS MOUNTED]
[BOOT: LOADING SWAPPABLE MONITOR, PASS1][OK]

RECONSTRUCT ROOT-DIRECTORY?
```

16. Type Y and press the RETURN key. This causes the backup copy of <ROOT– DIRECTORY> to be used.

```
RECONSTRUCT ROOT-DIRECTORY? Y(RET)

[RECONSTRUCTION PHASE 1 COMPLETED]

%%NO SETSPD

System restarting, wait...
ENTER CURRENT DATE AND TIME:
```

The system restarts and runs CHECKD to reconstruct the bit table. The bit table contains one bit for every page in the file system. If the bit is on, the page is available; if the bit is off, the page is in use.

17. Type the date and time and press the RETURN key. Type Y and press the RETURN key to confirm the date and time.

```
ENTER CURRENT DATE AND TIME: 1-JAN-81 0931(RET)

YOU HAVE ENTERED THURSDAY, 01-JANUARY-1981 9:31AM,
IS THIS CORRECT (Y,N) Y(RET)
WHY RELOAD?
```

18. Type OTHER and press the RETURN key.

```
WHY RELOAD? OTHER(RET)
[REBUILDING BIT TABLE]
```

NOTE

You should type a response to WHY RELOAD?, that reminds you of why you did this procedure. This response is stored in the <SYSTEM–ERROR>ERROR.SYS file. Refer to the *TOPS–20 KL10 Model B Installation Guide* for a complete list of valid abbreviations.

19. The system prints some standard messages, the output from CHECKD, RUNNING DDMP, and the output from SYSJOB and PTYCON.

```
[REBUILDING BIT TABLE]

[WORKING ON STRUCTURE - PS:]
```

output from CHECKD

```
RUNNING DDMP
```

output from SYSJOB and PTYCON

Refer to the *TOPS–20 Operator's Guide* for samples of the output from CHECKD, SYSJOB and PTYCON.

20. Log in as user OPERATOR.

```
TOPS-20 MEDIUM SYSTEM, TOPS-20 Monitor 4(2644)
@LOGIN (USER) OPERATOR (PASSWORD) ___(ACCOUNT) OPERATOR(RET)
Job 1 On TTY1 1-JAN-79 10:33:32
```

9.3.2 Rebuilding PS:<ROOT–DIRECTORY> (2020)

To rebuild PS:<ROOT–DIRECTORY> on the 2020, halt the central processor and perform Steps 23 through 37 in Chapter 2 of the *TOPS–20 Software Installation Guide*. The steps are shown below for reference.

1. Type CTRL/backslash on the operator's console; the system prints KS10>. If the KS10> prompt does not print, check the lock switch on the front panel to be sure it is not in the lock position.

2. Type SH for SHUTDOWN and press RETURN; a message appears to indicate the system has stopped.

```
KS10> SH(RET)
KS10>USR MOD

**HALTED**
```

3. Mount the TOPS–20 Installation Tape on MTA0:.

   If the TOPS–20 Installation Tape is not your most recent system backup tape, mount your PS:SYSTEM BACKUP TAPE that contains the KS10 microcode file, BOOTSTRAP routines, the monitor TOPS–20 Command Processor, DLUSER, DLUSER data, DUMPER, and the save sets containing <SYSTEM> and <SUBSYS>.

4. Type CTRL/backslash; the system prints the KS10> prompt.

   If your TOPS–20 Installation Tape or your PS:SYSTEM BACKUP TAPE cannot be mounted on tape drive 0, you must inform the system which drive you are using before you can boot the system from magnetic tape. To do this:

   - type the MS command to the KS10> prompt before the MT command (Step 5 below).
   - press RETURN to take the defaults for the questions >>UBA?, >>RHBASE?, >>TCU?, and >>DENS?
   - respond to the last question that appears, which is >>SLV?, by typing the physical tape drive number (this number appears on the flywheel on the front of the drive) on which your magnetic tape is mounted.
   - press RETURN. The system responds with the KS10 prompt.

   The following example shows this procedure:

```
KS10>MS(RET)
>>UBA?(RET)
>>RHBAS?(RET)
>>TCU?(RET)
>>DENS?(RET)
>>SLV?2(RET)

KS10>
```

   You can now boot the system from magnetic tape using the MT command. If you encounter any errors during this procedure (perhaps you mistype a character), you can easily recover by following Steps 29 through 37 in the *TOPS–20 Software Installation Guide.*

5. Type MT for magnetic tape boot and press RETURN; the system prints KS10>USR MOD and the magnetic tape boot prompt, MTBOOT>.

```
KS10>MT(RET)
KS10>USR MOD
MTBOOT>
```

6. Type /L and press RETURN; the system prints the MTBOOT> prompt again.

```
MTBOOT>/L(RET)
MTBOOT>
```

7. Type /G143 and press RETURN.

```
MTBOOT> /G143(RET)

[FOR ADDITIONAL INFORMATION TYPE "?" TO ANY
OF THE FOLLOWING QUESTIONS.]

DO YOU WANT TO REPLACE THE FILE SYSTEM ON
THE PUBLIC STRUCTURE?
```

NOTE

Read Step 8. carefully before answering this question. From now on you are not following the steps in the *TOPS–20 Software Installation Guide.*

8. Type N and press RETURN. You DO NOT want to clear all the information on the disk packs. If you want to retain all the information on the file system, always type N.

```
DO YOU WANT TO REPLACE THE FILE SYSTEM ON
THE PUBLIC STRUCTURE?  N(RET)

[PS MOUNTED]
RECONSTRUCT ROOT-DIRECTORY?
```

9. Type Y and press the RETURN key. This causes the backup copy of <ROOT–DIRECTORY> to be used.

```
RECONSTRUCT ROOT-DIRECTORY? Y(RET)

[RECONSTRUCTION PHASE 1 COMPLETED]

%%NO SETSPD

System restarting, wait...
ENTER CURRENT DATE AND TIME:
```

The system restarts and runs CHECKD to reconstruct the bit table. The bit table contains one bit for every page in the file system. If the bit is on, the page is available; if the bit is off, the page is in use.

10. Type the date and time and press RETURN. Type Y and press RETURN to confirm the date and time.

```
ENTER CURRENT DATE AND TIME: 22-JUN-79 0931(RET)

YOU HAVE ENTERED THURSDAY, 22-JUNE-1979 9:31AM,
 IS THIS CORRECT (Y,N) Y(RET)
WHY RELOAD?
```

11. Type RECONSTRUCT ROOT–DIRECTORY and press RETURN.

```
WHY RELOAD?  RECONSTRUCT ROOT-DIRECTORY(RET)
```

NOTE

You should type a response to WHY REBUILD?, for example, RECONSTRUCT or REBUILD, that reminds you of why you did this procedure. The response is stored in the PS:<SYSTEM> ERROR.SYS file.

12. The system prints some standard messages, the output from CHECKD, RUNNING DDMP, and the output from SYSJOB and PYTCON.

```
[REBUILDING BIT TABLE]
[WORKING ON STRUCTURE - PS:]
```

output from CHECKD

```
RUNNING DDMP
```

output from SYSJOB and PTYCON

Refer to the *TOPS–20 Operator's Guide* for samples of the output from CHECKD, SYSJOB and PTYCON.

13. Log in as user OPERATOR.

```
 TOPS-20 MEDIUM SYSTEM, TOPS-20 Monitor 4(2644)
@LOGIN (USER) OPERATOR (PASSWORD)___(ACCOUNT) OPERATOR(RET)
 Job 1 on TTY1 22-JUN-79 10:33:32
```

NOTE

The version and edit numbers in this manual may differ from the numbers printed on your console. The numbers on your console must be equal to or greater than the numbers in this manual.

9.4 Restoring the Entire File System

If you are still receiving random errors and cannot use the system, you may have to restore the entire file system on PS:. Before doing this, you should contact your software specialist to ensure that resorting to this procedure is necessary. The procedure requires shutting down the system and reinstalling the file system.

9.4.1 Re–creating the File Sytem On PS: (2040,2050,2060)

The following steps outline the procedure for restoring the file system on PS: on a DECSYSTEM–2040, 2050, or 2060:

1. Type CTRL/backslash to start the front–end command parser.

```
CTRL/backslash

PAR>
```

2. Type SHUTDOWN to stop the central processor; the system prints a few messages.

```
PAR>SHUTDOWN(RET)
**HALTED**

%DECSYSTEM-20 NOT RUNNING
```

3. Start at Step 9 in Chapter 2 of the *TOPS–20 KL10 Model B Installation Guide* and follow all the steps through Step 60.

   In Step 9, instead of mounting the TOPS–20 Installation Tape, mount your PS:SYSTEM BACKUP TAPE that contains the monitor, TOPS–20 Command Processor, DLUSER, DLUSER data, DUMPER, and save sets containg <SYSTEM> and <SUBSYS>.

4. After performing Step 60, restore your entire file system using DUMPER. First run DUMPER, then mount your first reel of the most recent full DUMPER tape and follow the procedures in the *TOPS–20 Operator's Guide.*

5. Re–create the front–end file system by following the directions in Chapter 4 of the *Software Installation Guide.*

6. Finally, restart the system by following the directions in Chapter 5 of the *Software Installation Guide.*

NOTE

Restore the incremental saves to obtain the most recently saved files. Before restoring each incremental tape, type DUMPER>CREATE to restore all the directories that were created since the last time you ran the DLUSER program.

9.4.2 Re–creating the File System on PS: (2020)

The following steps outline the procedure for restoring the file system on PS: on a DECSYSTEM–2020:

1. Type CTRL/backslash to start the microprocessor command parser.

```
CTRL/backslash

KS10>
```

2. Type SH for SHUTDOWN and press RETURN to halt the central processor; a message appears indicating the system has stopped.

```
KS10>SH(RET)
KS10>USR MOD
**HALTED**
```

3. Start at Step 28 in Chapter 2 of the *TOPS–20 Software Installation Guide* and follow all the steps through Step 60 with the following exception:

   In Step 28, instead of mounting the TOPS–20 Installation Tape, mount your PS:SYSTEM BACKUP TAPE that contains the KS10 microcode file, BOOTSTRAP routines, the monitor, TOPS–20 Command Processor, DLUSER, DLUSER data, DUMPER, and the save sets containing <SYSTEM> and <SUBSYS>.

4. After performing Step 62, restore your entire file system using DUMPER. First run DUMPER, then mount your first reel of the most recent DUMPER tape and follow the procedures in the *TOPS–20 Operator's Guide.*

5. Re-create the microprocessor file system by following the directions in Chapter 5 (running SMFILE) of the *TOPS–20 Software Installation Guide.*

6. Finally, restart the system by following the directions in Chapter 6 of the *TOPS–20 Software Installation Guide.*

NOTE

Restore the incremental saves to obtain the most recently saved files. Before restoring each incremental tape, type DUMPER>CREATE to restore all the directories that were created since the last time you ran the DLUSER program.

9.4.3 Re-creating Structures Other Than PS: (All Systems)

If, after your efforts to correct errors on a structure other than PS: (using the command RECONSTRUCT ROOT–DIRECTORY to CHECKD), you still cannot use the structure, you can re-create that structure and restore all the directories and files. You can restore structures other than PS: during timesharing.

To restore a non-PS: structure you must:

1. Give the SET STRUCTURE IGNORED command to the OPR program to prevent other users from mounting the structure while you are re-creating it.

2. Run CHECKD to create the structure.

3. Run DUMPER using the backup tapes for this structure. Give the CREATE and RESTORE commands to DUMPER to restore all the directories and files.

The *TOPS–20 Operator's Guide* details the procedures for running CHECKD, and DUMPER to re-create a structure.

9.5 Power Failures

Unfortunately, power failures and brown-outs sometimes occur at installations. You should be aware of the immediate steps to perform and transmit this information to your operations people. The kind of attention you should direct to this type of problem depends on the type of outage you have. These steps can protect your system from physical damage and perhaps unnecessary loss of files.

If your system experiences a total power failure, you should:

1. Immediately power-off all components of the system.
2. Inform your DIGITAL field service representative as to when you expect to resume power. The field service representative may request to be present while you bring your system back up.

If your system experiences a short instance of a power failure or brown-out, the system may recover on its own. You may not have any problems and usually, all your data remains intact. If you notice a problem, call your field service representative.

9.6 Remote Diagnostic Link (KLINIK)

You may occasionally have a problem with your system and cannot determine the cause. The remote diagnostic link, available on all systems, allows a DIGITAL Field Service engineer to access your system from a remote location and run diagnostic programs. This capability is called KLINIK. The DIGITAL engineer accesses KLINIK through a terminal and telephone line at the DIGITAL Service Center. The *TOPS–20 Operator's Guide* describes when and how to use the KLINIK capability.

Chapter 10
System Performance

The configurations of systems running TOPS–20 and the mix of jobs on these systems vary from installation to installation. TOPS–20 is designed to provide better response to interactive users than to provide higher throughput to computational tasks. On systems with a typical mix of interactive and batch jobs, this design provides both good response and adequate throughput. Therefore, if your system has many timesharing users and an average amount of batch processing jobs, you will find that your system's response is good and most users are satisfied.

If you have a mix of jobs or a configuration different from a typical system, you may want to change the response of your system to provide better service to specific users. TOPS–20 provides several tuning mechanisms that allow you to experiment with and change the behavior of your system. One mechanism allows you to favor classes of users by allocating each class a specified percentage of the CPU. Another mechanism allows you to favor computational jobs over interactive jobs. A third mechanism allows you to disable features that normally provide better performance, but that may not be applicable at your installation.

This chapter describes the mechanisms for tuning your system's response and provides guidelines for using these mechanisms. Because the response of your system can be felt during actual use only, you can expect system tuning to be an experimental and iterative process. By analyzing the statistics from the WATCH program and by gathering inputs from your users, you can determine the best way to tune your system.

The tuning mechanisms include:

- The class scheduler, which allows you to divide the central processor (CPU) time among groups of users
- Assigning low priority to batch jobs for installations that do not use the class scheduler

- Bias controls, which allow you to favor either interactive or compute-bound jobs on your system
- The program name cache for improving the startup time of frequently used programs
- Reinitializing disk packs in heavily used structures for improving file processing time

Each mechanism can be used independently. Unless otherwise noted, there is no interrelationship among them.

10.1 The Class Scheduler

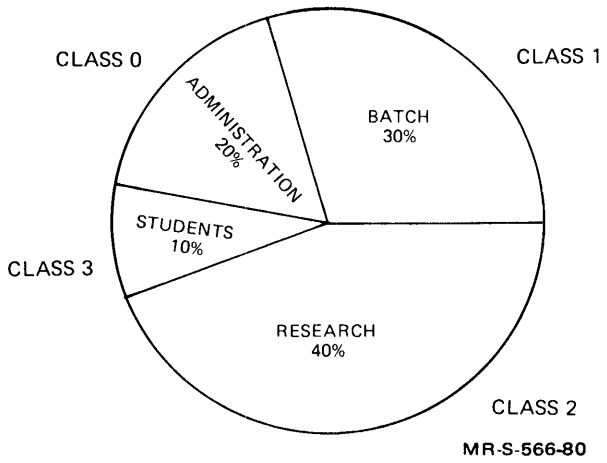
Occasionally, certain jobs monopolize the central processor time while other more critical jobs wait for the CPU. You may want to control the amount of CPU time a job receives. You can use the class scheduler to provide an even distribution of CPU time or to provide more CPU time to critical jobs on the system. Some system managers may want to allocate a larger amount of processor time to special users than to other system users. Sections 10.1.1 through 10.1.9 describe:

- an overview of the class scheduler
- who should use the class scheduler
- how to begin using the class scheduler
- turning on the class scheduler
- changing class percentages
- disabling the class scheduler
- getting information about class scheduler status
- a sample session using class scheduler commands
- an alternative to using the class scheduler with accounts.

10.1.1 Overview

The class scheduler allows you to allocate percentages of the central processor's time to individual classes of users. Each job in a class receives a portion of the class percentage. Therefore, by using the class scheduler, you can provide a consistent service to predefined groups of users.

The diagram below illustrates the concept of classes and percentages of CPU time allocated to each class. Note that you can set up a class to include all batch jobs. This allows you to control batch jobs separately from timesharing jobs. The batch class is given a high percentage or a low percentage. Now, each time a user submits a batch job, the scheduler uses the batch class and not the user's class. Section 10.1.4, Step 5., describes how to create a batch class.



You can define class memberships by using the TOPS–20 accounting facility or by writing your own access control program. Section 10.1.9 provides an overview of using an access control program to define classes. The following description applies to using the class scheduler with the accounting facility.

Using the accounting facility, you can associate a class number with each account on the system. Each account has only one class associated with it. Therefore, only a user who has access to more than one account can have access to more than one class. The user changes classes by changing accounts. To use this method, you must enable account validation so that users are required to use a valid account. Section 10.1.4 describes the accounting file entries.

The scheduler periodically computes the time used by classes and individual jobs within a class. The scheduler's first concern is that a class receives its share of the CPU time. The scheduler gives a greater percentage of time to the class that is the furthest away from its target (the total percentage allocated to the class). The scheduler's second concern is that each job within a class receives its fair share. Periodically, the scheduler computes the average amount of CPU time that a job has accrued. This quantity determines the job's priority within the class. A job that is requesting the CPU and is farthest way from its fair share within the class receives a greater percentage of time within that class.

Generally, the CPU has unused time. This unused time is called windfall. Windfall occurs if one or more classes has no logged-in jobs, or if one or more classes has an insufficient demand for the CPU to use all of its class percentage.

You can handle windfall in one of two ways; allocate it or withhold it. Allocating windfall means that the excess CPU time is awarded proportionately to the active classes. (In this discussion, the term active class refers to a class that has logged-in users requesting CPU time.) Higher percentage classes receive proportionately more of the available windfall than lower percentage classes. This means classes can receive slightly more than their allowed percentages when there is windfall available. Note that windfall is distributed proportionately to each class and not to each job within a class.

Withholding windfall means that the excess CPU time is idle time and it is not distributed to the active classes. It also means that classes do not receive more than the percentage allowed for them. Usually, it is better to allocate windfall, and not withhold it so you don't throw away valuable computer time.

10.1.2 Who Should Use The Class Scheduler?

As mentioned earlier, the class scheduler allows you to divide the user community into defined classes and allocate a percentage of CPU time to each class. This intended use may not be beneficial or practical for some systems. Tradeoffs do exist and some installations do not require class scheduling. Therefore, read the following paragraphs and determine if your system needs this type of control. Several instances that can benefit from using the class scheduler are:

- You can elect to sell portions of CPU time. For example, customer X can purchase some percentage of computer time at a proportional cost. You should, in this case, invite customer X to your installation and provide an environment that simulates the kind of response this customer can expect at this percentage. Because it is impossible to create the environment exactly as the customer will experience it at all times, customer X may decide later to change the purchase to a different percentage.
- If you have a natural division among the users of the system, you can divide these users into classes, then distribute CPU time to these classes with respect to their importance on the system. For example, in an educational environment, you may have administrative users (doing payroll, grades, etc.), faculty users (perhaps working on a grant), and student users (who are computer science majors). Using the class scheduler, you can establish three classes and distribute the CPU time according to the importance or pay rate of each class.
- If you wish to give preference to batch jobs, you can use the class scheduler to give the batch class a high percentage of the CPU.

Conversely, you may not want to use the class scheduler for some of the following reasons:

- If you have not realized a need in the past for class scheduling, or if you have a new system and do not see an immediate requirement for class scheduling, you should not set it up.
- Small systems with 256K or less of memory may experience an increase in swapping. This additional overhead may offset any advantage gained by using the class scheduler.
- In addition to other tasks, the scheduler constantly maintains usage values for each class and job. Because of this extra overhead, the overall system throughput decreases. Therefore, use the class scheduler if allocating percentages to individually defined classes outweighs the throughput depreciation.

- To give batch jobs a low priority on the system, you do not have to use the class scheduler. Section 10.2 describes placing the BATCH–BACKGROUND command in the n-CONFIG.CMD file to place all batch jobs in a low priority (or background) queue.

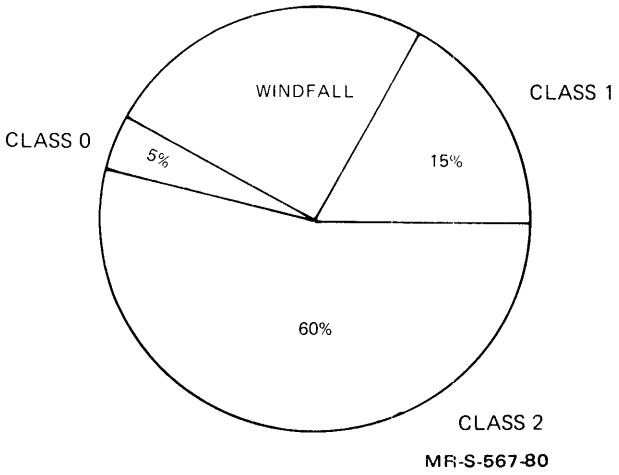
10.1.3 How To Begin Using The Class Scheduler

If you elect to use the class scheduler, first divide the system users into classes. Next, determine the amount of CPU time each class should receive. Percentages given to classes are in units of 5, that is, 5%, 10%, 15%, ...100%. The sum of the percentages given to all classes cannot exceed 100 percent. The result of these two steps depends on the reason you elected to use the class scheduler, and how you plan to use it at your installation. Because of this system dependency, step-by-step procedures for selecting classes and allocating the CPU cannot be given. The following discussion provides you with guidelines for setting up the class scheduler to meet your system's needs.

- Start with as few classes as possible, three or perhaps four. The class scheduler allows eight. Later, you can divide classes further, if necessary.

- Determine the number of logged-in users you expect in each class. Be sure that you do not overload a class, making it impossible to give a sufficient percentage of the CPU to all the jobs in the class. Also, you can limit the number of users in a class, but you cannot limit the number of jobs. For example, the SUBMIT command creates an additional job. Therefore, consider the type of work that users in a class perform.

- Estimate the percentage of CPU time (or time purchased) each class should receive. This is a difficult step; however, you can experiment with and later alter the percentages you choose. (Section 10.1.5 describes how you can change class percentages.) If a class consists of a large number of users who are generally logged in at the same time, be sure to give this class sufficient CPU time. For example, using the diagram below, class 2 has 60 members. It also has 60 percent of the CPU. This means that if approximately 60 jobs in class 2 are demanding the CPU, each job will receive approximately 1 percent. (It may be slightly greater than 1 percent if you allocate windfall.) This also means that on a per-job basis, users in class 2, with the higher percentage, may not get as much of the CPU as users in class 0 or class 1.



- The above diagram illustrates that you can allocate less than 100 percent of the CPU to classes. However, the total percentage for all classes cannot exceed 100 percent.
- The system uses class 0 as a default for any account that does not have a valid class assigned to it. Therefore, you can divide a portion of the system into well-defined classes and have all other users receive the percentage given to the default class.
- You can set up the class scheduler so that one class, perhaps the default class, receives only windfall. For example, suppose you allow some users to log into an account that is not yet assigned a class. These users are assigned to the default class 0. Or, suppose several users log in infrequently, read mail, and perform little computing. These users may log into an account that is not assigned a class (and, therefore, are assigned the default class, 0), or an account that is associated with class 0. You subsequently assign a zero percentage to class 0. These users may receive enough windfall to get their job done. However, they are not allocated CPU time and can have periods of extremely slow or no response. You may find, after experimenting with this type of procedure, that it is better to associate each account with a class and give the class a very low percentage of the CPU.
- Situations may arise that affect the scheduler's ability to give a class its percentage of the CPU. For example,the members of a class cannot use the amount of CPU time allocated to the class. Also, a situation may arise where the demands of the class exceed the percentage of CPU time given the class and windfall is available but not allocated. In either case, you should reevaluate both the percentages given to classes, and whether or not you want to continue withholding or allocating windfall.

10.1.4 Procedures To Turn On The Class Scheduler

After dividing users into classes and estimating the amount of CPU time, follow these steps to set up classes and turn on the class scheduler.

STEP PROCEDURE

1. Edit the PS:<ACCOUNTS>ACCOUNTS.CMD file (and the subaccount files, if any, that contain all your accounting data). Follow the procedure in Chapter 6, Creating Accounts, for modifying or creating each account with the /CLASS:n switch. For example,

```
ACCOUNT COMP1/CLASS:1
USERS KOHN,HOLLAND,MILLER
```

means that when users Kohn, Holland, and Miller log into or change their account to COMP1, they are placed in class 1. Each account has only one class associated with it. Therefore, only the user who has access to more than one account can have access to more than one class. A job can be in only one class at any one time.

2. Run the ACTGEN program. Give the TAKE command and specify the <ACCOUNTS>ACCOUNTS.CMD file (or the file containing the accounting data).

3. When ACTGEN returns its prompt, give the INSTALL command. This command updates the ACCOUNTS–TABLE.BIN file that the system uses to validate accounts.

4. Edit the n–CONFIG.CMD file to include the CREATE commands that specify the classes. Be sure account validation is enabled. That is, check that the DISABLE ACCOUNT VALIDATION command is *NOT* in the file. (The system default is ENABLED.) If account validation is disabled, users can use any account and therefore any class they choose. Enter the following commands in the order shown below.

   The CREATE command defines the class number and the percentage of CPU time the class should receive. For example,

```
CREATE 0 .40
```

means that the default class, 0, should receive 40 percent of the CPU. Enter the CREATE command for each of the classes that you defined in the ACCOUNTS.CMD file. For example,

```
CREATE 0 .40
CREATE 1 .20
CREATE 2 .30
```

NOTE

Remember that class 0 is the default class. Any user whose account (this includes subaccounts) is not associated with a specific class is placed in class 0.

5. Next, if you have decided to place all batch jobs in a separate class, enter the BATCH–CLASS command. For example,

```
BATCH-CLASS 2
```

means that all batch jobs will be placed in class 2 regardless of the user's associated class. You must use the CREATE command to define the percentage of CPU time that the batch class should receive. Note that timesharing users can be in the same class as well, if the account they use is associated with that class. The CREATE command in step 4. defines class 2 to receive 30 percent of the processor. If you do not place a BATCH command in the n–CONFIG.CMD file, the scheduler treats all batch jobs the same as timesharing jobs.

6. Finally, enter the ENABLE CLASS–SCHEDULING command. Be sure this is the last command entered. If you reverse the order, that is, you enter ENABLE CLASS–SCHEDULING before the CREATE commands and BATCH command, all users will receive zero percent of the CPU.

The ENABLE CLASS–SCHEDULING command turns on the class scheduler at the next system reload. It also specifies if you are using accounting or an access control program (policy program) for class scheduling and if you are allocating or withholding the windfall. The format of this command is:

```
ENABLE CLASS-SCHEDULING ACCOUNTS          WITHHELD
                        POLICY-PROGRAM    ALLOCATED
```

For example,

```
ENABLE CLASS-SCHEDULING ACCOUNTS ALLOCATED
```

means turn on the class scheduler, use the accounting method, and allocate the windfall. Always allocate the windfall. Do not withhold windfall unless you have a very good reason to do so, and you are sure that your class scheme works.

The entry,

```
ENABLE CLASS-SCHEDULING POLICY-PROGRAM ALLOCATED
```

means turn on the class scheduler, use the policy (access control) program, and allocate the windfall.

7. At the next system reload (the system is brought down and back up again) the new commands in the n–CONFIG.CMD file take effect.

10.1.5 Changing Class Percentages During Timesharing

During timesharing, you can change the percentage that a class receives by using the SET command to OPR. This change lasts until either the next system reload, or you make another change using the SET command. The format of the SET command appears:

```
SET SCHEDULER CLASS (number) m (to percent) n
```

n can be from 0–99 inclusive. Note that the decimal point required in the CREATE command is not allowed in this command.

To make the percentage that a class receives permanent, edit the CREATE commands in the n–CONFIG.CMD file. For example, you can edit the commands

```
CREATE 0 .60
CREATE 1 .30
CREATE 2 .10
```

to

```
CREATE 0 .10
CREATE 1 .05
CREATE 2 .80
```

Next, reload the system to process the commands in the n–CONFIG.CMD file. The classes that received a low percentage before you made the change now receive a higher percentage of CPU time.

Changing class percentages may be useful if you decide that users in one or two classes should receive a greater percentage of CPU time than other classes during the day. However, these high-percentage classes do not require this time during the evening shift.

10.1.6 Disabling the Class Scheduler During Timesharing

You can turn the class scheduler off and back on during timesharing by using the DISABLE and ENABLE commands to OPR. The class scheduler remembers the class percentages that were in effect before the DISABLE command was given. For example, suppose you use the SET command to change the percentage that class 1 should receive from 05 to 15 percent. Then, you give the DISABLE CLASS–SCHEDULER command, and later give the ENABLE CLASS–SCHEDULER command. The class scheduler uses 15 percent for class 1. If you reload the system after disabling the class scheduler, the class scheduler is enabled again and uses the percentages given in the n–CONFIG.CMD file.

10.1.7 Getting Information About Class Scheduler Status

Several TOPS–20 commands provide information about different class scheduler statistics.

The INFORMATION (ABOUT) SYSTEM–STATUS command informs you if:

- the class scheduler is enabled or disabled
- the accounting method or an access control program is being used
- windfall is allocated or withheld
- the system's batch jobs are in a separate class.

For example,

```
@INFORMATION (ABOUT) SYSTEM-STATUS(RET)

 CLASS SCHEDULING BY ACCOUNTS ENABLED, WINDFALL ALLOCATED, BATCH CLASS 1,
```

The INFORMATION (ABOUT) MONITOR–STATISTICS command provides a table that shows each active class, its target use of the CPU, its current use of the CPU, and the load averages for that class. For example,

```
@INFORMATION (ABOUT) MONITOR-STATISTICS(RET)

Class Share   Use   Loads

    0  0.80  0.79    6.56   4.36   3.57
    1  0.15  0.21    5.46   1.38    .95
    2  0.05  0.00    0.00   0.00   0.00
```

The SYSTAT command outputs load averages for either the entire system or a specific class. When the class scheduler is disabled, these averages represent the load of the entire system. For example,

```
@SYSTAT(RET)
 Wed 11-Jul-79 10:17:09  Up 11:38:12
 52+16 Jobs   Load av  5.30  4.03  4.86
```

The last three numbers following Load av indicate the average number of runnable processes over a period of one minute, five minutes, and fifteen minutes. These numbers start at zero. The higher the numbers the longer a job has to wait for CPU time. Using the example, over a fifteen minute period, a given job demanding the CPU waits approximately 4.86 times longer to run than it would if it were the only job running on the system.

When the class scheduler is enabled, these load averages represent the status of the job doing the SYSTAT command and not the entire system. For example,

```
@SYSTAT(RET)
 Wed 11-Jul-79 10:28:07  Up  11:49:12
 52+10 Jobs   Load av (class 1)    1.79   2.36   2.89
```

The last three numbers following Load av indicate the load averages of the class that the job giving the SYSTAT command is in. The SYSTAT command with the CLASS argument provides a breakdown of each job on the system. This breakdown includes the class each job is in, the average share of the class percentage that this job can receive, and how much CPU time the job is currently using.

The *TOPS–20 Commands Reference Manual* describes these commands in detail.

10.1.8 A Sample Session

The following examples show:

- The ACCOUNTS.CMD file after associating classes with accounts.
- The procedures that you follow after editing all your accounting files.
- A sample of the class scheduling commands that are placed in the 4-CONFIG.CMD file.

```
!The <ACCOUNTS>ACCOUNTS.CMD file has been edited.

$ TYPE (FILE) PS:<ACCOUNTS>ACCOUNTS.CMD(RET)

ACCOUNT MYBANK/CLASS:2
USERS BLOUNT,KONEN,ENGEL

ACCOUNT TRUST/CLASS:1
USERS BRAITHWAITE,HURLEY,HALL,CRISS

ACCOUNT OVERHEAD
USERS SAMBERG,BERKOWITZ,TAYLOR

ACCOUNT PROG/CLASS:3
USERS BLOUNT,KOHN,HOLLAND

$

!Notice that account OVERHEAD uses the default class, 0.

!Next, run the ACTGEN program.

$ RUN ACTGEN(RET)
ACTGEN> TAKE (COMMANDS FROM) PS:<ACCOUNTS>ACCOUNTS.CMD(RET)

!After ACTGEN finishes and returns its prompt, give the
!INSTALL command to create a new version of the
!<SYSTEM>ACCOUNTS-TABLE.BIN file

ACTGEN> INSTALL(RET)

!After ACTGEN finishes and returns its prompt, give the
!EXIT command

ACTGEN> EXIT(RET)
$
```

```
$TYPE SYSTEM:4-CONFIG.CMD<RET>

!Terminal Speeds
TERMINAL 1 SPEED 2400
TERMINAL 2 SPEED 9600
TERMINAL 3 SPEED 2400
TERMINAL 4 SPEED 2400
TERMINAL 5 SPEED 300
TERMINAL 6 SPEED 300
DEFINE SYS: PS:<SUBSYS>
DEFINE SYSTEM: PS:<SYSTEM>
DEFINE NEW: PS:<NEW>,SYS:
DEFINE OLD: PS:<OLD>,SYS:
DEFINE HLP: PS:<OLD>,SYS:
DEFINE HLP: SYS:
MAGTAPE 0 24
MAGTAPE 1 25
PRINTER 0 VFU SYS:NORMAL.VFU
PRINTER 0 LOWERCASE RAM SYS:LP96.RAM
TIMEZONE 6

!Commands for the class scheduler

CREATE 0 ,05
CREATE 1 ,45
CREATE 2 ,25
CREATE 3 ,15
BATCH 3
ENABLE CLASS-SCHEDULING ACCOUNTS ALLOCATED

$
```

To start the Class Scheduler, either reload the system, or give the ENABLE CLASS–SCHEDULER command to OPR. If you edited the n–CONFIG.CMD file to remove the DISABLE ACCOUNT VALIDATION command, you must reload the system so you can start validating accounts.

10.1.9 An Alternative To Using Accounts

Chapter 11 describes the access control program that is used to grant and restrict access to various system hardware and software. This same program can also include the appropriate monitor calls to handle class scheduler decisions. Some of the requests it can define are:

- classes
- class memberships
- the batch class
- class percentages
- changing class percentages
- windfall allocation per class

The monitor calls that are required in this program are described in the *TOPS–20 Monitor Calls Reference Manual*. DIGITAL distributes a sample Access Control Job program that includes class scheduler monitor calls.

This program is called ACJ.MEM and is located on the Distribution tape in the documentation area. The ACJ.MEM file provides you with a guide for writing your own access control program. Do not copy it and try to use it as is.

NOTE

You CANNOT run the access control program to implement class scheduler decisions at the same time you use the accounting method. You must use one or the other.

10.2 Scheduling Low Priority To Batch Jobs

The decision to favor batch jobs or to run them as background tasks depends on the type of batch environment you have at your installation. For example, if users submit batch jobs that are long and/or that do not require completion immediately, you can give batch jobs a low priority. Conversely, if batch jobs are the primary jobs on the system, you can give them a high priority.

Section 10.1 describes how to place batch jobs in either a high or low percentage class by including the BATCH n command in the n–CONFIG.CMD file. When a user submits a batch job, the scheduler uses the batch class and not the user's class.

You must use the class scheduler to give batch jobs a high priority. If you are not using the class scheduler, you can give batch jobs a low priority. To do this, enter the

```
BATCH-BACKGROUND
```

command in the n–CONFIG.CMD file. This command specifies that all batch jobs run on the lowest priority queue, also known as the background queue. This means that after processing all interactive jobs, the scheduler selects and runs batch jobs waiting in the queue. They receive left-over CPU time. You can enter the BATCH–BACKGROUND command into the n–CONFIG.CMD file. The command takes effect the next time you reload the system.

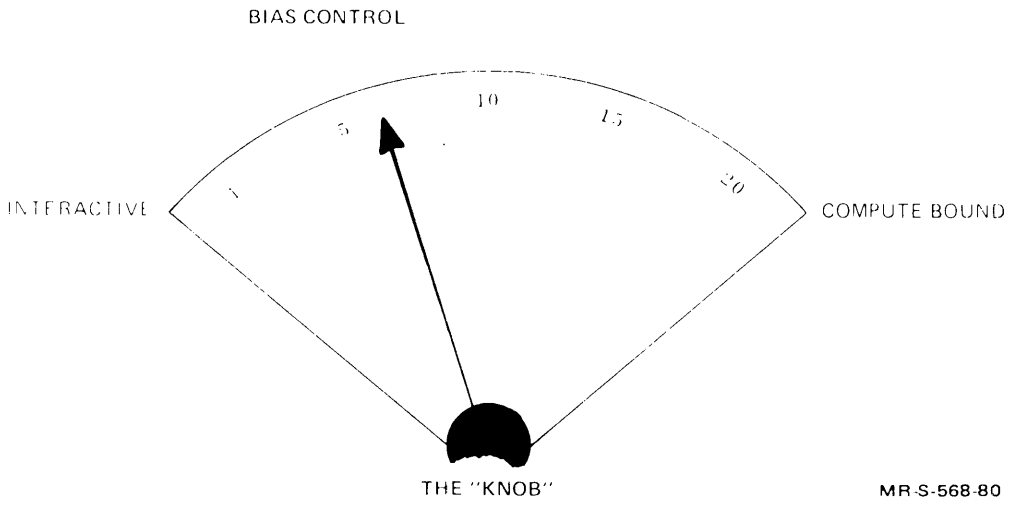
NOTE

The BATCH–BACKGROUND command is intended for those who are not using the class scheduler on their system but want to give the batch jobs a low priority. You should not use this command when you enable the class scheduler.

10.3 Favoring Interactive Versus Compute-Bound Programs

This section describes how you can influence the scheduler to favor either interactive or compute-bound programs. You do this by using the bias controls, which is analogous to turning a knob over a range of settings (from 1 to 20). When you select a lower number, the scheduler favors users running interactive programs. When you select a higher number, the scheduler favors users running computational programs. Figure 10–1 illustrates this concept.

Figure 10–1: Bias Control 'Knob'



NOTE

You can use bias controls with or without the class scheduler enabled. However, the effect of the bias control is less noticeable when used with the class scheduler.

After you install TOPS–20 software, the system uses the default bias control number 11. This setting distributes the scheduler's attention evenly to interactive and compute-bound programs. This setting causes the scheduler to favor programs in approximately the same manner as it did in releases of TOPS–20 previous to Version 4.

New users of TOPS–20 can use the default setting until they need to favor particular types of programs. Previous users of TOPS–20 may want to experiment with new control settings. For example, the bias controls can serve as a good tool for favoring different types of users at different times of the day. For instance, you can set the bias to a low number during the day to favor on-line users and to a high number during the evening to favor batch users The response you receive from your user community should determine the appropriateness of the selected settings.

Setting the bias toward interactive programs gives better response to the terminal user. Also, this setting may produce a higher system overhead, because the scheduler swaps jobs and switches from different tasks more often in its effort to favor interactive programs. Generally, setting the bias toward interactive programs is beneficial only if you have sufficient memory. Both the swapping rate and the scheduler overhead are likely to increase in small systems with too little memory. If your system has adequate memory, the scheduler overhead should be fairly constant over most of the bias settings.

Setting the bias toward computational programs should reduce system overhead and increase the total system throughput. However, the response to the terminal user may decrease slightly. In some cases, the improvement in system throughput more than compensates for the lessened response time, and users are more satisfied.

As you experiment with the bias settings, remember that setting the bias control to the extremes can prevent certain types of programs from running for long time periods.

After you select a bias control number that fits your system's operation, enter the BIAS CONTROL command in the n–CONFIG.CMD file or use the SET SCHEDULER BIAS–CONTROL command to the OPR program.

The format of the command in the n–CONFIG.CMD file is:

```
BIAS CONTROL m
```

The format of the command to OPR is:

```
SET SCHEDULER BIAS-CONTROL (TO) m
```

where m is the bias control number.

You can change the bias setting during timesharing by using the SET SCHEDULER BIAS–CONTROL command. However, when you reload the system, the bias setting in the n–CONFIG.CMD file takes effect. If you want your change to be permanent, edit the n–CONFIG.CMD file at a convenient time before you reload the system.

Any time you are unsure of the current setting of the bias controls, give the INFORMATION (ABOUT) SYSTEM–STATUS command to determine its setting.

10.4 Improving Program Startup Time

Some of the programs on your system are run frequently. This involves constant searching for the same file on disk, bringing the program pages into memory, and allocating swapping space when the program is swapped out of memory. By storing these programs in an easy-to-access area, some of the startup time is saved. To improve the startup time of the frequently used system programs, TOPS–20 keeps a program name cache.

The <SYSTEM>PROGRAM–NAME–CACHE.TXT file is placed in the directory PS:<SYSTEM> automatically at installation time, and contains a list of programs to be copied into the program name cache. These programs are:

SYS:PA1050.EXE
SYS:MACRO.EXE
SYS:EDIT.EXE
SYS:TV.EXE
SYS:LINK.EXE
SYSTEM:ERRMES.BIN

Each time you reload the system, the <SUBSYS>MAPPER.EXE program runs under the SYSJOB program at the operator's console. <SUBSYS>MAPPER.EXE reads the <SYSTEM>PROGRAM–NAME–CACHE.TXT file and loads the program name cache. Now, when requests are made for these programs, the system looks first in the program name cache to see if it can retrieve the required pages quickly.

To further improve system performance, you can edit the PROGRAM–NAME–CACHE.TXT file and add the filenames of your own frequently accessed programs. For example, if your system uses the FORTRAN language, you may want to add the files:

SYS:FORTRA.EXE
SYS:FOROTS.EXE
SYS:FORLIB.REL

Your list can contain the names of up to 16 executable files. Therefore, select the files to be placed in the program name cache carefully. You should consider only the executable files that are started frequently by a large number of users.

You can also add library files to the program name cache, for example, SYS:FORLIB.REL. However, these types of files use up swapping space. If you have too many or very large files, you may create a detrimental effect on your system's performance. The total library file pages that you cache should be no greater than 200 to 300. Give the VDIRECTORY command for the library files you are considering to cache and check the number of pages in each file.

If you edit the cache file, the revised file takes effect (MAPPER creates a new version of the cache) the next time you reload the system. Alternatively, you can create a new version of the cache immediately by entering the following commands at the operator's console.

```
^ESPEAK                          !talk to SYSJOB
KILL MAPPER                      !kill old version of cache
RUN SYS:MAPPER.EXE               !read new file and create
                                 !new version of cache
^Z                               !exit
```

10.5 Reinitializing Disk Packs

After many files are created, they are no longer contiguous on the structure (disk(s)). This scattering of files increases the time it takes to process them. You can decrease excessive processing time by reinitializing the disk packs in your heavily used structures. For example, the public structure may be a likely candidate for reinitialization. This procedure places files into a contiguous format and can be scheduled as part of a backup procedure. How often you reinitialize your packs depends on the work load of the system and whether you notice a difference in system performance after following this procedure.

Reinitializing disk packs requires that you dump all the directories and files to tape, re-create the structure (and, in the case of PS:, reinstall the system), and restore all the directories and files. Therefore, if you do not notice any appreciable difference in your system performance after doing this, don't schedule it on a regular basis, if at all.

The *TOPS–20 Operator's Guide* describes re-creating the public structure and other structures. Have the operator use this procedure for reinitializing disk packs. If possible, have the operator re-create the structure and restore the files to a different pack (or set of packs) from the structure that you dumped. This ensures that you do not lose your files should you have problems reading the tape back to disk. That is, you still have the original structure intact and can run DUMPER again to copy the files to another tape.

Chapter 11
Controlling Access To System Resources

The previous chapters deal with administrative policies for allocating resources. For example, Chapter 10 describes the policy decisions you can make regarding the scheduler, and Chapter 8 describes the policy decisions you can make regarding tape drive allocation and labeled tape support.

In addition, you can make policy decisions that govern the access to specific system resources. For instance, TOPS–20 allows a user to change the speed of a terminal, assign a device, log in at any time of day, mount a magnetic tape, and mount a disk structure. However, you may want to restrict or disallow use of some of these facilities. You may want only specified users at specified times of the day and, perhaps, at specified terminals, to use certain facilities. By using an access control mechanism, you can govern the access to many of your system's resources and services. You can reduce or prevent malicious access to system resources, and you have an additional means for collecting accounting or other information.

To use the access control mechanism, you must write an access control program that carries out your policy decisions. An access control program can control scheduling classes, the bias control, batch background queue, logging in, use of physical resources (tape drives, terminals, structures), and enabling capabilities. When a user requests a resource (like ASSIGN TTY34:), your program identifies the user, the user's controlling terminal, and the type of request being made. Your program can merely log this information in a file, or make a decision and tell the monitor to either grant or deny the request.

DIGITAL provides the necessary mechanisms (monitor calls) to implement a program at your installation. Your system programmer uses the appropriate monitor calls to write an access control program according to the requirements of your system. A sample access control program (ACJ.MEM) is distributed in the documentation area on the distribution tape. Your system programmer can use this program as a sample of the structure of an access control program and the parameters and decisions that can be controlled.

The following list defines the resources that an access control program can control and why you may want to control them.

Assign Devices

You can restrict users from assigning terminal lines or tape drives to their jobs. For example, if you have not enabled tape drive allocation (described in Chapter 8), you may still want to control which users are allowed to assign tape drives. Your program can also prevent one user from assigning all the available tape drives or terminals.

Enable Capabilities

You can allow users to enable their capabilities only in certain locations, or perhaps only at certain times of the day. For example, in a university environment, you may not want users enabling WHEEL capabilities in a terminal room used by students. In an environment where security is a major concern, you may want to allow a user to enable WHEEL capabilities only on a hard-copy terminal and only in a certain location. You can then collect the hard-copy output from the terminal. The access control program can also keep a file of the names of people who have enabled their capabilities.

Create Jobs

You can restrict the users who are able to write programs that create additional jobs via the CRJOB monitor call. You may want to limit this facility to certain applications only.

Allow Login

You can prevent users from logging in more than once, or permit a user to log in only at certain times of the day. For example, in a university environment, it may not be desirable to have students on the system during large production runs. Also, if you use your own accounting information (and not that provided with TOPS–20), you can use the access control program with the login function to recognize a user. In addition, the login function can be used to control the number of jobs that a user can create under PTYCON.

Create Inferior Processes (Forks)

You can prevent a user from creating more than a predefined number of processes. Also, you may want to charge users for using many inferior processes.

Set Terminal Baud Rate

You can control the input and output speed settings on all terminals. This control prevents users from changing the baud rate to a speed that is unsupported by the terminal, and as a result, rendering the terminal unusable until the operator resets the baud rate. You can also restrict the speed of a terminal to no more than a specified maximum, for example, 300 baud.

Logout

You can request the access control program to notify you or record information in an accounting file each time a user logs off the system. You may also want to keep track of the users who log out and are over their permanent disk page quota. The access control program can notify the operator that a migration-trim-run is needed for this directory to bring the directory back under its quota. If you use the login function with the logout function, you can give the user who is logging out information about time, resources, and perhaps money spent.

Set ENQ Quota

TOPS–20 allows enabled WHEELS to change the ENQ–DEQ quota. By using the ENQ quota function in your access control program, you can allow users other than WHEELS to change the ENQ–DEQ quota. (The *TOPS–20 Monitor Calls Reference Manual* describes the uses of ENQ–DEQ.)

Create Directory

You can prevent users from giving the BUILD or ^ECREATE command to create directories or change parameters. Or, you may simply request the access control program to notify you of the people who have used these commands. You may want the operator to police these directories and check the parameter changes.

Mount a Structure

You can control access to certain structures by allowing only a select group of users to give the MOUNT command for a particular structure(s). This facility is used in conjunction with regulated structures. (The *TOPS–20 Operator's Guide* describes REGULATED and NON–REGULATED structures.) Also, information about structure mounts can be recorded in accounting files.

Enter MDDT

You can disallow privileged users from entering MDDT mode. For example, during certain times of the day, you may not want enabled WHEELS looking at or fixing a problem in the monitor. You may also want to keep a record of who has used MDDT.

Class Assignment

You can prevent users from changing to unauthorized scheduling classes. The access control program determines the classes a job can use.

Set Class At Login

The access control program can set a user's class at log in. Your program can contain (or access a file that contains) the list of users and their associated class.

MT Access Request

You can have the access control program decide whether a user should be allowed to access a restricted labeled tape from a non–TOPS–20 system. For example, if a non–TOPS–20 labeled tape is mounted with a nonblank access code in the access field, you can have your program decide if this user can use this tape. (The *TOPS–20 Tape Processing Manual* describes the access fields on labeled tapes.)

ACCESS/CONNECT Request

The access control program can determine if an ACCESS or CONNECT request to a directory should succeed in cases where the request is denied by the monitor. For example, the TOPS–20 monitor allows an ACCESS or CONNECT request to succeed when appropriate criteria are met. These are:

- The requesting process has WHEEL or OPERATOR capabilities enabled.
- The target directory is in the same group as the job's "accessed" directory.
- The target structure is DOMESTIC and the target directory name matches the logged–in directory or the job.
- The correct password is specified.

If all of the above criteria fail, the monitor denies the request. The access control program can be called to approve or override the denial.

ATTACH Request

The access control program can prevent a user from attaching his terminal to another job. This function allows the access control program to control which terminals can attach to specific jobs.

Appendix A
The BUILD Command

The BUILD command allows nonprivileged users to create, change, and delete subdirectories. This description of how to use the BUILD command is provided here to allow you to distribute it to those users (or administrators) who are creating directories. It also enables these users to have the information they need without having to reference the *TOPS–20 System Manager's Guide* or the *TOPS–20 Operator's Guide.*

The structure of the BUILD command is similar to that of the ^ECREATE command that the operator uses to create directories. Therefore, any parameter that can be specified with the ^ECREATE command can be specified with the BUILD command. However, unlike ^ECREATE, the BUILD command does not require a user to have enabled capabilities to use it. All users who have been given the MAXIMUM–SUBDIRECTORIES (ALLOWED) with a parameter greater than zero in their directory can use the BUILD command. Users who do not have a positive value for this parameter in their directory and who attempt to use the BUILD command receive an error message.

Because the BUILD command is designed to be used for creating, changing, and deleting directories that are subordinate to another directory, users cannot give the BUILD command to delete or change parameters in their connected directory. This command can be used only for directories inferior to the connected directory.

IMPORTANT

The following descriptions of the BUILD command, the subcommands that can be used with BUILD, and the technical notes, should be read thoroughly before any attempt is made to create directories.

BUILD Command

Function

The BUILD command creates, changes, or deletes a directory that is subordinate to your connected directory.

Format

```
@BUILD (DIRECTORY NAME) <directory name>
[status]
@@subcommand
@@ .
@@ .
@
```

Where

<directory name>

is the name of the directory to be created, changed, or deleted. The directory name includes as a prefix the names of any directories superior to it. These names are separated by periods. For example, the name of a directory could be <MATH.SMITH>. The directory <MATH> is the directory to which you are now connected, and <MATH.SMITH> is the directory subordinate to your connected directory.

[status]

indicates whether the directory exists. The word [NEW] appears if the directory does not exist, and the word [OLD] appears if the directory does exist.

@@

indicates that you are entering subcommand mode.

subcommand

is a keyword chosen from the list on the following pages. Many of these keywords have default values that the system uses when you do not specify them.

BUILD Subcommands

ABORT

Ends the current BUILD command. If you are creating a directory and give the ABORT subcommand, the BUILD session ends, you are returned to the TOPS–20 command level, and no directory is created. If you are changing a directory and give this subcommand, the BUILD session ends, you are returned to the TOPS–20 command level, and any changes you made during this session are canceled. The directory is exactly the same as it was before you gave the BUILD command. If you are deleting a directory and give this subcommand, the BUILD session ends, you are returned to the TOPS–20 command level, and the directory is not deleted.

ABSOLUTE–ARPANET–SOCKETS (CAPABILITY)

Allows the owner of this directory to use absolute socket numbers when they are required in his programs. Your system must be running TOPS–20AN to use this capability. Also, you must have this capability yourself and be enabled at the time you are entering it into another user's directory.

ACCOUNT–DEFAULT (FOR LOGIN) account

Specifies that the given account is to be used as the default account if the owner of the directory logs in without specifying one. This account remains as the default until it is changed or until it is no longer valid. If this subcommand is not given, the owner of the directory must specify an account every time he logs in. Note that this command is useful only if a valid account is given. If an invalid account is given, the user receives an error message whenever he tries to log in using the default account.

ARPANET–WIZARD (CAPABILITY)

Gives the owner of this directory the ability to perform certain ARPA network privileged functions. These privileged functions are mentioned in the *TOPS–20AN Monitor Calls User's Guide.* Your system must be running TOPS–20AN to use this capability. Also, you must be an enabled ARPANET WIZARD to give this capability to another user.

CONFIDENTIAL (INFORMATION–ACCESS CAPABILITY)

Gives the owner of this directory the confidential information access capability. This capability allows the owner to obtain confidential information within the system via certain monitor calls. (Refer to Technical Note 13 in this appendix and to the *TOPS–20 Monitor Calls Reference Manual.*)

DEFAULT–FILE–PROTECTION (NUMBER) number

Specifies the file protection code to be used as the default if the user creates files in the directory without specifying a protection code. If this subcommand is not given, the system assumes a protection of 777700. Note that users can change the default file protection for their directories using the TOPS–20 SET command. (Refer to the *TOPS–20 User's Guide* for a description of the valid file protection codes.)

DIRECTORY–GROUP (NUMBER) number

Places this directory in the specified directory group. Users in this same user group can access this directory and its files (according to the protection codes set for both the directory and files) as group members. (Ask your system manager for a description of how to establish valid group relationships. Also, refer to the USER subcommand description in this list.)

ENQ–DEQ (CAPABILITY)

Gives the owner of this directory the ability to perform privileged ENQUEUE and DEQUEUE functions. (Refer to Technical Note 13 in this appendix and to the *TOPS–20 Monitor Calls Reference Manual.*)

FILES–ONLY

Specifies that the directory cannot be accessed via the LOGIN or ACCESS command. However, if you assign a password to the directory, users can access it via the CONNECT command. Access to a files-only directory is further governed by its assigned directory and file protection codes and by its membership in directory groups. (Refer to the PROTECTION and DIRECTORY–GROUP subcommands in this list and to the *TOPS–20 User's Guide.*)

GENERATIONS (TO KEEP) number

Specifies how many generations of each file are to be retained in the directory. Valid numbers are 0 to 15, where 0 means an infinite number. Users can set the number of generations to keep for individual files using the TOPS–20 SET command.

DEFAULT — 1

IPCF (CAPABILITY)

Gives the owner of this directory the ability to execute all privileged IPCF functions. (Refer to Technical Note 13 in this appendix, and to the *TOPS–20 Monitor Calls Reference Manual.*)

KILL (THIS DIRECTORY)

Deletes and expunges the directory and its files. You must confirm this subcommand with an extra carriage return. If this directory has directories inferior to it, however, the system does not allow you to delete it. You must first use the BUILD command with the KILL subcommand and delete the inferior directories, and then delete this directory. (Refer to Technical Note 20 in this appendix.)

LIST VERBOSE
FAST
NAME–ONLY

Outputs the parameters associated with the directory. The LIST VERBOSE subcommand lists all the subcommands that can be given to a directory, along with the parameters that have been assigned to this directory. The LIST FAST subcommand lists only those subcommands and parameters that have been given to this directory, either by default or assignment. LIST NAME–ONLY outputs the directory name.

DEFAULT — LIST FAST

MAINTENANCE (CAPABILITY)

Gives the owner of the directory the ability to execute certain functions that are needed to maintain the system. (Refer to Technical Note 13 in this appendix, and to the *TOPS–20 Monitor Calls Reference Manual.*)

MAXIMUM SUBDIRECTORIES (ALLOWED) number

Allows the owner of this new directory to create other directories that are inferior to it. The number given to this directory is subtracted from the number allowed for the immediately superior directory. If this subcommand is not given, the owner of the new directory cannot create any inferior directories. (Refer to Technical Note 14 in this appendix.)

NOT subcommand

Negates a previously given subcommand and its parameter, if one was given. The subcommands that can be used with NOT are:

NOT ABSOLUTE–ARPANET–SOCKETS
NOT ARPANET–WIZARD
NOT CONFIDENTIAL
NOT DIRECTORY–GROUP
NOT ENQ–DEQ
NOT FILES–ONLY
NOT IPCF
NOT KILL
NOT MAINTENANCE

NOT OPERATOR
NOT SUBDIRECTORY–USER–GROUP
NOT USER–GROUP
NOT WHEEL

If you need to negate a subcommand that is not listed above, enter the subcommand without specifying any parameter.

NUMBER (OF DIRECTORY) number

Assigns a number to the directory so the system can identify it. Because the system assigns a directory number, you need not use this subcommand. Note that you can specify a number only when you are creating a directory, not when you are changing one.

DEFAULT — The first available number that the system finds.

OPERATOR (CAPABILITY)

Gives the owner of this directory the ability to perform all the privileged commands that can be given to a user except for the ^EEDDT and ^EQUIT commands. (Refer to Technical Note 13 in this appendix.)

PASSWORD password

Gives the directory a password of 1 to 39 alphanumeric characters known only to you and the owner of the directory.

PERMANENT (DISK STORAGE PAGE LIMIT) number

Allocates the given number of pages of disk space to this directory. This amount is subtracted from the total disk space given to the immediately superior directory.

DEFAULT — 250 pages.

The default is not accepted if the immediately superior directory does not have 250 pages to allocate. (Refer to Technical Note 6 in this appendix.)

PROTECTION (OF DIRECTORY) number

Assigns a protection number to the directory. If this subcommand is not given, the system assumes a protection of 777700. Note that users can change the directory protection for directories by using the TOPS–20 SET command. (Refer to the *TOPS–20 User's Guide* for a description of the valid directory protection codes.)

REPEAT–LOGIN–MESSAGES

Causes all system messages to be printed on the user's terminal each time the user logs into this directory. This is useful when two or more users are sharing the same log-in directory. System messages are mail sent by privileged users, for example, the operator, to all system users. If you do not give this subcommand, the system prints only the messages created since the last time the user logged into this directory.

SUBDIRECTORY–USER–GROUP (ALLOWED) number

Allows the owner of this directory to establish group relationships among the directories he creates. He must use this number as the response to the USER–GROUP (NUMBER) subcommand for these directories. (Refer to Technical Note 15 in this appendix.)

USER–GROUP (NUMBER) number

Places the owner of this directory in the given user group. Remember to use only those group numbers that are in the superior directory's User Group List. The owner can then access any directory in the same directory group.

WHEEL (CAPABILITY)

Gives the owner of this directory the ability to perform all the privileged functions that can be given to a user. (Refer to Technical Note 13 in this appendix.)

WORKING (DISK STORAGE PAGE LIMIT) number

Allocates the given number of pages of disk space to this directory. This amount is subtracted from the total disk space given to the immediately superior directory.

DEFAULT — 250 pages.

The default is not accepted if the immediately superior directory does not have 250 pages to allocate. (Refer to Technical Note 6 in this appendix.)

BUILD Command Technical Notes

Technical notes 1 through 17 describe the procedures and considerations you use when creating directories. Technical notes 18 through 20 refer to the procedures and considerations for changing and deleting directories already built.

A subdirectory is just like any other directory. It can have an owner and a password; it can be a member of user and directory groups; it can have directory and default file protections; and it can have assigned working and permanent disk space. The only additional rights you have with this directory is the ability to delete it or change its parameters, (e.g., password, groups, and disk space quotas).

Creating New Directories

1. First, log into or give the CONNECT command to the directory for which you will be creating subdirectories.

2. Next, give the INFORMATION (ABOUT) DIRECTORY command for this directory. Several of the parameters that are shown in the output tell you the number of directories you can create (Maximum subdirectories allowed) and which numbers you can use if you want to place this directory in user groups (Subdirectory user groups allowed). For example, if you gave an INFORMATION (ABOUT) DIRECTORY command for a directory name <MATH>, the output might be similar to the following:

```
@INFORMATION (ABOUT) DIRECTORY <MATH>
Name PS:<MATH>
Password - None set
Working disk storage page limit 500
Permanent disk storage page limit 500
Number of directory 164
Account default for LOGIN UC.MATH
Maximum subdirectories allowed 15
Last LOGIN 29-Aug-77 08:21:45
User groups 2597
Subdirectory user groups allowed 2597,2598,2599
@
```

The directory <MATH> could be the directory that you always log into and store your files in, or a directory that was created by the operator for the sole purpose of having directories created under it. In the latter case, you would log into your own directory and issue the CONNECT command to <MATH>, giving its password.

NOTE

When you are giving an INFORMATION (ABOUT) DIRECTORY command for your connected directory, the password is not printed.

3. The next command you should give is the INFORMATION (ABOUT) DISK-USAGE. The first line of the output tells you how many pages of disk storage you have used and, therefore, how many pages of disk storage you have left to divide among the directories you build. Any disk space that you give to a directory is subtracted from your connected directory's (the directory that is superior to the one you are creating) disk space. The following example shows the output for the sample directory <MATH>.

```
@INFORMATION (ABOUT) DISK-USAGE (OF DIRECTORY)↵
 28 Pages assigned
 500 Working pages, 500 Permanent pages allowed
 2464 Pages free on PS:
```

The second line tells you that this directory was originally given 500 pages of working and 500 pages of permanent disk space. The first line tells you that you have used up 28 pages of this space, so you now have 472 pages left to create additional files in this directory or to assign to subdirectories.

After you are familiar with this information, you can begin creating your directories.

4. If you are creating directories for users, assign these people their user names. The user names should be as close as possible to the user's last names. This convention is very helpful when you give a SYSTAT command. You can easily identify who is using the system. Also, the system uses these names when recording the authors of files and sending MAIL to users.

5. Give the BUILD command and include the new directory name as the argument. For example, if you are connected to directory <MATH> and are creating a directory for the user J. Scheid, who has been given the user name SCHEID, the command line is:

```
@BUILD (DIRECTORY NAME) <MATH.SCHEID>↵
```

NOTE

> The directory names that you create cannot exceed 39 alphanumeric characters. The period that separates the levels of the directory name is counted as one of the 39 characters.

Because you are specifying this directory for the first time, the system responds with <NEW> and immediately places you in subcommand mode.

Entering Subcommands

6. To allow the owner of this directory to create files and perhaps directories of his own, you must give the directory a WORKING (DISK STORAGE PAGE LIMIT) and a PERMANENT (DISK STORAGE PAGE LIMIT). Take the

number of pages you have available in your connected directory (INFORMATION (ABOUT) DISK-USAGE, the allocation minus the used pages) and divide it according to how many directories you are going to create.

You should consider the requirements of the individuals who will be using these directories when estimating how much space they will need. Users who will be doing a considerable amount of debugging and sorting will require substantially more disk space than users who are creating and editing small files.

Remember that the total amount of disk space you allocate to the directories you create cannot exceed the total amount unused in the superior directory. You can use the default of 250 pages for working and permanent disk storage for your directories but be careful that this procedure does not deplete the superior directory's quota before you have created all the directories. For example, using the sample directory <MATH> that has 472 pages of available disk space, you could create only one subdirectory using the 250 page default.

7. Next, assign the directory a PASSWORD. The password can be 1 to 39 alphanumeric characters, known only to you and the owner of the directory. If you are creating a files-only directory, which users cannot access via the LOGIN or ACCESS command, you may or may not want to assign a password. (Refer to Technical Note 16 for more information about files-only directories.)

8. If you do not want to use the default (777700) for the directory and file protections, give the PROTECTION and DEFAULT FILE PROTECTION subcommands. (Refer to the *TOPS-20 User's Guide* for a description of the valid directory and file protections.) You can remind the owners of the directories you create that they can change their own directory and file protections using the TOPS-20 SET command.

9. Use the DIRECTORY-GROUP subcommand if you want to assign this directory to one or more directory groups. (You must make a separate entry for each group assigned.) Note that there are no constraints on the directory numbers you use. However, any user who is a user group member of a number you choose for a directory group will have access to that directory. Therefore, it is wise to choose directory group numbers corresponding to your user groups. If you are creating more than one level of directories, it is a good idea to place each directory you create in a directory group of which you are also a user member. This makes changing or deleting these directories much easier. (Refer to Technical Note 19 on changing directories.)

10. Enter the USER-GROUP subcommand if you want to place the owner of this directory into a user group. Remember that you can use only the numbers from your list of subdirectory user groups allowed. Ask your system manager how to best establish group relationships. Specifically, ask for a description of the teacher-student group relationship. If you do

not want the owners of the directories you create to read your files, you must be careful not to put the superior directory into a directory group that matches the user group numbers of its subdirectories.

11. Enter the ACCOUNT–DEFAULT command and an account that is valid for the owner of this directory. This allows the owner of this directory to log in without specifying his account. If the owner's account changes, he can use the TOPS–20 SET command to change his default account. Note that the account default parameter lasts as long as the account is valid for the owner of the directory, or until it is changed.

12. Give the GENERATIONS subcommand if the owner of the directory requires more than one copy of each file he creates to be retained on disk. Notice, however, that the more backup copies that are retained, the less file space the owner has to create additional files.

13. You can give the ABSOLUTE–ARPANET–SOCKETS, ARPANET WIZARD, CONFIDENTIAL, ENQ–DEQ, IPCF, MAINTENANCE, OPERATOR, and WHEEL capabilities to users who absolutely need them. However, you must have those capabilities yourself and you must have them enabled when you are entering them. The INFORMATION (ABOUT) DIRECTORY command for your own directory tells you which capabilities you have as a user. If you want to give the owner of a directory a capability that you do not have, create the directory without the capability and have the operator enter the capability later.

    You should confer with the system manager if you are giving powerful capabilities to users.

14. Enter the MAXIMUM–SUBDIRECTORIES subcommand if you want to allow the owner of the directory to create subdirectories of his own. The number you enter is subtracted from the quota given to the directory immediately superior to it. For example, if the directory <MATH> has a MAXIMUM SUBDIRECTORY quota of 40 and you give directory <MATH.LAB> a MAXIMUM SUBDIRECTORY quota of 10, the quota in directory <MATH> is now 29. Remember that you used one to create the directory <MATH.LAB>. If directory <MATH.LAB> gives away 4 to directory <MATH.LAB.SMITH>, the number of subdirectories that could be created under <MATH.LAB> is now 5.

    Give the INFORMATION (ABOUT) DIRECTORY command if you want to check the subdirectory quota given to a particular directory. To find out how many directories now exist, do an INFORMATION (ABOUT) DIRECTORY <directoryname.*> with the NAME–ONLY subcommand. (Refer to Technical Note 17.)

15. If you allow the owner of the directory to create directories inferior to his own (MAXIMUM SUBDIRECTORIES (ALLOWED)), you may also want to allow him to establish group relationships among those directories. If so, enter the SUBDIRECTORY USER GROUP (ALLOWED) subcommand. Take one or two numbers from your own subdirectory user groups allowed list and enter them, one at a time, for instance:

```
@@ SUBDIRECTORY-USER-GROUP (ALLOWED) 2567(RET)
@@ SUBDIRECTORY-USER-GROUP (ALLOWED) 2568(RET)
```

    The owner of the subdirectory can now create directories and enter these numbers in USER and DIRECTORY GROUP subcommands.

16. Give the FILES–ONLY subcommand if you do not want to associate this directory with a particular user. For example, you create a directory <MATH.LAB–EXER>, which several users can share as a library. These users can connect to <MATH.LAB–EXER> by giving its password. Alternatively, you can place this directory in the same group as the users who need to use the directory and perhaps not give it a password.

    You cannot log into a files-only directory or access it via the ACCESS command. Also, you cannot create log-in directories created under it. Therefore, you should not create a directory as files-only if you plan to create user directories inferior to it. For example, if you create the directory <MATH.LAB> and plan to have user directories under it (<MATH.LAB.JONES>), then do not create <MATH.LAB> as a files-only directory.

17. After you create directories, you can give the INFORMATION (ABOUT) DIRECTORY <directoryname.*> with the NAME–ONLY and VERBOSE subcommands (where directory name is your connected (superior) directory). If you do not enter the NAME–ONLY or VERBOSE subcommand, the parameters that you assigned to each subdirectory you created are typed under a separate heading. The NAME–ONLY subcommand lists only the names of the directories that were created. The VERBOSE subcommand lists all the directories you created and all the parameters about each directory.

IMPORTANT

This command works successfully only if you have placed the directories you created into a directory group that matches a user group number in the superior directory. Remember that subdirectories are just like any other directory. Unless you have access to these directories as a group member, you have "world" access when you attempt to give the INFORMATION command to list the names of your subdirectories or their contents. (Refer to the *TOPS–20 User's Guide* for a description of protection codes.)

Changing Directories

18. After you create a directory but before you give the last carriage return to complete the creation, you can give the LIST VERBOSE subcommand to check the parameters you entered for the directory. You can change any parameter at this time either by typing the appropriate subcommand with a different value or by negating a parameter using the NOT subcommand. (Refer to the description of the NOT subcommand for a list of the subcomands that accept NOT as a prefix.)

19. To change parameters of existing directories, you must first connect to the directory immediately superior to the one being changed. For example, <MATH> is immediately superior to <MATH.LAB> and <MATH.LAB> is immediately superior to <MATH.LAB.JONES>.

    Next, give the BUILD command with the name of the directory you are changing. Because the directory already exists, the system responds with <OLD> and places you in subcommand mode.

    You can now add parameters or change the ones previously given. To change a parameter, simply enter the appropriate subcommand with its new value. Follow the procedure in Technical Note 18 if you are deleting any parameters.

    If you change directories frequently on two or more levels below your connected directory, you can use the following procedure and save some typing and time. This same procedure is also helpful when you are listing the names of subdirectories as mentioned in Technical Note 17.

    - Place each directory in a directory group of which you are a user group member.
    - Verify that the group protection code (the middle two digits) on the directories is 77 in all cases. (77 allows owner privileges.)

    Now, each time you make a change to a directory that is several levels below your connected directory, you need not connect to its immediate superior. You have owner access to the directory as a group member.

NOTE

If the superior directory, (e.g., <MATH>) is not your logged-in directory and you give the CONNECT command to <MATH>, you do not have the user group privileges for that directory. Give the ACCESS command to <MATH> to gain your group relationship with the directories inferior to <MATH>. Note that you lose your own group access if <MATH> is on PS:.

Deleting Directories

20. You can delete a directory by using the KILL subcommand. First, give the BUILD command with the name of the directory to be deleted. The system responds with <OLD> and places you in subcommand mode. Enter the KILL subcommand. The system prompts you to confirm this subcommand with an extra carriage return. Give the carriage return to confirm the deletion. The system deletes the directory, then deletes and expunges all its files.

    If the directory being deleted has any inferior directories, you cannot delete it until you first delete the inferiors. For example, if you give the KILL subcommand for directory <MATH.LAB> and directory <MATH.LAB.JONES> still exists, you receive an error message. You must first connect to the immediate superior <MATH.LAB> (or use your group relationships), and give the BUILD command with the KILL subcommand to directory <MATH.LAB.JONES>. Now you can delete <MATH.LAB>.

    If you give the KILL subcommand for a directory and change your mind before you give the last carriage return to confirm it, you can give the NOT KILL subcommand and negate the entry. The following example illustrates this procedure:

```
@BUILD (DIRECTORY NAME) <MATH,JONES>(RET)
<OLD>
@@KILL(RET)
Confirm (RET)
@@NOT KILL(RET)
@@(RET)
@
```

    You can add or change any subcommands you wish or carriage return to exit this BUILD session.

Index

TOPS–20
System Manager's Guide
AD–4169F–T2

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

_____
_____
_____
_____
_____
_____
_____
_____
_____

Did you find errors in this manual? If so, specify the error and the page number.

_____
_____
_____
_____
_____
_____
_____
_____
_____

Please indicate the type of reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) _____

Name _____ Date _____

Organization _____ Telephone _____

Street _____

City _____ State _____ Zip Code _____
or Country

Do Not Tear – Fold Here and Tape

digital

No Postage Necessary if Mailed in the United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

SOFTWARE PUBLICATIONS
200 FOREST STREET MR1–2/L12
MARLBOROUGH, MASSACHUSETTS 01752

Do Not Tear – Fold Here and Tape

Cut Along Dotted Line

UPDATE NOTICE

TOPS–20
System Manager's Guide

AD-4169F-T1

December 1980

Insert this Update Notice in the *TOPS–20 System Manager's Guide* to maintain an up-to-date record of changes to the manual.

Changed Information

The changed pages contained in this update package reflect the RP20 Extension to TOPS–20 Version 4.

Software and manuals should be ordered by title and order number. In the United States, send orders to the nearest distribution center. Outside the United States, orders should be directed to the nearest DIGITAL Field Sales Office or representative.

NORTHEAST/MID-ATLANTIC REGION	CENTRAL REGION	WESTERN REGION
Technical Documentation Center Cotton Road Nashua, NH 03060 Telephone: (800) 258-1710 New Hampshire residents: (603) 884-6660	Technical Documentation Center 1050 East Remington Road Schaumburg, Illinois 60195 Telephone: (312) 640-5612	Technical Documentation Center 2525 Augustine Drive Santa Clara, California 95051 Telephone: (408) 984-0200

INSTRUCTIONS
AD-4169F-T1

The following list of page numbers specifies which pages are to be placed in the *TOPS–20 System Manager's Guide* as replacements for, or additions to, current pages.

Title page
Copyright page

4–1
4–2

4–9
4–10

7–3
7–4

KEEP THIS UPDATE NOTICE IN YOUR MANUAL TO MAINTAIN AN UP-TO-DATE RECORD OF CHANGES.

TYPE AND IDENTIFICATION OF DOCUMENTATION CHANGES

Five types of changes are used to update documents contained in the TOPS–20 software manuals. Change symbols and notations are used to specify where, when, and why alterations were made to each updated page. The five types of update changes and the manner in which each is identified are described in the following table.

The Following Symbols and/or Notations	Identify the Following Types of Update
1. Change bar in outside margin; version number and change date printed at bottom of page.	1. Changes were required by a new version of the software being described.
2. Change bar in outside margin; change date printed at bottom of page.	2. Changes were required to either clarify or correct the existing material.
3. Change date printed at bottom of page.	3. Changes were made for editorial purposes but use of the software is not affected.
4. Bullet (•) in outside margin; version number and change date printed at bottom of page.	4. Data was deleted to comply with a new version of the software being described.
5. Bullet (•) in outside margin; change date printed at bottom of page.	5. Data was deleted to either clarify or correct the existing material.

December 1980

UPDATE NOTICE

TOPS–20
System Manager's Guide
AD–4169F–T2

April 1982

Insert this Update Notice in the *TOPS–20 System Manager's Guide* to maintain an up-to-date record of changes to the manual.

Changed Information

The changed pages contained in this update package reflect TOPS–20 Version 5.0.

The instructions for inserting this update start on the next page.

digital software

INSTRUCTIONS
AD–4169F–T2

The following list of page numbers specifies which pages are to be placed in the *TOPS–20 System Manager's Guide* as replacements for, or additions to, current pages.

Title page Copyright page	3–21 Blank	5–33 5–34	9–3 9–6
Entire Contents	4–1 4–4	7–3 7–6	9–11 9–12
xi Blank	4–9 4–10	8–3 8–4	11–3 11–4
1–1 1–2	4–17 Blank	8–9 8–10	Entire Index
2–1 2–2	5–5 5–8	8–13 8–14	
3–1 3–20	5–15 5–16	8–17 Blank	

KEEP THIS UPDATE NOTICE IN YOUR MANUAL TO MAINTAIN AN UP-TO-DATE RECORD OF CHANGES.

TYPE AND IDENTIFICATION OF DOCUMENTATION CHANGES.

Five types of changes are used to update documents contained in the TOPS–20 software manuals. Change symbols and notations are used to specify where, when, and why alterations were made to each update page. The five types of update changes and the manner in which each is identified are described in the following table.

The Following Symbols and/or Notations	Identify the Following Types of Update Changes
1. Change bar in outside margin; version number and change date printed at bottom of page.	1. Changes were required by a new version of the software being described.
2. Change bar in outside margin; change date printed at bottom of page.	2. Changes were required to either clarify or correct the existing material.
3. Change date printed at bottom of page.	3. Changes were made for editorial purposes but use of the software is not affected.
4. Bullet (•) in outside margin; version number and change date printed at bottom of page.	4. Data was deleted to comply with a new version of the software being described.
5. Bullet (•) in outside margin; change date printed at bottom of page.	5. Data was deleted to either clarify or correct the existing material.

April 1982